A Complete Abstract Interpretation Framework for Coverability Properties of WSTS

Pierre Ganty[1], Jean-François Raskin[1], and Laurent Van Begin[2]

[1] Département d'Informatique, Université Libre de Bruxelles
[2] LIAFA, Université Paris 7

Abstract. We present an abstract interpretation based approach to solve the coverability problem of well-structured transition systems. Our approach distinguishes from other attempts in that (1) we solve this problem for the whole class of well-structured transition systems using a forward algorithm. So, our algorithm has to deal with possibly infinite downward closed sets. (2) Whereas other approaches have a non generic representation for downward closed sets of states, which turns out to be hard to devise in practice, we introduce a generic representation requiring no additional effort of implementation.

1 Introduction

Model-checking is nowadays widely accepted as a powerful technique for the automatic verification of reactive systems that have natural finite state abstractions. However, many reactive systems are only naturally modeled as infinite-state systems. This is why a large research effort was done in the recent years to allow the direct application of model-checking techniques to infinite-state models. This research line has shown successes for several interesting classes of infinite-state systems, for example: timed automata [1], hybrid automata [2], FIFO channel systems [3–5], extended Petri nets [6, 7], broadcast protocols [8], etc.

General decidability results hold for a large class of infinite-state systems called the *well-structured transition systems*, WSTS for short. WSTS are transition systems whose sets of states are well-quasi ordered and whose transition relations enjoy a monotonicity property with respect to the well-quasi order. Examples of WSTS are Petri nets [9], monotonic extensions of Petri nets (Petri nets with transfer arcs [10], Petri nets with reset arcs [11], and Petri nets with non-blocking arcs [12]), broadcast protocols [13], lossy channel systems [4]. For all those classes of infinite-state systems, we know that an interesting and large class of *safety properties* are decidable by reduction to the *coverability problem*. The coverability problem is defined as follows: "given a WSTS for the well-quasi order $\geq$, and two states c_1 and c_2, does there exist a state c_3 which is reachable from c_1 and such that $c_3 \geq c_2$?" (in that context, we say that c_3 covers c_2)

Broadly speaking, there are two ways to solve the coverability problem for WSTS. The first way is to explore *backwardly* the transition system by iterating the *pre* operator[3] starting from the set of states that are greater or equal to c_2. This simple procedure is effective when very mild assumptions are met. In fact, for any well-quasi ordered set $(X, \geq)$, the following nice property holds: every $\geq$-upward closed set can be *finitely* represented using its finite set of minimal elements[4]. This generic representation of $\geq$-upward closed set is adequate as union and inclusion are effective. The only further property that is needed for the procedure to be effective is that given a finite set of minimal elements M defining an $\geq$-upward closed set U, it must be possible to compute the finite set of minimal elements M' representing $pre(U)$. Higman's lemma on well-quasi orders ensure the termination of this procedure.

The second way is to explore forwardly the transition system from the initial state c_1. Here, the situation is more complicated. A saturation method that iterates the *post* operator[5] from c_0 can not lead to an algorithm as the reachability problem is undecidable for WSTS. Recently, we have shown that the coverability problem can be decided in a forward way by constructing two sequences of abstractions of the reachable states of the system, one from below and one from above [14]. The sequence of abstractions from below allows us to detect positive instances of the coverability problem and it is simply the bounded iteration of *post* from the initial state. The abstraction from above is the iteration of an overapproximation of *post* over downward closed set of states that becomes more and more precise. This sequence allows us to decide negative instances of the problem. This schema of algorithm is general but to be applicable to a given class of WSTS, the user has to provide a, so called, *adequate set of limits*. This set is in fact a (usually infinite) set of abstract values that allows to represent any downward closed set for $\geq$. The situation is less satisfactory than for upward-closed set where there exists, as we have seen above, a simple and generic way to represent upward-closed set by sets of minimal elements. Such a generic way of representing downward closed sets was missing and this problem is solved here.

The contributions of this paper are as follows. First, we show that for any well-quasi ordered set, there exists a generic and effective representation of downward closed sets. To the best of our knowledge, this is the first time that such a generic representation is proposed. An attempt in that direction was taken in [15, 16] but the result is a theory for designing symbolic representation of downward closed sets and not a generic symbolic representation of such sets. As a consequence, their theory has to be instantiated for the particular class of WSTS that is targeted and this is not a trivial task. Second, as downward closed sets are abstractions for sets of reachable sets of states as in the forward algorithm, we formalize our generic representation of downward closed set as a generic abstract domain. This allow us to rephrase in a simpler way our forward algorithm in the context of abstract interpretation. Third, we show how to automatically refine

[3] a function that returns all the states that have a one-step successor in a given set of states.

[4] or a finite set of its minimal elements if $\geq$ is not a partial order.

[5] a function that returns all the one-step successors states of a given set of states.

the abstract domain in order to obtain, in an efficient way, overapproximations that are guaranteed to be sufficiently precise to decide the coverability problem.

Our paper is organized as follows. Section 2 presents some preliminaries. Section 3 introduce the generic representation of downward closed sets. In Section 4 we will be concerned with the abstract interpretation of WSTS. Section 5 is devoted the refinement of the abstract domain. Section 6 shows on an example how these techniques work. Furthermore, missing proofs can be found in the appendix.

2 Preliminaries

2.1 Well-Quasi Ordered Sets

A *preorder* $\succeq$ is a binary relation over a set X which is reflexive, and transitive. The preorder $\succeq$ is a *well-quasi order* (wqo for short) if there is no infinite sequence $x_0, x_1, \ldots$, such that $x_i \not\succeq x_j$ for all $i > j \geq 0$. A set $M \subseteq X$ is said to be *canonical* if for any distinct $x, y \in M$ we have $x \not\succeq y$. We say that $M \subseteq S$ is a *minor set* of $S \subseteq X$, if for all $x \in S$ there exists $y \in M$ such that $x \succeq y$, and M is canonical.

Lemma 1 (From [17]). *Let $(X, \succeq)$ be a well-quasi ordered set (wqo-set for short). For any set $S \subseteq X$, S has at least one finite minor set M.*

We use min to denote a function which, given a set $S \subseteq X$, returns a minor set of S. Let $(X, \succeq)$ be a wqo-set, we call $x{\downarrow} = \{x' \in X \mid x \succeq x'\}$ and $x{\uparrow} = \{x' \in X \mid x' \succeq x\}$ the $\succeq$*-downward closure* and $\succeq$*-upward closure* of $x \in X$, respectively. This definition is naturally extended to sets in X. We define a set $S \subseteq X$ to be a $\succeq$-downward-closed set ($\succeq$*-dc-set* for short), respectively $\succeq$-upward-closed set ($\succeq$*-uc-set* for short), iff $S{\downarrow} = S$, respectively $S{\uparrow} = S$. Examples of such sets are given in Fig. 1. For any wqo-set $(X, \succeq)$, we define $DCS(X)$ ($UCS(X)$) to be the set of all $\succeq$-dc-sets ($\succeq$-uc-sets) in X. For any $x \in X$ we define the $\succeq$-equivalence class of x, denoted $[x]$, to be the set $x{\uparrow} \cap x{\downarrow}$, i.e. the set of elements that are $\succeq$-equivalent to x. For A and B subsets of X, we say that $A \equiv B$ if $A{\uparrow} = B{\uparrow}$. Observe that $A \equiv B$ iff for all $a \in A$ there is a $b \in B$ such that $a \succeq b$, and vice versa. We now recall a well-known lemma on $\succeq$-uc-sets and $\succeq$-dc-sets.

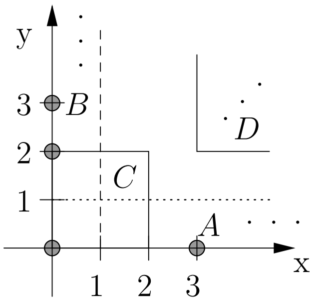
Lemma 2 (From [17]). *Let $(X, \succeq)$ a wqo-set and an infinite sequence of $\succeq$-uc-set $U_0 U_1 \ldots$ such that $\forall i \geq 0\colon U_i \subseteq U_{i+1}$. There exists $j \geq 0\colon \forall j' \geq j\colon U_j = U_{j'}$. Symmetrically, given an infinite sequence of $\succeq$-dc-sets $D_0 D_1 \ldots$ such that $\forall i \geq 0\colon D_i \supseteq D_{i+1}$, there exists $j \geq 0\colon \forall j' \geq j\colon D_j = D_{j'}$.*

We now introduce a lemma stating several facts about sets and their closure. These facts are merely of technical interest and will be used subsequently.

Lemma 3.

1. *For any $S, S' \subseteq X$, $S{\downarrow} \cap S'{\uparrow} \neq \emptyset \Leftrightarrow S{\downarrow} \cap S' \neq \emptyset \Leftrightarrow S \cap S'{\uparrow} \neq \emptyset$.*
2. *For any $S, S' \subseteq X$, $S{\uparrow} \subseteq S'{\uparrow} \Leftrightarrow \forall s \in S\ \exists s' \in S'\colon s \succeq s'$.*
3. *$\forall s \in X, S \in UCS(X)\colon s \in S \Leftrightarrow \exists s' \in \mathsf{min}(S)\colon s \succeq s'$.*

Fig. 1. $\succeq$-dc-sets and $\succeq$-uc-sets in $\mathbb{N}^2$.



The wqo $\succeq$ is defined as follows $(a_1, a_2) \succeq (b_1, b_2)$ if and only if $a_1 \geq b_1$ and $a_2 \geq b_2$. The $\succeq$-dc-sets A and B are infinite $\succeq$-dc-set : $A = \{(x, y) \in \mathbb{N}^2 \mid y \leq 1\}$, $B = \{(x, y) \in \mathbb{N}^2 \mid x \leq 1\}$. On the contrary, the $\succeq$-dc-set $C = \{(x, y) \in \mathbb{N}^2 \mid x \leq 2 \wedge y \leq 2\}$ is finite. The $\succeq$-uc-set D is given by $\{(x, y) \in \mathbb{N}^2 \mid x \geq 3 \wedge y \geq 2\}$. Note that D has exactly one minor set since $\succeq$ is a partial order.

Lemma 3.2 and 3.3 suggest an effective representation of $\succeq$-uc-sets: every $\succeq$-uc-set U can be finitely represented by $\mathsf{min}(U)$. For decidable well-quasi order $\succeq$, this readily gives us an effective procedure to check inclusion between two $\succeq$-uc-sets, to check membership and to compute union [18]. Sometimes we write s instead of the set $\{s\}$. Unless otherwise stated the transitive and reflexive closure f^* of a function f such that its domain and co-domain coincide is given by $\bigcup_{i \geq 0} f^i$ where f^0 is the identity and $f^{i+1} = f^i \circ f$.

