Generalized Strong Preservation by Abstract Interpretation

FRANCESCO RANZATO FRANCESCO TAPPARO
Dipartimento di Matematica Pura ed Applicata, Università di Padova
Via Belzoni 7, 35131 Padova, Italy
francesco.ranzato@unipd.it tapparo@math.unipd.it

Abstract

Standard abstract model checking relies on abstract Kripke structures which approximate concrete models by gluing together indistinguishable states, namely by a partition of the concrete state space. Strong preservation for a specification language $\mathscr{L}$ amounts to the equivalence of concrete and abstract model checking of formulas in $\mathscr{L}$. We show how abstract interpretation can be used to design generic abstract models that allow to view standard abstract Kripke structures as particular instances. Accordingly, strong preservation is generalized to abstract interpretation-based models and precisely related to the concept of completeness in abstract interpretation. The problem of minimally refining an abstract model in order to make it strongly preserving for some language $\mathscr{L}$ can be formulated as a minimal domain refinement in abstract interpretation in order to get completeness w.r.t. the logical/temporal operators of $\mathscr{L}$. It turns out that this refined strongly preserving abstract model always exists and can be characterized as a greatest fixed point. As a consequence, some well-known behavioural equivalences, like bisimulation, simulation and stuttering, and their corresponding partition refinement algorithms can be elegantly characterized in abstract interpretation as completeness properties and refinements.

Keywords: Abstract interpretation, abstract model checking, strong preservation, completeness, refinement, behavioural equivalence.

1 Introduction

Motivations

Formal verification by model checking is a well-known and successfully applied, also in industry, technique for hardware/software system verification. In a model checking tool, an hardware/software system is represented by a formal model M, typically a Kripke structure, and an algorithm checks whether a correctness specification φ, written as a formula of a temporal language, holds on the model M or not. Approximate verification by *abstract model checking* provides one important solution to the state explosion problem that arises in model checking systems with parallel components. In abstract model checking, approximation is encoded by an abstract model A that hides some details of the concrete model M so that it becomes more efficient verifying correctness specifications on A rather than on M. The design of an abstract model checking framework always includes a preservation result, roughly stating that for any formula φ specified in some temporal language $\mathscr{L}$, if φ holds on an abstract model A then φ also holds on the concrete model M. Thus, abstract verification of φ on A may yield false negatives due to the approximation of M to A. On the other hand, strong preservation means that a formula φ of $\mathscr{L}$ holds on A if and only if φ holds on M. Strong preservation is highly desirable since it allows to draw consequences from negative answers on the abstract side. See [10] for a standard reference to model checking.

Abstract interpretation is a well-known general theory extensively used for specifying the approximation of formal semantics of computational systems at different levels of abstraction [14, 15]. This paper follows the standard abstract interpretation approach where a concrete domain of computation C is approximated by a corresponding abstract domain A through a Galois connection, namely an abstraction map $\alpha : C \to A$ that encodes the approximation together with a concretization map $\gamma : A \to C$ that provides the concrete meaning of abstract objects. Thus, $\alpha(c) \leq_A a$ and $c \leq_C \gamma(a)$ both mean that a concrete value $c \in C$ is correctly approximated by the abstract object $a \in A$. Galois connections guarantee that any concrete value in C admits a best, that is optimal, abstraction in A. A concrete semantic function $\boldsymbol{f} : C \rightarrow C$ can then be correctly approximated on the abstract domain A by a corresponding abstract function $\boldsymbol{f}^\sharp : A \rightarrow A$ that safely mimics the behaviour of $\boldsymbol{f}$ on A, that is $\alpha(\boldsymbol{f}(c)) \leq_A \boldsymbol{f}^\sharp(\alpha(c))$. We refer to [13] for an excellent overview of abstract interpretation.

The relationship between abstract model checking and abstract interpretation has been the subject of a number of works (see e.g. [9, 11, 17, 18, 20, 21, 29, 35, 41, 42, 43, 45, 48, 49]). This paper aims at studying the notion of strong preservation in abstract model checking from a generalized abstract interpretation perspective. One main goal is to apply this abstract interpretation-based view of strong preservation for understanding some common principles in well-known algorithms that refine abstract Kripke structures in order to make them strongly preserving for some temporal language.

Main Results

Abstract Semantics of Languages. In this work, we deal with generic (temporal) languages $\mathscr{L}$ of state formulae that are inductively generated by some given sets of atomic propositions and logical/temporal operators, e.g. standard temporal operators like existential/universal next EX/AX, until EU/AU, globally EG/AG, etc. The semantics of a language is determined by a suitable semantic structure $\mathcal{S}$, e.g. a Kripke structure, on a concrete state space States, that provides an interpretation of atoms and operators in $\mathscr{L}$ as, respectively, elements and operators on the powerset $\wp(\mathit{States})$. Thus, $\mathcal{S}$ determines for any formula $\varphi \in \mathscr{L}$ a concrete semantics $[\![\varphi]\!]_{\mathcal{S}} \in \wp(\mathit{States})$, namely the set of states making φ true w.r.t. $\mathcal{S}$. Abstract interpretation provides a systematic technique for approximating a concrete semantics by an *abstract semantics* defined on some abstract domain. We consider abstract domains of the powerset $\wp(\mathit{States})$ that plays here the role of concrete semantic domain. An abstract domain A, related to $\wp(\mathit{States})$ by abstraction/concretization maps α/γ, induces an abstract semantic structure $\mathcal{S}^A$ where the interpretation of an atom $\boldsymbol{p}$ is abstracted to $\alpha(\boldsymbol{p})$ while a concrete semantic operator $\boldsymbol{f}$ is abstracted by its best correct approximation on A, that is $\alpha \circ \boldsymbol{f} \circ \gamma$. Thus, any abstract domain A systematically induces an abstract semantics $[\![\varphi]\!]^A_{\mathcal{S}} \in A$ that evaluates formulae $\varphi \in \mathscr{L}$ in the abstract domain A.

It turns out that this approach based on abstract semantics generalizes standard abstract model checking [9, 10]. Given a Kripke structure $\mathcal{K} = (\mathit{States}, \rightarrow)$, where $\rightarrow$ is the transition relation between states, a standard abstract model is specified as an abstract Kripke structure $\mathcal{A} = (\mathit{AStates}, \rightarrow^\sharp)$ where the set $\mathit{AStates}$ of abstract states is defined by a surjective map $h : \mathit{States} \rightarrow \mathit{AStates}$ that groups together indistinguishable concrete states. Thus, $\mathit{AStates}$ determines a partition of States and vice versa any partition of States can be viewed as a set of abstract states. We show that state partitions can be viewed as a particular class of abstract domains. In fact, it turns out that the whole lattice of partitions of States is an abstract interpretation of the whole lattice of abstract domains of $\wp(\mathit{States})$ so that any abstract state space $\mathit{AStates}$ corresponds to a particular abstract domain of $\wp(\mathit{States})$. Abstract domains that can be derived from a state partition are called *partitioning*.

Generalized Strong Preservation. In standard abstract model checking, given a language $\mathscr{L}$ and a corresponding interpretation of a Kripke structure $\mathcal{K}$, an abstract Kripke structure $\mathcal{A}$ strongly preserves $\mathscr{L}$ when for any $\varphi \in \mathscr{L}$ and $s \in \mathit{States}$, we have that $h(s) \models^{\mathcal{A}} \varphi \;\Leftrightarrow\; s \models^{\mathcal{K}} \varphi$.

It turns out that strong preservation can be generalized from standard abstract models to abstract interpretation-based models. A generalized abstract model is given as an abstract domain A of $\wp(\mathit{States})$ that systematically induces an abstract semantics $[\![\cdot]\!]^A_{\mathcal{S}}$. We therefore define the abstract semantics $[\![\cdot]\!]^A_{\mathcal{S}}$ to be strongly preserving for $\mathscr{L}$ when for any $\varphi \in \mathscr{L}$ and $S \in \wp(\mathit{States})$, $\alpha(S) \leq_A [\![\varphi]\!]^A_{\mathcal{S}} \;\Leftrightarrow\; S \subseteq [\![\varphi]\!]_{\mathcal{S}}$. Observe that strong preservation is an abstract domain property, meaning that it does not depend on the abstract interpretation of atoms and logical/temporal operators on the abstract domain A but only depends on A itself. Standard strong preservation becomes a particular instance, because it turns out that an abstract Kripke structure strongly preserves $\mathscr{L}$ if and only if the corresponding partitioning abstract domain strongly preserves $\mathscr{L}$ in the generalized sense. On the other hand, generalized strong preservation may work where standard strong preservation may fail. In fact, it may happen that although a strongly preserving abstract semantics on a partition P always exists this abstract semantics cannot be derived from a strongly preserving abstract Kripke structure on P.

Generalized Strong Preservation and Complete Abstract Interpretations. Given a language $\mathscr{L}$ and a Kripke structure $\mathcal{K} = (States, \rightarrow)$, a well-known key problem is to compute the smallest abstract state space $AStates_{\mathscr{L}}$, when this exists, such that one can define an abstract Kripke structure $\mathcal{A}_{\mathscr{L}} = (AStates_{\mathscr{L}}, \rightarrow^{\sharp})$ that strongly preserves $\mathscr{L}$. This problem admits solution for a number of well-known temporal languages like CTL (or, equivalently, the μ-calculus), ACTL and CTL-X (i.e. CTL without the next-time operator X). A number of algorithms for solving this problem exist, like those by Paige and Tarjan [44] for CTL, by Henzinger et al. [37], Bustan and Grumberg [5], Tan and Cleaveland [50] and Gentilini et al. [28] for ACTL, and Groote and Vaandrager [33] for CTL-X. These are coarsest partition refinement algorithms: given a language $\mathscr{L}$ and a state partition P, which is determined by a state labeling, these algorithms can be viewed as computing the coarsest partition $P_{\mathscr{L}}$ that refines P and strongly preserves $\mathscr{L}$. It is worth remarking that most of these algorithms have been designed for computing well-known behavioural equivalences used in process algebra like bisimulation (for CTL), simulation (for ACTL) and divergence-blind stuttering (for CTL-X) equivalence. Our abstract interpretation-based framework allows us to provide a generalized view of the above partition refinement algorithms. We show that the most abstract (i.e., least informative) domain, denoted by $\mathrm{AD}_{\mathscr{L}}$, that strongly preserves a given language $\mathscr{L}$ always exists. It turns out that $\mathrm{AD}_{\mathscr{L}}$ is a partitioning abstract domain if and only if $\mathscr{L}$ includes propositional logic, that is when $\mathscr{L}$ is closed under logical conjunction and negation. Otherwise, a proper loss of information occurs when abstracting $\mathrm{AD}_{\mathscr{L}}$ to the corresponding partition $P_{\mathscr{L}}$. Moreover, for some languages $\mathscr{L}$, it may happen that one cannot define an abstract Kripke structure on the abstract state space $P_{\mathscr{L}}$ that strongly preserves $\mathscr{L}$ whereas the most abstract strongly preserving semantics instead exists.

The concept of *complete* abstract interpretation is well known [15, 32]. This encodes an ideal situation where the abstract semantics coincides with the abstraction of the concrete semantics. We establish a precise correspondence between generalized strong preservation of abstract models and completeness in abstract interpretation. Our results are based on the notion of *forward complete* abstract domain. An abstract domain A is forward complete for a concrete semantic function $\boldsymbol{f}$ when for any $a \in A$, $\boldsymbol{f}(\gamma(a)) = \gamma(\alpha(\boldsymbol{f}(\gamma(a))))$, namely when no loss of precision occurs by approximating in A a computation $\boldsymbol{f}(\gamma(a))$. This notion of forward completeness is dual and orthogonal to the standard definition of completeness in abstract interpretation. Giacobazzi et al. [32] showed how complete abstract domains can be systematically and constructively derived from noncomplete abstract domains by minimal refinements. We show that this can be done for forward completeness as well. Given any domain A, the most abstract domain that refines A and is forward complete for $\boldsymbol{f}$ does exist and can be characterized as a greatest fixpoint. Such a domain is called the *forward complete shell* of A for $\boldsymbol{f}$. It turns out that strong preservation is related to forward completeness as follows. As described above, the most abstract domain $\mathrm{AD}_{\mathscr{L}}$ that strongly preserves $\mathscr{L}$ always exists. It turns out that $\mathrm{AD}_{\mathscr{L}}$ coincides with the forward complete shell for the logical/temporal operators of $\mathscr{L}$ of a basic abstract domain determined by the state labeling. This characterization provides an elegant generalization of partition refinement algorithms used in standard abstract model checking. As a consequence of these results, we derive a novel characterization of the corresponding behavioural equivalences in terms of forward completeness of abstract domains. For example, we show that a partition P is a bisimulation on some Kripke structure $\mathcal{K}$ if and only if the corresponding partitioning abstract domain A_P is forward complete for the standard predecessor transformer $\mathrm{pre}_{\rightarrow}$ for the transition relation of $\mathcal{K}$.

2 Background

2.1 Notation and Preliminaries

Notations. Let X be any set. $\mathrm{Fun}(X)$ denotes the set of functions $f : X^n \rightarrow X$, for any $n \geq 0$, called arity of f. Following a standard convention, when $n = 0$, f is meant to be a specific object of X. The arity of f is also denoted by $\sharp(f) \geq 0$. id denotes the identity map. If $F \subseteq \mathrm{Fun}(X)$ and $Y \subseteq X$ then $F(Y) \overset{\text{def}}{=} \{f(\vec{y}) \mid f \in F, \vec{y} \in Y^{\sharp(f)}\}$, namely $F(Y)$ is the set of images of Y for each function in F. If $f : X \rightarrow Y$ then the image of f is also denoted by $\mathrm{img}(f) = \{f(x) \in Y \mid x \in X\}$. If $f : X \rightarrow Y$ and $g : Y \rightarrow Z$ then $g \circ f : X \rightarrow Z$ denotes the composition of f and g, i.e. $g \circ f = \lambda x.g(f(x))$. The complement operator for the universe set X is $\complement : \wp(X) \to \wp(X)$, where $\complement(S) = X \smallsetminus S$. When writing a set S of subsets of a given set, like a partition, we often write S in a compact form like $\{1, 12, 13\}$ or $\{[1], [12], [13]\}$ that stand for $\{\{1\}, \{1, 2\}, \{1, 3\}\}$. Ord denotes the proper class of ordinals.

Orders and Fixpoints. Let $\langle P, \leq \rangle$ be a poset. Posets are often denoted by $P_{\leq}$. We use the symbol $\sqsubseteq$ to denote pointwise ordering between functions: If X is any set and $f, g : X \to P$ then $f \sqsubseteq g$ if for all $x \in X$, $f(x) \leq g(x)$. A mapping $f : P \to Q$ on posets is continuous when f preserves least upper bounds (lub's) of countable chains in P, while, dually, it is co-continuous when f preserves greatest lower bounds (glb's) of countable chains in P. A complete lattice $C_{\leq}$ is also denoted by $\langle C, \leq, \vee, \wedge, \top, \bot \rangle$ where $\vee$, $\wedge$, $\top$ and $\bot$ denote, respectively, lub, glb, greatest element and least element in C. A mapping $f : C \to D$ between complete lattices is additive (co-additive) when for any $Y \subseteq C$, $f(\vee_C Y) = \vee_D f(Y)$ ($f(\wedge_C Y) = \wedge_D f(Y)$). We denote by $\mathrm{lfp}(f)$ and $\mathrm{gfp}(f)$, respectively, the least and greatest fixpoint, when they exist, of an operator f on a poset. The well-known Knaster-Tarski's theorem states that any monotone operator $f : C \to C$ on a complete lattice C admits a least fixpoint and the following characterization holds:

$$\mathrm{lfp}(f) = \vee_{\alpha \in \mathrm{Ord}} f^{\alpha,\uparrow}(\bot)$$

where the upper iteration sequence $\{f^{\alpha,\uparrow}(x)\}_{\alpha \in \mathrm{Ord}}$ of f in $x \in C$ is defined by transfinite induction on α as usual:

- $\alpha = 0$: $f^{0,\uparrow}(x) = x$;
- successor ordinal $\alpha = \beta + 1$: $f^{\beta+1,\uparrow}(x) = f(f^{\beta,\uparrow}(x))$;
- limit ordinal α: $f^{\alpha,\uparrow}(x) = \vee_{\beta<\alpha} f^{\beta,\uparrow}(x)$.

It is well known that if f is continuous then $\mathrm{lfp}(f) = \vee_{n \in \mathbb{N}} f^{n,\uparrow}(\bot)$. Dually, f also admits a greatest fixpoint and the following characterization holds:

$$\mathrm{gfp}(f) = \wedge_{\alpha \in \mathrm{Ord}} f^{\alpha,\downarrow}(\top),$$

where the lower iteration sequence $\{f^{\alpha,\downarrow}(x)\}_{\alpha \in \mathrm{Ord}}$ of f in $x \in C$ is defined as the upper iteration sequence but for the case of limit ordinals: $f^{\alpha,\downarrow}(x) = \wedge_{\beta<\alpha} f^{\beta,\downarrow}(x)$.

Partitions. Let Σ be any set. A partition P of Σ is a set of nonempty subsets of Σ, called blocks, that are pairwise disjoint and whose union gives Σ. $\mathrm{Part}(\Sigma)$ denotes the set of partitions of Σ. If $\equiv\, \subseteq \Sigma \times \Sigma$ is an equivalence relation then we denote by $P_{\equiv} \in \mathrm{Part}(\Sigma)$ the corresponding partition of Σ. Vice versa, if $P \in \mathrm{Part}(\Sigma)$ then $\equiv_P \subseteq \Sigma \times \Sigma$ denotes the corresponding equivalence relation on Σ. $\mathrm{Part}(\Sigma)$ is endowed with the following standard partial order $\preccurlyeq$: $P_1 \preccurlyeq P_2$, i.e. P_2 is coarser than P_1 (or P_1 refines P_2) iff $\forall B \in P_1. \exists B' \in P_2.\, B \subseteq B'$. It is well known that $\langle \mathrm{Part}(\Sigma), \preccurlyeq \rangle$ is a complete lattice.

Transition Systems. A transition system $\mathcal{T} = (\Sigma, \to)$ consists of a (possibly infinite) set Σ of states and a transition relation $\to\, \subseteq \Sigma \times \Sigma$. As usual [10], we assume that the relation $\to$ is total, i.e., for any $s \in \Sigma$ there exists some $t \in \Sigma$ such that $s \to t$, so that any maximal path in $\mathcal{T}$ is necessarily infinite. $\mathcal{T}$ is finitely branching when for any $s \in \Sigma$, $\{t \in \Sigma \mid s \to t\}$ is a finite set. The pre/post transformers on $\wp(\Sigma)$ are defined as usual:

- $\mathrm{pre}_{\to} \overset{\text{def}}{=} \lambda Y. \{a \in \Sigma \mid \exists b \in Y.\, a \to b\}$ (predecessor operator)
- $\widetilde{\mathrm{pre}}_{\to} \overset{\text{def}}{=} \complement \circ \mathrm{pre}_{\to} \circ \complement = \lambda Y. \{a \in \Sigma \mid \forall b \in \Sigma. (a \to b \Rightarrow b \in Y)\}$ (dual predecessor operator)
- $\mathrm{post}_{\to} \overset{\text{def}}{=} \lambda Y. \{b \in \Sigma \mid \exists a \in Y.\, a \to b\}$ (successor operator)
- $\widetilde{\mathrm{post}}_{\to} \overset{\text{def}}{=} \complement \circ \mathrm{post}_{\to} \circ \complement = \lambda Y. \{b \in \Sigma \mid \forall a \in \Sigma. (a \to b \Rightarrow a \in Y)\}$ (dual successor operator)

Let us observe that $\mathrm{pre}_{\to}$ and $\mathrm{post}_{\to}$ are additive operators on $\wp(\Sigma)_{\subseteq}$ while $\widetilde{\mathrm{pre}}_{\to}$ and $\widetilde{\mathrm{post}}_{\to}$ are co-additive.

2.2 Abstract Interpretation and Completeness

2.2.1 Abstract Domains

In standard Cousot and Cousot's abstract interpretation, abstract domains can be equivalently specified either by Galois connections, i.e. adjunctions, or by upper closure operators (uco's) [14, 15]. Let us recall these standard notions.

Galois Connections and Insertions. If A and C are posets and $\alpha : C \to A$ and $\gamma : A \to C$ are monotone functions such that $\forall c \in C.\ c \leq_C \gamma(\alpha(c))$ and $\alpha(\gamma(a)) \leq_A a$ then the quadruple (α, C, A, γ) is called a Galois connection (GC for short) between C and A. If in addition $\alpha \circ \gamma = \lambda x.x$ then (α, C, A, γ) is a Galois insertion (GI for short) of A in C. In a GI, γ is 1-1 and α is onto. Let us also recall that the notion of GC is equivalent to that of adjunction: if $\alpha : C \to A$ and $\gamma : A \to C$ then (α, C, A, γ) is a GC iff $\forall c \in C. \forall a \in A.\ \alpha(c) \leq_A a \Leftrightarrow c \leq_C \gamma(a)$. The map α (γ) is called the left- (right-) adjoint to γ (α). It turns out that one adjoint map α/γ uniquely determines the other adjoint map γ/α as follows. On the one hand, a map $\alpha : C \to A$ admits a necessarily unique right-adjoint map $\gamma : A \to C$ iff α preserves arbitrary lub's; in this case, we have that $\gamma \overset{\text{def}}{=} \lambda a.\ \vee_C \{c \in C \mid \alpha(c) \leq_A a\}$. On the other hand, a map $\gamma : A \to C$ admits a necessarily unique left-adjoint map $\alpha : C \to A$ iff γ preserves arbitrary glb's; in this case, $\alpha \overset{\text{def}}{=} \lambda c.\ \wedge_A \{a \in A \mid c \leq_C \gamma(a)\}$. In particular, in any GC (α, C, A, γ) between complete lattices it turns out that α is additive and γ is co-additive.

We assume the standard abstract interpretation framework, where concrete and abstract domains, C and A, are complete lattices related by abstraction and concretization maps α and γ forming a GC (α, C, A, γ). A is called an abstraction of C and C a concretization of A. The ordering relations on concrete and abstract domains describe the relative precision of domain values: $x \leq y$ means that y is an approximation of x or, equivalently, x is more precise than y. Galois connections allow to relate the concrete and abstract notions of relative precision: an abstract value $a \in A$ approximates a concrete value $c \in C$ when $\alpha(c) \leq_A a$, or, equivalently (by adjunction), $c \leq_C \gamma(a)$. As a key consequence of requiring a Galois connection, it turns out that $\alpha(c)$ is the best possible approximation in A of c, that is $\alpha(c) = \wedge\{a \in A \mid c \leq_C \gamma(a)\}$ holds. If (α, C, A, γ) is a GI then each value of the abstract domain A is useful in representing C, because all the values in A represent distinct members of C, being γ 1-1. Any GC can be lifted to a GI by identifying in an equivalence class those values of the abstract domain with the same concretization. $\mathrm{Abs}(C)$ denotes the set of abstract domains of C and we write $A \in \mathrm{Abs}(C)$ to mean that the abstract domain A is related to C through a GI (α, C, A, γ). An abstract domain A is disjunctive when the corresponding concretization map γ is additive.

Closure Operators. An (upper) closure operator, or simply a closure, on a poset $P_{\leq}$ is an operator $\mu : P \to P$ that is monotone, idempotent and extensive, i.e., $\forall x \in P.\ x \leq \mu(x)$. Dually, lower closure operators are monotone, idempotent, and restrictive, i.e., $\forall x \in P.\ \mu(x) \leq x$. $\mathrm{uco}(P)$ denotes the set of closure operators on P. Let $\langle C, \leq, \vee, \wedge, \top, \bot \rangle$ be a complete lattice. A closure $\mu \in \mathrm{uco}(C)$ is uniquely determined by its image $\mathrm{img}(\mu)$, which coincides with its set of fixpoints, as follows: $\mu = \lambda y.\ \wedge \{x \in \mathrm{img}(\mu) \mid y \leq x\}$. Also, $X \subseteq C$ is the image of some closure operator μ_X on C iff X is a Moore-family of C, i.e., $X = \mathcal{M}(X) \overset{\text{def}}{=} \{\wedge S \mid S \subseteq X\}$ — where $\wedge\varnothing = \top \in \mathcal{M}(X)$. In other terms, X is a Moore-family of C when X is meet-closed. In this case, $\mu_X = \lambda y.\ \wedge \{x \in X \mid y \leq x\}$ is the corresponding closure operator on C. For any $X \subseteq C$, $\mathcal{M}(X)$ is called the Moore-closure of X in C, i.e., $\mathcal{M}(X)$ is the least (w.r.t. set inclusion) subset of C which contains X and is a Moore-family of C. Moreover, it turns out that for any $\mu \in \mathrm{uco}(C)$ and any Moore-family $X \subseteq C$, $\mu_{\mathrm{img}(\mu)} = \mu$ and $\mathrm{img}(\mu_X) = X$. Thus, closure operators on C are in bijection with Moore-families of C. This allows us to consider a closure operator $\mu \in \mathrm{uco}(C)$ both as a function $\mu : C \to C$ and as a Moore-family $\mathrm{img}(\mu) \subseteq C$. This is particularly useful and does not give rise to ambiguity since one can distinguish the use of a closure μ as function or set according to the context.

It turns out that $\langle \mu, \leq \rangle$ is a complete meet subsemilattice of C, i.e. $\wedge$ is its glb, but, in general, it is not a complete sublattice of C, since the lub in μ — defined by $\lambda Y \subseteq \mu.\ \mu(\vee Y)$ — might be different from that in C. In fact, it turns out that μ is a complete sublattice of C (namely, $\mathrm{img}(\mu)$ is also join-closed) iff μ is additive.

If C is a complete lattice then $\mathrm{uco}(C)$ endowed with the pointwise ordering $\sqsubseteq$ is a complete lattice denoted by $\langle \mathrm{uco}(C), \sqsubseteq, \sqcup, \sqcap, \lambda x.\top, \lambda x.x\rangle$, where for every $\mu, \eta \in \mathrm{uco}(C)$, $\{\mu_i\}_{i\in I} \subseteq \mathrm{uco}(C)$ and $x \in C$:

- $\mu \sqsubseteq \eta$ iff $\forall y \in C.\ \mu(y) \leq \eta(y)$ iff $\mathrm{img}(\eta) \subseteq \mathrm{img}(\mu)$;
- $(\sqcap_{i\in I}\mu_i)(x) = \wedge_{i\in I}\mu_i(x)$;
- $x \in \sqcup_{i\in I}\mu_i \;\Leftrightarrow\; \forall i \in I.\ x \in \mathrm{img}(\mu_i)$;
- $\lambda x.\top$ is the greatest element, whereas $\lambda x.x$ is the least element.

Thus, the glb in $\mathrm{uco}(C)$ is defined pointwise, while the lub of a set of closures $\{\mu_i\}_{i\in I} \subseteq \mathrm{uco}(C)$ is the closure whose image is given by the set-intersection $\cap_{i\in I}\mu_i$.

Closures are Equivalent to Galois Insertions. It is well known since [15] that abstract domains can be equivalently specified either as Galois insertions or as closures. These two approaches are completely equivalent. On the one hand, if $\mu \in \mathrm{uco}(C)$ and A is a complete lattice which is isomorphic to $\mathrm{img}(\mu)$, where $\iota : \mathrm{img}(\mu) \to A$ and $\iota^{-1} : A \to \mathrm{img}(\mu)$ provide the isomorphism, then $(\iota \circ \mu, C, A, \iota^{-1})$ is a GI. On the other hand, if (α, C, A, γ) is a GI then $\mu_A \stackrel{\text{def}}{=} \gamma \circ \alpha \in \mathrm{uco}(C)$ is the closure associated with A such that $\langle \mathrm{img}(\mu_A), \leq_C\rangle$ is a complete lattice which is isomorphic to $\langle A, \leq_A\rangle$. Furthermore, these two constructions are inverse of each other. Let us also remark that an abstract domain A is disjunctive iff μ_A is additive. Given an abstract domain A specified by a GI (α, C, A, γ), its associated closure $\gamma \circ \alpha$ on C can be thought of as the “logical meaning” of A in C, since this is shared by any other abstract representation for the objects of A. Thus, the closure operator approach is particularly convenient when reasoning about properties of abstract domains independently from the representation of their objects.

The Lattice of Abstract Domains. Abstract domains specified by GIs can be pre-ordered w.r.t. precision as follows: if $A_1, A_2 \in \mathrm{Abs}(C)$ then A_1 is more precise (or concrete) than A_2 (or A_2 is an abstraction of A_1), denoted by $A_1 \preceq A_2$, when $\mu_{A_1} \sqsubseteq \mu_{A_2}$. The pointwise ordering $\sqsubseteq$ between uco’s corresponds therefore to the standard ordering used to compare abstract domains with respect to their precision. Also, A_1 and A_2 are equivalent, denoted by $A_1 \simeq A_2$, when their associated closures coincide, i.e. $\mu_{A_1} = \mu_{A_2}$. Hence, the quotient $\mathrm{Abs}(C)_{/\simeq}$ gives rise to a poset that, by a slight abuse of notation, is simply denoted by $\langle \mathrm{Abs}(C), \sqsubseteq\rangle$. Thus, when we write $A \in \mathrm{Abs}(C)$ we mean that A is any representative of an equivalence class in $\mathrm{Abs}(C)_{/\simeq}$ and is specified by a Galois insertion (α, C, A, γ). It turns out that $\langle \mathrm{Abs}(C), \sqsubseteq\rangle$ is a complete lattice, called the lattice of abstract interpretations of C [14, 15], because it is isomorphic to the complete lattice $\langle \mathrm{uco}(C), \sqsubseteq\rangle$. Lub’s and glb’s in $\mathrm{Abs}(C)$ have therefore the following reading as operators on domains. Let $\{A_i\}_{i\in I} \subseteq \mathrm{Abs}(C)$: (i) $\sqcup_{i\in I}A_i$ is the most concrete among the domains which are abstractions of all the A_i’s; (ii) $\sqcap_{i\in I}A_i$ is the most abstract among the domains which are more concrete than every A_i — this latter domain is also known as reduced product [15] of all the A_i’s.

2.2.2 Completeness in Abstract Interpretation

Correct Abstract Interpretations. Let C be a concrete domain, $f : C \to C$ be a concrete semantic function[1] and $f^\sharp : A \to A$ be a corresponding abstract function on an abstract domain $A \in \mathrm{Abs}(C)$ specified by a GI (α, C, A, γ). Then, $\langle A, f^\sharp\rangle$ is a sound (or correct) abstract interpretation when $\alpha \circ f \sqsubseteq f^\sharp \circ \alpha$ holds. The abstract function $f^\sharp$ is called a correct approximation on A of f. This means that a concrete computation $f(c)$ can be correctly approximated in A by $f^\sharp(\alpha(c))$, namely $\alpha(f(c)) \leq_A f^\sharp(\alpha(c))$. An abstract function $f_1^\sharp : A \to A$ is more precise than $f_2^\sharp : A \to A$ when $f_1^\sharp \sqsubseteq f_2^\sharp$. Since $\alpha \circ f \sqsubseteq f^\sharp \circ \alpha$ holds iff $\alpha \circ f \circ \gamma \sqsubseteq f^\sharp$ holds, the abstract function $f^A \stackrel{\text{def}}{=} \alpha \circ f \circ \gamma : A \to A$ is called the best correct approximation of f in A.

[1]For simplicity of notation we consider here unary functions since the extension to generic n-ary functions is straightforward.

Complete Abstract Interpretations. Completeness in abstract interpretation corresponds to requiring that, in addition to soundness, no loss of precision occurs when $f(c)$ is approximated in A by $f^\sharp(\alpha(c))$. Thus, completeness of $f^\sharp$ for f is encoded by the equation $\alpha \circ f = f^\sharp \circ \alpha$. This is also called backward completeness because a dual form of forward completeness may be considered. As a very simple example, let us consider the abstract domain *Sign* representing the sign of an integer variable, namely $\mathit{Sign} = \{\bot, \mathbb{Z}_{\leq 0}, 0, \mathbb{Z}_{\geq 0}, \top\} \in \mathrm{Abs}(\wp(\mathbb{Z})_\subseteq)$. Let us consider the binary concrete operation of integer addition on sets of integers, that is $X + Y \overset{\text{def}}{=} \{x + y \mid x \in X,\, y \in Y\}$, and the square operator on sets of integers, that is $X^2 \overset{\text{def}}{=} \{x^2 \mid x \in X\}$. It turns out that the best correct approximation $+^{\mathit{Sign}}$ of integer addition in *Sign* is sound but not complete — because $\alpha(\{-1\} + \{1\}) = 0 <_{\mathit{Sign}} \top = \alpha(\{-1\}) +^{\mathit{Sign}} \alpha(\{1\})$ — while it is easy to check that the best correct approximation of the square operation in *Sign* is instead complete.

A dual form of completeness can be considered. The soundness condition $\alpha \circ f \sqsubseteq f^\sharp \circ \alpha$ can be equivalently formulated as $f \circ \gamma \sqsubseteq \gamma \circ f^\sharp$. Forward completeness for $f^\sharp$ corresponds to requiring that the equation $f \circ \gamma = \gamma \circ f^\sharp$ holds, and therefore means that no loss of precision occurs when a concrete computation $f(\gamma(a))$, for some abstract value $a \in A$, is approximated in A by $f^\sharp(a)$. Let us notice that backward and forward completeness are orthogonal concepts. In fact: (1) as observed above, we have that $+^{\mathit{Sign}}$ is not backward complete while it is forward complete because for any $a_1, a_2 \in \mathit{Sign}$, $\gamma(a_1) + \gamma(a_2) = \gamma(a_1 +^{\mathit{Sign}} a_2)$: for instance, $\gamma(\mathbb{Z}_{\geq 0}) + \gamma(\mathbb{Z}_{\geq 0}) = \mathbb{Z}_{\geq 0} = \gamma(\mathbb{Z}_{\geq 0} +^{\mathit{Sign}} \mathbb{Z}_{\geq 0})$; (2) the best correct approximation $(\cdot)^{2_{\mathit{Sign}}}$ of the square operator on *Sign* is not forward complete because $\gamma(\mathbb{Z}_{\geq 0})^2 \subsetneq \gamma(\mathbb{Z}_{\geq 0}) = \gamma((\mathbb{Z}_{\geq 0})^{2_{\mathit{Sign}}})$ while, as observed above, it is instead backward complete.

Completeness is an Abstract Domain Property. Giacobazzi et al. [32] observed that completeness uniquely depends upon the abstraction map, i.e. upon the abstract domain. This means that if $f^\sharp$ is backward complete for f then the best correct approximation f^A of f in A is backward complete as well, and, in this case, $f^\sharp$ indeed coincides with f^A. Hence, for any abstract domain A, one can define a backward complete abstract operation $f^\sharp$ on A if and only if f^A is backward complete. Thus, an abstract domain $A \in \mathrm{Abs}(C)$ is defined to be backward complete for f iff the equation $\alpha \circ f = f^A \circ \alpha$ holds. This simple observation makes backward completeness an abstract domain property, namely an intrinsic characteristic of the abstract domain. Let us observe that $\alpha \circ f = f^A \circ \alpha$ holds iff $\gamma \circ \alpha \circ f = \gamma \circ f^A \circ \alpha = \gamma \circ \alpha \circ f \circ \gamma \circ \alpha$ holds, so that A is backward complete for f when $\mu_A \circ f = \mu_A \circ f \circ \mu_A$. Thus, a closure $\mu \in \mathrm{uco}(C)$, that defines some abstract domain, is backward complete for f when $\mu \circ f = \mu \circ f \circ \mu$ holds. Analogous observations apply to forward completeness, which is also an abstract domain property: $A \in \mathrm{Abs}(C)$ is forward complete for f (or forward f-complete) when $f \circ \mu_A = \mu_A \circ f \circ \mu_A$, while a closure $\mu \in \mathrm{uco}(C)$ is forward complete for f when $f \circ \mu = \mu \circ f \circ \mu$ holds.

Fixpoint Completeness. Let us also recall that, by a well-known result (see, e.g., [15, Theorem 7.1.0.4], [1, Fact 2.3] and [22, Lemma 4.3]), backward complete abstract domains are "fixpoint complete" as well. This means that if $A \in \mathrm{Abs}(C)$ is backward complete for a concrete monotone function $f : C \to C$ then $\alpha(\mathrm{lfp}(f)) = \mathrm{lfp}(f^A)$. Moreover, if α and f are both co-continuous then this also holds for greatest fixpoints, namely $\alpha(\mathrm{gfp}(f)) = \mathrm{gfp}(f^A)$. As far as forward completeness is concerned, to the best of our knowledge, no similar results of fixpoint transfer are available. We thus prove the following result.

Lemma 2.1. *If $A \in \mathrm{Abs}(C)$ is forward complete for a monotone f then $\alpha(\mathrm{gfp}(f)) = \mathrm{gfp}(f^A)$. Moreover, if γ and f are both continuous and $\gamma(\bot_A) = \bot_C$ then $\alpha(\mathrm{lfp}(f)) = \mathrm{lfp}(f^A)$.*

Proof. Let us show that $\alpha(\mathrm{gfp}(f)) = \mathrm{gfp}(f^A)$. On the one hand, since $\mathrm{gfp}(f) \leq \gamma(\alpha(\mathrm{gfp}(f)))$, we have that $\mathrm{gfp}(f) = f(\mathrm{gfp}(f)) \leq f(\gamma(\alpha(\mathrm{gfp}(f))))$, therefore, by using forward completeness, $\mathrm{gfp}(f) \leq \gamma(f^A(\alpha(\mathrm{gfp}(f))))$. Thus, $\alpha(\mathrm{gfp}(f)) \leq f^A(\alpha(\mathrm{gfp}(f)))$, from which follows that $\alpha(\mathrm{gfp}(f)) \leq \mathrm{gfp}(f^A)$. On the other hand, by using forward completeness, $f(\gamma(\mathrm{gfp}(f^A))) = \gamma(f^A(\mathrm{gfp}(f^A))) = \gamma(\mathrm{gfp}(f^A))$, so that $\gamma(\mathrm{gfp}(f^A)) \leq \mathrm{gfp}(f)$, and therefore, by applying α, we obtain that $\mathrm{gfp}(f^A) = \alpha(\gamma(\mathrm{gfp}(f^A))) \leq \alpha(\mathrm{gfp}(f))$.
Assume now that γ and f are both continuous and $\gamma(\bot_A) = \bot_C$. Let us show by induction on k that for any $k \in \mathbb{N}$, $\gamma((f^A)^{k,\uparrow}(\bot_A)) = f^{k,\uparrow}(\bot_C)$.
($k = 0$): By hypothesis, $\gamma((f^A)^{0,\uparrow}(\bot_A)) = \gamma(\bot_A) = \bot_C = f^{0,\uparrow}(\bot_C)$.

$(k+1)$:

$$\begin{aligned}
\gamma((f^A)^{k+1,\uparrow}(\bot_A)) &= \\
\gamma(f^A((f^A)^{k,\uparrow}(\bot_A))) &= && \text{[by forward completeness]} \\
f(\gamma((f^A)^{k,\uparrow}(\bot_A))) &= && \text{[by inductive hypothesis]} \\
f(f^{k,\uparrow}(\bot_C)) &= \\
f^{k+1,\uparrow}(\bot_C)).
\end{aligned}$$

Thus, by applying α, we obtain that for any $k \in \mathbb{N}$,

$$(f^A)^{k,\uparrow}(\bot_A) = \alpha(f^{k,\uparrow}(\bot_C)). \qquad (\dagger)$$

Since γ and f are continuous and α is always additive, we have that $f^A = \alpha \circ f \circ \gamma$ is continuous because it is a composition of continuous functions. Hence:

$$\begin{aligned}
\mathrm{lfp}(f^A) &= && \text{[by Knaster-Tarski's theorem]} \\
\vee_{k\in\mathbb{N}}(f^A)^{k,\uparrow}(\bot_A) &= && \text{[by } (\dagger)\text{]} \\
\vee_{k\in\mathbb{N}}\alpha(f^{k,\uparrow}(\bot_C)) &= && \text{[as } \alpha \text{ is additive]} \\
\alpha(\vee_{k\in\mathbb{N}}f^{k,\uparrow}(\bot_C)) &= && \text{[by Knaster-Tarski's theorem]} \\
\alpha(\mathrm{lfp}(f))
\end{aligned}$$

and this concludes the proof. □

It is worth noting that concretization maps of abstract domains which satisfies the ascending chain conditions (i.e., every ascending chain is eventually stationary) are always trivially continuous.

2.2.3 Shells

Refinements of abstract domains have been studied from the beginning of abstract interpretation [14, 15] and led to the notion of shell of an abstract domain [27, 30, 32]. Given a generic poset $P_\leq$ of semantic objects — where $x \leq y$ intuitively means that x is a "refinement" of y — and a property $\mathcal{P} \subseteq P$ of these objects, the generic notion of *shell* goes as follows: the $\mathcal{P}$-shell of an object $x \in P$ is defined to be an object $s_x \in P$ such that:

(i) s_x satisfies the property $\mathcal{P}$,

(ii) s_x is a refinement of x, and

(iii) s_x is the greatest among the objects in P satisfying (i) and (ii).

Note that if a $\mathcal{P}$-shell exists then it is unique. Moreover, if the $\mathcal{P}$-shell exists for any object in P then it turns out that the operator that maps any $x \in P$ to its $\mathcal{P}$-shell is a lower closure operator on $\mathcal{P}$, being monotone, idempotent and reductive: this is called the *$\mathcal{P}$-shell refinement* operator. We will be interested in shells of abstract domains and partitions, namely shells in the complete lattices of abstract domains and partitions. Given a state space Σ and a partition property $\mathcal{P} \subseteq \mathrm{Part}(\Sigma)$, the $\mathcal{P}$-shell of $P \in \mathrm{Part}(\Sigma)$ is the coarsest refinement of P satisfying $\mathcal{P}$, when this exists. Also, given a concrete domain C and a domain property $\mathcal{P} \subseteq \mathrm{Abs}(C)$, the $\mathcal{P}$-shell of $A \in \mathrm{Abs}(C)$, when this exists, is the most abstract domain that satisfies $\mathcal{P}$ and refines A. Giacobazzi et al. [32] gave a constructive characterization of backward complete abstract domains, under the assumption of dealing with continuous concrete functions. As a consequence, they showed that backward complete shells always exist when the concrete functions are continuous. In Section 6 we will follow this same idea for forward completeness and this will provide the link between strongly preserving abstract models and complete abstract interpretations.

2.3 Abstract Model Checking and Strong Preservation

Kripke Structures. Standard temporal languages like CTL, CTL*, ACTL, the μ-calculus, LTL, etc., are interpreted on models specified as Kripke structures. Given a set AP of atomic propositions (of some language), a Kripke structure $\mathcal{K} = (\Sigma, \rightarrow, \ell)$ over AP consists of a transition system $(\Sigma, \rightarrow)$ together with a state labeling function $\ell : \Sigma \rightarrow \wp(AP)$. We use the following notation: for any $s \in \Sigma$, $[s]_\ell \overset{\text{def}}{=} \{s' \in \Sigma \mid \ell(s) = \ell(s')\}$, while $P_\ell \overset{\text{def}}{=} \{[s]_\ell \mid s \in \Sigma\} \in \mathrm{Part}(\Sigma)$ denotes the state partition that is induced by ℓ. The notation $s {\models^{\mathcal{K}}} \varphi$ means that a state $s \in \Sigma$ satisfies in $\mathcal{K}$ a state formula φ of some language $\mathscr{L}$, where the specific definition of the satisfaction relation $\models^{\mathcal{K}}$ depends on the language $\mathscr{L}$ (interpretations of standard logical/temporal operators can be found in [10]).

Abstract Kripke Structures and Strong Preservation. Standard abstract model checking [9, 10] relies on abstract Kripke structures that are defined over partitions of the concrete state space Σ. A set A of abstract states is related to Σ by a surjective abstraction $h : \Sigma \rightarrow A$ that maps concrete states into abstract states and thus gives rise to a state partition $P_h \overset{\text{def}}{=} \{h^{-1}(a) \mid a \in A\} \in \mathrm{Part}(\Sigma)$. Thus, in standard abstract model checking, formulae are interpreted on an abstract Kripke structure $\mathcal{A} = (A, \rightarrow^\sharp, \ell^\sharp)$ whose states are an abstract representation in A of some block of the partition P_h. Given a specification language $\mathscr{L}$ of state formulae, a weak preservation result for $\mathscr{L}$ guarantees that if a formula in $\mathscr{L}$ holds on an abstract Kripke structure $\mathcal{A}$ then it also holds on the corresponding concrete structure $\mathcal{K}$: for any $\varphi \in \mathscr{L}$, $a \in A$ and $s \in \Sigma$ such that $h(s) = a$, if $a {\models^{\mathcal{A}}} \varphi$ then $s {\models^{\mathcal{K}}} \varphi$. Moreover, strong preservation (s.p. for short) for $\mathscr{L}$ encodes the equivalence of abstract and concrete validity for formulae in $\mathscr{L}$: for any $\varphi \in \mathscr{L}$, $a \in A$ and $s \in \Sigma$ such that $h(s) = a$, $a {\models^{\mathcal{A}}} \varphi$ if and only if $s {\models^{\mathcal{K}}} \varphi$.

The definition of weakly/strongly preserving abstract Kripke structures depends on the language $\mathscr{L}$. Let us recall some well-known examples [9, 10, 34]. Let $\mathcal{K} = (\Sigma, \rightarrow, \ell)$ be a concrete Kripke structure and $h : \Sigma \rightarrow A$ be a surjection.

(i) Consider the language ACTL*. If $P_h \preccurlyeq P_\ell$ then the abstract Kripke structure $\mathcal{A} = (A, \rightarrow_h^{\exists\exists}, \ell_h)$ weakly preserves ACTL*, where $\ell_h(a) = \cup\{\ell(s) \mid s \in \Sigma,\ h(s) = a\}$ and $\rightarrow_h^{\exists\exists} \subseteq A \times A$ is defined as: $h(s_1) \rightarrow_h^{\exists\exists} h(s_2) \;\Leftrightarrow\; \exists s_1', s_2'.\, h(s_1') = h(s_1) \;\&\; h(s_2') = h(s_2) \;\&\; s_1' \rightarrow s_2'$.

(ii) Let $P_{\text{sim}} \in \mathrm{Part}(\Sigma)$ be the partition induced by simulation equivalence on $\mathcal{K}$. If $P_h = P_{\text{sim}}$ (this also holds when $P_h \preccurlyeq P_{\text{sim}}$) then the abstract Kripke structure $\mathcal{A} = (A, \rightarrow_h^{\forall\exists}, \ell_h)$ strongly preserves ACTL*, where $h(s_1) \rightarrow_h^{\forall\exists} h(s_2) \;\Leftrightarrow\; \forall s_1'.\, h(s_1') = h(s_1).\, \exists s_2'.\, h(s_2') = h(s_2) \;\&\; s_1' \rightarrow s_2'$.

(iii) Let $P_{\text{bis}} \in \mathrm{Part}(\Sigma)$ be the partition induced by bisimulation equivalence on $\mathcal{K}$. If $P_h = P_{\text{bis}}$ (this also holds when $P_h \preccurlyeq P_{\text{bis}}$) then the abstract Kripke structure $\mathcal{A} = (A, \rightarrow_h^{\exists\exists}, \ell_h)$ strongly preserves CTL*.

Strongly Preserving Partitions. Following Dams [20, Section 6.1] and Henzinger et al. [38, Section 2.2], the notion of strong preservation can be also given w.r.t. a mere state partition rather than w.r.t. an abstract Kripke structure. Let $[\![\cdot]\!]_{\mathcal{K}} : \mathscr{L} \rightarrow \wp(\Sigma)$ be the semantic function of state formulae in $\mathscr{L}$ w.r.t. a Kripke structure $\mathcal{K} = (\Sigma, \rightarrow, \ell)$, i.e., $[\![\varphi]\!]_{\mathcal{K}} \overset{\text{def}}{=} \{s \in \Sigma \mid s {\models^{\mathcal{K}}} \varphi\}$. Then, the semantic interpretation of $\mathscr{L}$ on $\mathcal{K}$ induces the following logical equivalence $\equiv_{\mathscr{L}}^{\mathcal{K}} \subseteq \Sigma \times \Sigma$:

$$s \equiv_{\mathscr{L}}^{\mathcal{K}} s' \ \text{ iff } \ \forall \varphi \in \mathscr{L}.\, s \in [\![\varphi]\!]_{\mathcal{K}} \;\Leftrightarrow\; s' \in [\![\varphi]\!]_{\mathcal{K}}.$$

Let $P_{\mathscr{L}} \in \mathrm{Part}(\Sigma)$ be the partition induced by $\equiv_{\mathscr{L}}^{\mathcal{K}}$ (the index $\mathcal{K}$ denoting the Kripke structure is omitted). Then, a partition $P \in \mathrm{Part}(\Sigma)$ is strongly preserving[2] for $\mathscr{L}$ (when interpreted on $\mathcal{K}$) if $P \preccurlyeq P_{\mathscr{L}}$. Thus, $P_{\mathscr{L}}$ is the coarsest partition that is strongly preserving for $\mathscr{L}$. For a number of well known temporal languages, like ACTL*, CTL* (see, respectively, the above points (ii) and (iii)), CTL*-X and the fragments of the μ-calculus described by Henzinger et al. [38], it turns out that if P is strongly preserving for $\mathscr{L}$ then the abstract Kripke structure $(P, \rightarrow^{\exists\exists}, \ell_{\mathscr{L}})$ is strongly preserving for $\mathscr{L}$, where, for any $B, B' \in P$, $B \rightarrow^{\exists\exists} B'$ iff $\exists s \in B. \exists s' \in B'.\, s \rightarrow s'$, and $\ell_{\mathscr{L}}(B) = \cup_{s \in B} \ell(s)$. In particular, $(P_{\mathscr{L}}, \rightarrow^{\exists\exists}, \ell_{\mathscr{L}})$

[2]Dams [20] uses the term "fine" instead of "strongly preserving".

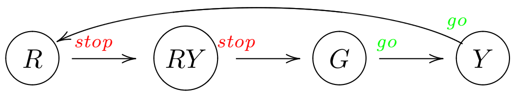

Figure 1: A U.K. traffic light.

is strongly preserving for $\mathscr{L}$ and, additionally, $P_{\mathscr{L}}$ is the smallest possible abstract state space, namely if $\mathcal{A} = (A, \to^\sharp, \ell^\sharp)$ is an abstract Kripke structure that strongly preserves $\mathscr{L}$ then $|P_{\mathscr{L}}| \leq |A|$.

However, given a language $\mathscr{L}$ and a Kripke structure $\mathcal{K}$ where formulae of $\mathscr{L}$ are interpreted, the following example shows that it is not always possible to define an abstract Kripke structure $\mathcal{A}$ on the partition $P_{\mathscr{L}}$ such that $\mathcal{A}$ strongly preserves $\mathscr{L}$.

Example 2.2. Consider the following simple language $\mathscr{L}$:

$$\mathscr{L} \ni \varphi ::= \mathit{stop} \mid \mathit{go} \mid \mathrm{AXX}\varphi$$

and the Kripke structure $\mathcal{K}$ depicted in Figure 1, where superscripts determine the labeling function. $\mathcal{K}$ models a four-state traffic light controller (like in the U.K. and in Germany): Red $\to$ RedYellow $\to$ Green $\to$ Yellow. According to the standard semantics of AXX, we have that $s|\!=^{\mathcal{K}}\mathrm{AXX}\varphi$ iff for any path $s_0s_1s_2\ldots$ starting from $s_0 = s$, it happens that $s_2|\!=^{\mathcal{K}}\varphi$. It turns out that $[\![\mathrm{AXX}\mathit{stop}]\!]_{\mathcal{K}} = \{G, Y\}$ and $[\![\mathrm{AXX}\mathit{go}]\!]_{\mathcal{K}} = \{R, RY\}$. Thus, we have that $P_{\mathscr{L}} = \{\{R, RY\}, \{G, Y\}\}$. However, let us show that there exists no abstract transition relation $\to^\sharp \subseteq P_{\mathscr{L}} \times P_{\mathscr{L}}$ such that the abstract Kripke structure $\mathcal{A} = (P_{\mathscr{L}}, \to^\sharp, \ell_{\mathscr{L}})$ strongly preserves $\mathscr{L}$. Assume by contradiction that such an abstract Kripke structure $\mathcal{A}$ exists. Let $B_1 = \{R, RY\} \in P_{\mathscr{L}}$ and $B_2 = \{G, Y\} \in P_{\mathscr{L}}$. Since $R|\!=^{\mathcal{K}}\mathrm{AXX}\mathit{go}$ and $G|\!=^{\mathcal{K}}\mathrm{AXX}\mathit{stop}$, by strong preservation, it must be that $B_1|\!=^{\mathcal{A}}\mathrm{AXX}\mathit{go}$ and $B_2|\!=^{\mathcal{A}}\mathrm{AXX}\mathit{stop}$. Hence, necessarily, $B_1\to^\sharp B_2$ (otherwise B_1 can never reach the state B_2 where the atom go holds) and $B_2\to^\sharp B_1$ (otherwise B_2 can never reach the state B_1 where the atom stop holds). This leads to the contradiction $B_1 \not\models^{\mathcal{A}}\mathrm{AXX}\mathit{go}$. In fact, if $\to^\sharp = \{(B_1, B_2), (B_2, B_1)\}$ then we would have that $B_1 \not\models^{\mathcal{A}}\mathrm{AXX}\mathit{go}$. On the other hand, if, instead, $B_1\to^\sharp B_1$ (the case $B_2\to^\sharp B_2$ is analogous), then we would still have that $B_1 \not\models^{\mathcal{A}}\mathrm{AXX}\mathit{go}$. Even more, along the same lines it is not hard to show that no proper abstract Kripke structure that strongly preserves $\mathscr{L}$ can be defined, because even if either B_1 or B_2 is split we still cannot define an abstract transition relation that is strongly preserving for $\mathscr{L}$. □

3 Partitions as Abstract Domains

Let Σ be any (possibly infinite) set of states. Following [16, Section 5], a partition $P \in \mathrm{Part}(\Sigma)$ can be viewed as an abstraction of $\wp(\Sigma)_\subseteq$ as follows: any $S \subseteq \Sigma$ is over approximated by the unique minimal cover of S in P, namely by the union of all the blocks $B \in P$ such that $B \cap S \neq \varnothing$. A graphical example is depicted on the left-hand side of Figure 2. This abstraction is formalized by a GI $(\alpha_P, \wp(\Sigma)_\subseteq, \wp(P)_\subseteq, \gamma_P)$ where:

$$\alpha_P(S) \overset{\text{def}}{=} \{B \in P \mid B \cap S \neq \varnothing\} \qquad \gamma_P(\mathcal{B}) \overset{\text{def}}{=} \cup_{B\in\mathcal{B}}\, B.$$

Hence, any partition $P \in \mathrm{Part}(\Sigma)$ induces an abstract domain $\mathrm{ad^p}(P) \in \mathrm{Abs}(\wp(\Sigma))$, and an abstract domain $A \in \mathrm{Abs}(\wp(\Sigma))$ is called *partitioning* when A is equivalent to $\mathrm{ad^p}(P)$ for some partition P. Observe that the closure $\mathrm{ad^p}(P) = \gamma_P \circ \alpha_P$ associated to a partitioning abstract domain is defined as $\mathrm{ad^p}(P) = \lambda S. \cup\{B \in P \mid B \cap S \neq \varnothing\}$. Accordingly, a closure $\mu \in \mathrm{uco}(\wp(\Sigma))$ that coincides with $\gamma_P \circ \alpha_P$, for some partition P, is called partitioning. We denote by $\mathrm{Abs^{par}}(\wp(\Sigma))$ and $\mathrm{uco^{par}}(\wp(\Sigma))$ the sets of, respectively, partitioning abstract domains and closures on $\wp(\Sigma)$. As noted in [17], a surjective abstraction $h : \Sigma \to A$ used in standard abstract model checking that maps concrete states into abstract states (cf. Section 2.3) gives rise to a partitioning Galois insertion $(\alpha_h, \wp(\Sigma)_\subseteq, \wp(A)_\subseteq, \gamma_h)$ where $\alpha_h \overset{\text{def}}{=} \lambda S \subseteq \Sigma.\{h(s) \in A \mid s \in S\}$ and $\gamma_h \overset{\text{def}}{=} \lambda X \subseteq A.\{s \in \Sigma \mid h(s) \in X\}$.