2.2 Well-Structured Transitions Systems

In this paper we follow [19] in the definition of well-structured transition systems.

Definition 1. *A* well-structured transition system *(*WSTS*)* $\mathcal{S}$ *is a tuple* $(X, \delta, \succeq)$ *where* X *is a (possibly) infinite set of states,* $\delta \subseteq X \times X$ *is a transition relation between states — we use the notation* $x \to x'$ *if* $(x, x') \in \delta$ *—, and* $\succeq \subseteq X \times X$ *is a preorder between states such that the two following conditions hold:* (i) $\succeq$ *is a wqo; and* (ii) $\forall x_1, x_2, x_3\ \exists x_4\colon (x_3 \succeq x_1 \wedge x_1 \to x_2) \Rightarrow (x_3 \to^* x_4 \wedge x_4 \succeq x_2)$*, where* $\to^*$ *is the reflexive and transitive closure of the transition relation (upward compatibility)*[6]*. Moreover, we define an* initialized WSTS *(* IWSTS*) to be a pair* $(\mathcal{S}, x_0)$ *where* $\mathcal{S} = (X, \delta, \succeq)$ *is a* WSTS *and* $x_0 \in X$ *is the* initial state*. We adhere to the convention that if* $\mathcal{S}_0$ *is an* IWSTS *then* $\mathcal{S}$ *is its* WSTS*.*

Let $\mathcal{S} = (X, \delta, \succeq)$ be a WSTS and $T \subseteq X$, $post[\mathcal{S}](T) \stackrel{\text{def}}{=} \{x' \mid \exists x \in T\colon x \to x'\}$. Analogously, we define $pre[\mathcal{S}](T)$ as $\{x \mid \exists x' \in T\colon x \to x'\}$. We define $minpre[\mathcal{S}](T) \stackrel{\text{def}}{=} \mathsf{min}((pre[\mathcal{S}](T\!\uparrow))\!\uparrow)$. To shorten notation, we write *pre*,*minpre* and *post* if the WSTS is clear from the context. The following definition follows [17].

Definition 2. *An* effective WSTS *is a* WSTS $\mathcal{S} = (X, \delta, \succeq)$ *where both* $\succeq$ *and* $\to$ *are decidable and for all* $x \in X\colon minpre[\mathcal{S}](x)$ *is computable.*

2.3 The Coverability Problem

The verification of safety properties on IWSTS reduces to the so called coverability problem.

[6] Upward compatibility is more general than the compatibility used in [17]

Problem 1. The *coverability problem* for IWSTS is defined as follows: "*Given an* IWSTS $((X, \delta, \succeq), x_0)$ *and* bad $\in UCS(X)$, $post^*(x_0) \cap \mathsf{bad} = \emptyset$*?*"

In general, bad is an upward-closed set of states where errors occur.

Two solutions to the coverability problem can be found in the literature. The first one (see [17, 19]) is a backward approach based on the following two lemmas:

Lemma 4 (From [19]). *Given a* WSTS $\mathcal{S} = (X, \delta, \succeq)$ *and* $U \in UCS(X)$, *(a)* $pre^*(U) \in UCS(X)$, *and (b)* $minpre^*(\mathsf{min}(U))\!\uparrow = pre^*(U)$.

Lemma 4, together with Lemma 1 and 2, show how to (symbolically) compute the (possibly) infinite set $pre^*(U)$ using the minor sets of $\succeq$-uc-sets. Once $pre^*(U)$ is computed, or rather a finite representation using one of its minor set, one can decide the coverability problem by testing if the initial state is in $pre^*(U)$ by using Lemma 3.3.

The second approach is a forward approach based on the notion of *covering set* [20, 13, 4]. The covering set $Cover(\mathcal{S}_0)$ of an IWSTS $\mathcal{S}_0 = (\mathcal{S}, x_0)$ is given by $Cover(\mathcal{S}_0) \stackrel{\text{def}}{=} post^*(x_0)\!\downarrow$. The following lemma shows the usefulness of covering sets to solve the coverability problem:

Lemma 5. *Given an* IWSTS $\mathcal{S}_0 = ((X, \delta, \succeq), x_0)$ *and* bad $\in UCS(X)$, $Cover(\mathcal{S}_0) \cap \mathsf{bad} = \emptyset$ *if and only if* $post^*(x_0) \cap \mathsf{bad} = \emptyset$.

As already mentioned in the introduction, there are two difficulties to overcome when trying to design a forward algorithm for the coverability problem:

1. Currently, there are no generic way to effectively represent $\succeq$-dc-sets (as the one shown above for $\succeq$-uc-sets). So, for every wqo-set $(X, \succeq)$ one has to design a symbolic representation for the sets in $DCS(X)$.
2. The set $Cover(\mathcal{S}_0)$ is in general not effectively constructable, see [11] for details. As a consequence, all the algorithms based on its construction (except the well-known Karp-Miller algorithm on Petri nets) may fail to terminate.

To overcome those two difficulties:

1. In [16], the authors propose a methodology to design a symbolic representation of dc-sets. However the design of such a symbolic data-structure is far from being trivial.
2. The authors of this paper proposed, in [21], an algorithmic schema called *expand, enlarge and check* which can be instantiated for any class of WSTS as long as a symbolic representation of dc-sets is provided (called there an adequate set of limits).

In this paper, we provide, in our opinion, a much more satisfactory answer to those two difficulties by providing, in the form of a generic abstract domain and a generic abstract analysis, a completely generic algorithm to solve the coverability problem for WSTS.

3 A Generic Abstract Domain

In this section, we present a parametrized abstract domain that allows us to represent any $\succeq$-dc-set in a wqo-set $(X, \succeq)$. The parameter D is a finite subset of X and it defines the precision of the abstract domain. We also show that this parametrized abstract domain enjoys the following properties: (i) our parametrized abstract domain defines a complete lattice, (ii) we define an abstraction and a concretisation function that is shown to be a Galois insertion, (iii) any $\succeq$-dc-set can be exactly represented by our parametrized abstract domain provided an adequate value for the parameter D is used, and (iv) each $\succeq$-dc-set has a finite representation.

Recall that the powerset lattice $PL(A)$ associated to a set A is the complete lattice having the powerset of A as carrier, and union and intersection as least upper bound and greatest lower bound, respectively. In our setting the concrete lattice is the powerset lattice $PL(X)$ of the set of states X.

Fix a finite set $D \subseteq X$ which is called the *domain*, the abstract lattice $DPL(D)$ has $DCS(D)$ as a carrier, $\sqcup_D$ as the *least upper bound* operator, $\sqcap_D$ as the *greatest lower bound* operator, and D and $\emptyset$ are the $\sqsubseteq_D$-maximal and $\sqsubseteq_D$-minimal element, respectively. We define the relation $\sqsubseteq_D$ over $DCS(D) \times DCS(D)$ such that for all $P_1, P_2 \in DCS(D)$: $P_1 \sqsubseteq_D P_2$ if and only if $P_1 \subseteq P_2$, $P_1 \sqcup_D P_2 \stackrel{\text{def}}{=} P_1 \cup P_2$, $P_1 \sqcap_D P_2 \stackrel{\text{def}}{=} P_1 \cap P_2$. Notice that $DPL(D)$ is complete because the union and the intersection operations are closed in $DPL(D)$. Given an abstract lattice $DPL(D)$, the abstraction and concretisation mappings are given as follows:

$$\begin{aligned} \forall E \in PL(X)\colon \quad & \alpha[D](E) \stackrel{\text{def}}{=} E\!\downarrow \cap D \\ \forall P \in DPL(D)\colon \quad & \gamma[D](P) \stackrel{\text{def}}{=} \{x \in X \mid x\!\downarrow \cap D \subseteq P\} \ . \end{aligned}$$

The set between brackets defines the *parameter* of the function and the set between parentheses is its *argument*. For simplicity of notation, we also write $\gamma(P)$, $\alpha(E)$, $\sqsubseteq$, $\sqcup$ and $\sqcap$ if the parameter is clear from the context.

We next show through an example that the domain D actually parametrizes the precision of the abstract domain with respect to the concrete domain.

Example 1. Let us consider the $\succeq$-dc-sets of Fig. 1 and consider the following domain $D = \{(0,0), (3,0), (0,2), (0,3)\}$ depicted by the grey dots. Applying α on the $\succeq$-dc-sets A, B and C give, respectively, the (abstract) sets $\alpha(A) = \{(0,0), (3,0)\}, \alpha(B) = \{(0,0), (0,2), (0,3)\}, \alpha(C) = \{(0,0), (0,2)\}$. A and C are exactly represented, i.e. $\gamma(\alpha(A)) = A$ and $\gamma(\alpha(C)) = C$, but B is not: $\gamma(\alpha(B)) = \{(x,y) \in \mathbb{N}^2 \mid x \leq 2\}$. But, if we add $(2,0)$ to D then B is also exactly representable with D.

This generic abstract domain is a generalization of the ideas exposed in [22] for finite states systems.

Fix a domain D, the *concrete* $PL(X)$ and *abstract* $DPL(D)$ domains and the *abstraction* $\alpha\colon PL(X) \mapsto DPL(D)$ and *concretisation* $\gamma\colon DPL(D) \mapsto PL(X)$ maps form a *Galois insertion*, denoted by $PL(X) \underset{\gamma}{\overset{\alpha}{\rightleftharpoons}} DPL(D)$.

Proposition 1. *For every domain D, $PL(X) \underset{\gamma}{\overset{\alpha}{\rightleftharpoons}} DPL(D)$.*

Proof. Fix a domain D. It follows immediately from the definitions that α is monotonic (i.e., $C \subseteq C'$ implies $\alpha(C) \sqsubseteq \alpha(C')$) and γ as well. Indeed, $\gamma(P_1) \subseteq \gamma(P_2) \Leftrightarrow \{c \mid c{\downarrow} \cap D \subseteq P_1\} \subseteq \{c \mid c{\downarrow} \cap D \subseteq P_2\} \Leftrightarrow P_1 \subseteq P_2 \Leftrightarrow P_1 \sqsubseteq P_2$. So, it suffices to prove (a) and (b) below:

(a) $C \subseteq (\gamma \circ \alpha)(C)$ for every $C \in PL(X)$.

$$\begin{aligned}(\gamma \circ \alpha)(C) &= \{c \in X \mid c{\downarrow} \cap D \subseteq C{\downarrow} \cap D\}\\ &\supseteq \{c \in C \mid c{\downarrow} \cap D \subseteq C{\downarrow} \cap D\}\\ &= C\end{aligned}$$

(b) $(\alpha \circ \gamma)(P) = P$ for every $P \in DPL(D)$.

$$\begin{aligned}(\alpha \circ \gamma)(P) &= \{c \mid c{\downarrow} \cap D \subseteq P\}{\downarrow} \cap D &&\\ &= \{c \mid c{\downarrow} \cap D \subseteq P\} \cap D && \gamma(P){\downarrow} = \gamma(P)\\ &= \{c \in D \mid c{\downarrow} \cap D \subseteq P\} &&\\ &= P && P \subseteq D \text{ and } P \in DCS(D) \quad \square\end{aligned}$$

We now prove some properties on the precision of our abstract domain. The next lemma states that any $\succeq$-dc-set of X can be represented exactly using a domain D and a set $P \in DCS(D)$.