Partitions can be also viewed as dual abstractions when a set S is under approximated by the union of all the blocks $B \in P$ such that $B \subseteq S$. A graphical example of this under approximation is depicted on the right-hand side of Figure 2. This dual abstraction is formalized by the GI $(\widetilde{\alpha}_P, \wp(\Sigma)_\supseteq, \wp(P)_\supseteq, \widetilde{\gamma}_P)$ where

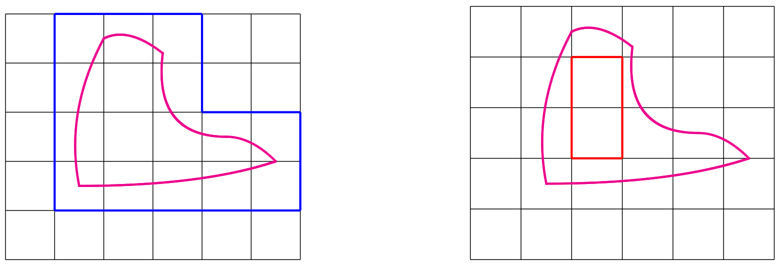

Figure 2: Partitions as abstract domains: over-approximation on the left and under-approximation on the right.

the ordering on the concrete domain $\wp(\Sigma)$ is given by the subset relation and

$$\widetilde{\alpha}_P(S) \overset{\text{def}}{=} \{B \in P \mid B \subseteq S\} \qquad \widetilde{\gamma}_P(\mathcal{B}) \overset{\text{def}}{=} \cup_{B \in \mathcal{B}} B.$$

In the following, we will be interested in viewing partitions as over approximations, that is partitions as abstract domains of $\wp(\Sigma)_\subseteq$.

Thus, partitions can be viewed as representations of abstract domains. On the other hand, it turns out that abstract domains can be abstracted to partitions. An abstract domain $A \in \mathrm{Abs}(\wp(\Sigma)_\subseteq)$ induces a state equivalence $\equiv_A$ on Σ by identifying those states that cannot be distinguished by A:

$$s \equiv_A s' \quad \text{iff} \quad \alpha(\{s\}) = \alpha(\{s'\}).$$

For any $s \in \Sigma$, $[s]_A \overset{\text{def}}{=} \{s' \in \Sigma \mid \alpha(\{s\}) = \alpha(\{s'\})\}$ is a block of the state partition $\mathrm{par}(A)$ induced by A:

$$\mathrm{par}(A) \overset{\text{def}}{=} \{[s]_A \mid s \in \Sigma\}.$$

Thus, $\mathrm{par} : \mathrm{Abs}(\wp(\Sigma)) \to \mathrm{Part}(\Sigma)$ is a mapping from abstract domains to partitions.

Example 3.1. Let $\Sigma = \{1, 2, 3, 4\}$ and let us specify abstract domains as uco's on $\wp(\Sigma)$. The uco's $\mu_1 = \{\varnothing, 12, 3, 4, 1234\}$, $\mu_2 = \{\varnothing, 12, 3, 4, 34, 1234\}$, $\mu_3 = \{\varnothing, 12, 3, 4, 34, 123, 124, 1234\}$, $\mu_4 = \{12, 123, 124, 1234\}$ and $\mu_5 = \{\varnothing, 12, 123, 124, 1234\}$ all induce the same partition $P = \mathrm{par}(\mu_i) = \{12, 3, 4\} \in \mathrm{Part}(\Sigma)$. For example, $\mu_5(\{1\}) = \mu_5(\{2\}) = \{1, 2\}$, $\mu_5(\{3\}) = \{1, 2, 3\}$ and $\mu_5(\{4\}) = \{1, 2, 3, 4\}$ so that $\mathrm{par}(\mu_5) = P$. Observe that μ_3 is the only partitioning abstract domain because $\mathrm{ad}^{\mathrm{p}}(P) = \mu_3$. $\square$

Abstract domains of $\wp(\Sigma)$ carry additional information other than the underlying state partition and this additional information allows us to distinguish them. It turns out that this can be precisely stated by abstract interpretation since the above mappings par and ad^{p} allows us to show that the whole lattice of partitions of Σ can be viewed as a ("higher-order") abstraction of the lattice of abstract domains of $\wp(\Sigma)$.

Theorem 3.2. $(\mathrm{par}, \mathrm{Abs}(\wp(\Sigma))_\sqsupseteq, \mathrm{Part}(\Sigma)_\succeq, \mathrm{ad}^{\mathrm{p}})$ *is a Galois insertion.*

Proof. Let $A \in \mathrm{Abs}(\wp(\Sigma))$ and $P \in \mathrm{Part}(\Sigma)$ and let $\mu_A \in \mathrm{uco}(\wp(\Sigma))$ be the closure associated with the abstract domain A. Let us prove that $P \preccurlyeq \mathrm{par}(A) \Leftrightarrow \mathrm{ad}^{\mathrm{p}}(P) \sqsubseteq \mu_A$.
($\Rightarrow$) For $S \in \wp(\Sigma)$ we have to prove that $\mathrm{ad}^{\mathrm{p}}(P)(S) \subseteq \mu_A(S)$. Consider $s \in \mathrm{ad}^{\mathrm{p}}(P)(S)$. Hence, there exists some $B \in P$ such that $s \in B$ and $B \cap S \neq \varnothing$. Let $q \in B \cap S$. Since $P \preccurlyeq \mathrm{par}(A)$, there exists some block $[r]_A \in \mathrm{par}(A)$ such that $B \subseteq [r]_A$. Thus, for any $x, y \in B$, $\alpha(\{x\}) = \alpha(\{r\}) = \alpha(\{y\})$, in particular, $\alpha(\{s\}) = \alpha(\{q\})$. Consequently, since $q \in S$ and therefore $\mu_A(\{q\}) \subseteq \mu_A(S)$, we have that $\mu_A(\{s\}) = \mu_A(\{q\}) \subseteq \mu_A(S)$, so that $s \in \mu_A(S)$.
($\Leftarrow$) Consider a block $B \in P$ and some $s \in B$. We show that $B \subseteq [s]_A$, namely if $s', s'' \in B$ then $\alpha(\{s'\}) = \alpha(\{s''\})$. Since $\mathrm{ad}^{\mathrm{p}}(P) \sqsubseteq \mu_A$, if $s', s'' \in B$ then $\mathrm{ad}^{\mathrm{p}}(P)(\{s'\}) = B \subseteq \mu_A(\{s'\})$ so that $s'' \in \mu_A(\{s'\})$ and therefore $\mu_A(\{s''\}) \subseteq \mu_A(\{s'\})$. Likewise, $\mu_A(\{s'\}) \subseteq \mu_A(\{s''\})$ so that $\mu_A(\{s'\}) = \mu_A(\{s''\})$ and in turn $\alpha(\{s'\}) = \alpha(\{s''\})$.
Finally, observe that ad^{p} is 1-1 so that the above adjunction is indeed a Galois insertion. $\square$

Let us observe that, as recalled in Section 2.2, the adjoint maps par and ad^{p} give rise to an order isomorphism between the lattices $\langle \mathrm{Part}(\Sigma), \preccurlyeq \rangle$ and $\langle \mathrm{Abs}^{\mathrm{par}}(\wp(\Sigma)), \sqsubseteq \rangle$.

Corollary 3.3. *Let $A \in \mathrm{Abs}(\wp(\Sigma))$. The following statements are equivalent:*
(1) *A is partitioning.*
(2) *γ is additive and $\{\gamma(\alpha(\{s\}))\}_{s\in\Sigma}$ is a partition of Σ. In this case, $\mathrm{par}(A) = \{\gamma(\alpha(\{s\}))\}_{s\in\Sigma}$.*
(3) *A is forward complete for the complement operator $\complement$.*

Proof. Let $A \in \mathrm{Abs}(\wp(\Sigma))$ and let $\mu_A = \gamma \circ \alpha \in \mathrm{uco}(\wp(\Sigma))$ be the corresponding uco.
(1) $\Rightarrow$ (2) By Theorem 3.2, $A \in \mathrm{Abs}^{\mathrm{par}}(\wp(\Sigma))$ iff $\mathrm{ad}^{\mathrm{p}}(\mathrm{par}(A)) = A$. Thus, if $\mathrm{ad}^{\mathrm{p}}(\mathrm{par}(A)) = A$ then $\mu_A = \gamma \circ \alpha$ is obviously additive. Moreover, $s \equiv_A s'$ iff $\alpha(\{s\}) = \alpha(\{s'\})$ iff $\gamma(\alpha(\{s\})) = \gamma(\alpha(\{s'\}))$, so that, for any $s \in \Sigma$, $[s]_A = \gamma(\alpha(\{s\}))$ and therefore $\mathrm{par}(A) = \{\gamma(\alpha(\{s\}))\}_{s\in\Sigma}$.
(2) $\Rightarrow$ (1) Since $\{\gamma(\alpha(\{s\}))\}_{s\in\Sigma} = P \in \mathrm{Part}(\Sigma)$ we have that for any $s \in \Sigma$, $[s]_A = \gamma(\alpha(\{s\}))$: in fact, if $s' \in \gamma(\alpha(\{s\}))$ then $\alpha(\{s'\}) \leq \alpha(\{s\})$, hence $\gamma(\alpha(\{s'\})) \subseteq \gamma(\alpha(\{s\}))$ and therefore $\gamma(\alpha(\{s'\})) = \gamma(\alpha(\{s\}))$. Thus, $\mathrm{par}(A) = P$. Moreover, since γ is additive, for any $S \subseteq \Sigma$, $\cup_{s\in S}\gamma(\alpha(\{s\})) = \gamma(\vee_{s\in S}\alpha(\{s\})) = \gamma(\alpha(S)) \in \mu_A$. Hence, since $\mathrm{ad}^{\mathrm{p}}(P) = \{\cup_{s\in S}\gamma(\alpha(\{s\})) \mid S \subseteq \Sigma\}$ we have that $\mathrm{ad}^{\mathrm{p}}(\mathrm{par}(A)) = A$.
(1) $\Rightarrow$ (3) Assume that $A \in \mathrm{Abs}^{\mathrm{par}}(\wp(\Sigma))$. It is enough to prove that for any $s \in \Sigma$, $\complement(\mu_A(\{s\})) \in \mu_A$: in fact, by (1) $\Rightarrow$ (2), γ is additive and therefore μ_A is additive (because it is a composition of additive maps) and therefore if $S \in \mu_A$ then $S = \cup_{s\in S}\mu_A(\{s\})$ so that $\complement(S) = \cap_{s\in S}\complement(\mu_A(\{s\}))$. Let us observe the following fact $(*)$: for any $s, s' \in \Sigma$, $s \notin \mu_A(\{s'\}) \Leftrightarrow \mu_A(\{s\}) \cap \mu_A(\{s'\}) = \varnothing$; this is a consequence of the fact that, by (1) $\Rightarrow$ (2), $\{\mu_A(\{s\})\}_{s\in\Sigma}$ is a partition. For any $s \in \Sigma$, we have that $\complement(\mu_A(\{s\})) \in \mu_A$ because:

$$\begin{aligned}
\mu_A(\complement(\mu_A(\{s\}))) &= \mu_A(\{s' \in \Sigma \mid s' \notin \mu_A(\{s\})\}) && \text{[by additivity of } \mu_A\text{]}\\
&= \cup\{\mu_A(\{s'\}) \mid s' \notin \mu_A(\{s\})\} && \text{[by the above fact } (*)\text{]}\\
&= \cup\{\mu_A(\{s'\}) \mid \mu_A(\{s'\}) \cap \mu_A(\{s\}) = \varnothing\}\\
&= \cup\{\mu_A(\{s'\}) \mid \mu_A(\{s'\}) \subseteq \complement(\mu_A(\{s\}))\}\\
&\subseteq \complement(\mu_A(\{s\}))
\end{aligned}$$

(3) $\Rightarrow$ (1) Assume that μ_A is forward complete for $\complement$, i.e. μ_A is closed under complements. By (2) $\Rightarrow$ (1), it is enough to prove that γ is additive and that $\{\mu_A(\{s\})\}_{s\in\Sigma} \in \mathrm{Part}(\Sigma)$.
(i) γ is additive. Observe that γ is additive iff μ_A is additive iff μ_A is closed under arbitrary unions. If $\{S_i\}_{i\in I} \subseteq \mu_A$ then $\cup_i S_i = \complement(\cap_i \complement(S_i)) \in \mu_A$, because, μ_A is closed under complements (and arbitrary intersections).
(ii) $\{\mu_A(\{s\})\}_{s\in\Sigma} \in \mathrm{Part}(\Sigma)$. Clearly, we have that $\cup_{s\in\Sigma}\mu_A(\{s\}) = \Sigma$. Consider now $s, r \in \Sigma$ such that $\mu_A(\{s\}) \cap \mu_A(\{r\}) \neq \varnothing$. Let us show that $\mu_A(\{s\}) = \mu_A(\{r\})$. In order to show this, let us prove that $s \in \mu_A(\{r\})$. Notice that $\mu_A(\{s\}) \smallsetminus \mu_A(\{r\}) = \mu_A(\{s\}) \cap \complement(\mu_A(\{r\})) \in \mu_A$, because μ_A is closed under complements. If $s \notin \mu_A(\{r\})$ then we would have that $s \in \mu_A(\{s\}) \smallsetminus \mu_A(\{r\}) \in \mu_A$, and this would imply $\mu_A(\{s\}) \subseteq \mu_A(\{s\}) \smallsetminus \mu_A(\{r\}) \subseteq \mu_A(\{s\})$, namely $\mu_A(\{s\}) = \mu_A(\{s\}) \smallsetminus \mu_A(\{r\})$. Thus, we would obtain the contradiction $\mu_A(\{s\}) \cap \mu_A(\{r\}) = \varnothing$. Hence, we have that $s \in \mu_A(\{r\})$ and therefore $\mu_A(\{s\}) \subseteq \mu_A(\{r\})$. By swapping the roles of s and r, we also obtain that $\mu_A(\{r\}) \subseteq \mu_A(\{s\})$, so that $\mu_A(\{s\}) = \mu_A(\{r\})$. □

Let us remark that $\mathbb{P} \overset{\mathrm{def}}{=} \mathrm{ad}^{\mathrm{p}} \circ \mathrm{par}$ is a lower closure operator on $\langle \mathrm{Abs}(\wp(\Sigma)), \sqsubseteq \rangle$ and that for any $A \in \mathrm{Abs}(\wp(\Sigma))$, A is partitioning iff $\mathbb{P}(A) = A$. Hence, $\mathbb{P}$ is exactly the partitioning-shell refinement, namely $\mathbb{P}(A)$ is the most abstract refinement of A that is partitioning.

4 Abstract Semantics of Languages

4.1 Concrete Semantics

We consider temporal specification languages $\mathscr{L}$ whose state formulae φ are inductively defined by:

$$\mathscr{L} \ni \varphi ::= p \mid f(\varphi_1, ..., \varphi_n)$$

where p ranges over a (typically finite) set of atomic propositions AP, while f ranges over a finite set Op of operators. AP and Op are also denoted, respectively, by $AP_{\mathscr{L}}$ and $Op_{\mathscr{L}}$. Each operator $f \in Op$ has an arity[3] $\sharp(f) > 0$.

Formulae in $\mathscr{L}$ are interpreted on a *semantic structure* $\mathcal{S} = (\Sigma, I)$ where Σ is any (possibly infinite) set of states and I is an interpretation function $I : AP \cup Op \to \mathrm{Fun}(\wp(\Sigma))$ that maps $p \in AP$ to $I(p) \in \wp(\Sigma)$ and $f \in Op$ to $I(f) : \wp(\Sigma)^{\sharp(f)} \to \wp(\Sigma)$. $I(p)$ and $I(f)$ are also denoted by, respectively, $\boldsymbol{p}$ and $\boldsymbol{f}$. Moreover, $\boldsymbol{AP} \stackrel{\mathrm{def}}{=} \{\boldsymbol{p} \in \wp(\Sigma) \mid p \in AP\}$ and $\boldsymbol{Op} \stackrel{\mathrm{def}}{=} \{\boldsymbol{f} : \wp(\Sigma)^{\sharp(f)} \to \wp(\Sigma) \mid f \in Op\}$. Note that the interpretation I induces a state labeling $\ell_I : \Sigma \to \wp(AP)$ by $\ell_I(s) \stackrel{\mathrm{def}}{=} \{p \in AP \mid s \in I(p)\}$. The *concrete state semantic function* $[\![\cdot]\!]_{\mathcal{S}} : \mathscr{L} \to \wp(\Sigma)$ evaluates a formula $\varphi \in \mathscr{L}$ to the set of states making φ true w.r.t. the semantic structure $\mathcal{S}$:

$$[\![p]\!]_{\mathcal{S}} = \boldsymbol{p} \quad \text{and} \quad [\![f(\varphi_1, ..., \varphi_n)]\!]_{\mathcal{S}} = \boldsymbol{f}([\![\varphi_1]\!]_{\mathcal{S}}, ..., [\![\varphi_n]\!]_{\mathcal{S}}).$$

Semantic structures generalize the role of Kripke structures. In fact, in standard model checking a semantic structure is usually defined through a Kripke structure $\mathcal{K}$ so that the interpretation of logical/temporal operators is defined in terms of standard logical operators and paths in $\mathcal{K}$. In the following, we freely use standard logical and temporal operators together with their corresponding usual interpretations: for example, $I(\wedge) = \cap$, $I(\vee) = \cup$, $I(\neg) = \complement$, $I(\mathrm{EX}) = \mathrm{pre}_{\rightarrow}$, $I(\mathrm{AX}) = \widetilde{\mathrm{pre}}_{\rightarrow}$, etc. As an example, consider the standard semantics of CTL:

$$\mathrm{CTL} \ni \varphi ::= p \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \mathrm{AX}\varphi \mid \mathrm{EX}\varphi \mid \mathrm{AU}(\varphi_1, \varphi_2) \mid \mathrm{EU}(\varphi_1, \varphi_2) \mid \mathrm{AR}(\varphi_1, \varphi_2) \mid \mathrm{ER}(\varphi_1, \varphi_2)$$

with respect to a Kripke structure $\mathcal{K} = (\Sigma, R, \ell)$. Hence, $\mathcal{K}$ determines a corresponding interpretation I for atoms in AP and operators of Op_{CTL}, namely $I(\mathrm{AX}) = \widetilde{\mathrm{pre}}_{\rightarrow}$, $I(\mathrm{EX}) = \mathrm{pre}_{\rightarrow}$, etc., and this defines the concrete semantic function $[\![\cdot]\!]_{\mathcal{K}} : \mathrm{CTL} \to \wp(\Sigma)$.

If g is any operator with arity $\sharp(g) = n > 0$, whose interpretation is given by $\boldsymbol{g} : \wp(\Sigma)^n \to \wp(\Sigma)$, and $\mathcal{S} = (\Sigma, I)$ is a semantic structure then we say that a language $\mathscr{L}$ is *closed under* g for $\mathcal{S}$ when for any $\varphi_1, ..., \varphi_n \in \mathscr{L}$ there exists some $\psi \in \mathscr{L}$ such that $\boldsymbol{g}([\![\varphi_1]\!]_{\mathcal{S}}, ..., [\![\varphi_n]\!]_{\mathcal{S}}) = [\![\psi]\!]_{\mathcal{S}}$. For instance, if $Op_{\mathscr{L}}$ includes EX and negation with their standard interpretations, i.e. $I(\mathrm{EX}) = \mathrm{pre}_{\rightarrow}$ and $I(\neg) = \complement$, then $\mathscr{L}$ is closed under AX with its standard interpretation $\widetilde{\mathrm{pre}}_{\rightarrow}$ because $\widetilde{\mathrm{pre}}_{\rightarrow} = \complement \circ \mathrm{pre}_{\rightarrow} \circ \complement$. This notion can be extended in a straightforward way to infinitary operators: for instance, $\mathscr{L}$ is closed under infinite logical conjunction for $\mathcal{S}$ iff for any $\Phi \subseteq \mathscr{L}$, there exists some $\psi \in \mathscr{L}$ such that $\bigcap_{\varphi \in \Phi} [\![\varphi]\!]_{\mathcal{S}} = [\![\psi]\!]_{\mathcal{S}}$. In particular, let us note that if $\mathscr{L}$ is closed under infinite logical conjunction then it must exist some $\psi \in \mathscr{L}$ such that $\cap\varnothing = \Sigma = [\![\psi]\!]_{\mathcal{S}}$, namely $\mathscr{L}$ is able to express the tautology *true*. Let us also remark that if the state space Σ is finite and $\mathscr{L}$ is closed under logical conjunction then we always mean that there exists some $\psi \in \mathscr{L}$ such that $\cap\varnothing = \Sigma = [\![\psi]\!]_{\mathcal{S}}$. Finally, note that if $\mathscr{L}$ is closed under negation and (infinite) logical conjunction then $\mathscr{L}$ is closed under (infinite) logical disjunction as well.

4.2 Abstract Semantics

In the following, we apply the standard abstract interpretation approach for defining abstract semantics [14, 15]. Let $\mathscr{L}$ be a language and $\mathcal{S} = (\Sigma, I)$ be a semantic structure for $\mathscr{L}$. An *abstract semantic structure* $\mathcal{S}^\sharp = (A, I^\sharp)$ is given by an abstract domain $A \in \mathrm{Abs}(\wp(\Sigma)_{\subseteq})$ and by an abstract interpretation function $I^\sharp : AP \cup Op \to \mathrm{Fun}(A)$. An abstract semantic structure $\mathcal{S}^\sharp$ therefore induces an *abstract semantic function* $[\![\cdot]\!]_{\mathcal{S}^\sharp} : \mathscr{L} \to A$ that evaluates formulae in $\mathscr{L}$ to abstract values in A. The abstract interpretation $I^\sharp$ is a correct over-approximation (respectively, under-approximation) of I on A when for any $p \in AP$, $\gamma(I^\sharp(p)) \supseteq I(p)$ (respectively, $\gamma(I^\sharp(p)) \subseteq I(p)$) and for any $f \in Op$, $\gamma \circ I^\sharp(f) \sqsupseteq I(f) \circ \gamma$ (respectively, $\gamma \circ I^\sharp(f) \sqsubseteq I(f) \circ \gamma$). If $I^\sharp$ is a correct over-approximation (respectively, under-approximation) of I and the semantic operations in $\boldsymbol{Op}$ are monotone then the abstract semantics is an over-approximation (respectively, under-approximation) of the concrete semantics, namely for any $\varphi \in \mathscr{L}$, $\gamma([\![\varphi]\!]_{\mathcal{S}^\sharp}) \supseteq [\![\varphi]\!]_{\mathcal{S}}$ (respectively, $\gamma([\![\varphi]\!]_{\mathcal{S}^\sharp}) \subseteq [\![\varphi]\!]_{\mathcal{S}}$).

[3] It would be possible to consider generic operators whose arity is any possibly infinite ordinal, thus allowing, for example, infinite conjunctions or disjunctions.

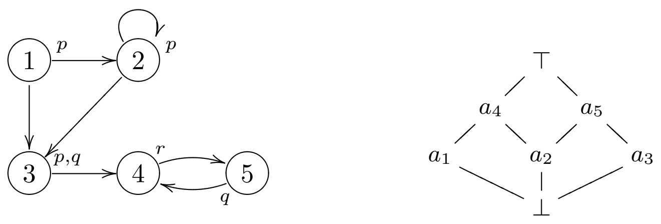

Figure 3: A Kripke structre on the left and an abstract domain on the right.

In particular, the abstract domain A always induces an abstract semantic structure $\mathcal{S}^A = (A, I^A)$ where I^A is the best correct approximation of I on A, i.e. I^A interprets atoms p and operators f as best correct approximations on A of, respectively, $\boldsymbol{p}$ and $\boldsymbol{f}$: for any $p \in AP$ and $f \in Op$,

$$I^A(p) \overset{\text{def}}{=} \alpha(\boldsymbol{p}) \quad \text{and} \quad I^A(f) \overset{\text{def}}{=} \boldsymbol{f}^A.$$

Thus, the abstract domain A systematically induces an abstract semantic function $[\![\cdot]\!]_{\mathcal{S}^A} : \mathscr{L} \to A$, also denoted by $[\![\cdot]\!]^A_{\mathcal{S}}$, which is therefore defined by:

$$[\![p]\!]^A_{\mathcal{S}} = \alpha(\boldsymbol{p}) \quad \text{and} \quad [\![f(\varphi_1, ..., \varphi_n)]\!]^A_{\mathcal{S}} = \boldsymbol{f}^A([\![\varphi_1]\!]^A_{\mathcal{S}}, ..., [\![\varphi_n]\!]^A_{\mathcal{S}}).$$

As usual in abstract interpretation, observe that the concrete semantics is a particular abstract semantics, namely it is the abstract semantics induced by the "identical abstraction" $(\mathrm{id}, \wp(\Sigma), \wp(\Sigma), \mathrm{id})$.

Example 4.1. Let $\mathscr{L} \ni \varphi ::= p \mid q \mid r \mid \varphi_1 \wedge \varphi_2 \mid \mathrm{EX}\varphi$. Let us consider the Kripke structure $\mathcal{K} = (\Sigma, \to, \ell)$ and the lattice A both depicted in Figure 3. Let $\mathcal{S}$ be the semantic structure induced by the Kripke structure $\mathcal{K}$ so that $\mathbf{EX} = \mathrm{pre}_{\to}$. Let us consider the formulae $\mathrm{EX}r$ and $\mathrm{EX}(p \wedge q)$, whose concrete semantics are as follows: $[\![\mathrm{EX}r]\!]_{\mathcal{S}} = \{3, 5\}$ and $[\![\mathrm{EX}(p \wedge q)]\!]_{\mathcal{S}} = \{1, 2\}$. A is an abstract domain of $\wp(\Sigma)$ where the Galois insertion $(\alpha, \wp(\Sigma), A, \gamma)$ is determined by the following concretization map:

$$\begin{array}{l} \gamma(\bot) = \varnothing;\ \ \gamma(a_1) = \{1,2\};\ \ \gamma(a_2) = \{3\};\ \ \gamma(a_3) = \{3,4\}; \\ \gamma(a_4) = \{1,2,3\};\ \ \gamma(a_5) = \{3,4,5\};\ \ \gamma(\top) = \{1,2,3,4,5\}. \end{array}$$

Note that, by Corollary 3.3, A is not partitioning because γ is not additive: $\gamma(a_2) \cup \gamma(a_3) = \{3, 4\} \subsetneq \{3, 4, 5\} = \gamma(a_2 \vee a_3)$. It turns out that:

$$\begin{array}{ll} [\![\mathrm{EX}r]\!]^A_{\mathcal{S}} & = \alpha(\mathrm{pre}_{\to}(\gamma([\![r]\!]^A_{\mathcal{S}})) = \alpha(\mathrm{pre}_{\to}(\gamma(\alpha(\boldsymbol{r})))) = \alpha(\mathrm{pre}_{\to}(\gamma(a_3))) \\ & = \alpha(\mathrm{pre}_{\to}(\{3,4\})) = \alpha(\{1,2,3,5\}) = \top; \\ [\![\mathrm{EX}(p \wedge q)]\!]^A_{\mathcal{S}} & = \alpha(\mathrm{pre}_{\to}(\gamma([\![p]\!]^A_{\mathcal{S}} \wedge [\![q]\!]^A_{\mathcal{S}}))) = \alpha(\mathrm{pre}_{\to}(\gamma(\alpha(\boldsymbol{p}) \wedge \alpha(\boldsymbol{q})))) \\ & = \alpha(\mathrm{pre}_{\to}(\gamma(a_4 \wedge a_5))) = \alpha(\mathrm{pre}_{\to}(\gamma(a_2))) = \alpha(\mathrm{pre}_{\to}(3)) = \alpha(\{1,2\}) = a_1. \end{array}$$

Observe that the abstract semantics $[\![\mathrm{EX}r]\!]^A_{\mathcal{S}}$ is a proper over-approximation of $[\![\mathrm{EX}r]\!]_{\mathcal{S}}$ because $[\![\mathrm{EX}r]\!]_{\mathcal{S}} \subsetneq \gamma([\![\mathrm{EX}r]\!]^A_{\mathcal{S}})$. On the other hand, the concrete semantics $[\![\mathrm{EX}(p \wedge q)]\!]_{\mathcal{S}}$ is precisely represented in A because $\gamma([\![\mathrm{EX}(p \wedge q)]\!]^A_{\mathcal{S}}) = [\![\mathrm{EX}(p \wedge q)]\!]_{\mathcal{S}}$. □

5 Generalized Strong Preservation

We showed in Section 3 how a state partition P can be viewed as a partitioning abstract domain $\mathrm{ad}^{\mathrm{p}}(P)$ specified by the GI $(\alpha_P, \wp(\Sigma)_{\subseteq}, \wp(P)_{\subseteq}, \gamma_P)$. Thus, given a language $\mathscr{L}$ and a corresponding semantic structure $\mathcal{S} = (\Sigma, I)$, it turns out that any partition $P \in \mathrm{Part}(\Sigma)$ systematically induces a corresponding abstract semantics $[\![\cdot]\!]^P_{\mathcal{S}} \overset{\text{def}}{=} [\![\cdot]\!]^{\mathrm{ad}^{\mathrm{p}}(P)}_{\mathcal{S}} : \mathscr{L} \to \mathrm{ad}^{\mathrm{p}}(P)$ that evaluates a formula in $\mathscr{L}$ to a (possibly empty) union of blocks of P. Strong preservation for a partition P can be characterized in terms of the corresponding abstract domain $\mathrm{ad}^{\mathrm{p}}(P)$ as follows.

Lemma 5.1. *$P \in \mathrm{Part}(\Sigma)$ is s.p. for $\mathscr{L}$ iff $\forall \varphi \in \mathscr{L}$ and $S \subseteq \Sigma$, $\alpha_P(S) \subseteq [\![\varphi]\!]^P_{\mathcal{S}} \Leftrightarrow S \subseteq [\![\varphi]\!]_{\mathcal{S}}$.*

Proof. ($\Rightarrow$): Let us first observe that for any $\varphi \in \mathscr{L}$, $\gamma_P(\alpha_P(\llbracket\varphi\rrbracket_{\mathcal{S}})) = \llbracket\varphi\rrbracket_{\mathcal{S}}$: in fact, for any $s \in \llbracket\varphi\rrbracket_{\mathcal{S}}$, $\alpha_P(\{s\})$ is the block of P containing s; since $P \preccurlyeq P_{\mathscr{L}}$, we have that $\alpha_P(\{s\}) \subseteq \llbracket\varphi\rrbracket_{\mathcal{S}}$, and from this $\alpha_P(\llbracket\varphi\rrbracket_{\mathcal{S}}) \subseteq \llbracket\varphi\rrbracket_{\mathcal{S}}$ and in turn $\gamma_P(\alpha_P(\llbracket\varphi\rrbracket_{\mathcal{S}})) = \llbracket\varphi\rrbracket_{\mathcal{S}}$.
Let us now prove by structural induction on $\varphi \in \mathscr{L}$ that $\llbracket\varphi\rrbracket_{\mathcal{S}} = \gamma_P(\llbracket\varphi\rrbracket^P_{\mathcal{S}})$:

– $\varphi \equiv p \in AP_{\mathscr{L}}$: by using the above observation, $\llbracket p\rrbracket_{\mathcal{S}} = \gamma_P(\alpha_P(\llbracket p\rrbracket_{\mathcal{S}})) = \gamma_P(\llbracket p\rrbracket^P_{\mathcal{S}})$.

– $\varphi \equiv f(\varphi_1, \ldots, \varphi_n)$:

$$\begin{aligned}
\llbracket f(\varphi_1,\ldots,\varphi_n)\rrbracket_{\mathcal{S}} = {} & \text{[by the above observation]}\\
\gamma_P(\alpha_P(\llbracket f(\varphi_1,\ldots,\varphi_n)\rrbracket_{\mathcal{S}})) = {} & \text{[by definition]}\\
\gamma_P(\alpha_P(\boldsymbol{f}(\llbracket\varphi_1\rrbracket_{\mathcal{S}},\ldots,\llbracket\varphi_n\rrbracket_{\mathcal{S}}))) = {} & \text{[by inductive hypothesis]}\\
\gamma_P(\alpha_P(\boldsymbol{f}(\gamma_P(\llbracket\varphi_1\rrbracket^P_{\mathcal{S}}),\ldots,\gamma_P(\llbracket\varphi_n\rrbracket^P_{\mathcal{S}})))) = {} & \text{[by definition]}\\
\gamma_P(\llbracket f(\varphi_1,\ldots,\varphi_n)\rrbracket^P_{\mathcal{S}}). &
\end{aligned}$$

Now, consider any $\varphi \in \mathscr{L}$. If $S \subseteq \llbracket\varphi\rrbracket_{\mathcal{S}}$ then $\alpha_P(S) \subseteq \alpha_P(\llbracket\varphi\rrbracket_{\mathcal{S}}) = \alpha_P(\gamma_P(\llbracket\varphi\rrbracket^P_{\mathcal{S}})) = \llbracket\varphi\rrbracket^P_{\mathcal{S}}$. Conversely, if $\alpha_P(S) \subseteq \llbracket\varphi\rrbracket^P_{\mathcal{S}}$ then $S \subseteq \gamma_P(\llbracket\varphi\rrbracket^P_{\mathcal{S}}) = \llbracket\varphi\rrbracket_{\mathcal{S}}$.
($\Leftarrow$): Consider a block $B \in P$ and $s, s' \in B$ so that $\alpha_P(\{s\}) = B = \alpha_P(\{s'\})$. By hypothesis, for any $\varphi \in \mathscr{L}$, we have that $s \in \llbracket\varphi\rrbracket_{\mathcal{S}}$ iff $\alpha_P(\{s\}) \subseteq \llbracket\varphi\rrbracket^P_{\mathcal{S}}$ iff $\alpha_P(\{s'\}) \subseteq \llbracket\varphi\rrbracket^P_{\mathcal{S}}$ iff $s' \in \llbracket\varphi\rrbracket_{\mathcal{S}}$. Thus, $s \equiv_{\mathscr{L}} s'$. □

This states that a partition $P \in \mathrm{Part}(\Sigma)$ is s.p. for $\mathscr{L}$ if and only if to check whether some set S of states satisfies some formula $\varphi \in \mathscr{L}$, i.e. $S \subseteq \llbracket\varphi\rrbracket_{\mathcal{S}}$, is equivalent to check whether the abstract state $\alpha_P(S)$ is more precise than the abstract semantics $\llbracket\varphi\rrbracket^P_{\mathcal{S}}$, that is S is over-approximated by $\llbracket\varphi\rrbracket^P_{\mathcal{S}}$. The key observation here is that in our abstract interpretation-based framework partitions are particular abstract domains. Lemma 5.1 allows us to generalize the notion of strong preservation from partitions to generic abstract semantic functions as follows.

Definition 5.2. Let $\mathscr{L}$ be a language, $\mathcal{S} = (\Sigma, I)$ be a semantic structure for $\mathscr{L}$ and $\mathcal{S}^\sharp = (A, I^\sharp)$ be a corresponding abstract semantic structure where the GI is $(\alpha, \wp(\Sigma)_\subseteq, A_\leq, \gamma)$. The abstract semantics $\llbracket\cdot\rrbracket_{\mathcal{S}^\sharp}$ is *strongly preserving* for $\mathscr{L}$ (w.r.t. $\mathcal{S}$) if for any $\varphi \in \mathscr{L}$ and $S \subseteq \Sigma$,

$$\alpha(S) \leq \llbracket\varphi\rrbracket_{\mathcal{S}^\sharp} \;\Leftrightarrow\; S \subseteq \llbracket\varphi\rrbracket_{\mathcal{S}}. \quad \square$$

Definition 5.2 generalizes standard strong preservation from partitions, as characterized by Lemma 5.1, both to an arbitrary abstract domain $A \in \mathrm{Abs}(\wp(\Sigma))$ and to a corresponding abstract interpretation function $I^\sharp$. Likewise, standard weak preservation can be generalized as follows. Let $\mathcal{K} = (\Sigma, R, \ell)$ be a concrete Kripke structure that induces the concrete semantics $\llbracket\varphi\rrbracket_{\mathcal{K}} = \{s \in \Sigma \mid s \models^{\mathcal{K}} \varphi\}$. Let $h : \Sigma \to A$ be a surjective abstraction and let $(\alpha_h, \wp(\Sigma), \wp(A), \gamma_h)$ be the corresponding partitioning abstract domain. Let $\mathcal{A} = (A, R^\sharp, \ell^\sharp)$ be an abstract Kripke structure on A that gives rise to the abstract semantics $\llbracket\varphi\rrbracket_{\mathcal{A}} = \{a \in A \mid a \models^{\mathcal{A}} \varphi\}$. Then, $\mathcal{A}$ weakly preserves $\mathscr{L}$ when

$$\forall\phi \in \mathscr{L}. \forall S \subseteq \Sigma.\ \alpha_h(S) \subseteq \llbracket\varphi\rrbracket_{\mathcal{A}} \;\Rightarrow\; S \subseteq \llbracket\varphi\rrbracket_{\mathcal{K}}.$$

Hence, weak preservation can be generalized to generic abstract domains and abstract semantics accordingly to Definition 5.2.

5.1 Strong Preservation is an Abstract Domain Property

Definition 5.2 is a direct and natural generalization of the standard notion of strong preservation in abstract model checking. It can be equivalently stated as follows.

Lemma 5.3. $\llbracket\cdot\rrbracket_{\mathcal{S}^\sharp}$ *is s.p. for* $\mathscr{L}$ *iff for any* $\varphi \in \mathscr{L}$, $\llbracket\varphi\rrbracket_{\mathcal{S}} = \gamma(\llbracket\varphi\rrbracket_{\mathcal{S}^\sharp})$.

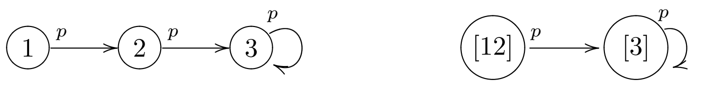

Figure 4: A Kripke structure $\mathcal{K}$ on the left and an abstract Kripke structure $\mathcal{A}$ on the right.

Proof. ($\Rightarrow$) On the one hand, $\gamma([\![\varphi]\!]_{\mathcal{S}^\sharp}) \subseteq [\![\varphi]\!]_{\mathcal{S}}$ iff $\alpha(\gamma([\![\varphi]\!]_{\mathcal{S}^\sharp})) \leq [\![\varphi]\!]_{\mathcal{S}^\sharp}$ iff $[\![\varphi]\!]_{\mathcal{S}^\sharp} \leq [\![\varphi]\!]_{\mathcal{S}^\sharp}$, which is trivially true. On the other hand, $[\![\varphi]\!]_{\mathcal{S}} \subseteq \gamma([\![\varphi]\!]_{\mathcal{S}^\sharp})$ iff $\alpha([\![\varphi]\!]_{\mathcal{S}}) \leq [\![\varphi]\!]_{\mathcal{S}^\sharp}$ iff $[\![\varphi]\!]_{\mathcal{S}} \subseteq [\![\varphi]\!]_{\mathcal{S}}$, that is trivially true.
($\Leftarrow$) We have that $S \subseteq [\![\varphi]\!]_{\mathcal{S}}$ iff $S \subseteq \gamma([\![\varphi]\!]_{\mathcal{S}^\sharp})$ iff $\alpha(S) \leq [\![\varphi]\!]_{\mathcal{S}^\sharp}$. $\square$

In particular, it is worth noting that if $[\![\cdot]\!]_{\mathcal{S}^\sharp}$ is s.p. for $\mathscr{L}$ then $[\![\cdot]\!]_{\mathcal{S}^\sharp} = \alpha \circ [\![\cdot]\!]_{\mathcal{S}}$ holds.

Lemma 5.4. *Let $A \in \mathrm{Abs}(\wp(\Sigma))$.*
(1) Let $\mathcal{S}_1^\sharp = (A, I_1^\sharp)$ and $\mathcal{S}_2^\sharp = (A, I_2^\sharp)$ be abstract semantic structures on A. If $[\![\cdot]\!]_{\mathcal{S}_1^\sharp}$ and $[\![\cdot]\!]_{\mathcal{S}_2^\sharp}$ are both s.p. for $\mathscr{L}$ then $[\![\cdot]\!]_{\mathcal{S}_1^\sharp} = [\![\cdot]\!]_{\mathcal{S}_2^\sharp}$.
(2) Let $\mathcal{S}^\sharp = (A, I^\sharp)$ be an abstract semantic structure on A. If $[\![\cdot]\!]_{\mathcal{S}^\sharp}$ is s.p. for $\mathscr{L}$ then $[\![\cdot]\!]_{\mathcal{S}}^A$ is s.p. for $\mathscr{L}$.

Proof. (1) By Lemma 5.3, for any $\varphi \in \mathscr{L}$, $\gamma([\![\varphi]\!]_{\mathcal{S}_1^\sharp}) = [\![\varphi]\!]_{\mathcal{S}} = \gamma([\![\varphi]\!]_{\mathcal{S}_2^\sharp})$, so that, by applying α, $[\![\varphi]\!]_{\mathcal{S}_1^\sharp} = \alpha(\gamma([\![\varphi]\!]_{\mathcal{S}_1^\sharp})) = \alpha([\![\varphi]\!]_{\mathcal{S}}) = \alpha(\gamma([\![\varphi]\!]_{\mathcal{S}_2^\sharp})) = [\![\varphi]\!]_{\mathcal{S}_2^\sharp}$.
(2) Let us first observe that for any $\varphi \in \mathscr{L}$, $\gamma(\alpha([\![\varphi]\!]_{\mathcal{S}})) = [\![\varphi]\!]_{\mathcal{S}}$. In fact, $\gamma(\alpha([\![\varphi]\!]_{\mathcal{S}})) \subseteq [\![\varphi]\!]_{\mathcal{S}} \Leftrightarrow \alpha(\gamma(\alpha([\![\varphi]\!]_{\mathcal{S}}))) \leq [\![\varphi]\!]_{\mathcal{S}^\sharp} \Leftrightarrow \alpha([\![\varphi]\!]_{\mathcal{S}}) \leq [\![\varphi]\!]_{\mathcal{S}^\sharp} \Leftrightarrow [\![\varphi]\!]_{\mathcal{S}} \subseteq [\![\varphi]\!]_{\mathcal{S}}$. As a consequence of this fact, by structural induction on $\varphi \in \mathscr{L}$, analogously to the proof of Lemma 5.1, it is easy to prove that $\gamma([\![\varphi]\!]_{\mathcal{S}}^A) = [\![\varphi]\!]_{\mathcal{S}}$. Thus, by Lemma 5.3, $[\![\cdot]\!]_{\mathcal{S}}^A$ is s.p. for $\mathscr{L}$. $\square$

Thus, it turns out that strong preservation is an *abstract domain property*. This means that given any abstract domain $A \in \mathrm{Abs}(\wp(\Sigma))$, by Lemma 5.4 (2), it is possible to define an abstract semantic structure $\mathcal{S}^\sharp = (A, I^\sharp)$ on A such that the corresponding abstract semantics $[\![\cdot]\!]_{\mathcal{S}^\sharp}$ is s.p. for $\mathscr{L}$ if and only if the induced abstract semantics $[\![\cdot]\!]_{\mathcal{S}}^A : \mathscr{L} \to A$ is s.p. for $\mathscr{L}$. In particular, this also holds for the standard approach: if $\mathcal{A} = (A, R^\sharp, \ell^\sharp)$ is an abstract Kripke structure for $\mathscr{L}$, where $h : \Sigma \to A$ is the corresponding surjection, then the standard abstract semantics $[\![\cdot]\!]_{\mathcal{A}}$ strongly preserves $\mathscr{L}$ if and only if the abstract semantics induced by the partitioning abstract domain $(\alpha_h, \wp(\Sigma), \wp(A), \gamma_h)$ strongly preserves $\mathscr{L}$, and in this case, by Lemma 5.4 (1), this abstract semantics coincides with the standard abstract semantics $[\![\cdot]\!]_{\mathcal{A}}$. Strong preservation is an abstract domain property and therefore can be defined without loss of generality as follows.

Definition 5.5. An abstract domain $A \in \mathrm{Abs}(\wp(\Sigma))$ is *strongly preserving for $\mathscr{L}$* (w.r.t. a semantic structure $\mathcal{S}$) when $[\![\cdot]\!]_{\mathcal{S}}^A$ is s.p. for $\mathscr{L}$ (w.r.t. $\mathcal{S}$). We denote by $\mathrm{SP}_{\mathscr{L}} \subseteq \mathrm{Abs}(\wp(\Sigma))$ the set of abstract domains that are s.p. for $\mathscr{L}$. $\square$

Example 5.6. Let us consider Example 4.1. It turns out that the abstract domain A is not s.p. for $\mathscr{L}$ because, by Lemma 5.3, $[\![\mathrm{EX}r]\!]_{\mathcal{S}} = \{3, 5\} \subsetneq \{1, 2, 3, 4, 5\} = \gamma(\top) = \gamma([\![\mathrm{EX}r]\!]_{\mathcal{S}}^A)$. $\square$

Example 5.7. Let us consider the simple language $\mathcal{L} \ni \varphi ::= p \mid \mathrm{EX}\varphi$ and the Kripke structure $\mathcal{K}$ depicted in Figure 4. The Kripke structure $\mathcal{K}$ induces the semantic structure $\mathcal{S} = (\{1, 2, 3\}, I)$ such that $I(p) = \{1, 2, 3\}$ and $I(\mathrm{EX}) = \mathrm{pre}_{\to}$. Hence, we have that $[\![p]\!]_{\mathcal{S}} = \{1, 2, 3\}$, $[\![\mathrm{EX}p]\!]_{\mathcal{S}} = \{1, 2, 3\}$ and, for $k > 1$, $[\![\mathrm{EX}^k p]\!]_{\mathcal{S}} = \{1, 2, 3\}$. Let us consider the partitioning abstract domain A induced by the partition $P = \{[12], [3]\}$ and related to $\wp(\Sigma)$ by α and γ. Let us consider two different abstract semantic structures on A.

- The abstract semantic structure $\mathcal{S}^A = (A, I^A)$ is induced as best correct approximation of I by A.
- The abstract semantic structure $\mathcal{S}^{\mathcal{A}} = (A, I^{\mathcal{A}})$ is instead induced by the abstract Kripke structure $\mathcal{A} = (A, \to^\sharp, \ell^\sharp)$ in Figure 4. Hence, $I^{\mathcal{A}}(p) = \{[12], [3]\}$ and $I^{\mathcal{A}}(\mathrm{EX}) = \mathrm{pre}_{\to^\sharp}$.

$\mathcal{S}^A$ is different from $\mathcal{S}^{\mathcal{A}}$ because $I^A(\mathrm{EX}) \neq I^{\mathcal{A}}(\mathrm{EX})$. In fact, $I^A(\mathrm{EX})(\{[12]\}) = \alpha(\mathrm{pre}_{\rightarrow}(\gamma(\{[12]\}))) = \alpha(\mathrm{pre}_{\rightarrow}(\{1,2\})) = \alpha(\{1\}) = \{[12]\}$, while $I^{\mathcal{A}}(\mathrm{EX})(\{[12]\}) = \mathrm{pre}_{\rightarrow^\sharp}(\{[12]\}) = \varnothing$.
Let us show that both the abstract semantics $[\![\cdot]\!]^A_{\mathcal{S}}$ and $[\![\cdot]\!]_{\mathcal{S}^\sharp}$ are s.p. for $\mathscr{L}$.

– We have that $[\![p]\!]^A_{\mathcal{S}} = \{[12],[3]\}$, $[\![\mathrm{EX}p]\!]^A_{\mathcal{S}} = \alpha(\mathrm{pre}_{\rightarrow}(\{1,2,3\})) = \alpha(\{1,2,3\}) = \{[12],[3]\}$ and, for $k > 1$, $[\![\mathrm{EX}^k p]\!]^A_{\mathcal{S}} = \{[12],[3]\}$. Thus, for any $\varphi \in \mathcal{L}$, $[\![\varphi]\!]_{\mathcal{S}} = \gamma([\![\varphi]\!]^A_{\mathcal{S}})$.

– We have that $[\![p]\!]_{\mathcal{S}^{\mathcal{A}}} = \{[12],[3]\}$, $[\![\mathrm{EX}p]\!]_{\mathcal{S}^{\mathcal{A}}} = \mathrm{pre}_{\rightarrow^\sharp}(\{[12],[3]\}) = \{[12],[3]\}$ and, for $k > 1$, $[\![\mathrm{EX}^k p]\!]_{\mathcal{S}^{\mathcal{A}}} = \{[12],[3]\}$. Thus, for any $\varphi \in \mathcal{L}$, $[\![\varphi]\!]_{\mathcal{S}} = \gamma([\![\varphi]\!]_{\mathcal{S}^{\mathcal{A}}})$.

Consequently, by Lemma 5.3, both abstract semantics are s.p. for $\mathscr{L}$. □

5.2 The Most Abstract Strongly Preserving Domain

As recalled in Section 2.3, a language $\mathscr{L}$ and a semantic structure $\mathcal{S}$ for $\mathscr{L}$ induce a corresponding logical partition $P_{\mathscr{L}} \in \mathrm{Part}(\Sigma)$. By Lemma 5.1, it turns out that $P_{\mathscr{L}}$ is the coarsest strongly preserving partitioning abstract domain for $\mathscr{L}$. This can be generalized to arbitrary abstract domains as follows. Let us define $\mathrm{AD}_{\mathscr{L}}$ by:

$$\mathrm{AD}_{\mathscr{L}} \stackrel{\mathrm{def}}{=} \mathcal{M}(\{[\![\varphi]\!]_{\mathcal{S}} \mid \varphi \in \mathscr{L}\}).$$

Hence, $\mathrm{AD}_{\mathscr{L}}$ is the closure under arbitrary intersections of the set of concrete semantics of formulae in $\mathscr{L}$. Observe that $\mathrm{AD}_{\mathscr{L}} \in \mathrm{Abs}(\wp(\Sigma))$ because it is a Moore-family of $\wp(\Sigma)$.

Theorem 5.8. *For any $A \in \mathrm{Abs}(\wp(\Sigma))$, $A \in \mathrm{SP}_{\mathscr{L}}$ iff $A \sqsubseteq \mathrm{AD}_{\mathscr{L}}$.*

Proof. Let $\mu = \gamma \circ \alpha \in \mathrm{uco}(\wp(\Sigma))$ and let $\mu_{\mathscr{L}} \in \mathrm{uco}(\wp(\Sigma))$ be the uco associated to $\mathrm{AD}_{\mathscr{L}}$, that is $\mu_{\mathscr{L}}(S) = \cap\{[\![\varphi]\!]_{\mathcal{S}} \mid \varphi \in \mathscr{L},\ S \subseteq [\![\varphi]\!]_{\mathcal{S}}\}$. Recall that $A \sqsubseteq \mathrm{AD}_{\mathscr{L}}$ iff for any $\varphi \in \mathscr{L}$, $[\![\varphi]\!]_{\mathcal{S}} \in \mu$.
($\Rightarrow$) For any $\varphi \in \mathscr{L}$, we have that $\gamma(\alpha([\![\varphi]\!]_{\mathcal{S}})) = [\![\varphi]\!]_{\mathcal{S}}$ because, by Lemma 5.3, $\gamma(\alpha([\![\varphi]\!]_{\mathcal{S}})) = \gamma(\alpha(\gamma([\![\varphi]\!]^A_{\mathcal{S}}))) = \gamma([\![\varphi]\!]^A_{\mathcal{S}}) = [\![\varphi]\!]_{\mathcal{S}}$.
($\Leftarrow$) By hypothesis, $\gamma(\alpha([\![\varphi]\!]_{\mathcal{S}})) = [\![\varphi]\!]_{\mathcal{S}}$ for any $\varphi \in \mathscr{L}$. Let us show by structural induction on $\varphi \in \mathscr{L}$ that $[\![\varphi]\!]_{\mathcal{S}} = \gamma([\![\varphi]\!]^A_{\mathcal{S}})$.