Lemma 6 (Completeness of the representation). *For each $E \in DCS(X)$ there exists a domain D such that $(\gamma \circ \alpha)(E) = E$.*

Proof. Given E, we define the domain D to be $D = \mathsf{min}(X \setminus E)$. We proof $(\gamma \circ \alpha)(E) = E$.

Let us show that $(\gamma \circ \alpha)(E) \subseteq E$. For that, suppose by contradiction that there exists $p \in (\gamma \circ \alpha)(E) \wedge p \notin E$.

$$\begin{aligned}& p \notin E &&\\ \Leftrightarrow\ & p{\downarrow} \not\subseteq E && E \in DCS(X)\\ \Leftrightarrow\ & p{\downarrow} \cap (X \setminus E) \neq \emptyset && A \cap B = \emptyset \text{ iff } A \subseteq (X \setminus B)\\ \Leftrightarrow\ & p{\downarrow} \cap \mathsf{min}(X \setminus E) \neq \emptyset && \text{Lem. 3.1}\\ \Leftrightarrow\ & \exists p' : p' \in p{\downarrow} \wedge p' \in \mathsf{min}(X \setminus E) &&\\ \Rightarrow\ & \exists p' : p' \in p{\downarrow} \wedge p' \in \mathsf{min}(X \setminus E) \wedge \exists p'' \in [p'] : p'' \in D && \text{def. of } D\\ \Leftrightarrow\ & \exists p'' : p'' \in p{\downarrow} \wedge p'' \in D \wedge p'' \notin E && p, p' \succeq p''; p'{\downarrow} \cap E = \emptyset\end{aligned} \tag{1}$$

$$\begin{aligned}& p \in (\gamma \circ \alpha)(E) &&\\ \Leftrightarrow\ & p{\downarrow} \cap D \subseteq \alpha(E) && \text{def. of } \gamma\\ \Leftrightarrow\ & p{\downarrow} \cap D \subseteq E{\downarrow} \cap D && \text{def. of } \alpha\\ \Leftrightarrow\ & p{\downarrow} \cap D \subseteq E \cap D && E \in DCS(X)\\ \Leftrightarrow\ & p{\downarrow} \cap D \subseteq E &&\\ \Leftrightarrow\ & \forall p' : p' \in p{\downarrow} \wedge p' \in D \Rightarrow p' \in E &&\\ \Leftrightarrow\ & \neg\neg\left(\forall p' : p' \in p{\downarrow} \wedge p' \in D \Rightarrow p' \in E\right) &&\\ \Leftrightarrow\ & \neg\left(\exists p' : p' \in p{\downarrow} \wedge p' \in D \wedge p' \notin E\right) &&\end{aligned} \tag{2}$$

From (1) and (2) follows a contradiction.

$E \subseteq (\gamma \circ \alpha)(E)$ is immediate by property of Galois insertion. So, we have proved that $(\gamma \circ \alpha)(E) = E$. □

We then show that the more elements you put into the domain D, the more $\succeq$-dc-sets the abstract domain is able to represent exactly. Prior to this we need the following proposition that provides an equivalent definition for $\gamma[D](P)$:

Proposition 2. *Fix a domain D, for every $P \in DPL(D)$ we find that $\gamma(P) = X \setminus (D \setminus P)\uparrow$.*

Proof.

$$\begin{aligned}
\gamma(P) &= \{x \in X \mid x\downarrow \cap D \subseteq P\} && \text{def. of } \gamma \\
X \setminus \gamma(P) &= \{x \in X \mid \neg\,(x\downarrow \cap D \subseteq P)\} \\
&= \{x \in X \mid x\downarrow \cap D \cap (X \setminus P) \neq \emptyset\} && A \subseteq B \text{ iff } A \cap (X \setminus B) = \emptyset \\
&= \{x \in X \mid x\downarrow \cap (D \setminus P) \neq \emptyset\} && D \cap (X \setminus P) = D \setminus P \\
&= \{x \in X \mid \{x\} \cap (D \setminus P)\uparrow \neq \emptyset\} && \text{Lem. 3.1} \\
&= \{x \in X \mid x \in (D \setminus P)\uparrow\} \\
&= (D \setminus P)\uparrow
\end{aligned}$$

Hence, $\gamma(P) = X \setminus (D \setminus P)\uparrow$. □

Proposition 3. *Fix two domains D and D' such that $D \subset D'$. For every $P \in DPL(D)$, there exists a $P' \in DPL(D')$ such that $\gamma[D](P) = \gamma[D'](P')$.*

Proof. We define the set $P' = D' \cap \gamma[D](P)$, thus

$$\begin{aligned}
\gamma[D'](P') &= \{c \mid c\downarrow \cap D' \subseteq P'\} && \text{def. of } \gamma \\
&= \{c \mid c\downarrow \cap D' \subseteq \gamma[D](P) \cap D'\} \\
&= \{c \mid c\downarrow \cap D' \subseteq \gamma[D](P)\} \\
&= \{c \mid c\downarrow \cap D' \cap (D \setminus P)\uparrow = \emptyset\} && \text{prop. 2 and set theory} \\
&= \{c \mid c\downarrow \cap D' \cap (D \setminus P) = \emptyset\} && (c\downarrow \cap D') \in DCS(X), \text{Lem. 3.1} \\
&= \{c \mid c\downarrow \cap (D \setminus P) = \emptyset\} && D \subset D' \\
&= \{c \mid \{c\} \cap (D \setminus P)\uparrow = \emptyset\} && \text{Lem.3.1} \\
&= \{c \mid c \notin (D \setminus P)\uparrow\} \\
&= X \setminus (D \setminus P)\uparrow \\
&= \gamma[D](P) && \text{prop. 2} \quad \square
\end{aligned}$$

Effectiveness. It is worth pointing that since we impose finiteness of D then $\sqcup_D$, $\sqcap_D$ are effective and $\sqsubseteq_D$ is decidable. So, given a domain D, the complete lattice $DPL(D)$ represents an effective way to manipulate (infinite) $\succeq$-dc-sets.

In Sect. 5 we need to decide the intersection emptiness between an $\succeq$-uc-set and a $\succeq$-dc-set. In input of this problem we are given an effective representation of these two sets. Then we solve the problem using the result of Lemma 3.1 together with the following proposition.

Proposition 4. *Fix a domain D, for all $P \in DPL(D)$ there exists an effective procedure to answer the membership test, i.e. "given $c \in X$, $c \in \gamma(P)$?".*

Proof.

$$\begin{aligned} c \in \gamma(P) &\Leftrightarrow c \in X \setminus (D \setminus P){\uparrow} && \text{prop. 2} \\ &\Leftrightarrow c \notin (D \setminus P){\uparrow} \\ &\Leftrightarrow \neg(\exists c' \in (D \setminus P) \colon c \succeq c') \end{aligned}$$

Since both D and P are finite sets, the decidability of $\succeq$ shows that we have an effective procedure to answer the question: $c \in \gamma(P)$? □

4 Abstract Interpretation

In this section, we define the forward abstract interpretation of a WSTS using an abstract domain parametrized by D as defined in the previous section.

Let $\mathcal{S}$ be a WSTS and D be a domain, $post^{\#}[\mathcal{S}, D] \colon DPL(D) \mapsto DPL(D)$ is the function defined as follows: $post^{\#}[\mathcal{S}, D] \stackrel{\text{def}}{=} \lambda P.(\alpha[D] \circ post[\mathcal{S}] \circ \gamma[D])(P)$. The function $post^{\#}[\mathcal{S}, D]^* \colon DPL(D) \mapsto DPL(D)$ is defined as follows: $post^{\#}[\mathcal{S}, D]^* \stackrel{\text{def}}{=} \lambda P. \sqcup_{i \geq 0} post^{\#}[\mathcal{S}, D]^i(P)$. We shorten $post^{\#}[\mathcal{S}, D]$ to $post^{\#}$ and $post^{\#}[\mathcal{S}, D]^*$ to $(post^{\#})^*$ if the WSTS and the domain are clear from the context.

The following lemma establishes the soundness of our abstract interpretation of WSTS which follows by property of Galois connection:

Lemma 7. *Given a WSTS $(X, \delta, \succeq)$ with $I \subseteq X$ and a domain D, (i) $post(I) \subseteq (\gamma \circ post^{\#} \circ \alpha)(I)$ and (ii) $post^*(I) \subseteq (\gamma \circ (post^{\#})^* \circ \alpha)(I)$.*

The next proposition shows that we can improve the precision of the analysis by improving the precision of the abstract domain.

Proposition 5 ($post^{\#}$ Monotonicity). *Given a WSTS $\mathcal{S} = (X, \delta, \succeq)$, two domains D, D' with $D \subseteq D'$, and two sets $C, C' \subseteq X$ with $C \subseteq C'$, we find that, (1) $(\gamma[D'] \circ post^{\#}[\mathcal{S}, D'] \circ \alpha[D'])(C) \subseteq (\gamma[D] \circ post^{\#}[\mathcal{S}, D] \circ \alpha[D])(C')$; and (2) $(\gamma[D'] \circ post^{\#}[\mathcal{S}, D']^* \circ \alpha[D'])(C) \subseteq (\gamma[D] \circ post^{\#}[\mathcal{S}, D]^* \circ \alpha[D])(C')$.*

Let us now show that if we fix a domain D, then $post^{\#}$ is computable for any effective WSTS but first we need the following lemma:

Lemma 8. *Given a WSTS $\mathcal{S} = (X, \delta, \succeq)$, $\forall x, x' \in X \colon x \in pre(x'{\uparrow}) \Leftrightarrow x' \in post(x){\downarrow}$.*

Proof. $x \in pre(x'{\uparrow}) \Leftrightarrow \exists x'' \colon x'' \succeq x' \wedge x \to x'' \Leftrightarrow x' \in post(x){\downarrow}$ □

We have the following characterization of $post^{\#}$.