– $\varphi \equiv p \in AP_{\mathscr{L}}$: by using the hypothesis, $[\![p]\!]_{\mathcal{S}} = \gamma_P(\alpha_P([\![p]\!]_{\mathcal{S}})) = \gamma_P([\![p]\!]^A_{\mathcal{S}})$.

– $\varphi \equiv f(\varphi_1, \ldots, \varphi_n)$:

$$\begin{aligned}
[\![f(\varphi_1,\ldots,\varphi_n)]\!]_{\mathcal{S}} = {} & \text{[by hypothesis]}\\
\gamma(\alpha([\![f(\varphi_1,\ldots,\varphi_n)]\!]_{\mathcal{S}})) = {} & \text{[by definition]}\\
\gamma(\alpha(\boldsymbol{f}([\![\varphi_1]\!]_{\mathcal{S}},\ldots,[\![\varphi_n]\!]_{\mathcal{S}}))) = {} & \text{[by inductive hypothesis]}\\
\gamma(\alpha(\boldsymbol{f}(\gamma([\![\varphi_1]\!]^A_{\mathcal{S}}),\ldots,\gamma([\![\varphi_n]\!]^A_{\mathcal{S}})))) = {} & \text{[by definition]}\\
\gamma([\![f(\varphi_1,\ldots,\varphi_n)]\!]^A_{\mathcal{S}}). &
\end{aligned}$$

Thus, by Lemma 5.3, $A \in \mathrm{SP}_{\mathscr{L}}$. □

Thus, $\mathrm{AD}_{\mathscr{L}}$ is the most abstract domain that is s.p. for $\mathscr{L}$ w.r.t. $\mathcal{S}$. As a consequence, it turns out that A is s.p. for $\mathscr{L}$ if and only if A represents with no loss of precision the concrete semantics of any formula in $\mathscr{L}$, that is $\forall \varphi \in \mathscr{L}.\ \gamma(\alpha([\![\varphi]\!]_{\mathcal{S}})) = [\![\varphi]\!]_{\mathcal{S}}$. Lemma 5.4 states that if a s.p. abstract semantics on a given abstract domain exists then this is unique. Nevertheless, Example 5.7 shows that this unique s.p. abstract semantics may be induced from different abstract semantic structures, i.e. different abstract interpretation functions. However, when $\mathscr{L}$ is closed under conjunction, it turns out that on the most abstract s.p. domain $\mathrm{AD}_{\mathscr{L}}$, the abstract interpretation function is unique and is given by the best correct approximation $I^{\mathrm{AD}_{\mathscr{L}}}$.

Theorem 5.9. *Let $\mathscr{L}$ be closed under infinite logical conjunction and let $\mathcal{S}^\sharp = (\mathrm{AD}_{\mathscr{L}}, I^\sharp)$ be an abstract semantic structure on $\mathrm{AD}_{\mathscr{L}}$. If $[\![\cdot]\!]_{\mathcal{S}^\sharp}$ is s.p. for $\mathscr{L}$ then $I^\sharp = I^{\mathrm{AD}_{\mathscr{L}}}$.*

Proof. Since $\mathscr{L}$ is closed under arbitrary logical conjunctions we have that $\mathrm{AD}_{\mathscr{L}} = \{[\![\varphi]\!]_{\mathcal{S}} \mid \varphi \in \mathscr{L}\}$. Thus, for any $a \in \mathrm{AD}_{\mathscr{L}}$, there exists some $\varphi \in \mathscr{L}$ such that $a = [\![\varphi]\!]_{\mathcal{S}^\sharp} = [\![\varphi]\!]_{\mathcal{S}}^{\mathrm{AD}_{\mathscr{L}}}$. In fact, if $a \in \mathrm{AD}_{\mathscr{L}}$ then $a = [\![\varphi]\!]_{\mathcal{S}}$, for some $\varphi \in \mathscr{L}$, so that, by Lemmata 5.3 and 5.4, $a = [\![\varphi]\!]_{\mathcal{S}} = \gamma([\![\varphi]\!]_{\mathcal{S}^\sharp}) = [\![\varphi]\!]_{\mathcal{S}^\sharp} = [\![\varphi]\!]_{\mathcal{S}}^{\mathrm{AD}_{\mathscr{L}}}$.
Let $p \in AP$. Then, by Lemma 5.4, $[\![p]\!]_{\mathcal{S}^\sharp} = [\![p]\!]_{\mathcal{S}}^{\mathrm{AD}_{\mathscr{L}}}$ so that $I^\sharp(p) = I^{\mathrm{AD}_{\mathscr{L}}}(p)$.
Let $f \in Op$. Then,

$$\begin{aligned}
I^\sharp(f)(a_1,...,a_n) = &\quad \text{[by the observation above]}\\
I^\sharp(f)([\![\varphi_1]\!]_{\mathcal{S}^\sharp},...,[\![\varphi_n]\!]_{\mathcal{S}^\sharp}) = &\quad \text{[by definition]}\\
[\![f(\varphi_1,...,\varphi_n)]\!]_{\mathcal{S}^\sharp} = &\quad \text{[by Lemma 5.4]}\\
[\![f(\varphi_1,...,\varphi_n)]\!]_{\mathcal{S}}^{\mathrm{AD}_{\mathscr{L}}} = &\quad \text{[by definition]}\\
I^{\mathrm{AD}_{\mathscr{L}}}(f)([\![\varphi_1]\!]_{\mathcal{S}}^{\mathrm{AD}_{\mathscr{L}}},...,[\![\varphi_n]\!]_{\mathcal{S}}^{\mathrm{AD}_{\mathscr{L}}}) = &\quad \text{[by the observation above]}\\
I^{\mathrm{AD}_{\mathscr{L}}}(f)(a_1,...,a_n).&
\end{aligned}$$

Thus, $I^\sharp = I^{\mathrm{AD}_{\mathscr{L}}}$. □

Hence, there is a unique choice for interpreting atoms and operations of $\mathscr{L}$ for the most abstract s.p. domain $\mathrm{AD}_{\mathscr{L}}$.

In our generalized framework, strong preservation for partitions becomes a particular instance through the Galois insertion $\mathrm{par}/\mathrm{ad}^{\mathrm{p}}$. Moreover, when $\mathscr{L}$ is closed under infinite conjunction, it turns out that the most abstract s.p. domain $\mathrm{AD}_{\mathscr{L}}$ is partitioning if and only if $\mathscr{L}$ is also closed under negation.

Proposition 5.10.
(1) $P_{\mathscr{L}} = \mathrm{par}(\mathrm{AD}_{\mathscr{L}})$ *and* $\mathrm{ad}^{\mathrm{p}}(P_{\mathscr{L}}) = \mathbb{P}(\mathrm{AD}_{\mathscr{L}})$.
(2) *P is strongly preserving for $\mathscr{L}$ iff $P \preccurlyeq \mathrm{par}(\mathrm{AD}_{\mathscr{L}})$ iff $\mathrm{ad}^{\mathrm{p}}(P) \sqsubseteq \mathrm{AD}_{\mathscr{L}}$.*
(3) *Let $\mathscr{L}$ be closed under infinite logical conjunction. Then, $\mathrm{AD}_{\mathscr{L}}$ is partitioning iff $\mathscr{L}$ is closed under logical negation.*

Proof. (1) Let $\mu_{\mathscr{L}} \in \mathrm{uco}(\wp(\Sigma))$ be the uco associated to $\mathrm{AD}_{\mathscr{L}}$. We have that $\mathrm{par}(\mathrm{AD}_{\mathscr{L}}) = \{[s]_{\mathrm{AD}_{\mathscr{L}}} \mid s \in \Sigma\}$, where $[s]_{\mathrm{AD}_{\mathscr{L}}} = \{s' \in \Sigma \mid \mu_{\mathscr{L}}(\{s'\}) = \mu_{\mathscr{L}}(\{s\})\}$. We also have that $s \equiv_{\mathscr{L}} s'$ iff $\forall \varphi \in \mathscr{L}. s \in [\![\varphi]\!]_{\mathcal{S}} \Leftrightarrow s' \in [\![\varphi]\!]_{\mathcal{S}}$ iff $\mu_{\mathscr{L}}(\{s\}) = \mu_{\mathscr{L}}(\{s'\})$, so that $P_{\mathscr{L}} = \mathrm{par}(\mathrm{AD}_{\mathscr{L}})$. Moreover, $\mathrm{ad}^{\mathrm{p}}(P_{\mathscr{L}}) = \mathrm{ad}^{\mathrm{p}}(\mathrm{par}(\mathrm{AD}_{\mathscr{L}})) = \mathbb{P}(\mathrm{AD}_{\mathscr{L}})$.
(2) P is s.p. for $\mathscr{L}$ iff $P \preccurlyeq P_{\mathscr{L}}$ iff, by Point (1), $P \preccurlyeq \mathrm{par}(A_{\mathscr{L}})$ iff, by Theorem 3.2, $\mathrm{ad}^{\mathrm{p}}(P) \sqsubseteq \mathrm{AD}_{\mathscr{L}}$.
(3) Since $\mathscr{L}$ is closed under infinite logical conjunction, $\mathrm{AD}_{\mathscr{L}} = \{[\![\varphi]\!]_{\mathcal{S}} \mid \varphi \in \mathscr{L}\}$. Thus, $\mathscr{L}$ is closed under logical negation iff $\mathrm{AD}_{\mathscr{L}}$ is closed under complementation $\complement$ and this exactly means that $\mathrm{AD}_{\mathscr{L}}$ is forward complete for the complement $\complement$. By Corollary 3.3, this latter fact happens iff $\mathrm{AD}_{\mathscr{L}}$ is partitioning. □

In particular, when $\mathscr{L}$ is closed under infinite conjunction but not under negation, it turns out that $\mathrm{ad}^{\mathrm{p}}(P_{\mathscr{L}}) \sqsubset \mathrm{AD}_{\mathscr{L}}$, i.e. a proper loss of information occurs when the domain $\mathrm{AD}_{\mathscr{L}}$ is abstracted to the partition $\mathrm{par}(\mathrm{AD}_{\mathscr{L}}) = P_{\mathscr{L}}$. On the other hand, when $\mathscr{L}$ is closed under negation and infinite conjunction, we have that $\mathrm{ad}^{\mathrm{p}}(P_{\mathscr{L}}) = \mathrm{AD}_{\mathscr{L}}$ and therefore, by Theorem 5.9, the abstract interpretation function on the partitioning abstract domain $\mathrm{ad}^{\mathrm{p}}(P_{\mathscr{L}})$ is uniquely determined.

Example 5.11. Let us consider the traffic light controller $\mathcal{K}$ in Example 2.2. As already observed, formulae of $\mathscr{L}$ have the following semantics in $\mathcal{K}$:

$$[\![\mathit{stop}]\!]_{\mathcal{K}} = \{R, RY\}; \quad [\![\mathit{go}]\!]_{\mathcal{K}} = \{G, Y\}; \quad [\![\mathrm{AXX}\mathit{stop}]\!]_{\mathcal{K}} = \{G, Y\}; \quad [\![\mathrm{AXX}\mathit{go}]\!]_{\mathcal{K}} = \{R, RY\}$$

so that

$$\mathrm{AD}_{\mathscr{L}} = \mathcal{M}(\{[\![\varphi]\!]_{\mathcal{K}} \mid \varphi \in \mathscr{L}\}) = \{\varnothing, \{R, RY\}, \{G, Y\}, \{R, RY, G, Y\}\}$$

and $P_{\mathscr{L}} = \mathrm{par}(\mathrm{AD}_{\mathscr{L}}) = \{\{R, RY\}, \{G, Y\}\}$. We denote by $\mu_{\mathscr{L}}$ the uco associated to $\mathrm{AD}_{\mathscr{L}}$. As shown in Example 2.2, it turns out that no abstract Kripke structure that properly abstracts $\mathcal{K}$ and strongly preserves $\mathscr{L}$ can be defined. In our approach, the abstract domain $\mathrm{AD}_{\mathscr{L}}$ induces a corresponding strongly

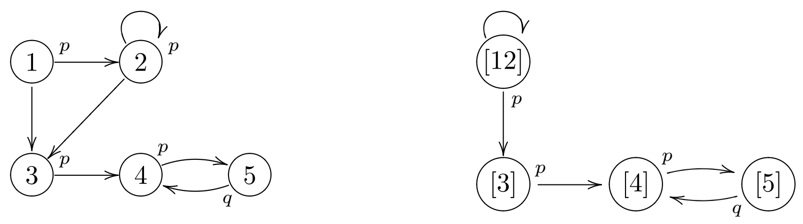

Figure 5: Concrete (on the left) and abstract (on the right) Kripke structures.

preserving abstract semantics $[\![\cdot]\!]^{\mathrm{AD}_{\mathscr{L}}}_{\mathcal{K}} : \mathscr{L} \to \mathrm{AD}_{\mathscr{L}}$, where the best correct approximation of the operator $\mathbf{AXX} : \wp(\Sigma) \to \wp(\Sigma)$ on $\mathrm{AD}_{\mathscr{L}}$ is:

$$\mu_{\mathscr{L}} \circ \mathbf{AXX} = \{\varnothing \mapsto \varnothing, \{R, RY\} \mapsto \{G, Y\}, \{G, Y\} \mapsto \{R, RY\},\\ \{R, RY, G, Y\} \mapsto \{R, RY, G, Y\}\}. \qquad \square$$

Example 5.12. Consider the language CTL and the Kripke structure $\mathcal{K} = (\Sigma, R, \ell)$ depicted in Figure 5, where the interpretation of temporal operators of CTL on $\mathcal{K}$ is standard. It is well known that the coarsest s.p. partition P_{CTL} can be obtained by refining the initial partition $P = \{1234, 5\}$ induced by the labeling ℓ through the Paige-Tarjan [44] algorithm, since P_{CTL} coincides with bisimulation equivalence on $\mathcal{K}$. It is easy to check that $P_{\mathrm{CTL}} = \{12, 3, 4, 5\}$. This partition determines (see point (2) in Section 2.3) the s.p. abstract Kripke structure depicted in Figure 5. Since CTL is closed under conjunction and negation, by Proposition 5.10 (1) and (3), it turns out that the most abstract s.p. domain A_{CTL} is partitioning and coincides with the following partitioning closure:

$$\mathrm{ad}^{\mathrm{p}}(P_{\mathrm{CTL}}) = \{\varnothing, 12, 3, 4, 5, 34, 35, 45, 122, 124, 125, 345, 1234, 1235, 1245, 12345\}.$$

Let us now consider the following language $\mathscr{L} \ni \varphi ::= p \mid q \mid \varphi_1 \wedge \varphi_2 \mid \mathrm{EF}_{[0,2]}\varphi$, where $\mathrm{EF}_{[0,2]}$ is a time bounded reachability operator that is useful for quantitative temporal analysis [25], e.g., of discrete real-time systems [10, Chapter 16]. The standard interpretation of $\mathrm{EF}_{[0,2]}$ is as follows: $s \models^{\mathcal{K}} \mathrm{EF}_{[0,2]}\varphi$ iff there exists a path $s_0 s_1 s_2 s_3 \ldots$ in $\mathcal{K}$ starting from $s = s_0$ and some $n \in [0, 2]$ such that $s_n \models^{\mathcal{K}} \varphi$. Let us characterize the semantics of formulae in $\mathscr{L}$:

$$[\![p]\!]_{\mathcal{K}} = \{1, 2, 3, 4\}; \quad [\![q]\!]_{\mathcal{K}} = \{5\}; \quad [\![\mathrm{EF}_{[0,2]}p]\!]_{\mathcal{K}} = \{1, 2, 3, 4, 5\};$$
$$[\![\mathrm{EF}_{[0,2]}q]\!]_{\mathcal{K}} = \{3, 4, 5\}; \quad [\![\mathrm{EF}_{[0,2]}(\mathrm{EF}_{[0,2]}q)]\!]_{\mathcal{K}} = \{1, 2, 3, 4, 5\};$$
$$[\![p \wedge \mathrm{EF}_{[0,2]}q]\!]_{\mathcal{K}} = \{3, 4\}; \quad [\![\mathrm{EF}_{[0,2]}(p \wedge \mathrm{EF}_{[0,2]}q)]\!]_{\mathcal{K}} = \{1, 2, 3, 4, 5\}.$$

Thus, $\mathrm{AD}_{\mathscr{L}} = \mathcal{M}(\{[\![\varphi]\!]_{\mathcal{K}} \mid \varphi \in \mathscr{L}\}) = \{\varnothing, 5, 34, 345, 1234, 12345\}$. On the other hand, by Proposition 5.10 (1), $P_{\mathscr{L}} = \mathrm{par}(\mathrm{AD}_{\mathscr{L}}) = \{12, 34, 5\}$. In this case, it turns out that $\mathrm{ad}^{\mathrm{p}}(P_{\mathscr{L}}) \sqsubset \mathrm{AD}_{\mathscr{L}}$. Moreover, analogously to Example 2.2, let us show that there exists no abstract transition relation $\to^{\sharp} \subseteq P_{\mathscr{L}} \times P_{\mathscr{L}}$ that determines an abstract Kripke structure $\mathcal{A} = (P_{\mathscr{L}}, \to^{\sharp}, \ell_{\mathscr{L}})$ which strongly preserves $\mathscr{L}$. Let $B = \{1, 2\}$, $B' = \{3, 4\}$ and $B'' = \{5\}$ be the blocks in $P_{\mathscr{L}}$. Assume by contradiction that such an abstract Kripke structure $\mathcal{A}$ exists.

(i) On the concrete model $\mathcal{K}$ we have that $3 \models^{\mathcal{K}} \mathrm{EF}_{[0,2]}q$. Thus, by strong preservation, it must be that $B' \models^{\mathcal{A}} \mathrm{EF}_{[0,2]}q$. On the other hand, if $B' \to^{\sharp} B$ and $B \to^{\sharp} B''$ then $B \models^{\mathcal{A}} \mathrm{EF}_{[0,2]}q$ and therefore, by weak preservation, we would have that $1 \models^{\mathcal{K}} \mathrm{EF}_{[0,2]}q$, which is a contradiction. Thus, necessarily, $B' \to^{\sharp} B''$.

(ii) Let us observe that $1 \models^{\mathcal{K}} \mathrm{EF}_{[0,2]}\mathrm{EF}_{[0,2]}q$. Hence, by strong preservation, $B \models^{\mathcal{A}} \mathrm{EF}_{[0,2]}\mathrm{EF}_{[0,2]}q$. If $B \to^{\sharp} B''$ then, as in point (i), we would still have that $1 \models^{\mathcal{K}} \mathrm{EF}_{[0,2]}q$, i.e. a contradiction. Hence, necessarily, $B \to^{\sharp} B'$.

(iii) From $B \to^{\sharp} B'$ and $B' \to^{\sharp} B''$, we would obtain that $B \models^{\mathcal{A}} \mathrm{EF}_{[0,2]}q$ that, as observed in point (ii), is a contradiction.

Thus, this shows that it is not possible to define an abstract Kripke structure on the abstract state space $P_{\mathscr{L}}$ that strongly preserves $\mathscr{L}$. The abstract domain $\mathrm{AD}_{\mathscr{L}}$ induces a corresponding abstract semantics $[\![\cdot]\!]_{\mathcal{K}}^{\mathrm{AD}_{\mathscr{L}}}$ that instead strongly preserves $\mathscr{L}$. In this case, the best correct approximation of the operator $\mathbf{EF}_{[0,2]}$ on $\mathrm{AD}_{\mathscr{L}}$ is:

$$\mu_{\mathscr{L}} \circ \mathbf{EF}_{[0,2]} = \{\varnothing \mapsto \varnothing,\ 5 \mapsto 345,\ 34 \mapsto 12345,\ 345 \mapsto 12345,\\ 1234 \mapsto 12345,\ 12345 \mapsto 12345\}. \qquad \square$$

6 Strong Preservation and Completeness

In this section we establish a precise correspondence between generalized strong preservation of abstract models and completeness of abstract interpretations, so that the problem of minimally refining an abstract model in order to get strong preservation can be formulated as a complete domain refinement in abstract interpretation.

6.1 Forward Complete Shells

Let us consider forward completeness of abstract domains $A \in \mathrm{Abs}(C)$ for generic n-ary concrete operations $f : C^n \to C$, with $n \geq 0$. Hence, A is forward complete for f, or simply f-complete, when $f \circ \langle \mu_A, ..., \mu_A\rangle = \mu_A \circ f \circ \langle \mu_A, ..., \mu_A\rangle$, that is, for any $\vec{x} \in C^n$, $f(\mu_A(x_1), ..., \mu_A(x_n)) = \mu_A(f(\mu_A(x_1), ..., \mu_A(x_n)))$. Equivalently, A is f-complete when for any $\vec{a} \in A^n$, $f(\gamma(a_1), ..., \gamma(a_n)) = \gamma(\alpha(f(\gamma(a_1), ..., \gamma(a_n))))$. For a set of operations $F \subseteq \mathrm{Fun}(C)$, A is F-complete when A is f-complete for each $f \in F$. Observe that F-completeness for an abstract domain A means that the associated closure μ_A is closed under the image of functions in F, namely $F(\mu_A) \subseteq \mu_A$. Also note that when $k : C^0 \to C$, i.e. $k \in C$ is a constant, A is k-complete iff k is precisely represented in A, i.e. $\gamma(\alpha(k)) = k$. Let us also note that an abstract domain $A \in \mathrm{Abs}(C)$ is always forward meet-complete because any uco is Moore-closed.

Let us first note that forward F-complete shells always exist. Let $\mathscr{S}_F : \mathrm{Abs}(C) \to \mathrm{Abs}(C)$ be defined as $\mathscr{S}_F(A) \overset{\mathrm{def}}{=} \sqcup \{X \in \mathrm{Abs}(C) \mid X \sqsubseteq A,\ X \text{ is } F\text{-complete}\}$.

Lemma 6.1. *$\mathscr{S}_F(A)$ is the F-complete shell of A.*

Proof. Let $\eta = \sqcup\{\rho \in \mathrm{uco}(C) \mid \rho \sqsubseteq \mu_A,\ \rho \text{ is } F\text{-complete}\} = \sqcap\{\rho \in \mathrm{uco}(C) \mid \rho \sqsubseteq \mu_A,\ \rho \text{ is } F\text{-complete}\}$. Let $f \in F$, with $\sharp(f) = n > 0$ (if $\sharp(f) = 0$ then, trivially, $f \in \eta$) and $\vec{c} \in \eta^n$. Consider any $\rho \in \mathrm{uco}(C)$ that is F-complete and such that $\rho \sqsubseteq \mu$. Since $\eta \subseteq \rho$, we have that $\vec{c} \in \rho^n$ and therefore $f(\vec{c}) \in \rho$ because ρ is F-complete. Thus, $f(\vec{c}) \in \eta$, i.e., η is F-complete. $\square$

A forward complete shell $\mathscr{S}_F(A)$ is a more concrete abstraction than A. How to characterize $\mathscr{S}_F(A)$? It is here useful to view abstract domains as closure operators on the concrete domain, i.e. as subsets of C. Hence, A is viewed as the subset $\mathrm{img}(\mu_A) = \gamma(A)$ of the concrete domain C so that $\mathscr{S}_F(A)$ can be characterized as the least Moore-closed subset of C that contains $\mathrm{img}(\mu_A)$ and is forward F-complete. We need to characterize the least amount of concrete information that must be added to $\gamma(A)$ in order to get forward completeness. It turns out that forward complete shells admit a constructive fixpoint characterization. Let $F^{\mathrm{uco}} : \mathrm{uco}(C) \to \mathrm{uco}(C)$ be defined as follows: $F^{\mathrm{uco}}(\rho) \overset{\mathrm{def}}{=} \mathcal{M}(F(\rho))$, namely $F^{\mathrm{uco}}(\rho)$ is the most abstract domain that contains the image of F on ρ. Observe that the operator $\lambda\rho.\mu_A \sqcap F^{\mathrm{uco}}(\rho) : \mathrm{uco}(C) \to \mathrm{uco}(C)$ is monotone.

Lemma 6.2. *$\mathscr{S}_F(A) = \mathrm{gfp}(\lambda\rho.\mu_A \sqcap F^{\mathrm{uco}}(\rho))$.*

Proof. Observe that a uco ρ is F-complete iff $F(\rho) \subseteq \rho$ iff $\mathcal{M}(F(\rho)) = F^{\mathrm{uco}}(\rho) \subseteq \rho$ iff $\rho \sqsubseteq F^{\mathrm{uco}}(\rho)$. Thus, we have that $\mathscr{S}_F(A) = \sqcup\{\rho \in \mathrm{uco}(C) \mid \rho \sqsubseteq \mu_A,\ \rho \text{ is } F\text{-complete}\} = \sqcup\{\rho \in \mathrm{uco}(C) \mid \rho \sqsubseteq \mu_A,\ \rho \sqsubseteq F^{\mathrm{uco}}(\rho)\} = \sqcup\{\rho \in \mathrm{uco}(C) \mid \rho \sqsubseteq \mu_A \sqcap F^{\mathrm{uco}}(\rho)\}$ and this last lub is precisely the greatest fixpoint $\mathrm{gfp}(\lambda\rho.\mu_A \sqcap F^{\mathrm{uco}}(\rho))$. $\square$

Thus, it turns out that the lower iteration sequence of $\lambda\rho.\mu_A \sqcap F^{\mathrm{uco}}(\rho)$ in $\mathrm{uco}(C)$ converges to the complete shell $\mathscr{S}_F(\mu_A)$.

Example 6.3. Let us consider the square operator on sets of integers $\mathrm{sq} : \wp(\mathbb{Z}) \to \wp(\mathbb{Z})$, i.e. $\mathrm{sq}(X) = X^2 = \{x^2 \mid x \in X\}$, and the abstract domain $\mathit{Sign} = \{\varnothing, \mathbb{Z}_{\leq 0}, \{0\}, \mathbb{Z}_{\geq 0}, \mathbb{Z}\}$. As observed in Section 2.2.2, Sign is not forward complete for the square operator. Let us apply Lemma 6.2 in order to compute the forward complete shell $\mathscr{S}_{\mathrm{sq}}(\mathit{Sign})$. Observe that

$$\varnothing^2 = \varnothing \in \mathit{Sign}; \quad \{0\}^2 = \{0\} \in \mathit{Sign}; \quad \mathbb{Z}^2_{\leq 0} = \mathbb{Z}^2_{\geq 0} = \mathbb{Z}^2 \notin \mathit{Sign}.$$

Thus, the first step of iteration refines Sign to $\mathit{Sign} \cup \{\mathbb{Z}^2\}$ (notice that this is an abstract domain because it is Moore-closed). Then, $(\mathbb{Z}^2)^2 = \mathbb{Z}^{2^2} \notin \mathit{Sign} \cup \{\mathbb{Z}^2\}$, so that on the second step of iteration we obtain $\mathit{Sign} \cup \{\mathbb{Z}^2, \mathbb{Z}^{2^2}\}$. In general, for $n \geq 1$, the n-th step of iteration provides $\mathit{Sign} \cup \{\mathbb{Z}^{2^k} \mid k \in [1, n]\}$, so that the complete shell $\mathscr{S}_{\mathrm{sq}}(\mathit{Sign})$ coincides with the least fixpoint $\mathit{Sign} \cup \{\mathbb{Z}^{2^n} \mid n \geq 1\}$. □

Finally, the following easy observation will be useful later on.

Lemma 6.4. *Let $F, G \subseteq \mathrm{Fun}(C)$. Then, $\mathscr{S}_F = \mathscr{S}_G$ if and only if for any $A \in \mathrm{Abs}(C)$, A is F-complete $\Leftrightarrow$ A is G-complete.*

Proof. ($\Rightarrow$) If A is F-complete then $A = \mathscr{S}_F(A) = \mathscr{S}_G(A)$ and therefore A is G-complete as well.
($\Leftarrow$) This follows from $\mathscr{S}_F(A) = \sqcup\{X \in \mathrm{Abs}(C) \mid X \sqsubseteq A,\ X \text{ is } F\text{-complete}\} = \sqcup\{X \in \mathrm{Abs}(C) \mid X \sqsubseteq A,\ X \text{ is } G\text{-complete}\} = \mathscr{S}_G(A)$. □

6.2 Strong Preservation and Complete Shells

Let $\mathscr{L}$ be a language with atoms in $AP_{\mathscr{L}}$ and operators in $Op_{\mathscr{L}}$ and let $\mathcal{S} = (\Sigma, I)$ be a semantic structure for $\mathscr{L}$ so that $\boldsymbol{AP}_{\mathscr{L}}$ and $\boldsymbol{Op}_{\mathscr{L}}$ denote, respectively, the corresponding sets of semantic interpretations of atoms and operators. It turns out that forward completeness for $\boldsymbol{AP}_{\mathscr{L}}$ and $\boldsymbol{Op}_{\mathscr{L}}$ implies strong preservation for $\mathscr{L}$.

Lemma 6.5. *If $A \in \mathrm{Abs}(\wp(\Sigma))$ is forward complete for $\boldsymbol{AP}_{\mathscr{L}}$ and $\boldsymbol{Op}_{\mathscr{L}}$ then A is s.p. for $\mathscr{L}$.*

Proof. By Theorem 5.8, it suffices to show that $A \sqsubseteq \mathrm{AD}_{\mathscr{L}}$. Let us show by induction that for any $\varphi \in \mathscr{L}$, $[\![\varphi]\!]_{\mathcal{S}} = \gamma(\alpha([\![\varphi]\!]_{\mathcal{S}}))$.

– $\varphi \equiv p \in AP_{\mathscr{L}}$: since A is forward complete for $\boldsymbol{p}$, $[\![p]\!]_{\mathcal{S}} = \boldsymbol{p} = \gamma(\alpha(\boldsymbol{p})) = \gamma(\alpha([\![p]\!]_{\mathcal{S}}))$.

– $\varphi \equiv f(\varphi_1, \ldots, \varphi_n)$ with $f \in Op_{\mathscr{L}}$:

$$\begin{aligned} [\![f(\varphi_1, ..., \varphi_n)]\!]_{\mathcal{S}} = &\ \text{[by definition]} \\ \boldsymbol{f}([\![\varphi_1]\!]_{\mathcal{S}}, ..., [\![\varphi_n]\!]_{\mathcal{S}}) = &\ \text{[by inductive hypothesis]} \\ \boldsymbol{f}(\gamma(\alpha([\![\varphi_1]\!]_{\mathcal{S}})), ..., \gamma(\alpha([\![\varphi_n]\!]_{\mathcal{S}}))) = &\ \text{[since } A \text{ is forward complete for } \boldsymbol{f}] \\ \gamma(\alpha(\boldsymbol{f}(\gamma(\alpha([\![\varphi_1]\!]_{\mathcal{S}})), ..., \gamma(\alpha([\![\varphi_n]\!]_{\mathcal{S}}))))) = &\ \text{[by inductive hypothesis and by definition]} \\ \gamma(\alpha([\![f(\varphi_1, ..., \varphi_n)]\!]_{\mathcal{S}})). &\end{aligned}$$

□

On the other hand, the converse is not true, that is strong preservation does not imply forward completeness, as shown by the following example.

Example 6.6. Let us consider again Example 5.7 where we showed that the partitioning abstract domain $A = \wp(P)_{\subseteq}$ is s.p. for $\mathscr{L}$. However, A is not forward complete for $\boldsymbol{Op}_{\mathscr{L}} = \{\mathrm{pre}_{\to}\}$. In fact: $\gamma(\alpha(\mathrm{pre}_{\to}(\gamma(\alpha(\{3\}))))) = \gamma(\alpha(\mathrm{pre}_{\to}(\{3\}))) = \gamma(\alpha(\{2,3\})) = \{1,2,3\}$ while $\mathrm{pre}_{\to}(\gamma(\alpha(\{3\}))) = \mathrm{pre}_{\to}(\{3\}) = \{2,3\}$. □

Instead, it turns out that most abstract s.p. domains can be characterized as forward complete shells.

6.2.1 Complete Shells as Strongly Preserving Abstract Domains

Partition refinement algorithms for computing behavioural equivalences like bisimulation [44], simulation equivalence [5, 37, 50] and (divergence blind) stuttering equivalence [33] are used in standard abstract model checking to compute the coarsest strongly preserving partition of temporal languages like CTL* or the μ-calculus for the case of bisimulation equivalence, ACTL* for simulation equivalence and CTL*-X for stuttering equivalence.

Given a language $\mathscr{L}$ and a concrete state space Σ, these partition refinement algorithms work by iteratively refining an initial partition P within the lattice of partitions $\mathrm{Part}(\Sigma)$ until the fixpoint $P_{\mathscr{L}}$ is reached. The input partition P determines the set AP_P of atoms and their interpretation I_P as follows: $AP_P \stackrel{\text{def}}{=} \{p_B \mid B \in P\}$ and $I_P(p_B) \stackrel{\text{def}}{=} B$. More in general, any $\mathcal{X} \subseteq \wp(\Sigma)$ determines a set $\{p_X\}_{X\in\mathcal{X}}$ of atoms with interpretation $I_{\mathcal{X}}(p_X) = X$. In particular, this can be done for an abstract domain $A \in \mathrm{Abs}(\wp(\Sigma))$ by considering its concretization $\gamma(A) \subseteq \Sigma$, namely A is viewed as a set of atoms with interpretation $I_A(a) = \gamma(a)$. Thus, an abstract domain $A \in \mathrm{Abs}(\wp(\Sigma))$ together with a set of functions $F \subseteq \mathrm{Fun}(\wp(\Sigma))$ determine a language $\mathscr{L}_{A,F}$, with atoms in A, operations in F and endowed with a semantic structure $\mathcal{S}_{A,F} = (\Sigma, I_A \cup I_F)$ such that for any $a \in A$, $I_A(a) = \gamma(a)$ and for any $f \in F$, $I_F(f) = f$. Therefore, the most abstract s.p. domain $\mathrm{AD}_{\mathscr{L}_{A,F}}$ defined in Section 5.2 generalizes in our framework the output of a partition refinement algorithm for some language. Accordingly, we aim at characterizing $\mathrm{AD}_{\mathscr{L}_{A,F}}$ as the output of a refinement process of the initial domain A within the lattice $\mathrm{Abs}(\wp(\Sigma))$ of abstract domains. The following result shows that forward completeness for the operations in F is the right notion of refinement to be used for the case of abstract domains.

Theorem 6.7. *Let $A \in \mathrm{Abs}(\wp(\Sigma))$, $F \subseteq \mathrm{Fun}(\wp(\Sigma))$ and assume that $\mathscr{L}_{A,F}$ is closed under infinite logical conjunction. Then, $\mathrm{AD}_{\mathscr{L}_{A,F}} = \mathscr{S}_F(A)$.*

Proof. Since $\mathscr{L}_{A,F}$ is closed under conjunction we have that $\mathrm{AD}_{\mathscr{L}_{A,F}} = \{[\![\varphi]\!]_{\mathcal{S}_{A,F}} \mid \varphi \in \mathscr{L}_{A,F}\}$. Let us first prove that $\{[\![\varphi]\!]_{\mathcal{S}_{A,F}} \mid \varphi \in \mathscr{L}_{A,F}\} \subseteq \mathscr{S}_F(A)$ by structural induction on $\varphi \in \mathscr{L}_{A,F}$:

- $\varphi \equiv a \in A$: $[\![a]\!]_{\mathcal{S}_{A,F}} = I_A(a) = \gamma(a) \in \gamma(A) \subseteq \mathscr{S}_F(A)$.
- $\varphi \equiv f(\varphi_1, ..., \varphi_n)$ with $f \in F$: $[\![f(\varphi_1, ..., \varphi_n)]\!]_{\mathcal{S}_{A,F}} = f([\![\varphi_1]\!]_{\mathcal{S}_{A,F}}, ..., [\![\varphi_n]\!]_{\mathcal{S}_{A,F}})$, where, by inductive hypothesis, $[\![\varphi_i]\!]_{\mathcal{S}_{A,F}} \in \mathscr{S}_F(A)$. Therefore, since $\mathscr{S}_F(A)$ is forward f-complete, we have that $f([\![\varphi_1]\!]_{\mathcal{S}_{A,F}}, ..., [\![\varphi_n]\!]_{\mathcal{S}_{A,F}}) \in \mathscr{S}_F(A)$.

Let us now prove the opposite inclusion. Let us first observe that $\mathrm{AD}_{\mathscr{L}_{A,F}}$ is forward F-complete. For simplicity of notation, consider $f \in F$ with $\sharp(f) = 1$. If $[\![\varphi]\!]_{\mathcal{S}_{A,F}} \in \mathrm{AD}_{\mathscr{L}_{A,F}}$, where $\varphi \in \mathscr{L}_{A,F}$, then, $f(\varphi) \in \mathscr{L}_{A,F}$ and $f([\![\varphi]\!]_{\mathcal{S}_{A,F}}) = [\![f(\varphi)]\!]_{\mathcal{S}_{A,F}} \in \mathrm{AD}_{\mathscr{L}_{A,F}}$.
By Lemma 6.2, we know that $\mathscr{S}_F(A) = \sqcap_{\alpha\in\mathrm{Ord}}(\lambda\rho.\mu_A \sqcap \mathcal{M}(F(\rho)))^{\alpha,\downarrow}(\top_{\mathrm{uco}(\wp(\Sigma))})$, so that it is sufficient to prove by transfinite induction on $\alpha \in \mathrm{Ord}$ that

$$(\lambda\rho.\mu_A \sqcap \mathcal{M}(F(\rho)))^{\alpha,\downarrow}(\top_{\mathrm{uco}(\wp(\Sigma))}) \subseteq \mathrm{AD}_{\mathscr{L}_{A,F}}\,.$$

- $\alpha = 0$: $(\lambda\rho.\mu_A \sqcap \mathcal{M}(F(\rho)))^{0,\downarrow}(\top_{\mathrm{uco}(\wp(\Sigma))}) = \top_{\mathrm{uco}(\wp(\Sigma))} = \{\Sigma\} \in \gamma(A) \subseteq \mathrm{AD}_{\mathscr{L}_{A,F}}$.
- $\alpha + 1$: By inductive hypothesis, $(\lambda\rho.\mu_A \sqcap \mathcal{M}(F(\rho)))^{\alpha,\downarrow}(\top_{\mathrm{uco}(\wp(\Sigma))}) \subseteq \mathrm{AD}_{\mathscr{L}_{A,F}}$. Moreover, $\mathrm{AD}_{\mathscr{L}_{A,F}}$ is Moore-closed and forward F-complete (hence closed under F). Thus, $\mathcal{M}(F((\lambda\rho.\mu_A \sqcap \mathcal{M}(F(\rho)))^{\alpha,\downarrow}(\top_{\mathrm{uco}(\wp(\Sigma))}))) \subseteq \mathrm{AD}_{\mathscr{L}_{A,F}}$, namely $(\lambda\rho.\mu_A \sqcap \mathcal{M}(F(\rho)))^{\alpha+1,\downarrow}(\top_{\mathrm{uco}(\wp(\Sigma))}) \subseteq \mathrm{AD}_{\mathscr{L}_{A,F}}$.
- limit ordinal α: This follows from

$$(\lambda\rho.\mu_A \sqcap \mathcal{M}(F(\rho)))^{\alpha,\downarrow}(\top_{\mathrm{uco}(\wp(\Sigma))}) = \sqcap_{\beta<\alpha}(\lambda\rho.\mu_A \sqcap \mathcal{M}(F(\rho)))^{\beta,\downarrow}(\top_{\mathrm{uco}(\wp(\Sigma))})$$

  because, by inductive hypothesis, $(\lambda\rho.\mu_A \sqcap \mathcal{M}(F(\rho)))^{\beta,\downarrow}(\top_{\mathrm{uco}(\wp(\Sigma))}) \subseteq \mathrm{AD}_{\mathscr{L}_{A,F}}$, for any $\beta < \alpha$.

$\square$

6.2.2 Strongly Preserving Abstract Domains as Complete Shells

Let us consider a language $\mathscr{L}$, with atoms in $AP_{\mathscr{L}}$ and operators in $Op_{\mathscr{L}}$, and a semantic structure $\mathcal{S} = (\Sigma, I)$. As an immediate consequence of Theorem 6.7, the most abstract s.p. domain $\mathrm{AD}_{\mathscr{L}}$ for $\mathscr{L}$ w.r.t. $\mathcal{S}$ can be characterized as the forward $\boldsymbol{AP}_{\mathscr{L}} \cup \boldsymbol{Op}_{\mathscr{L}}$-complete shell of the most abstract domain $\{\Sigma\}$.

Corollary 6.8. *Let $\mathscr{L}$ be closed under infinite logical conjunction. Then,* $\mathrm{AD}_{\mathscr{L}} = \mathscr{S}_{\boldsymbol{AP}_{\mathscr{L}} \cup \boldsymbol{Op}_{\mathscr{L}}}(\{\Sigma\})$.

Let us also observe that $\mathrm{AD}_{\mathscr{L}}$ can be equivalently characterized as the forward $\boldsymbol{Op}_{\mathscr{L}}$-complete shell of an initial abstract domain $\mathcal{M}(\boldsymbol{AP}_{\mathscr{L}})$ induced by atoms: $\mathrm{AD}_{\mathscr{L}} = \mathscr{S}_{\boldsymbol{Op}_{\mathscr{L}}}(\mathcal{M}(\boldsymbol{AP}_{\mathscr{L}}))$.

6.2.3 Strongly Preserving Partitions

Theorem 6.7 and Corollary 6.8 provide an elegant generalization of partition refinement algorithms for strong preservation from an abstract interpretation perspective.

Given a language $\mathscr{L}$ with operators in $Op_{\mathscr{L}}$ and a corresponding semantic structure $\mathcal{S} = (\Sigma, I)$, as recalled in Section 6.2.1, an input partition $P \in \mathrm{Part}(\Sigma)$ for a partition refinement algorithm determines the set $AP_{\mathscr{L}} = \{p_B \mid B \in P\}$ of atoms of $\mathscr{L}$ and their interpretation $I(p_B) = B$. Thus, $\mathcal{M}(\boldsymbol{AP}_{\mathscr{L}}) = \mathcal{M}(P) = P \cup \{\varnothing, \Sigma\}$. It turns out that the coarsest s.p. partition $P_{\mathscr{L}}$ for $\mathscr{L}$ can be characterized in our abstract domain-based approach as follows.

Corollary 6.9. *Let $\mathscr{L}$ be closed under infinite logical conjunction.*
(1) $P_{\mathscr{L}} = \mathrm{par}(\mathscr{S}_{\boldsymbol{Op}_{\mathscr{L}}}(\mathcal{M}(P)))$.
(2) *Let $\mathscr{L}$ be closed under logical negation. Then,* $\mathrm{ad}^{\mathrm{p}}(P_{\mathscr{L}}) = \mathscr{S}_{\boldsymbol{Op}_{\mathscr{L}}}(\mathcal{M}(P))$.

Proof. (1) By Corollary 6.8, $\mathrm{AD}_{\mathscr{L}} = \mathscr{S}_{\boldsymbol{Op}_{\mathscr{L}}}(\mathcal{M}(P))$ and by Proposition 5.10 (1), $P_{\mathscr{L}} = \mathrm{par}(\mathrm{AD}_{\mathscr{L}}) = \mathrm{par}(\mathscr{S}_{\boldsymbol{Op}_{\mathscr{L}}}(\mathcal{M}(P)))$.
(2) By Proposition 5.10 (1) and (3), Corollary 6.8 and point (1), $\mathrm{ad}^{\mathrm{p}}(P_{\mathscr{L}}) = \mathrm{ad}^{\mathrm{p}}(\mathrm{par}(\mathrm{AD}_{\mathscr{L}})) = \mathrm{AD}_{\mathscr{L}} = \mathscr{S}_{\boldsymbol{Op}_{\mathscr{L}}}(\mathcal{M}(P))$. □

It is worth remarking that when $\mathscr{L}$ is not closed under negation, by Proposition 5.10 (3) and Corollary 6.9 (2), it turns out that $\mathrm{ad}^{\mathrm{p}}(P_{\mathscr{L}}) \sqsubset \mathscr{S}_{\boldsymbol{Op}_{\mathscr{L}}}(\mathcal{M}(P))$. This means that when $\mathscr{L}$ is not closed under negation the output partition $P_{\mathscr{L}}$ of any partition refinement algorithm for achieving strong preservation for $\mathscr{L}$ is not optimal within the lattice of abstract domains.

Example 6.10. Let us consider the language $\mathscr{L}$ and the concrete Kripke structure $\mathcal{K}$ in Example 5.12. The labeling determines the initial partition $P = \{\boldsymbol{p} = 1234, \boldsymbol{q} = 5\} \in \mathrm{Part}(\Sigma)$, so that $\mathcal{M}(P) = \{\varnothing, 1234, 5, 12345\} \in \mathrm{Abs}(\wp(\Sigma))$. Here, $Op_{\mathscr{L}} = \{\wedge, \mathrm{EF}_{[0,2]}\}$. Abstract domains are Moore-closed so that $\mathscr{S}_{\boldsymbol{Op}_{\mathscr{L}}} = \mathscr{S}_{\mathbf{EF}_{[0,2]}}$. Let us compute $\mathscr{S}_{\mathbf{EF}_{[0,2]}}(\mathcal{M}(P))$.

$$\begin{aligned}
A_0 &= \mathcal{M}(P) = \{\varnothing, 1234, 5, 12345\} \\
A_1 &= A_0 \sqcap \mathcal{M}(\mathbf{EF}_{[0,2]}(A_0)) = \mathcal{M}(A_0 \cup \mathbf{EF}_{[0,2]}(A_0)) \\
&= \mathcal{M}(\{\varnothing, 1234, 5, 12345\} \cup \{\mathbf{EF}_{[0,2]}(\{5\}) = 345\}) = \{\varnothing, 5, 34, 1234, 12345\} \\
A_2 &= A_1 \quad \text{(fixpoint)}
\end{aligned}$$

As already observed in Example 5.12, $P_{\mathscr{L}} = \{12, 34, 5\}$ is such that $\mathrm{ad}^{\mathrm{p}}(P_{\mathscr{L}}) \sqsubset \mu_{\mathscr{L}}$ and it is not possible to define a strongly preserving abstract Kripke structure on the abstract space $P_{\mathscr{L}}$. □

7 An Application to some Behavioural Equivalences

It is well known that some temporal languages like CTL, ACTL and CTL-X induce state logical equivalences that coincide with standard behavioural equivalences like bisimulation equivalence for CTL, (divergence blind) stuttering equivalence for CTL-X and simulation equivalence for ACTL. Also, these behavioural equivalences can be computed through well-known coarsest partition refinement algorithms like those by Paige and Tarjan [44], Groote and Vaandrager [33] and Henzinger et al. [37]. We derive here a novel characterization of these behavioural equivalences and corresponding algorithms in terms of forward completeness of abstract interpretations.

7.1 Bisimulation Equivalence

Let $\mathcal{K} = (\Sigma, \to, \ell)$ be a Kripke structure over some set AP of atomic propositions. A relation $R \subseteq \Sigma \times \Sigma$ is a bisimulation on $\mathcal{K}$ if for any $s, s' \in \Sigma$ such that sRs':

(1) $\ell(s) = \ell(s')$;

(2) For any $t \in \Sigma$ such that $s \to t$, there exists $t' \in \Sigma$ such that $s' \to t'$ and tRt';

(3) $s'Rs$, i.e. R is symmetric.

Since the empty relation is a bisimulation and bisimulations are closed under union, it turns out that the largest (as a set) bisimulation relation exists. This largest bisimulation is an equivalence relation called bisimulation equivalence and is denoted by $\sim_{\mathrm{bis}}$ while $P_{\mathrm{bis}} \in \mathrm{Part}(\Sigma)$ denotes the corresponding partition. Thus, a partition $P \in \mathrm{Part}(\Sigma)$ is a bisimulation on $\mathcal{K}$ when $P \preccurlyeq P_{\mathrm{bis}}$.

It is well known [4] that when $\mathcal{K}$ is finitely branching, bisimulation equivalence coincides with the state equivalence induced by CTL, i.e., $P_{\mathrm{bis}} = P_{\mathrm{CTL}}$ (the same holds for CTL* and the μ-calculus, see e.g. [20, Lemma 6.2.0.5]). Moreover, it is known (see e.g. [51, Section 12]) that it is enough to consider finitary Hennessy-Milner logic [36], i.e. a language $\mathscr{L}_1$ including propositional logic and the existential next operator in order to have that $P_{\mathscr{L}_1} = P_{\mathrm{bis}}$:

$$\mathscr{L}_1 \ni \varphi ::= p \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \mathrm{EX}\varphi$$

where, as usual, the interpretation $\mathbf{EX}$ of EX in $\mathcal{K}$ is $\mathrm{pre}_{\to}$. A number of algorithms for computing bisimulation equivalence exists [3, 24, 40, 44]. The Paige-Tarjan algorithm [44] runs in $O(|{\to}| \log(|\Sigma|))$-time and is the most time-efficient algorithm that computes bisimulation equivalence.

We recalled above that $P_{\mathscr{L}_1} = P_{\mathrm{CTL}}$. In our framework, this can be obtained as a consequence of the fact that the most abstract s.p. domains for CTL and $\mathscr{L}_1$ coincide.

Lemma 7.1. *Let $\mathcal{K}$ be finitely branching. Then,* $\mathrm{AD}_{\mathrm{CTL}} = \mathrm{AD}_{\mathscr{L}_1} = \mathrm{ad}^{\mathrm{p}}(P_{\mathrm{bis}})$.

Proof. Let $\boldsymbol{Op}_{\mathrm{CTL}} = \{\cap, \complement, \mathbf{AX}, \mathbf{EX}, \mathbf{AU}, \mathbf{EU}, \mathbf{AR}, \mathbf{ER}\}$ be the set of standard interpretations of the operators of CTL on $\mathcal{K}$, so that $\mathbf{AX} = \widetilde{\mathrm{pre}}_{\to}$ and $\mathbf{EX} = \mathrm{pre}_{\to}$. We show that $\mu \in \mathrm{uco}(\wp(\Sigma))$ is forward complete for $\boldsymbol{Op}_{\mathrm{CTL}}$ iff μ is forward complete for $\{\complement, \mathrm{pre}_{\to}\}$. Assume that μ is forward complete for $\{\complement, \mathrm{pre}_{\to}\}$. Let us first prove that μ is forward complete for $\widetilde{\mathrm{pre}}_{\to} = \mathbf{AX}$:

$$\begin{aligned}
\mu \circ \widetilde{\mathrm{pre}}_{\to} \circ \mu &= && [\text{by definition of } \widetilde{\mathrm{pre}}_{\to}]\\
\mu \circ \complement \circ \mathrm{pre}_{\to} \circ \complement \circ \mu &= && [\text{as } \mu \text{ is complete for } \complement]\\
\mu \circ \complement \circ \mathrm{pre}_{\to} \circ \mu \circ \complement \circ \mu &= && [\text{as } \mu \text{ is complete for } \mathrm{pre}_{\to}]\\
\mu \circ \complement \circ \mu \circ \mathrm{pre}_{\to} \circ \mu \circ \complement \circ \mu &= && [\text{as } \mu \text{ is complete for } \complement]\\
\complement \circ \mu \circ \mathrm{pre}_{\to} \circ \mu \circ \complement \circ \mu &= && [\text{as } \mu \text{ is complete for } \mathrm{pre}_{\to}]\\
\complement \circ \mathrm{pre}_{\to} \circ \mu \circ \complement \circ \mu &= && [\text{as } \mu \text{ is complete for } \complement]\\
\complement \circ \mathrm{pre}_{\to} \circ \complement \circ \mu &= && [\text{by definition of } \widetilde{\mathrm{pre}}_{\to}]\\
\widetilde{\mathrm{pre}}_{\to} \circ \mu & &&
\end{aligned}$$

The following fixpoint characterizations are well known [10]:

- $\mathbf{AU}(S_1, S_2) = \mathrm{lfp}(\lambda Z. S_2 \cup (S_1 \cap \widetilde{\mathrm{pre}}_{\to}(Z)))$;
- $\mathbf{EU}(S_1, S_2) = \mathrm{lfp}(\lambda Z. S_2 \cup (S_1 \cap \mathrm{pre}_{\to}(Z)))$;
- $\mathbf{AR}(S_1, S_2) = \mathrm{gfp}(\lambda Z. S_2 \cap (S_1 \cup \widetilde{\mathrm{pre}}_{\to}(Z)))$;
- $\mathbf{ER}(S_1, S_2) = \mathrm{gfp}(\lambda Z. S_2 \cap (S_1 \cup \mathrm{pre}_{\to}(Z)))$.