Proposition 6. *Fix a domain D, and an effective WSTS $\mathcal{S} = (X, \delta, \succeq)$. For every $x \in D$ and $P \in DPL(D)$:*

$$x \in post^{\#}(P) \Leftrightarrow (x \in D \wedge \neg(pre(x{\uparrow}) \subseteq (D \setminus P){\uparrow})) \quad .$$

Proof.

$$
\begin{aligned}
& x \in post^{\#}(P) \\
& \Leftrightarrow x \in (\alpha \circ post \circ \gamma)(P) && \text{def. of } post^{\#} \\
& \Leftrightarrow x \in D \wedge x \in (\downarrow \circ post \circ \gamma)(P) && \text{def. of } \alpha \\
& \Leftrightarrow x \in D \wedge (pre(x\uparrow) \cap \gamma(P)) \neq \emptyset && \text{Lem 8} \qquad (3) \\
& \Leftrightarrow x \in D \wedge (pre(x\uparrow) \cap X \setminus (D \setminus P)\uparrow) \neq \emptyset && \text{Prop. 2} \\
& \Leftrightarrow x \in D \wedge \neg(pre(x\uparrow) \subseteq (D \setminus P)\uparrow) && \square
\end{aligned}
$$

The sets $pre(x\uparrow)\uparrow$ and $(D \setminus P)\uparrow$ are $\succeq$-uc-sets which have as finite minor set $minpre(x)$ and $(D\setminus P)$ respectively. Lemma 3.2 shows that if both $minpre(x)$ and $(D \setminus P)$ are finite sets and $\succeq$ is decidable then we have an *effective* procedure to decide if $pre(x\uparrow)\uparrow\subseteq (D \setminus P)\uparrow$ which is equivalent to $pre(x\uparrow) \subseteq (D \setminus P)\uparrow$. Furthermore, since the complete lattice $DPL(D)$ is finite, it follows that:

Corollary 1. *For any effective* IWSTS $\mathcal{S}_0 = (\mathcal{S}, x_0)$*, and any domain* D*,* $((post^{\#}[\mathcal{S}, D])^* \circ \alpha)(x_0)$ *can be effectively computed.*

5 Domain Refinements

In this section, we show that the abstract interpretation that we have defined previously can be made sufficiently precise to decide the coverability problem of (effective) IWSTS. We present two ways of achieving completeness of the abstract interpretation. Both are based on abstract domain refinement. The first (and naïve) way is through enumeration of domains.

5.1 Enumerate Finite Domains

In Sect. 3, we showed that any $\succeq$-dc-set can be represented using a well chosen domain (Lemma 6). In particular, the covering set can be represented using a domain D. Hereunder, Theorem 1 asserts that the abstract interpretation of an IWSTS $\mathcal{S}_0$ using a domain D that allows to represent exactly the covering set of $\mathcal{S}_0$ leads to the construction of that set. To prove Theorem 1, we need the following lemma:

Lemma 9. *Given* $Cover(\mathcal{S}_0)$*, the covering set of an* IWSTS $\mathcal{S}_0$*, and a domain* D *such that* $(\gamma \circ \alpha)(Cover(\mathcal{S}_0)) = Cover(\mathcal{S}_0)$*. For any* $I \subseteq Cover(\mathcal{S}_0)$*, we have* $(\gamma \circ post^{\#} \circ \alpha)(I) \subseteq Cover(\mathcal{S}_0)$*.*

Proof. We first show that $(\gamma \circ \alpha)(I) \subseteq Cover(\mathcal{S}_0)$

$$
\begin{aligned}
I &\subseteq Cover(\mathcal{S}_0) && \text{by hyp.} \\
\alpha(I) &\subseteq \alpha(Cover(\mathcal{S}_0)) && \text{monotonicity of } \alpha \\
(\gamma \circ \alpha)(I) &\subseteq (\gamma \circ \alpha)(Cover(\mathcal{S}_0)) && \text{monotonicity of } \gamma \\
(\gamma \circ \alpha)(I) &\subseteq Cover(\mathcal{S}_0) && \text{by hyp.} \qquad (4)
\end{aligned}
$$

By definition of covering set and by monotonicity of $(\downarrow \circ post)$, we have

$$\forall S\colon\ (S \in DCS(X) \land S \subseteq Cover(\mathcal{S}_0)) \Rightarrow (\downarrow \circ post)(S) \subseteq Cover(\mathcal{S}_0) \qquad (5)$$

We proceed as follows:

$$\begin{aligned}
(post^{\#} \circ \alpha)(I) &= (\alpha \circ post \circ \gamma \circ \alpha)(I) && \text{def. of } post^{\#} \\
&= (\downarrow \circ post \circ \gamma \circ \alpha)(I) \cap D && \text{def. of } \alpha \\
&\subseteq Cover(\mathcal{S}_0) \cap D && \text{by (4) and (5)} \\
&= \alpha(Cover(\mathcal{S}_0)) && \text{def. of } \alpha,\ Cover(\mathcal{S}_0)\!\downarrow = Cover(\mathcal{S}_0)
\end{aligned}$$

From what has already been proved, we see that:

$$\begin{aligned}
(\gamma \circ post^{\#} \circ \alpha)(I) &\subseteq (\gamma \circ \alpha)(Cover(\mathcal{S}_0)) && \text{monotonicity of } \gamma \\
&= Cover(\mathcal{S}_0) && \text{by hyp.} \quad \square
\end{aligned}$$

The lemma is useful for the proof by induction of the following theorem.

Theorem 1. *Given $Cover(\mathcal{S}_0)$, the covering set of an* IWSTS $\mathcal{S}_0$*, and a domain D such that $(\gamma \circ \alpha)(Cover(\mathcal{S}_0)) = Cover(\mathcal{S}_0)$, for any $I \subseteq Cover(\mathcal{S}_0)$ we have $(\gamma \circ (post^{\#})^* \circ \alpha)(I) = Cover(\mathcal{S}_0)$.*

Thanks to this theorem and the results of [14] Algorithm 1 decides the coverability problem for an effective IWSTS$\mathcal{S}_0 = (\mathcal{S}, x_0)$ and a $\succeq$-uc-set bad. The main idea underlying the algorithm is to iteratively analyze an underapproximation of the reachable states (line 1) followed by an overapproximation (line 2). For

Algorithm 1: Enumeration

```
   Let D_0, D_1, ... be an enumeration of the finite subsets of X
   for i = 0, 1, 2, ... do
1      if ∃x_0, ..., x_k : x_0 → ... → x_k with ∀0 ≤ j ≤ k : x_j ∈ D_i ∧ x_k ∈ bad then
           return REACHABLE
2      else if (γ[D_i] ∘ (post#[S, D_i])* ∘ α[D_i])(x_0) ∩ bad = ∅ then
           return UNREACHABLE
   end
```

this algorithm to be effective, we only need the (mild) additional assumption that elements of X are enumerable.

In the next subsection, we show that this assumption can be dropped and propose a more sophisticated way to obtain a domain D which is precise enough to solve the coverability problem. Our refinement technique is based on the analysis of the states leading to bad.

5.2 Eliminate overapproximations leading to bad

Let us first consider the following lemma that can be considered as an alternative to Lemma 9 to achieve completeness.

Lemma 10. *Given a* WSTS $(X, \delta, \succeq)$, bad $\in UCS(X)$*, and a set $\mathcal{O} \in DCS(X)$ such that $post(\mathcal{O}) \subseteq \mathcal{O}$. Fix a domain D such that $(\gamma \circ \alpha)(\mathcal{O}) = \mathcal{O}$ and* $\min(pre^*(\mathsf{bad})) \cap \mathcal{O} \subseteq D$*. For every $P \in DPL(D)$ such that $\gamma(P) \subseteq \mathcal{O}$ we find that $\gamma(P) \cap pre^*(\mathsf{bad}) = \emptyset \Rightarrow \gamma(post^{\#}(P)) \cap pre^*(\mathsf{bad}) = \emptyset$.*

Proof.

$$\begin{array}{lr}
\gamma(P) \cap pre^*(\mathsf{bad}) = \emptyset & \\
\Leftrightarrow (\downarrow \circ post \circ \gamma)(P) \cap pre^*(\mathsf{bad}) = \emptyset & \text{Lem. 8 and } pre(pre^*(\mathsf{bad})) = pre^*(\mathsf{bad}) \\
\Rightarrow (\alpha \circ post \circ \gamma)(P) \cap pre^*(\mathsf{bad}) = \emptyset & (\alpha \circ post \circ \gamma)(P) \subseteq (\downarrow \circ post \circ \gamma)(P) \\
\Leftrightarrow post^{\#}(P) \cap pre^*(\mathsf{bad}) = \emptyset & \text{def. of } post^{\#} \\
\Leftrightarrow pre^*(\mathsf{bad}) \subseteq (X \setminus post^{\#}(P)) & A \cap B = \emptyset \text{ iff } A \subseteq (X \setminus B)
\end{array}$$

So we have established

$$\gamma(P) \cap pre^*(\mathsf{bad}) = \emptyset \Rightarrow pre^*(\mathsf{bad}) \subseteq (X \setminus post^{\#}(P)) \ . \tag{6}$$

$$\begin{array}{lr}
\text{Moreover, } \gamma(P) \subseteq \mathcal{O} & \\
\Rightarrow (\downarrow \circ post \circ \gamma)(P) \subseteq \mathcal{O} & (\downarrow \circ post)(\mathcal{O}) \subseteq \mathcal{O}, \downarrow \circ\ post \text{ is monotonic} \\
\Rightarrow (\alpha \circ post \circ \gamma)(P) \subseteq \alpha(\mathcal{O}) & \text{Monotonicity of } \alpha \text{ and } \alpha \circ \downarrow = \alpha \\
\Leftrightarrow post^{\#}(P) \subseteq \alpha(\mathcal{O}) & \text{def. of } post^{\#} \\
\Rightarrow \gamma(post^{\#}(P)) \subseteq (\gamma \circ \alpha)(\mathcal{O}) & \text{Monotonicity of } \gamma \\
\Rightarrow \gamma(post^{\#}(P)) \subseteq \mathcal{O} & \text{by hyp.}
\end{array} \tag{7}$$

Now, let us consider $\gamma(post^{\#}(P))$:

$$\begin{array}{lr}
\gamma(post^{\#}(P)) & \\
= \{c \mid c{\downarrow} \cap D \subseteq post^{\#}(P)\} & \text{definition of } \gamma \\
= \{c \in \mathcal{O} \mid c{\downarrow} \cap D \subseteq post^{\#}(P)\} & \text{by (7)} \\
\subseteq \{c \in \mathcal{O} \mid c{\downarrow} \cap \mathsf{min}(pre^*(\mathsf{bad})) \cap \mathcal{O} \subseteq post^{\#}(P)\} & \text{def. of } D \\
= \{c \in \mathcal{O} \mid c{\downarrow} \cap \mathsf{min}(pre^*(\mathsf{bad})) \subseteq post^{\#}(P)\} & c \in \mathcal{O} \text{ implies } c{\downarrow} \subseteq \mathcal{O} \\
= \{c \in \mathcal{O} \mid c{\downarrow} \cap \mathsf{min}(pre^*(\mathsf{bad})) \cap (X \setminus post^{\#}(P)) = \emptyset\} & \\
\subseteq \{c \in \mathcal{O} \mid c{\downarrow} \cap \mathsf{min}(pre^*(\mathsf{bad})) \cap pre^*(\mathsf{bad}) = \emptyset\} & \text{By (6)} \\
= \{c \in \mathcal{O} \mid c{\downarrow} \cap \mathsf{min}(pre^*(\mathsf{bad})) = \emptyset\} & \mathsf{min}(A) \subseteq A \text{ if } A \in UCS(X) \\
= \{c \in \mathcal{O} \mid \{c\} \cap pre^*(\mathsf{bad}) = \emptyset\} & \text{Lem. 3.1} \\
= \{c \in \mathcal{O} \mid c \notin pre^*(\mathsf{bad})\} &
\end{array}$$

Hence, $\gamma(post^{\#}(P)) \cap pre^*(\mathsf{bad}) = \emptyset$. □

Using the previous lemma and induction we can establish the following theorem.