Let us show that μ is forward complete for $\mathbf{AU}$. The proofs for the remaining operators in $\boldsymbol{Op}_{\mathrm{CTL}}$ are analogous. We need to show that $\mu(\mathrm{lfp}(\lambda Z.\mu(S_2) \cup (\mu(S_1) \cap \widetilde{\mathrm{pre}}_{\to}(Z)))) = \mathrm{lfp}(\lambda Z.\mu(S_2) \cup (\mu(S_1) \cap \widetilde{\mathrm{pre}}_{\to}(Z)))$. Let us show that μ is forward complete for the function $\lambda Z.\mu(S_2) \cup (\mu(S_1) \cap \widetilde{\mathrm{pre}}_{\to}(Z))$:

$$\begin{aligned}
\mu(\mu(S_2) \cup (\mu(S_1) \cap \widetilde{\mathrm{pre}}_{\to}(\mu(Z)))) &= && \text{[as } \mu \text{ is complete for } \widetilde{\mathrm{pre}}_{\to}] \\
\mu(\mu(S_2) \cup (\mu(S_1) \cap \mu(\widetilde{\mathrm{pre}}_{\to}(\mu(Z))))) &= && \text{[as } \mu \text{ is complete for } \cap] \\
\mu(\mu(S_2) \cup \mu(\mu(S_1) \cap \mu(\widetilde{\mathrm{pre}}_{\to}(\mu(Z))))) &= && \text{[as } \mu \text{ is complete for } \cup] \\
\mu(S_2) \cup \mu(\mu(S_1) \cap \mu(\widetilde{\mathrm{pre}}_{\to}(\mu(Z)))) &= && \text{[as } \mu \text{ is complete for } \cap] \\
\mu(S_2) \cup (\mu(S_1) \cap \mu(\widetilde{\mathrm{pre}}_{\to}(\mu(Z)))) &= && \text{[as } \mu \text{ is complete for } \widetilde{\mathrm{pre}}_{\to}] \\
\mu(S_2) \cup (\mu(S_1) \cap \widetilde{\mathrm{pre}}_{\to}(\mu(Z))). &
\end{aligned}$$

Observe that since μ is additive (and therefore continuous) we have that $\mu(\varnothing) = \varnothing$. Moreover, let us show that from the hypothesis that $\mathcal{K}$ is finitely branching it follows that $\widetilde{\mathrm{pre}}_{\to}$ is continuous. First, notice that $\widetilde{\mathrm{pre}}_{\to}$ is continuous iff $\mathrm{pre}_{\to}$ is co-continuous. Hence, let us check that $\mathrm{pre}_{\to}$ is co-continuous. Let $\{X_i\}_{i\in\mathbb{N}}$ be a decreasing chain of subsets of Σ and let $x \in \cap_{i\in\mathbb{N}} \mathrm{pre}_{\to}(X_i)$. Since $\mathcal{K}$ is finitely branching, $\mathrm{post}_{\to}(\{x\})$ is finite so that there exists some $k \in \mathbb{N}$ such that for any $j > 0$, $\mathrm{post}_{\to}(\{x\}) \cap X_k = \mathrm{post}_{\to}(\{x\}) \cap X_{k+j}$. Hence, there exists some $z \in \cap_{i\in\mathbb{N}} X_i \cap \mathrm{post}_{\to}(\{x\})$, so that $x \in \mathrm{pre}_{\to}(\cap_{i\in\mathbb{N}} X_i)$. Therefore, since $\widetilde{\mathrm{pre}}_{\to}$ is continuous we also have that $\lambda Z.\mu(S_2) \cup (\mu(S_1) \cap \widetilde{\mathrm{pre}}_{\to}(Z))$ is continuous. We can therefore apply Lemma 2.1 so that $\mu(\mathrm{lfp}(\lambda Z.\mu(S_2) \cup (\mu(S_1) \cap \widetilde{\mathrm{pre}}_{\to}(Z)))) = \mathrm{lfp}(\lambda Z.\mu(S_2) \cup (\mu(S_1) \cap \widetilde{\mathrm{pre}}_{\to}(Z)))$.
Thus, by Lemma 6.4, $\mathscr{S}_{\{\mathtt{C},\mathrm{pre}_{\to}\}} = \mathscr{S}_{\boldsymbol{Op}_{\mathrm{CTL}}}$, so that, by Corollary 6.8, $\mathrm{AD}_{\mathscr{L}_1} = \mathrm{AD}_{\mathrm{CTL}}$. Finally, since $\mathcal{K}$ is finitely branching and $\mathscr{L}_1$ is closed under conjunction and negation, $\mathrm{ad}^{\mathrm{p}}(P_{\mathscr{L}_1}) = \mathrm{ad}^{\mathrm{p}}(P_{\mathrm{bis}}) = \mathrm{ad}^{\mathrm{p}}(P_{\mathscr{L}_1}) = \mathrm{AD}_{\mathscr{L}_1}$. ☐

As a consequence of this and of the results in Section 6 (in particular of Corollary 6.9), any partition refinement algorithm $\mathrm{Alg}_{\mathrm{bis}}$ for computing bisimulation equivalence on a finitely branching Kripke structure, like those in [3, 24, 40, 44], can be characterized as a complete shell refinement as follows:

$$\mathrm{Alg}_{\mathrm{bis}}(P) = \mathrm{par}(\mathscr{S}_{\{\mathtt{C},\mathrm{pre}_{\to}\}}(\mathcal{M}(P))).$$

Thus, $\mathrm{Alg}_{\mathrm{bis}}$ is viewed as an algorithm for computing a particular abstraction, that is par, of a particular complete shell, that is $\mathscr{S}_{\{\mathtt{C},\mathrm{pre}_{\to}\}}$. In particular, this holds for the Paige-Tarjan algorithm [44] and leads to design a generalized Paige-Tarjan-like procedure for computing most abstract strongly preserving domains [47].

Finally, our abstract intepretation-based approach allows us to give the following nice characterization of bisimulation for a partition P in terms of forward completeness for the corresponding partitioning abstract domain $\mathrm{ad}^{\mathrm{p}}(P)$.

Theorem 7.2. *Let $P \in \mathrm{Part}(\Sigma)$. Then, P is a bisimulation on $\mathcal{K}$ iff $\mathrm{ad}^{\mathrm{p}}(P)$ is forward complete for $\{\boldsymbol{p} \mid p \in AP\} \cup \{\mathrm{pre}_{\to}\}$.*

Proof. We view $\mathrm{ad}^{\mathrm{p}}(P)$ as a uco so that $\mathrm{ad}^{\mathrm{p}}(P) = \{\cup_i B_i \in \wp(\Sigma) \mid \{B_i\} \subseteq P\}$. Let us first observe that $P \preccurlyeq P_\ell$ iff $\mathrm{ad}^{\mathrm{p}}(P)$ is forward complete for $\{\boldsymbol{p} \subseteq \Sigma \mid p \in AP\}$. On the one hand, since $\boldsymbol{p} = \{s \in \Sigma \mid p \in \ell(s)\}$, if $s \in \boldsymbol{p}$ and $s \in B$, for some $B \in P$, then $B \subseteq [s]_\ell \subseteq \boldsymbol{p}$. Hence, $\boldsymbol{p}$ is a union of some blocks of P and therefore $\boldsymbol{p} \in \mathrm{ad}^{\mathrm{p}}(P)$. On the other hand, if $\mathrm{ad}^{\mathrm{p}}(P)$ contains $\{\boldsymbol{p} \subseteq \Sigma \mid p \in AP\}$ then, for any $p \in AP$, $\boldsymbol{p}$ is a union of some blocks in P. Thus, for any $B \in P$, either $B \subseteq \boldsymbol{p}$ or $B \cap \boldsymbol{p} = \varnothing$. Consequently, if $s \in B$ then $B \subseteq [s]_\ell \in P_\ell$.
Let us now note that $\mathrm{ad}^{\mathrm{p}}(P)$ is forward complete for $\mathrm{pre}_{\to}$ iff for any block $B \in P$, $\mathrm{pre}_{\to}(B)$ is a (possibly empty) union of blocks of P: this holds because $\mathrm{pre}_{\to}$ is additive, and therefore if $\{B_i\} \subseteq P$ then $\mathrm{pre}_{\to}(\cup_i B_i) = \cup_i \mathrm{pre}_{\to}(B_i)$. The fact that, for some $B \in P$, $\mathrm{pre}_{\to}(B) = \cup_i B_i$, for some blocks $\{B_i\} \subseteq P$, implies that if $s \in \mathrm{pre}_{\to}(B)$, i.e., $s \to t$ for some $t \in B$, then $s \in B_j$, for some j, and if $s' \in B_j$ then $s' \in \mathrm{pre}_{\to}(B)$, i.e., $s' \to t'$ for some $t' \in B$, namely condition (2) of bisimulation for P holds. On the other hand, if condition (2) of bisimulation for P holds then if $s, s' \in B'$ and $s \in \mathrm{pre}_{\to}(B)$, for some $B, B' \in P$, then $s' \to t'$ for some $t' \in B$, i.e., $s' \in \mathrm{pre}_{\to}(B)$, and therefore $\mathrm{pre}_{\to}(B)$ is a union of blocks of P. This closes the proof. ☐

7.1.1 On the Smallest Abstract Transition Relation

As recalled in Section 2.3, the abstract Kripke structure $\mathcal{A} = (P_{\text{bis}}, \rightarrow^{\exists\exists}, \ell^{\exists})$ strongly preserves CTL, where $B_1 \rightarrow^{\exists\exists} B_2$ iff there exist $s_1 \in B_1$ and $s_2 \in B_2$ such that $s_1 \rightarrow s_2$, and $\ell^{\exists}(B) = \cup_{s \in B} \ell(s)$. As a simple and elegant consequence of our approach, it is easy to show that $\rightarrow^{\exists\exists}$ is the *unique* (and therefore the smallest) abstract transition relation on P_{bis} that induces strong preservation for CTL.

Let $\mathcal{K} = (\Sigma, \rightarrow, \ell)$ be finitely branching so that, by Lemma 7.1, $\text{AD}_{\mathscr{L}_1} = \text{ad}^{\text{p}}(P_{\text{bis}}) = \wp(P_{\text{bis}})$. Recall that the concrete interpretation I induced by $\mathcal{K}$ is such that $I(\text{EX}) = \text{pre}_{\rightarrow}$. By Theorem 5.9, the unique interpretation of atoms and operations in $\mathscr{L}_1$ on the abstract domain $\wp(P_{\text{bis}})$ that gives rise to a s.p. abstract semantics is the best correct approximation $I^{\wp(P_{\text{bis}})}$. Hence, if $\mathcal{A} = (P_{\text{bis}}, \rightarrow^{\sharp}, \ell^{\sharp})$ is strongly preserving for CTL then the interpretation $\text{pre}_{\rightarrow^{\sharp}}$ of EX induced by $\mathcal{A}$ must coincide with $I^{\wp(P_{\text{bis}})}(\text{EX})$. Consequently, $\text{pre}_{\rightarrow^{\sharp}} = \alpha \circ \text{pre}_{\rightarrow} \circ \gamma$ so that for any $B_1, B_2 \in P_{\text{bis}}$, we have that $B_1 \rightarrow^{\sharp} B_2$ iff $B_1 \in \alpha(\text{pre}_{\rightarrow}(\gamma(\{B_2\})))$. Therefore, we conclude by observing that $B_1 \in \alpha(\text{pre}_{\rightarrow}(\gamma(\{B_2\})))$ iff $B_1 \rightarrow^{\exists\exists} B_2$.

We believe that a similar reasoning could be also useful for other languages $\mathscr{L}$ in order to prove that the smallest abstract transition relation on $P_{\mathscr{L}}$ that induces strong preservation exists. For example, this has been proved for the case of ACTL by Bustan and Grumberg [5].

7.2 Stuttering Equivalence

Lamport's criticism [39] of the next-time operator X in CTL/CTL* is well known. This motivated the study of temporal logics CTL-X/CTL*-X obtained from CTL/CTL* by removing the next-time operator and this led to study notions of behavioural *stuttering*-based equivalences [4, 23, 33]. We are interested here in *divergence blind stuttering* (dbs for short) equivalence. Let $\mathcal{K} = (\Sigma, \rightarrow, \ell)$ be a Kripke structure over a set AP of atoms. A relation $R \subseteq \Sigma \times \Sigma$ is a divergence blind stuttering relation on $\mathcal{K}$ if for any $s, s' \in \Sigma$ such that sRs':

(1) $\ell(s) = \ell(s')$;

(2) If $s \rightarrow t$ then there exist $t_0, ..., t_k \in \Sigma$, with $k \geq 0$, such that: (i) $t_0 = s'$; (ii) for all $i \in [0, k-1]$, $t_i \rightarrow t_{i+1}$ and sRt_i; (iii) tRt_k;

(3) $s'Rs$, i.e. R is symmetric.

Observe that condition (2) allows the case $k = 0$ and this simply boils down to requiring that tRs'. Since the empty relation is a dbs relation and dbs relations are closed under union, it turns out that the largest dbs relation exists. It turns out that this largest dbs relation is an equivalence relation called dbs equivalence and is denoted by $\sim_{\text{dbs}}$ while $P_{\text{dbs}} \in \text{Part}(\Sigma)$ denotes the corresponding partition. In particular, a partition $P \in \text{Part}(\Sigma)$ is a dbs relation on $\mathcal{K}$ when when $P \preccurlyeq P_{\text{dbs}}$.

De Nicola and Vaandrager [23, Theorem 3.2.5] showed that for finite Kripke structures and for an interpretation of universal/existential path quantifiers over all the, possibly finite, prefixes, dbs equivalence coincides with the state equivalence induced from the language CTL-X (this also holds for CTL*-X), that is $P_{\text{dbs}} = P_{\text{CTL-X}}$. This is not true with the standard interpretation of path quantifiers over infinite paths, since this requires a divergence sensitive notion of stuttering (see the details in [23]). Groote and Vaandrager [33] presented a partition refinement algorithm that computes the partition P_{dbs} in $O(|\Sigma||\rightarrow|)$-time.

We provide a characterization of divergence blind stuttering equivalence as the state equivalence induced by the following language $\mathscr{L}_2$ that includes propositional logic and the existential until operator EU, where the interpretation of the existential path quantifier is standard, i.e. over infinite paths:

$$\mathscr{L}_2 \ni \varphi ::= \; p \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \text{EU}(\varphi_1, \varphi_2)$$

Since the transition relation $\rightarrow$ is assumed to be total, let us recall that the standard semantics $\mathbf{EU}_{\rightarrow} : \wp(\Sigma)^2 \rightarrow \wp(\Sigma)$ of the existential until operator is as follows:

$$\begin{aligned}\mathbf{EU}_{\rightarrow}(S_1, S_2) = S_2 \cup \{s \in S_1 \mid \exists s_0, ..., s_n \in \Sigma,\ &\text{with } n \geq 0, \text{ such that (i) } s_0 = s,\\ &\text{(ii) } \forall i \in [0, n-1].\, s_i \in S_1 \text{ and } s_i \rightarrow s_{i+1},\ \text{(iii) } s_n \in S_2\}.\end{aligned}$$

The following result characterizes a dbs partition P in terms of forward completeness for the corresponding partitioning abstract domain $\mathrm{ad}^{\mathrm{p}}(P)$.

Theorem 7.3. *Let $P \in \mathrm{Part}(\Sigma)$. Then, $P \in \mathrm{Part}(\Sigma)$ is a dbs partition on $\mathcal{K}$ iff $\mathrm{ad}^{\mathrm{p}}(P)$ is forward complete for $\{\boldsymbol{p} \mid p \in AP\} \cup \{\mathbf{EU}_{\to}\}$.*

Proof. As already shown in the proof of Theorem 7.2, it turns out that $P \preccurlyeq P_\ell$ iff $\mathrm{ad}^{\mathrm{p}}(P)$ is forward complete for $\{\boldsymbol{p} \subseteq \Sigma \mid p \in AP\}$. Thus, it remains to show $P \in \mathrm{Part}(\Sigma)$ satisfies condition (2) of the definition of dbs relation iff $\mathrm{ad}^{\mathrm{p}}(P)$ is forward complete for $\mathbf{EU}_{\to}$. Let us first observe that $P \in \mathrm{Part}(\Sigma)$ satisfies this condition (2) iff for any $B_1, B_2 \in P$, $\mathbf{EU}_{\to}(B_1, B_2) \in \{B_2, B_1 \cup B_2\}$.
($\Rightarrow$) If $B_1 = B_2$ then $\mathbf{EU}_{\to}(B_2, B_2) = B_2$. Otherwise, assume that $B_1 \neq B_2$. If $B_2 \subsetneq \mathbf{EU}_{\to}(B_1, B_2) \subseteq B_1 \cup B_2$ then there exists $s \in \mathbf{EU}_{\to}(B_1, B_2)$ such that $s \in B_1$. Thus, if $s' \in B_1$ then, by condition (2), $s' \in \mathbf{EU}_{\to}(B_1, B_2)$. This implies that $\mathbf{EU}_{\to}(B_1, B_2) = B_1 \cup B_2$.
($\Leftarrow$) Let $B \in P$, $s, s' \in B$ and $s{\to}t$. If $t \in B$ then condition (2) is satisfied. Otherwise, $t \in B'$, for some $B' \in P$, with $B \neq B'$. We have that $\mathbf{EU}_{\to}(B, B') = B \cup B'$ so that $s' \in \mathbf{EU}_{\to}(B, B')$ and therefore condition (2) is satisfied.
To complete the proof it is now sufficient to show that if, for any $B_1, B_2 \in P$, $\mathbf{EU}_{\to}(B_1, B_2) = B_1 \cup B_2$ then $\mathrm{ad}^{\mathrm{p}}(P)$ is forward complete for $\mathbf{EU}_{\to}$, i.e., for any $\{B_i\}_{i\in I}, \{B_j\}_{j\in J} \subseteq P$, $\mathbf{EU}_{\to}(\cup_i B_i, \cup_j B_j) = \cup_k B_k$, for some $\{B_k\}_{k\in K} \subseteq P$. The function $\mathbf{EU}_{\to}$ is additive in its second argument, thus we only need to show that, for any $B \in P$, $\mathbf{EU}_{\to}(\cup_i B_i, B) = \cup_k B_k$, namely if $s \in \mathbf{EU}_{\to}(\cup_i B_i, B)$ and $s \in B'$, for some $B' \in P$, then $B' \subseteq \mathbf{EU}_{\to}(\cup_i B_i, B)$. If $s \in \mathbf{EU}_{\to}(\cup_i B_i, B)$ and $s \in B'$, for some $B' \in \{B_i\}_{i\in I}$, then there exist $n \geq 0$ and $s_0, ..., s_n \in \Sigma$ such that $s_0 = s$, $\forall j \in [0, n-1].s_j \in \cup_i B_i$ and $s_j{\to}s_{j+1}$, and $s_n \in B$. Let us prove by induction on $n \in \mathbb{N}$ that if $s' \in B'$ then $s' \in \mathbf{EU}_{\to}(\cup_i B_i, B)$.

($n = 0$): In this case $s \in \cup_i B_i$ and $s \in B = B'$. Hence, for some k, $s \in B_k = B = B'$ and therefore $s \in \mathbf{EU}_{\to}(B, B)$. By hypothesis, $\mathbf{EU}_{\to}(B, B) = B$. Moreover, $\mathbf{EU}_{\to}$ is monotone on its first component and therefore $B' = B = \mathbf{EU}_{\to}(B, B) \subseteq \mathbf{EU}_{\to}(\cup_i B_i, B)$.

($n{+}1$): Suppose that there exist $s_0, ..., s_{n+1} \in \Sigma$ such that $s_0 = s$, $\forall j \in [0, n].s_j \in \cup_i B_i$ and $s_j{\to}s_{j+1}$, and $s_{n+1} \in B$. Let $s_n \in B_k$, for some $B_k \in \{B_i\}_{i\in I}$. Then, $s \in \mathbf{EU}_{\to}(\cup_i B_i, B_k)$ and $s = s_0{\to}s_1{\to}...{\to}s_n$. Since this finite path has length n, by inductive hypothesis, $s' \in \mathbf{EU}_{\to}(\cup_i B_i, B_k)$. Hence, there exist $r_0, ..., r_m \in \Sigma$, with $m \geq 0$, such that $s' = r_0$, $\forall j \in [0, m-1].r_j \in \cup_i B_i$ and $r_j{\to}r_{j+1}$, and $r_m \in B_k$. Moreover, since $s_n{\to}s_{n+1}$, we have that $s_n \in \mathbf{EU}_{\to}(B_k, B)$. By hypothesis, $\mathbf{EU}_{\to}(B_k, B) = B_k \cup B$, and therefore $r_m \in \mathbf{EU}_{\to}(B_k, B)$. Thus, there exist $q_0, ..., q_l \in \Sigma$, with $l \geq 0$, such that $r_m = q_0$, $\forall j \in [0, l-1].q_j \in B_k$ and $q_j{\to}q_{j+1}$, and $q_l \in B$. We have thus found the following finite path: $s' = r_0{\to}r_1{\to}...{\to}r_m = q_0{\to}q_1{\to}...{\to}q_l$, where all the states in the sequence but the last one q_l belong to $\cup_i B_i$, while $q_l \in B$. This means that $s' \in \mathbf{EU}_{\to}(\cup_i B_i, B)$. $\square$

As a consequence, we obtain a characterization of dbs equivalence as the state equivalence induced by the standard interpretation of the language $\mathscr{L}_2$.

Corollary 7.4. *Let Σ be finite. Then, $P_{\mathrm{dbs}} = P_{\mathscr{L}_2}$.*

Proof. By definition, $P_{\mathrm{dbs}} = \curlyvee_{\mathrm{Part}(\Sigma)}\{P \in \mathrm{Part}(\Sigma) \mid P \text{ is a dbs relation on } \mathcal{K}\}$. By Theorem 7.3, $P_{\mathrm{dbs}} = \curlyvee_{\mathrm{Part}(\Sigma)}\{P \in \mathrm{Part}(\Sigma) \mid \mathrm{ad}^{\mathrm{p}}(P) \text{ is complete for } \{\boldsymbol{p} \mid p \in AP\} \cup \{\mathbf{EU}_{\to}\}\}$. By Theorem 3.2, ad^{p} is co-additive on $\mathrm{Part}(\Sigma)_{\succcurlyeq}$, that is ad^{p} preserves lub's in $\mathrm{Part}(\Sigma)_{\preccurlyeq}$. Hence, $\mathrm{ad}^{\mathrm{p}}(P_{\mathrm{dbs}}) = \sqcup_{\mathrm{Abs}(\wp(\Sigma))}\{\mathrm{ad}^{\mathrm{p}}(P) \in \mathrm{Abs}(\wp(\Sigma)) \mid P \in \mathrm{Part}(\Sigma),\ \mathrm{ad}^{\mathrm{p}}(P) \text{ is complete for } \{\boldsymbol{p} \mid p \in AP\} \cup \{\mathbf{EU}_{\to}\}\}$. By Theorem 3.2, $\mathrm{Abs}^{\mathrm{par}}(\wp(\Sigma)) = \{\mathrm{ad}^{\mathrm{p}}(P) \mid P \in \mathrm{Part}(\Sigma)\}$ so that $\mathrm{ad}^{\mathrm{p}}(P_{\mathrm{dbs}}) = \sqcup_{\mathrm{Abs}(\wp(\Sigma))}\{A \in \mathrm{Abs}^{\mathrm{par}}(\wp(\Sigma)) \mid A \text{ is complete for } \{\boldsymbol{p} \mid p \in AP\} \cup \{\mathbf{EU}_{\to}\}\}$. By Corollary 3.3, $A \in \mathrm{Abs}^{\mathrm{par}}(\wp(\Sigma))$ iff A is forward complete for $\complement$, so that $\mathrm{ad}^{\mathrm{p}}(P_{\mathrm{dbs}}) = \sqcup_{\mathrm{Abs}(\wp(\Sigma))}\{A \in \mathrm{Abs}(\wp(\Sigma)) \mid A \text{ is complete for } \{\boldsymbol{p} \mid p \in AP\} \cup \{\complement, \mathbf{EU}_{\to}\}\}$. Then, we note that A is forward complete for $\{\boldsymbol{p} \mid p \in AP\}$ iff $A \sqsubseteq \mathcal{M}(\{\boldsymbol{p} \mid p \in AP\})$. Hence, $\mathrm{ad}^{\mathrm{p}}(P_{\mathrm{dbs}}) = \sqcup_{\mathrm{Abs}(\wp(\Sigma))}\{A \in \mathrm{Abs}(\wp(\Sigma)) \mid A \sqsubseteq \mathcal{M}(\{\boldsymbol{p} \mid p \in AP\}),\ A \text{ is complete for } \{\complement, \mathbf{EU}_{\to}\}\} = \mathscr{S}_{\{\complement, \mathbf{EU}_{\to}\}}(\mathcal{M}(\{\boldsymbol{p} \mid p \in AP\}))$. Finally, since Σ is finite and therefore closure under infinite conjunction boils down to closure under finite conjunction, by Corollary 6.8, $\mathscr{S}_{\{\complement, \mathbf{EU}_{\to}\}}(\mathcal{M}(\{\boldsymbol{p} \mid p \in AP\})) = \mathrm{AD}_{\mathscr{L}_2}$. Thus, by Proposition 5.10 (1), $\mathrm{ad}^{\mathrm{p}}(P_{\mathrm{dbs}}) = \mathrm{AD}_{\mathscr{L}_2}$, so that $P_{\mathrm{dbs}} = \mathrm{par}(\mathrm{ad}^{\mathrm{p}}(P_{\mathrm{dbs}})) = \mathrm{par}(\mathrm{AD}_{\mathscr{L}_2}) = P_{\mathscr{L}_2}$. $\square$

As a consequence of Corollary 6.9, the Groote-Vaandrager algorithm [33] GV for computing dbs equivalence on a finite Kripke structure can be characterized as a complete shell refinement as follows:

$$\mathrm{GV}(P) = \mathrm{par}(\mathscr{S}_{\{\complement, \mathbf{EU}_{\to}\}}(\mathcal{M}(P))).$$

7.3 Simulation Preorder and Equivalence

Simulations are possibly nonsymmetric bisimulations, that is $R \subseteq \Sigma \times \Sigma$ is a simulation on a Kripke structure $\mathcal{K} = (\Sigma, \to, \ell)$ if for any $s, s' \in \Sigma$ such that sRs':

(1) $\ell(s') \subseteq \ell(s)$;

(2) For any $t \in \Sigma$ such that $s \to t$, there exists $t' \in \Sigma$ such that $s' \to t'$ and tRt'.

The empty relation is a simulation and simulation relations are closed under union, so that the largest simulation relation exists. It turns out that the largest simulation is a preorder relation on Σ, i.e. a reflexive and transitive relation on Σ, called similarity preorder (on $\mathcal{K}$). $\mathrm{PreOrd}(\Sigma)$ denotes the set of preorder relations on Σ and the similarity preorder is denoted by $R_{\mathrm{sim}} \in \mathrm{PreOrd}(\Sigma)$. Therefore, a preorder relation $R \in \mathrm{PreOrd}(\Sigma)$ is a simulation on $\mathcal{K}$ when $R \subseteq R_{\mathrm{sim}}$. Simulation equivalence $\sim_{\mathrm{simeq}} \subseteq \Sigma \times \Sigma$ is defined as the symmetric reduction of R_{sim}: $s \sim_{\mathrm{simeq}} s'$ iff there exist two simulation relations R_1 and R_2 such that sR_1s' and $s'R_2s$. $P_{\mathrm{simeq}} \in \mathrm{Part}(\Sigma)$ denotes the partition corresponding to $\sim_{\mathrm{simeq}}$.

A number of algorithms for computing simulation equivalence have been proposed [2, 5, 12, 28, 37] and some of them like [2, 37] first compute the similarity preorder and then from it they obtain simulation equivalence. The problem of computing simulation equivalence is important in model checking because, as recalled in Section 2.3, simulation equivalence strongly preserves ACTL so that $P_{\mathrm{simeq}} = P_{\mathrm{ACTL}}$ (see [34, Section 4]). Recall that ACTL is obtained by restricting CTL, as defined in Section 4.1, to universal quantifiers and by allowing negation on atomic propositions only:

$$\mathrm{ACTL} \ni \varphi ::= p \mid \neg p \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \mathrm{AX}\varphi \mid \mathrm{AU}(\varphi_1, \varphi_2) \mid \mathrm{AR}(\varphi_1, \varphi_2)$$

It turns out that the most abstract s.p. domain for ACTL can be obtained as the most abstract s.p. domain for the following sublanguage $\mathscr{L}_3$:

$$\mathscr{L}_3 \ni \varphi ::= p \mid \neg p \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \mathrm{AX}\varphi$$

Lemma 7.5. *Let $\mathcal{K}$ be finitely branching. Then,* $\mathrm{AD}_{\mathrm{ACTL}} = \mathrm{AD}_{\mathscr{L}_3}$.