Theorem 2. *Given a* WSTS $(X, \delta, \succeq)$, $\mathsf{bad} \in UCS(X)$*, and a set* $\mathcal{O} \in DCS(X)$ *such that* $post(\mathcal{O}) \subseteq \mathcal{O}$*. Fix a domain* D *with* $(\gamma \circ \alpha)(\mathcal{O}) = \mathcal{O}$ *and* $\mathsf{min}(pre^*(\mathsf{bad})) \cap \mathcal{O} \subseteq D$*. For every* $I \subseteq X$ *such that* $(\gamma \circ (post^{\#})^* \circ \alpha)(I) \subseteq \mathcal{O}$*, we have* $I \cap pre^*(\mathsf{bad}) = \emptyset \Leftrightarrow (\gamma \circ (post^{\#})^* \circ \alpha)(I) \cap \mathsf{bad} = \emptyset$.

We are nearly in position to define our refinement-based algorithm. We first define the following operator parametrized by $\mathcal{O} \subseteq X$ which is applied to a finite subset of states $T \subseteq X$: $minpre[\mathcal{S},\mathcal{O}](T) \stackrel{\text{def}}{=} minpre[\mathcal{S}](T) \cap \mathcal{O}$.We also write $minpre[\mathcal{O}](T)$ instead of $minpre[\mathcal{S},\mathcal{O}](T)$ if the WSTS is clear from the context.

In the remainder of this section we adopt the following convention: a set A acting as the argument of *minpre* should be read as $\mathsf{min}(A)$. A direct consequence of the definition of *minpre* is the following, for any $\mathcal{O} \subseteq \mathcal{O}' \subseteq X$ and $A \subseteq X$ we have:

$$minpre[\mathcal{O}]^*(A) \subseteq minpre[\mathcal{O}']^*(A) \ . \tag{8}$$

The main ideas underlying our refinement-based algorithm (Algorithm 2) are as follows. In a first approximation, we consider a domain D_0 that contains a minor set of bad. With this set, we compute a first overapproximation of the reachable states of $\mathcal{S}_0$, noted $\mathcal{O}_0$. If this overapproximation is fine enough to prove that we are in presence of a negative instance of the problem then we conclude at line 2. If it is not the case, we compute R'_0 that represents all the states within $\mathcal{O}_0$ that can reach bad in one step. If this set contains x_0 then we conclude that bad is reachable. Otherwise, we refine the domain D_0 into D_1 to ensure at the next iteration that our overapproximation will be more precise (Prop. 5.2) and that $(\gamma[D_1] \circ post^{\#}[\mathcal{S}, D_1] \circ \alpha[D_1])(x_0))$ will not intersect with bad. So, we have excluded **all** spurious counter-examples of length one. We then proceed with this enlarged domain.

Since $\mathsf{min}(pre^*(\mathsf{bad}))$ is computable, Theorem 2 intuitively shows that our algorithm terminates. We formally establish the correctness of our technique as stated in the next lemmas which proof soundness, completeness, and termination of Algorithm 2.

Algorithm 2: Refinement loop

Input: An IWSTS $\mathcal{S}_0$, a set $\mathsf{bad} \in UCS(X)$
Let $D_0 \supseteq (\mathsf{min}(\mathsf{bad}))$
for $i = 0, 1, 2, \ldots$ **do**
1 Compute R_i defined to be $((post^{\#}[\mathcal{S}, D_i])^* \circ \alpha[D_i])(x_0)$
Let $\mathcal{O}_i$ denote $\gamma[D_i](R_i)$
2 **if** $\mathcal{O}_i \cap \mathsf{bad} = \emptyset$ **then return** UNREACHABLE
else
3 Compute R'_i defined to be $\mathsf{min}\left(\bigcup_{k=0}^{i+1} minpre[\mathcal{S}, \mathcal{O}_i]^k(\mathsf{bad})\right)$
4 **if** $\{x_0\} \cap R'_i\!\uparrow = \emptyset$ **then**
5 **choose** $D_{i+1} \supseteq D_i \cup R'_i$
else return REACHABLE
end
end

Lemma 11 (Soundness). *If Algorithm 2 says "*REACHABLE*" then we have* $post^*(x_0) \cap \mathsf{bad} \neq \emptyset$.

Proof. Let c be the value of variable i when the algorithm says "REACHABLE". $minpre[\mathcal{O}_c]^*(\mathsf{bad})\!\uparrow \subseteq minpre[X]^*(\mathsf{bad})\!\uparrow = pre^*(\mathsf{bad})$, the inclusion follows from $\mathcal{O}_c \subseteq X$, (8) and $\uparrow$ is monotonic, and the equality follows from Lemma 4.b. Since $\{x_0\} \cap pre^*(\mathsf{bad}) \neq \emptyset$ iff $post^*(x_0) \cap \mathsf{bad} \neq \emptyset$, $minpre[\mathcal{O}_c]^c(\mathsf{bad})\!\uparrow \subseteq pre^*(\mathsf{bad})$ shows that $post^*(x_0) \cap \mathsf{bad} \neq \emptyset$, by $\{x_0\} \cap minpre[\mathcal{O}_c]^c(\mathsf{bad})\!\uparrow \neq \emptyset$ (line 4). $\square$

Lemma 12 (Completeness). *If Algorithm 2 says "*UNREACHABLE*" then we have* $post^*(x_0) \cap \mathsf{bad} = \emptyset$.

Proof. Fix a domain D, by Lemma 7 we have that $post^*(x_0) \subseteq (\gamma \circ (post^{\#})^* \circ \alpha)(x_0)$. Let c be the value of variable i when the algorithm says "UNREACHABLE" at line 2. We conclude from $(\gamma[D_c] \circ (post^{\#}[\mathcal{S}, D_c])^* \circ \alpha[D_c])(x_0) \cap \mathsf{bad} = \emptyset$ that $post^*(x_0) \cap \mathsf{bad} = \emptyset$ which is the desired conclusion. □

Lemma 13 (Termination). *Given an effective* IWSTS $\mathcal{S}_0$ *and* $\mathsf{bad} \in UCS(X)$*, Algorithm 2 always terminates.*

Proof. It is routine to check that each domain D_i is finite. Hence, since $\sqcup_{D_i}$ is computable because the D_i's are finite and $post^{\#}[\mathcal{S}, D_i]$ is computable following Proposition 6 (notice that $\alpha[D_i](x_0)$ is computable since $\succeq$ is assumed to be decidable), the fixpoint computation of line 1 finishes after a finite amount of time.

Suppose, contrary to our claim, that the algorithm does not terminate. Since each line is evaluated in a finite amount of time, it follows that the algorithm executes the main loop infinitely many times. From line 5, we conclude that the algorithm considers an infinite sequence of domains $D_0 \subseteq D_1 \subseteq \cdots$ From Proposition 5.2, we know that $\mathcal{O}_0 \supseteq \mathcal{O}_1 \supseteq \cdots$ From Lemma 2, we conclude that there exists $i \geq 0$ such that $\mathcal{O}_i = \mathcal{O}_{i+1} = \cdots$

Let us consider the iteration i of the algorithm such that $\mathcal{O}_i = \mathcal{O}_{i+1} = \cdots$ We have the infinite sequence $R'_i\!\uparrow \subseteq R'_{i+1}\!\uparrow \subseteq \cdots$. From Lemma 2, we conclude that there exists $j \geq i$ such that $R'_j\!\uparrow = R'_{j+1}\!\uparrow = \cdots$. Hence, following line 5 of the algorithm, D contains $\mathsf{min}(minpre[\mathcal{O}_i]^*(\mathsf{bad}))$ (or rather D contains equivalent states to those of $\mathsf{min}(minpre[\mathcal{O}_i]^*(\mathsf{bad}))$) after the j^{th} iteration.

Let us now prove that **(a)** $\mathsf{min}(minpre[\mathcal{O}_{j+1}]^*(\mathsf{bad})) \equiv \mathsf{min}(minpre[X]^*(\mathsf{bad})) \cap \mathcal{O}_{j+1}$. Indeed, if it is not the case there exist $l \geq 0, c, c' \in X$ such that $c \in minpre[X]^l(\mathsf{bad})$, $c' \in minpre[X](c)$, $c \notin \mathcal{O}_{j+1}$ and $c' \in \mathcal{O}_{j+1}$. Hence, $post(c') \not\subseteq \mathcal{O}_{j+1}$ since $post(c') \cap c\!\uparrow \neq \emptyset$ and $\mathcal{O}_{j+1}$ is a $\succeq$-dc-set. But, $\forall \overline{c} \in \mathcal{O}_{j+1} : post(\overline{c}) \subseteq \mathcal{O}_{j+1}$. From this follows a contradiction.

Moreover, **(b)** $\mathsf{min}((minpre[X]^*(\mathsf{bad}))\!\uparrow) \equiv \mathsf{min}(pre^*(\mathsf{bad}))$ holds by Lemma 4.b and by definition of $\equiv$. We conclude, following line 5 of the algorithm, that D_{j+1} contains equivalent states to those of $\mathsf{min}(pre^*(\mathsf{bad})) \cap \mathcal{O}_{j+1}$.

By applying Theorem 2, we have $\{x_0\} \cap pre^*(\mathsf{bad}) = \emptyset$ iff $\mathcal{O}_{j+1} \cap \mathsf{bad} = \emptyset$. We consider two cases: (i) $\{x_0\} \cap pre^*(\mathsf{bad}) = \emptyset$, then we have $\mathcal{O}_{j+1} \cap \mathsf{bad} = \emptyset$ and the algorithm terminates since the test of line 2 is evaluated to true; (ii) $\{x_0\} \cap pre^*(\mathsf{bad}) \neq \emptyset$, then $\mathcal{O}_{j+1} \cap \mathsf{bad} \neq \emptyset$. Following **(a)** and **(b)** at line 3 of the algorithm $R'_{j+1} \equiv \mathsf{min}(pre^*(\mathsf{bad})) \cap \mathcal{O}_{j+1}$. Since $\{x_0\} \cap pre^*(\mathsf{bad}) \neq \emptyset$, there exists, on account of Lemma 3.3, $x \in \mathsf{min}(pre^*(\mathsf{bad}))\colon x_0 \succeq x$. $x_0 \in \mathcal{O}_{j+1}$ and $\mathcal{O}_{j+1} \in DCS(X)$ shows that $x \in \mathcal{O}_{j+1}$. We conclude from $R'_{j+1} \equiv \mathsf{min}(pre^*(\mathsf{bad})) \cap \mathcal{O}_{j+1}$ that $[x] \cap R'_{j+1} \neq \emptyset$, hence that $\{x_0\} \cap R'_{j+1}\!\uparrow \neq \emptyset$, and finally that the test of line 4 is evaluated to false which yields the algorithm to terminate. □

Remark 1. Let us notice that the practical efficiency of Algorithm 2 depends on (i) the preciseness of the overapproximations $\mathcal{O}_i$ and (ii) the time (and space) needed to build those overapproximations. Point (i) is crucial since rough approximations will lead to the computation of $\mathsf{min}(pre^*(\mathsf{bad}))$, which is time and space consuming in practice [23]. Point (ii) is important because an inefficient computation of overapproximations leads to an inefficient algorithm. Hence, a trade-off between (i) and (ii) must be chosen. This problem exceeds the scope of this paper and will be addressed in future works.

To ensure termination we require, at line 5, that the domain is enlarged by, at least, the states of R'_i. The algorithm remains correct if we add more states.

6 Illustrations

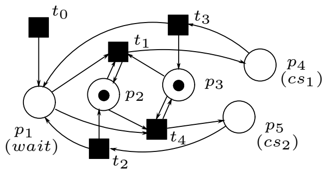
We have produced a prototype that implements Algorithm 2. We describe in this section the execution of that prototype when applied on a toy example. The example of IWSTS $\mathcal{S}_0$ is represented through a Petri net (see [9] for details), depicted in Fig. 2, which models a very simple mutual exclusion protocol. We want to check for safety of the protocol, that is check that *there is never more than one process in the critical sections.* The markings that violates the property, denoted bad, are given by $\{\langle 0,0,0,1,1\rangle, \langle 0,0,0,0,2\rangle, \langle 0,0,0,2,0\rangle\}\uparrow$. It is worth pointing that we want to establish the safety for any number of processes taking part in the protocol (recall that t_0 spawns processes).

Execution of the prototype. We describe the execution of the prototype iteration by iteration. On account of remark 1, we do not take $\mathsf{min}(\mathsf{bad})$ as initial domain but its downward closure instead and we do not add the set R'_i to D_i at the i^{th} iteration but its downward closure instead. Taking the $\succeq$-downward-closure of the sets allows us to efficiently proof the safeness of the protocol.
Initialisation. As mentioned before, the initial value of the domain, which is referred as D_0, is given by $\{\langle 0,0,0,1,1\rangle, \langle 0,0,0,0,2\rangle, \langle 0,0,0,2,0\rangle\}\downarrow$.
Iteration 1 (i=0). After the fixpoint computation of line 1, we have $R_0 = D_0$, and so $\mathcal{O}_0 = X$. Hence the test of line 2 fails and we compute $R'_0 = D_0 \cup \{\langle 1,1,1,0,1\rangle, \langle 1,1,1,1,0\rangle\}$ which corresponds to $\mathsf{min}(\mathsf{bad} \cup pre(\mathsf{bad}))$. Because the test of line 4 fails, we execute line 5 and we set D_1 to $R'_0\downarrow$.
Iteration 2 (i=1). The fixpoint computation of line 1 ends up with $R_1 = D_1$, hence $\mathcal{O}_1 = X$. Again we perform a refinement step by (i) computing $R'_1 = \mathsf{min}(\mathsf{bad}) \cup \{\langle 0,1,1,0,1\rangle, \langle 0,1,1,1,0\rangle, \langle 2,2,1,0,0\rangle, \langle 2,1,2,0,0\rangle\}$ (which corresponds to $\mathsf{min}(\mathsf{bad} \cup pre(\mathsf{bad}) \cup pre^2(\mathsf{bad})))$ and (ii) adding tuples of $R'_1\downarrow$ with the ones of D_1 to obtain D_2.
Iteration 3 (i=2). The fixpoint computation of line 1 finishes with a set R_2 such that the test of line 2 ($\mathcal{O}_2 \cap \mathsf{bad} = \emptyset$) succeeds and the system is proved to be safe.
Indeed $\mathcal{O}_2 = \{(p_1,p_2,p_3,p_4,p_5) \in \mathbb{N}^5 \mid (\bigwedge_{i=2}^{5} p_i \leq 1) \wedge p_4 + p_5 \leq 1 \wedge p_3 + p_4 \leq 1 \wedge p_2 + p_5 \leq 1\}$ which is equal to $Cover(\mathcal{S}_0)$. Since $Cover(\mathcal{S}_0)$ is, in general, not computable ([11]), the equality does always not hold. Notice that $pre^*(\mathsf{bad})$ is computed in five iterations with the classical algorithm of [17]. Hence, the forward analysis allows to drastically cut the backward search.

References

1. Alur, R., Dill, D.: A Theory of Timed Automata. Theoretical Computer Science **126** (1994) 183–236

Fig. 2. A simple mutual exclusion protocol.



The processes (the tokens in place p_1) can access some critical section (place p_4 or p_5) provided they acquired some lock (the tokens in places p_2 and p_3). The initial marking is given by $\langle 0,1,1,0,0\rangle$. Transition t_0 spawns processes.

2. Henzinger, T.A.: The theory of hybrid automata. In: Proceedings of the 11th Symposium on Logic in Computer Science (LICS '96), IEEE Computer Society (1996) 278
3. Abdulla, P., Jonsson, B.: Verifying Programs with Unreliable Channels. In: Proceedings of the 8th IEEE International Symposium in Logic in Computer Science (LICS'93), IEEE Computer Society Press (1993) 160–170
4. Abdulla, P., Bouajjani, A., Jonsson, B.: On-the-Fly Analysis of Systems with Unbounded, Lossy FIFO Channels. In: Proceedings of the 10th International Conference on Computer Aided Verification (CAV'98). Volume 1427 of LNCS., Springer (1998) 305–318
5. Abdulla, P., Annichini, A., Bouajjani, A.: Symbolic verification of lossy channel systems: Application to the bounded retransmission protocol. In: Proc. 5th Intern. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'99). Number 1579 in LNCS, Springer-Verlag (1999) 208–222
6. Delzanno, G., Raskin, J.F., Van Begin, L.: Towards the Automated Verification of Multithreaded Java Programs. In: Proceedings of the International Conference on Tools and Algorithms for Construction and Analysis of Systems (TACAS 2002). Volume 2280 of LNCS., Springer (2002) 173–187
7. Bardin, S., Finkel, A., Leroux, J., Petrucci, L.: FAST: Fast acceleration of symbolic transition systems. In: Proceedings of the 15th International Conference on Computer Aided Verification (CAV'03). will be published in LNCS, Springer (2003)
8. Esparza, J., Finkel, A., Mayr, R.: On the Verification of Broadcast Protocols. In: Proceedings of the 14th Annual Symposium on Logic in Computer Science (LICS'99), IEEE Computer Society Press (1999) 352–359
9. Reisig, W.: Petri Nets. An introduction. Springer (1986)
10. Ciardo, G.: Petri nets with marking-dependent arc multiplicity: properties and analysis. In: Proceedings of the 15th International Conference on Applications and Theory of Petri Nets (ICATPN 94). Volume 815 of LNCS., Springer (1994) 179–198
11. Dufourd, C., Finkel, A., Schnoebelen, P.: Reset Nets Between Decidability and Undecidability. In: In Proceedings of the 25th International Colloquium on Automata, Languages, and Programming (ICALP'98). Volume 1443 of LNCS., Springer (1998) 103–115
12. Raskin, J.F., Van Begin, L.: Petri Nets with Non-blocking Arcs are Difficult to Analyse. In: Proceedings of the 5th International Workshop on Verification of Infinite-state Systems (INFINITY 2003). Volume 96 of ENTCS., Elsevier (2003)
13. Emerson, E.A., Namjoshi, K.S.: On Model Checking for Non-deterministic Infinite-state Systems. In: Proceedings of the 13th Annual Symposium on Logic in Computer Science (LICS '98), IEEE Computer Society Press (1998) 70–80
14. Geeraerts, G., Raskin, J.F., Van Begin, L.: Expand, Enlarge and Check: new algorithms for the coverability problem of WSTS. In: FSTTCS. Volume 3328 of LNCS., Springer-Verlag (2004) 287–298
15. Abdulla, P.A., Deneux, J., Mahata, P., Nylén, A.: Downward Closed Language Generators. Technical Report 2003-001, Department of Information Technology, Uppsala University (2003)

16. Abdulla, P., Deneux, J., Mahata, P., Nylen, A.: Forward Reachability Analysis of Timed Petri Nets. In: Proceedings of Formats-FTRTFT '04. Joint Conference of Formal Modelling and Analysis of Timed Systems and Formal Techniques in Real-Time and Fault-Tolerant Systems. Volume 3253 of Lecture Notes in Computer Science., Springer (2004) 343—??
17. Abdulla, P.A., Cerans, K., Jonsson, B., Tsay, Y.K.: General decidability theorems for infinite-state systems. In: LICS. (1996) 313–321
18. Delzanno, G., Raskin, J.F., Van Begin, L.: Covering Sharing Trees: Efficient Data Structurs for the Automated Verification of Parametrized Systems. accepted for publication in Software Tools for Technology Transfer Manuscript **5** (2004) 268–297
19. Finkel, A., Schnoebelen, Ph.: Well-structured transition systems everywhere! Theoretical Computer Science **256** (2001) 63–92
20. Finkel, A.: Reduction and Covering of Infinite Reachability Trees. Information and Computation **89** (1990) 144–179
21. Geeraerts, G., Raskin, J.F., Van Begin, L.: Expand, enlarge and check: new algorithms for the coverability problem of WSTS. In: Proceedings of the 24th International Conference on Fondation of Software Technology and Theoretical Computer Science (FSTTCS 04). Volume 3328 of LNCS., Springer (2004) 287–298
22. Esparza, J., Ganty, P., Schwoon, S.: Locality-based abstractions. In: Proc. of the 12th Int. Symp. on Static Analysis (SAS). Volume 3672 of Lecture Notes in Computer Science., Springer-Verlag (2005) 118–134
23. Van Begin, L.: Efficient Verification of Counting Abstractions for Parametric Systems. PhD thesis, Université Libre de Bruxelles (2003)

A Proofs

Lemma 3

1. *For any* $S, S' \subseteq X$, $S{\downarrow} \cap S'{\uparrow} \neq \emptyset \Leftrightarrow S{\downarrow} \cap S' \neq \emptyset \Leftrightarrow S \cap S'{\uparrow} \neq \emptyset$.
2. *For any* $S, S' \subseteq X$, $S{\uparrow} \subseteq S'{\uparrow} \Leftrightarrow \forall s \in S\ \exists s' \in S' \colon s \succeq s'$.
3. $\forall s \in X, S \in UCS(X) \colon s \in S \Leftrightarrow \exists s' \in \mathsf{min}(S) \colon s \succeq s'$.