Proof. Let $\boldsymbol{Op}_{\mathrm{ACTL}} = \{\cap, \cup, \mathbf{AX}, \mathbf{AU}, \mathbf{AR}\}$ be the set of standard interpretations of the operators of ACTL on $\mathcal{K}$, so that $\mathbf{AX} = \widetilde{\mathrm{pre}}_{\to}$. Analogously to the proof of Lemma 7.1, as a consequence of the least/greatest fixpoint characterizations of $\mathbf{AU}$ and $\mathbf{AR}$, it turns out that for any $A \in \mathrm{Abs}(\wp(\Sigma))$, A is forward complete for $\boldsymbol{Op}_{\mathrm{ACTL}}$ iff A is forward complete for $\{\cup, \widetilde{\mathrm{pre}}_{\to}\}$. Thus, by Lemma 6.4, $\mathscr{S}_{\{\cup, \widetilde{\mathrm{pre}}_{\to}\}} = \mathscr{S}_{\boldsymbol{Op}_{\mathrm{ACTL}}}$, so that, by Corollary 6.8, $\mathrm{AD}_{\mathscr{L}_3} = \mathrm{AD}_{\mathrm{ACTL}}$. □

Thus, by Proposition 5.10 (1), $P_{\mathrm{ACTL}} = \mathrm{par}(\mathrm{AD}_{\mathrm{ACTL}}) = \mathrm{par}(\mathrm{AD}_{\mathscr{L}_3}) = P_{\mathscr{L}_3}$, so that $P_{\mathrm{simeq}} = P_{\mathscr{L}_3}$. As a further consequence, by Corollary 6.9, any algorithm $\mathrm{Alg}_{\mathrm{simeq}}$ that computes simulation equivalence can be viewed as a partitioning abstraction of the $\{\cup, \widetilde{\mathrm{pre}}_{\to}\}$-complete shell refinement:

$$\mathrm{Alg}_{\mathrm{simeq}}(P) = \mathrm{par}(\mathscr{S}_{\{\cup, \widetilde{\mathrm{pre}}_{\to}\}}(\mathcal{M}(P))).$$

7.3.1 Preorders as Abstract Domains

Simulations give rise to preorders rather than equivalences like in the case of bisimulations and dbs relations. Thus, in order to characterize simulation for preorders as forward completeness for abstract domains we need to view preorders as abstract domains. This can be obtained by generalizing the abstraction in Section 3 from partitions to preorders.

Let $R \in \mathrm{PreOrd}(\Sigma)$ and for any $x \in \Sigma$ let us define $R^{\mathrm{pre}} \stackrel{\mathrm{def}}{=} \{\mathrm{pre}_R(\{x\}) \subseteq \Sigma \mid x \in \Sigma\}$. The preorder R gives rise to an abstract domain $\wp(R^{\mathrm{pre}})_\subseteq$ which is related to $\wp(\Sigma)_\subseteq$ through the following abstraction and concretization maps:

$$\alpha_R(S) \stackrel{\mathrm{def}}{=} \{\mathrm{pre}_R(\{x\}) \subseteq \Sigma \mid x \in S\} \qquad \gamma_R(\mathcal{X}) \stackrel{\mathrm{def}}{=} \cup_{X \in \mathcal{X}} X.$$

It is easy to check that from the hypothesis that R is a preorder it follows that $(\alpha_R, \wp(\Sigma)_\subseteq, \wp(R^{\mathrm{pre}})_\subseteq, \gamma_R)$ is indeed a GI. Hence, any $R \in \mathrm{PreOrd}(\Sigma)$ induces an abstract domain denoted by $\mathrm{ad}^{\mathrm{d}}(R) \in \mathrm{Abs}(\wp(\Sigma))$. Also, note that $\gamma_R \circ \alpha_R = \mathrm{pre}_R$, namely pre_R is the closure associated to $\mathrm{ad}^{\mathrm{d}}(R)$. The notation ad^{d} comes from the fact that an abstract domain A is equivalent to some $\mathrm{ad}^{\mathrm{d}}(R)$ if and only if A is disjunctive.

Lemma 7.6. $\{\mathrm{ad}^{\mathrm{d}}(R) \in \mathrm{Abs}(\wp(\Sigma)) \mid R \in \mathrm{PreOrd}(\Sigma)\} = \{A \in \mathrm{Abs}(\wp(\Sigma)) \mid A \text{ is disjunctive}\}$.

Proof. Observe that γ_R is trivially additive, so that any $\mathrm{ad}^{\mathrm{d}}(R)$ is disjunctive. On the other hand, let $A \in \mathrm{Abs}(\wp(\Sigma))$ be disjunctive and consider the relation $R^A = \{(x, y) \mid \alpha(\{x\}) \leq_A \alpha(\{y\})\}$ which is trivially a preorder. Thus, $\mathrm{ad}^{\mathrm{d}}(R^A)$ is disjunctive so that in order to conclude that $\mathrm{ad}^{\mathrm{d}}(R^A)$ is equivalent to A it is enough to observe that for any $y \in \Sigma$, $\mathrm{pre}_{R^A}(\{y\}) = \gamma(\alpha(\{y\}))$: this is true because $\gamma(\alpha(\{y\})) = \{x \in \Sigma \mid \alpha(\{x\}) \leq_A \alpha(\{y\})\} = \mathrm{pre}_{R^A}(\{y\})$. ☐

Let us observe that ad^{d} indeed generalizes ad^{p} from partitions to preorders because for any $P \in \mathrm{Part}(\Sigma)$, $\mathrm{ad}^{\mathrm{p}}(P) = \mathrm{ad}^{\mathrm{d}}(R)$: this is a simple consequence of the fact that for a partition P viewed as an equivalence relation and for $x \in \Sigma$, P_x is exactly a block of P so that $\alpha_P(S) = \{\mathrm{pre}_P(\{x\}) \mid x \in S\}$. On the other hand, an abstract domain $A \in \mathrm{Abs}(\wp(\Sigma))$ induces a preorder relation $\mathrm{preord}(A) \in \mathrm{PreOrd}(\Sigma)$ as follows:

$$(x, y) \in \mathrm{preord}(A) \quad \text{iff} \quad \alpha(\{x\}) \leq_A \alpha(\{y\}).$$

It turns out that the maps ad^{d} and preord allows to view the lattice of preorder relations as an abstraction of the lattice of abstract domains.

Theorem 7.7. $(\mathrm{preord}, \mathrm{Abs}(\wp(\Sigma))_\sqsupseteq, \mathrm{PreOrd}(\Sigma)_\supseteq, \mathrm{ad}^{\mathrm{d}})$ *is a GC.*

Proof. Let $A \in \mathrm{Abs}(\wp(\Sigma))$ and $R \in \mathrm{PreOrd}(\Sigma)$. Let us prove that $R \subseteq \mathrm{preord}(A) \;\Leftrightarrow\; \mathrm{ad}^{\mathrm{d}}(R) \sqsubseteq \gamma \circ \alpha$.
($\Rightarrow$) Let $S \subseteq \Sigma$ and let us show that $\mathrm{ad}^{\mathrm{d}}(R)(S) = \mathrm{pre}_R(S) \subseteq \gamma(\alpha(S))$. If $x \in \mathrm{pre}_R(S)$ then xRy for some $y \in S$, so that $(x, y) \in \mathrm{preord}(A)$, i.e. $\alpha(\{x\}) \leq_A \alpha(\{y\})$. Thus, by applying γ, $x \in \gamma(\alpha(\{x\})) \subseteq \gamma(\alpha(\{y\})) \subseteq \gamma(\alpha(S))$.
($\Leftarrow$) Let $(x, y) \in R$ and let us show that $\alpha(\{x\}) \leq_A \alpha(\{y\})$. Note that $x \in \mathrm{pre}_R(\{y\}) = \mathrm{ad}^{\mathrm{d}}(R)(\{y\}) \subseteq \gamma(\alpha(\{y\}))$, so that $\alpha(\{x\}) \leq_A \alpha(\{y\})$, namely $(x, y) \in \mathrm{preord}(A)$. ☐

Let us remark that $\mathbb{D} \stackrel{\mathrm{def}}{=} \mathrm{ad}^{\mathrm{d}} \circ \mathrm{preord}$ is a lower closure operator on $\langle \mathrm{Abs}(\wp(\Sigma)), \sqsubseteq \rangle$ and that, by Lemma 7.6, for any $A \in \mathrm{Abs}(\wp(\Sigma))$, A is disjunctive iff $\mathbb{D}(A) = A$. Hence, $\mathbb{D}$ coincides with the disjunctive-shell refinement, also known as disjunctive completion [15, 31], namely $\mathbb{D}(A)$ is the most abstract disjunctive refinement of A.

We can now provide a characterization of simulation preorders in terms of forward completeness.

Theorem 7.8. *Let* $R \in \mathrm{PreOrd}(\Sigma)$. *Then, R is a simulation on $\mathcal{K}$ iff* $\mathrm{ad}^{\mathrm{d}}(R)$ *is forward complete for* $\{\boldsymbol{p} \mid p \in AP\} \cup \{\widetilde{\mathrm{pre}}_{\rightarrow}\}$.

Proof. Recall that pre_R is the closure associated to $\mathrm{ad}^{\mathrm{d}}(R)$. We first observe that $(sRs' \;\Rightarrow\; \ell(s') \subseteq \ell(s))$ iff pre_R is forward complete for $\boldsymbol{AP}$. On the one hand, if $\boldsymbol{p} \in \boldsymbol{AP}$ and $s \in \mathrm{pre}_R(\boldsymbol{p})$ then sRs' for some $s' \in \boldsymbol{p}$, so that, from $\ell(s') \subseteq \ell(s)$, we obtain $s \in \boldsymbol{p}$, and therefore $\mathrm{pre}_R(\boldsymbol{p}) = \boldsymbol{p}$. On the other hand, if sRs' and $s' \in \boldsymbol{p}$, for some $\boldsymbol{p} \in \boldsymbol{AP}$, then $s' \in \boldsymbol{p} = \mathrm{pre}_R(\boldsymbol{p})$ so that $\mathrm{pre}_R(\{s'\}) \subseteq \mathrm{pre}_R(\mathrm{pre}_R(\boldsymbol{p})) = \mathrm{pre}_R(\boldsymbol{p}) = \boldsymbol{p}$ and therefore from $s \in \mathrm{pre}_R(\{s'\})$ we obtain $s \in \boldsymbol{p}$.
Thus, it remains to show that R satisfies condition (2) of the definition of simulation iff pre_R is forward complete for $\mathrm{pre}_{\rightarrow}$.
($\Rightarrow$) We prove that for any S, $\mathrm{pre}_R(\widetilde{\mathrm{pre}}_{\rightarrow}(\mathrm{pre}_R(S))) \subseteq \widetilde{\mathrm{pre}}_{\rightarrow}(\mathrm{pre}_R(S))$. Let $x \in \mathrm{pre}_R(\widetilde{\mathrm{pre}}_{\rightarrow}(\mathrm{pre}_R(S)))$ so that there exists some $y \in \widetilde{\mathrm{pre}}_{\rightarrow}(\mathrm{pre}_R(S))$ such that xRy. If $x \rightarrow x'$, for some x', then, by simulation, there exists some y' such that $y \rightarrow y'$ and $x'Ry'$. Hence, $y' \in \mathrm{pre}_R(S)$ and this together with $x'Ry'$, as R is transitive, gives $x' \in \mathrm{pre}_R(S)$. Therefore, $x \in \widetilde{\mathrm{pre}}_{\rightarrow}(\mathrm{pre}_R(S))$.

($\Leftarrow$) Observe that in order to show that R is a simulation it is enough to show that if xRy then $x \in \widetilde{\mathrm{pre}}_{\to}(\mathrm{pre}_R(\mathrm{post}_{\to}(\{y\})))$. The following implications hold, where $\mathrm{post}_{\to}(\{y\}) \subseteq \mathrm{pre}_R(\mathrm{post}_{\to}(\{y\}))$ holds because pre_R is a uco:

$$\begin{array}{rcr}
\mathrm{post}_{\to}(\{y\}) \subseteq \mathrm{pre}_R(\mathrm{post}_{\to}(\{y\})) & \Rightarrow & [\text{as } \widetilde{\mathrm{pre}}_{\to} \text{ is monotone}] \\
\widetilde{\mathrm{pre}}_{\to}(\mathrm{post}_{\to}(\{y\})) \subseteq \widetilde{\mathrm{pre}}_{\to}(\mathrm{pre}_R(\mathrm{post}_{\to}(\{y\}))) & \Rightarrow & [\text{as } y \in \widetilde{\mathrm{pre}}_{\to}(\mathrm{post}_{\to}(\{y\}))] \\
\{y\} \subseteq \widetilde{\mathrm{pre}}_{\to}(\mathrm{pre}_R(\mathrm{post}_{\to}(\{y\}))) & \Rightarrow & [\text{as } \mathrm{pre}_R \text{ is monotone}] \\
\mathrm{pre}_R(\{y\}) \subseteq \mathrm{pre}_R(\widetilde{\mathrm{pre}}_{\to}(\mathrm{pre}_R(\mathrm{post}_{\to}(\{y\})))) & \Rightarrow & [\text{as } \mathrm{pre}_R \text{ is forward complete for } \widetilde{\mathrm{pre}}_{\to}] \\
\mathrm{pre}_R(\{y\}) \subseteq \widetilde{\mathrm{pre}}_{\to}(\mathrm{pre}_R(\mathrm{post}_{\to}(\{y\}))) & \Rightarrow & [\text{as } x \in \mathrm{pre}_R(\{y\})] \\
x \in \widetilde{\mathrm{pre}}_{\to}(\mathrm{pre}_R(\mathrm{post}_{\to}(\{y\}))) & &
\end{array}$$

and this closes the proof. □

8 Related work

Loiseaux et al. [41] generalized the standard approach to abstract model checking to more general abstract models where an abstraction relation $\sigma \subseteq States \times A$ is used instead of a surjective function $h : States \to A$. However, the results of strong preservation given there (cf. [41, Theorems 3 and 4]) require the hypothesis that the relation σ is difunctional, i.e. $\sigma = \sigma\sigma^{-1}\sigma$. In this case the abstraction relation σ can indeed be derived from a function, so that the class of strongly preserving abstract models in Loiseaux et al.'s framework is not really larger than the class of standard partition-based abstract models (see the detailed discussion by Dams et al. [21, Section 8.1]).

Giacobazzi and Quintarelli [29] first noted that strong preservation is related to completeness in abstract interpretation by studying the relationship between complete abstract interpretations and Clarke et al.'s [6, 7, 8] spurious counterexamples. Given a formula φ of ACTL, a model checker running on a standard abstract Kripke structure defined over a state partition P may provide a spurious counterexample $\pi^\sharp$ for φ, namely a path of abstract states, namely blocks of P, which does not correspond to a real concrete counterexample. In this case, by exploiting the spurious counterexample $\pi^\sharp$, the partition P is refined to P' by splitting a single block of P. As a result, this refined partition P' does not admit the spurious counterexample $\pi^\sharp$ for φ so that P' is given as a new refined abstract model for φ to the model checker. Giacobazzi and Quintarelli [29] cast spurious counterexamples for a partition P as a lack of (standard) completeness in the abstract interpretation sense for the corresponding partitioning abstract domain $\mathrm{ad}^{\mathrm{p}}(P)$. Then, by applying the results in [32] they put forward a method for systematically refining abstract domains in order to eliminate spurious counterexamples. The relationship between completeness and spurious counterexamples was further studied in [19], where it is also shown that a block splitting operation in Paige and Tarjan [44] partition refinement algorithm can be characterized in terms of complete abstract interpretations. More in general, the idea of systematically enhancing the precision of abstract interpretations by refining the underlying abstract domains dates back to the early works by Cousot and Cousot [15], and evolved to the systematic design of abstract interpretations by abstract domain refinements [27, 30, 32].

9 Conclusion

This work shows how the abstract interpretation technique allows to generalize the notion of strong preservation from standard abstract models specified as abstract Kripke structures to generic domains in abstract interpretation. For any inductively defined language $\mathscr{L}$, it turns out that strong preservation of $\mathscr{L}$ in a standard abstract model checking framework based on partitions of the space state Σ becomes a particular instance of the property of forward completeness of abstract domains w.r.t. the semantic operators of the language $\mathscr{L}$. In particular, a generalized abstract model can always be refined through a fixpoint iteration to the most abstract domain that strongly preserves $\mathscr{L}$. This generalizes in our framework the idea of partition refinement algorithms that reduce the state space Σ in order to obtain a minimal abstract Kripke structure that is strongly preserving for some temporal language.

This work deals with generic temporal languages consisting of state formulae only. As future work, it would be interesting to study whether the ideas of our abstract interpretation-based approach can be applied to linear languages like LTL consisting of formulae that are interpreted as sets of paths of a Kripke structure. The idea here is to investigate whether standard strong preservation of LTL can be generalized to abstract interpretations of the powerset of traces and to the corresponding completeness properties. Fairness can be also an interesting topic of investigation, namely to study whether our abstract interpretation-based framework allows to handle fair semantics and fairness constraints [10].

Finally, let us mention that the results presented in this paper led to design a generalized Paige-Tarjan refinement algorithm based on abstract interpretation for computing most abstract strongly preserving domains [47]. As shown in Section 6, a most abstract strongly preserving domain can be characterized as a greatest fixpoint computation in $\mathrm{Abs}(\wp(\Sigma))$. It is shown in [47] that the Paige-Tarjan algorithm [44] can be viewed exactly as a corresponding abstract greatest fixpoint computation in $\mathrm{Part}(\Sigma)$. This leads to an abstract interpretation-based Paige-Tarjan-like refinement algorithm that is parameteric on any abstract interpretation of the lattice $\mathrm{Abs}(\wp(\Sigma))$ of abstract domains of $\wp(\Sigma)$ and on any generic inductive language $\mathscr{L}$.

Acknowledgments. We wish to thank Mila Dalla Preda and Roberto Giacobazzi who contributed to the early stage of this work. This paper is an extended and revised version of [46]. This work was partially supported by the FIRB Project "Abstract interpretation and model checking for the verification of embedded systems" and by the COFIN2004 Project "AIDA: Abstract Interpretation Design and Applications".

References

[1] K.R. Apt and G.D. Plotkin. Countable nondeterminism and random assignment. *J. ACM*, 33(4):724–767, 1986.

[2] B. Bloom and R. Paige. Transformational design and implementation of a new efficient solution to the ready simulation problem. *Sci. Comp. Program.*, 24(3):189–220, 1995.

[3] A. Bouajjani, J.-C. Fernandez and N. Halbwachs. Minimal model generation. In *Proc. of the 2nd Internat. Conf. on Computer Aided Verification (CAV'90)*, LNCS 531, pp. 197–203, Springer, 1990.

[4] M.C. Browne, E.M. Clarke and O. Grumberg. Characterizing finite Kripke structures in propositional temporal logic. *Theoret. Comp. Sci.*, 59:115–131, 1988.

[5] D. Bustan and O. Grumberg. Simulation-based minimization. *ACM Trans. Comput. Log.*, 4(2):181–204, 2003.

[6] E.M. Clarke, O. Grumberg, S. Jha, Y. Lu and H. Veith. Counterexample-guided abstraction refinement. In *Proc. of the 12th Internat. Conf. on Computer Aided Verification (CAV'00)*, LNCS 1855, pp. 154–169, Springer, 2000.

[7] E.M. Clarke, O. Grumberg, S. Jha, Y. Lu and H. Veith. Counterexample-guided abstraction refinement for symbolic model checking. *J. ACM*, 50(5):752–794, 2003.

[8] E.M. Clarke, S. Jha, Y. Lu and H. Veith. Tree-like counterexamples in model checking. In *Proc. of the 17th IEEE Symp. on Logic in Computer Science (LICS'02)*, pp. 19–29, IEEE Press, 2002.

[9] E.M. Clarke, O. Grumberg and D. Long. Model checking and abstraction. *ACM Trans. Program. Lang. Syst.*, 16(5):1512–1542, 1994.

[10] E.M. Clarke, O. Grumberg and D.A. Peled. *Model checking*. The MIT Press, 1999.

[11] R. Cleaveland, S.P. Iyer, D. Yankelevich. Optimality in abstractions of model checking. In *Proc. 2nd Intern. Static Analysis Symposium (SAS'95)*, LNCS 983, pp. 51–63, Springer, 1995.

[12] R. Cleaveland, J. Parrow and B. Steffen. The Concurrency Workbench: a semantics based tool for the verification of concurrent systems. *ACM Trans. Program. Lang. Syst.*, 15(1):36–72, 1993.

[13] P. Cousot. Abstract interpretation based formal methods and future challenges. In *Informatics, 10 Years Back, 10 Years Ahead*, LNCS 2000, pp. 138–156, Springer, 2001.

[14] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proc. 4th ACM POPL*, pp. 238–252, 1977.

[15] P. Cousot and R. Cousot. Systematic design of program analysis frameworks. In *Proc. 6th ACM POPL*, pp. 269–282, 1979.

[16] P. Cousot and R. Cousot. Higher-order abstract interpretation (and application to comportment analysis generalizing strictness, termination, projection and PER analysis of functional languages). In *Proc. IEEE Int. Conf. on Computer Languages (ICCL'94)*, pp. 95–112, 1994.

[17] P. Cousot and R. Cousot. Refining model checking by abstract interpretation. *Automated Software Engineering Journal*, 6(1):69–95, 1999.

[18] P. Cousot and R. Cousot. Temporal abstract interpretation. In *Proc. 27th ACM POPL*, pp. 12–25, 2000.

[19] M. Dalla Preda. *Completeness and stability in abstract model checking*. Laurea Thesis (in Italian), Univ. of Verona, Italy, 2003.

[20] D. Dams. *Abstract interpretation and partition refinement for model checking*. Ph.D. Thesis, Eindhoven University of Technology, The Netherlands, 1996.

[21] D. Dams, O. Grumberg and R. Gerth. Abstract interpretation of reactive systems. *ACM Trans. Program. Lang. Syst.*, 16(5):1512–1542, 1997.

[22] J.W. De Bakker, J.-J.C. Meyer and J.I. Zucker. On infinite computations in denotational semantics. *Theoret. Comp. Sci.*, 26(1-2):53–82, 1983.

[23] R. De Nicola and F. Vaandrager. Three logics for branching bisimulation. *J. ACM*, 42(2):458–487, 1995

[24] A. Dovier, C. Piazza and A. Policriti. An efficient algorithm for computing bisimulation equivalence. *Theoret. Comp. Sci.*, 311(1-3):221–256, 2004.

[25] E.A. Emerson, A.K. Mok, A.P. Sistla and J. Srinivasen. Quantitative temporal reasoning. In *Proc. of the 2nd Internat. Conf. on Computer Aided Verification (CAV'90)*, LNCS 531, pp. 136–145, Springer, 1990.

[26] E.A. Emerson and E.M. Clarke. Characterizing correctness properties of parallel programs using fixpoints. In *Proc. ICALP'80*, LNCS 85, pp. 169–181, Springer, 1980.

[27] G. Filé, R. Giacobazzi and F. Ranzato. A unifying view of abstract domain design. *ACM Comput. Surv.*, 28(2):333–336, 1996.

[28] R. Gentilini, C. Piazza and A. Policriti. From bisimulation to simulation: coarsest partition problems. *J. Automated Reasoning*, 31(1):73-103, 2003.

[29] R. Giacobazzi and E. Quintarelli. Incompleteness, counterexamples and refinements in abstract model checking. In *Proc. 8th Intern. Static Analysis Symposium (SAS'01)*, LNCS 2126, pp. 356–373, Springer, 2001.

[30] R. Giacobazzi and F. Ranzato. Refining and compressing abstract domains. In *Proc. 24th ICALP*, LNCS 1256, pp. 771–781, Springer, 1997.

[31] R. Giacobazzi and F. Ranzato. Optimal domains for disjunctive abstract interpretation. *Sci. Comp. Program.*, 32:177–210, 1998.

[32] R. Giacobazzi, F. Ranzato and F. Scozzari. Making abstract interpretations complete. *J. ACM*, 47(2):361–416, 2000.

[33] J.F. Groote and F. Vaandrager. An efficient algorithm for branching bisimulation and stuttering equivalence. In *Proc. ICALP'90*, LNCS 443, pp. 626-638, Springer, 1990.

[34] O. Grumberg and D.E. Long. Model checking and modular verification. *ACM Trans. Program. Lang. Syst.*, 16(3):843–871, 1994.

[35] B.S. Gulavani and S.K. Rajamani. Counterexample driven refinement for abstract interpretation. In *Proc. 12th Intern. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'06)*, LNCS 3920, pp. 474–488, Springer, 2006.

[36] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *J. ACM*, 32(1):137–161, 1985.

[37] M.R. Henzinger, T.A. Henzinger and P.W. Kopke. Computing simulations on finite and infinite graphs. In *Proc. 36th FOCS*, pp. 453–462, IEEE Press, 1995.

[38] T.A. Henzinger, R. Maujumdar and J.-F. Raskin. A classification of symbolic transition systems. *ACM Trans. Comput. Log.*, 6(1):1–31, 2005.

[39] L. Lamport. What good is temporal logic? In *Information Processing '83*, pp. 657–668, IFIP North-Holland, 1983.

[40] D. Lee and M. Yannakakis. Online minimization of transition systems. In *Proc. 24th ACM STOC*, pp. 264–274, 1992.

[41] C. Loiseaux, S. Graf, J. Sifakis, A. Bouajjani and S. Bensalem. Property preserving abstractions for the verification of concurrent systems. *Formal Methods in System Design*, 6:1–36, 1995.

[42] D. Massé. Semantics for abstract interpretation-based static analyzes of temporal properties. In *Proc. 9th Intern. Static Analysis Symposium (SAS'02)*, LNCS 2477, pp. 428–443, Springer, 2002.

[43] D. Massé. Abstract domains for property checking driven analysis of temporal properties. In *Proc. 10th Intern. Conf. on Algebraic Methodology and Software Technology (AMAST'04)*, LNCS 3116, pp. 349–363, Springer, 2004.

[44] R. Paige and R.E. Tarjan. Three partition refinement algorithms. *SIAM J. Comput.*, 16(6):973–989, 1987

[45] F. Ranzato and F. Tapparo. Making abstract model checking strongly preserving. In *Proc. 9th Intern. Static Analysis Symposium (SAS'02)*, LNCS 2477, pp. 411–427, Springer, 2002.

[46] F. Ranzato and F. Tapparo. Strong preservation as completeness in abstract interpretation. In *Proc. 13th European Symposium on Programming (ESOP'04)*, LNCS. 2986, pp. 18–32, Springer, 2004.

[47] F. Ranzato and F. Tapparo. An abstract interpretation-based refinement algorithm for strong preservation. In *Proc. 11th Intern. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'05)*, LNCS 3440, pp. 140–156, Springer, 2005.

[48] D.A. Schmidt. Closed and logical relations for over- and under-approximation of powersets. In *Proc. 11th Intern. Static Analysis Symposium (SAS'04)*, LNCS 3148, pp. 22–37, Springer, 2004.

[49] D.A. Schmidt. Underapproximating predicate transformers. In *Proc. 13th Intern. Static Analysis Symposium (SAS'06)*, LNCS, Springer, 2006.

[50] L. Tan and R. Cleaveland. Simulation revisited. In In *Proc. 7th Intern. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'01)*, LNCS 2031, pp. 480-495, Springer, 2001.

[51] R.J. van Glabbeek. The linear time - branching time spectrum. In *Handbook of Process Algebra*, pp. 3–99, Elsevier, 2001.