Proof. Proof of item 1.

$$
\begin{aligned}
& S{\downarrow} \cap S'{\uparrow} \neq \emptyset \\
& \Leftrightarrow \{c \mid \exists x \in S \colon x \succeq c\} \cap \{c \mid \exists y \in S' \colon c \succeq y\} \neq \emptyset \\
& \Leftrightarrow \{c \mid \exists x \in S, \exists y \in S' \colon x \succeq c \wedge c \succeq y\} \neq \emptyset \\
& \Leftrightarrow \{c \mid \exists x \in S, \exists y \in S' \colon x \succeq c \wedge c = y\} \neq \emptyset \\
& \Leftrightarrow \{c \mid \exists x \in S \colon x \succeq c\} \cap S' \neq \emptyset \\
& \Leftrightarrow S{\downarrow} \cap S' \neq \emptyset
\end{aligned}
$$

A similar reasoning is used to obtain $S{\downarrow} \cap S'{\uparrow} \neq \emptyset \Leftrightarrow S \cap S'{\uparrow} \neq \emptyset$.

Proof of item 2. On account of set theory we have $S{\uparrow} \subseteq S'{\uparrow}$ iff $S{\uparrow} \cap (X \setminus S'{\uparrow}) = \emptyset$.

$$
\begin{aligned}
S{\uparrow} \cap (X \setminus S'{\uparrow}) = \emptyset & \Leftrightarrow \{c \mid \exists s \in S \colon c \succeq s\} \cap \{c \mid \neg(\exists s' \in S' \colon c \succeq s')\} = \emptyset \\
& \Leftrightarrow \forall c \colon \neg\left(\exists s \in S \colon c \succeq s \wedge \neg(\exists s' \in S' \colon c \succeq s')\right) \\
& \Leftrightarrow \forall c\, \exists s \in S \colon c \succeq s \Rightarrow \exists s' \in S' \colon c \succeq s' \\
& \Leftrightarrow \forall s \in S \Rightarrow \exists s' \in S' \colon s \succeq s'
\end{aligned}
$$

Proof of item 3. obvious. □

Proposition 5 (*$post^{\#}$* **Monotonicity**) *Given a* WSTS $\mathcal{S} = (X, \delta, \succeq)$*, two domains D and D' such that $D \subseteq D'$, and two sets $C, C' \subseteq X$ such that $C \subseteq C'$, we find that,*

1. $(\gamma[D'] \circ post^{\#}[\mathcal{S}, D'] \circ \alpha[D'])(C) \subseteq (\gamma[D] \circ post^{\#}[\mathcal{S}, D] \circ \alpha[D])(C')$*;*
2. $(\gamma[D'] \circ post^{\#}[\mathcal{S}, D']^* \circ \alpha[D'])(C) \subseteq (\gamma[D] \circ post^{\#}[\mathcal{S}, D]^* \circ \alpha[D])(C')$*.*

Proof. **Proof of item 1.**

$$(\gamma[D'] \circ post^{\#}[\mathcal{S}, D'] \circ \alpha[D'])(C) \subseteq (\gamma[D] \circ post^{\#}[\mathcal{S}, D] \circ \alpha[D])(C')$$

can be reformulated, following the definition of $post^{\#}$, by

$$(\gamma[D'] \circ \alpha[D'] \circ post \circ \gamma[D'] \circ \alpha[D'])(C) \subseteq (\gamma[D] \circ \alpha[D] \circ post \circ \gamma[D] \circ \alpha[D])(C') \ .$$

Hence, it is sufficient to prove that

$$\begin{aligned}
&(\gamma[D'] \circ \alpha[D'])(T) \subseteq (\gamma[D] \circ \alpha[D])(T') \text{ with } T \subseteq T' \\
&\Leftrightarrow X \setminus (D' \setminus T{\downarrow} \cap D'){\uparrow} \subseteq X \setminus (D \setminus T'{\downarrow} \cap D){\uparrow} && \text{Prop. 2, def. of } \alpha \\
&\Leftrightarrow (D \setminus T'{\downarrow} \cap D){\uparrow} \subseteq (D' \setminus T{\downarrow} \cap D'){\uparrow} \\
&\Leftrightarrow (D \setminus T'{\downarrow}){\uparrow} \subseteq (D' \setminus T{\downarrow}){\uparrow} && A \setminus (B \cap A) = A \setminus B \\
&\Leftrightarrow \forall c \in (D \setminus T'{\downarrow})\ \exists c' \in (D' \setminus T{\downarrow}) : c \succeq c' && \text{Lem. 3.2}
\end{aligned}$$

which is true since $D \subseteq D'$ and $T \subseteq T'$, i.e. $D \setminus T'{\downarrow} \subseteq D' \setminus T{\downarrow}$ (take $c' = c$).

Proof of item 2. We prove by induction on i that

$$(\gamma[D'] \circ \sqcup_{0 \le j \le i} post^{\#}[\mathcal{S}, D']^j \circ \alpha[D'])(C) \subseteq (\gamma[D] \circ \sqcup_{0 \le j \le i} post^{\#}[\mathcal{S}, D]^j \circ \alpha[D])(C')$$

and

$$(\gamma[D'] \circ post^{\#}[\mathcal{S}, D']^i \circ \alpha[D'])(C) \subseteq (\gamma[D] \circ post^{\#}[\mathcal{S}, D]^i \circ \alpha[D])(C') \ .$$

Base case ($i = 0$): By applying the same reasoning as for item 1, we obtain that $(\gamma[D'] \circ \alpha[D'])(C) \subseteq (\gamma[D] \circ \alpha[D])(C')$.

Inductive case ($i + 1$): We have by induction hypothesis that

$$(\gamma[D'] \circ \sqcup_{0 \le k \le i} post^{\#}[\mathcal{S}, D']^k \circ \alpha[D'])(C) \subseteq (\gamma[D] \circ \sqcup_{0 \le k \le i} post^{\#}[\mathcal{S}, D]^k \circ \alpha[D])(C') \tag{9}$$

and

$$(\gamma[D'] \circ post^{\#}[\mathcal{S}, D']^i \circ \alpha[D'])(C) \subseteq (\gamma[D] \circ post^{\#}[\mathcal{S}, D]^i \circ \alpha[D])(C') \ . \tag{10}$$

Using item 1 of the lemma on (10), we obtain

$$\begin{gathered}
(\gamma[D'] \circ post^{\#}[\mathcal{S}, D'] \circ \alpha[D'] \circ \gamma[D'] \circ post^{\#}[\mathcal{S}, D']^i \circ \alpha[D'])(C) \\
\subseteq \\
(\gamma[D] \circ post^{\#}[\mathcal{S}, D] \circ \alpha[D] \circ \gamma[D] \circ post^{\#}[\mathcal{S}, D]^i \circ \alpha[D])(C')
\end{gathered}$$

which can be reformulated as follows because of Galois insertion:

$$(\gamma[D'] \circ post^{\#}[\mathcal{S}, D']^{i+1} \circ \alpha[D'])(C) \subseteq (\gamma[D] \circ post^{\#}[\mathcal{S}, D]^{i+1} \circ \alpha[D])(C') \ . \tag{11}$$

Let $\overline{E} = (post^{\#}[\mathcal{S}, D']^{i+1} \circ \alpha[D'])(C) \sqcup (\sqcup_{0\leq k\leq i} post^{\#}[\mathcal{S}, D']^{k} \circ \alpha[D'])(C)$ and $E = (post^{\#}[\mathcal{S}, D]^{i+1} \circ \alpha[D])(C') \sqcup (\sqcup_{0\leq k\leq i} post^{\#}[\mathcal{S}, D]^{k} \circ \alpha[D])(C')$.

We now prove that $\gamma[D'](\overline{E}) \subseteq \gamma[D](E)$. Since $D \subseteq D'$, we know, following Prop. 3, that there exists $P \in DPL(D')$ such that $\gamma[D'](P) = \gamma[D](E)$. By property of the least upper bound operator $\sqcup_D$, we have

$$\gamma[D]((post^{\#}[\mathcal{S}, D]^{i+1} \circ \alpha[D]))(C')) \subseteq \gamma[D](E) \tag{12}$$

and

$$\gamma[D]((\sqcup_{0\leq k\leq i} post^{\#}[\mathcal{S}, D]^{k} \circ \alpha[D])(C')) \subseteq \gamma[D](E) \ . \tag{13}$$

We conclude from (11) and (12) that:

$$(\gamma[D'] \circ post^{\#}[\mathcal{S}, D']^{i+1} \circ \alpha[D'])(C) \subseteq \gamma[D](E)$$

hence that,

$$(post^{\#}[\mathcal{S}, D']^{i+1} \circ \alpha[D'])(C) \sqsubseteq_{D'} P \ .$$

Similarly, we conclude from (9) and (13) that :

$$(\gamma[D'] \circ \sqcup_{0\leq k\leq i} post^{\#}[\mathcal{S}, D']^{k} \circ \alpha[D'])(C) \subseteq \gamma[D](E)$$

hence that,

$$(\sqcup_{0\leq k\leq i} post^{\#}[\mathcal{S}, D']^{k} \circ \alpha[D'])(C) \sqsubseteq_{D'} P \ .$$

Since $\sqcup_{D'}$ is a least upper bound operator and by property of Galois insertion, we deduce that $\overline{E} \sqsubseteq_{D'} P$, hence that $\gamma[D'](\overline{E}) \subseteq \gamma[D'](P) = \gamma[D](E)$ which is the desired conclusion.

Finally, the lemma follows from the fact that there exists $j \geq 0$ such that $(\gamma[D'] \circ \sqcup_{0\leq k\leq j} post^{\#}[\mathcal{S}, D']^{k} \circ \alpha[D'])(C) = (\gamma[D'] \circ post^{\#}[\mathcal{S}, D']^{*} \circ \alpha[D'])(C)$ and $(\gamma[D] \circ \sqcup_{0\leq k\leq j} post^{\#}[\mathcal{S}, D]^{k} \circ \alpha[D])(C') = (\gamma[D] \circ post^{\#}[\mathcal{S}, D]^{*} \circ \alpha[D])(C')$. □

Theorem 1 *Given $Cover(\mathcal{S}_0)$, the covering set of an* IWSTS $\mathcal{S}_0$*, and a domain D such that $(\gamma \circ \alpha)(Cover(\mathcal{S}_0)) = Cover(\mathcal{S}_0)$, for any $I \subseteq Cover(x_0)$ we have*

$$(\gamma \circ (post^{\#})^{*} \circ \alpha)(I) = Cover(\mathcal{S}_0) \ .$$

Proof. "$\supseteq$". Trivial by soundness of the abstract interpretation.
"$\subseteq$". We prove by induction that $(\gamma \circ \sqcup_{0\leq j\leq i} (post^{\#})^{j} \circ \alpha)(I) \subseteq Cover(\mathcal{S}_0)$.

Base case ($i = 0$): $I \subseteq Cover(\mathcal{S}_0)$ is true by hypothesis. $(\gamma \circ \alpha)(I) \subseteq (\gamma \circ \alpha)(Cover(\mathcal{S}_0)) = Cover(\mathcal{S}_0)$, the inclusion holds by monotonicity of α and γ and the equality holds by hypothesis.

Inductive case ($i + 1$): By induction hypothesis, $(\gamma \circ \sqcup_{0\leq k\leq i} (post^{\#})^{k} \circ \alpha)(I) \subseteq Cover(\mathcal{S}_0)$, which implies, by Galois insertion, that

$$(\sqcup_{0\leq k\leq i} (post^{\#})^{k} \circ \alpha)(I) \sqsubseteq \alpha(Cover(\mathcal{S}_0)) \ . \tag{14}$$

Moreover, $(\gamma \circ (post^{\#})^{i+1} \circ \alpha)(I) \subseteq Cover(\mathcal{S}_0)$ is obtained from Lemma 9 and the fact that $(\gamma \circ (post^{\#})^{i} \circ \alpha)(I) \subseteq (\gamma \circ \sqcup_{0\leq k\leq i} (post^{\#})^{k} \circ \alpha)(I) \subseteq Cover(\mathcal{S}_0)$. Since we have a Galois insertion, we find that

$$((post^{\#})^{i+1} \circ \alpha)(I) \sqsubseteq \alpha(Cover(\mathcal{S}_0)) \ . \tag{15}$$

We conclude from (14) and (15) that

$$((post^{\#})^{i+1} \circ \alpha)(I) \sqcup (\sqcup_{0 \leq k \leq i} (post^{\#})^k \circ \alpha)(I) \sqsubseteq \alpha(Cover(\mathcal{S}_0))$$

i.e.

$$(\gamma \circ \sqcup_{0 \leq k \leq i+1} (post^{\#})^k \circ \alpha)(I) \subseteq (\gamma \circ \alpha)(Cover(\mathcal{S}_0)) = Cover(\mathcal{S}_0) \ .$$

The theorem follows from the fact that there exists $j \geq 0$ such that $(\gamma \circ \sqcup_{0 \leq k \leq j} (post^{\#})^k \circ \alpha)(I) = (\gamma \circ (post^{\#})^* \circ \alpha)(I)$. □

Lemma 14. *Fix a domain D and $U \in UCS(X)$ such that $\mathsf{min}(U) \subseteq D$. Let $A, B \in DPL(D)$, we find that*

$$\gamma(A) \cap U = \emptyset \wedge \gamma(B) \cap U = \emptyset \Leftrightarrow \gamma(A \sqcup B) \cap U = \emptyset \ .$$

Proof.

$\Leftarrow$. We conclude from the monotonicity of γ that $\gamma(A) \subseteq \gamma(A \sqcup B)$, hence that $\gamma(A \sqcup B) \cap U = \emptyset$ implies that $\gamma(A) \cap U = \emptyset$. Applying the same reasoning to B completes the proof.

$\Rightarrow$. Since $U \in UCS(X)$, we have $A \cap U = \emptyset$ for otherwise $\gamma(A) \cap U \neq \emptyset$ which is a contradiction. The same reasoning applies for B.

$$\begin{aligned}
\gamma(A \sqcup B) &= \gamma(A \cup B) && \text{by def. of } \sqcup \\
&= X \setminus (D \setminus (A \cup B))\!\uparrow && \text{Prop. 2} \\
&\subseteq X \setminus \mathsf{min}(U)\!\uparrow && (A \cup B) \cap U = \emptyset \\
&= X \setminus U && U \in UCS(X)
\end{aligned}$$

Hence $\gamma(A \sqcup B) \cap U = \emptyset$. □

Theorem 2 *Given a* WSTS $(X, \delta, \succeq)$, $\mathsf{bad} \in UCS(X)$, *and a set $\mathcal{O} \in DCS(X)$ such that $post(\mathcal{O}) \subseteq \mathcal{O}$. Fix a domain D with $(\gamma \circ \alpha)(\mathcal{O}) = \mathcal{O}$ and $\mathsf{min}(pre^*(\mathsf{bad})) \cap \mathcal{O} \subseteq D$. For every $I \subseteq X$ such that $(\gamma \circ (post^{\#})^* \circ \alpha)(I) \subseteq \mathcal{O}$, we have*

$$I \cap pre^*(\mathsf{bad}) = \emptyset \Leftrightarrow (\gamma \circ (post^{\#})^* \circ \alpha)(I) \cap \mathsf{bad} = \emptyset \ .$$

Proof.

$\Leftarrow$. We know that $I \cap pre^*(\mathsf{bad}) = \emptyset$ iff $post^*(I) \cap \mathsf{bad} = \emptyset$. By Lemma 7, we know that

$$post^*(I) \subseteq (\gamma \circ (post^{\#})^* \circ \alpha)(I)$$

which completes the proof.

$\Rightarrow$. We prove a stronger form of the theorem,

$$I \cap pre^*(\mathsf{bad}) = \emptyset \Rightarrow (\gamma \circ (post^\#)^* \circ \alpha)(I) \cap pre^*(\mathsf{bad}) = \emptyset \ .$$

We proceed by induction on the the number of applications of $post^\#$.

Base case (i=0) First, we have

$$(\gamma \circ \alpha)(I) \subseteq \mathcal{O} \ . \tag{16}$$

Indeed, that property is directly implied by $(\gamma\circ\alpha)(I) \subseteq (\gamma\circ(post^\#)^*\circ\alpha)(I) \subseteq \mathcal{O}$.
We now proof $I \cap pre^*(\mathsf{bad}) = \emptyset$ implies that $(\gamma \circ \alpha)(I) \cap pre^*(\mathsf{bad}) = \emptyset$.
Lemma 3.1 shows that (i) $I \cap pre^*(\mathsf{bad}) = \emptyset$ iff $I{\downarrow} \cap \mathsf{min}(pre^*(\mathsf{bad})) = \emptyset$.
By hypothesis, we have $\mathsf{min}(pre^*(\mathsf{bad})) \cap \mathcal{O} \subseteq D$, and so by (i)

$$\mathsf{min}(pre^*(\mathsf{bad})) \cap \mathcal{O} \subseteq D \setminus (I{\downarrow}) \ . \tag{17}$$

Hence,

$$\begin{aligned}
(\gamma \circ \alpha)(I) &= X \setminus (D \setminus \alpha(I)){\uparrow} && \text{Prop. 2}\\
&= X \setminus (D \setminus (I{\downarrow} \cap D)){\uparrow} && \text{def. of } \alpha\\
&= X \setminus (D \setminus (I{\downarrow})){\uparrow} && A \setminus (B \cap A) = A \setminus B\\
&\subseteq X \setminus (\mathsf{min}(pre^*(\mathsf{bad})) \cap \mathcal{O}){\uparrow} && \text{By (17)}
\end{aligned}$$

Since $A \subseteq B$ is equivalent to $A \cap (X \setminus B) = \emptyset$ we write

$$\begin{aligned}
&(\gamma \circ \alpha)(I) \cap (\mathsf{min}(pre^*(\mathsf{bad})) \cap \mathcal{O}){\uparrow} = \emptyset\\
&\Leftrightarrow (\gamma \circ \alpha)(I) \cap \mathsf{min}(pre^*(\mathsf{bad})) \cap \mathcal{O} = \emptyset && \text{Lem. 3.1}\\
&\Leftrightarrow (\gamma \circ \alpha)(I) \cap \mathsf{min}(pre^*(\mathsf{bad})) = \emptyset && \text{By (16)}\\
&\Leftrightarrow (\gamma \circ \alpha)(I) \cap pre^*(\mathsf{bad}) = \emptyset && \text{Lem. 3.1}
\end{aligned}$$

Hence $(\gamma \circ \alpha)(I) \cap pre^*(\mathsf{bad}) = \emptyset$.

Inductive case (i+1)

$$\begin{aligned}
&\gamma\left(\bigsqcup_{0\le k\le i} (post^\#)^k(\alpha(I))\right) \cap pre^*(\mathsf{bad}) = \emptyset && \text{Induction hypothesis}\\
&\Rightarrow \gamma\left((post^\#)^i(\alpha(I))\right) \cap pre^*(\mathsf{bad}) = \emptyset && \text{property of } \sqcup\\
&\Rightarrow \gamma\left((post^\#)^{i+1}(\alpha(I))\right) \cap pre^*(\mathsf{bad}) = \emptyset && \text{Lem. 10}
\end{aligned} \tag{18}$$

From (18), induction hypothesis, Lemma 14 and $(pre^*(\mathsf{bad}) \cap \mathcal{O}){\uparrow} \subseteq pre^*(\mathsf{bad})$, we obtain

$$\gamma\left(\bigsqcup_{0\le k\le i+1} (post^\#)^k(\alpha(I))\right) \cap (pre^*(\mathsf{bad}) \cap \mathcal{O}){\uparrow} = \emptyset \ . \tag{19}$$

Since $\sqcup$ is the least upper bound operator, we have that $((\sqcup_{0\leq k\leq i+1} post^{\#})^k)\circ \alpha)(I) \sqsubseteq ((post^{\#})^* \circ \alpha)(I)$. Moreover, $(\gamma\circ((post^{\#})^*)\circ\alpha)(I) \subseteq \mathcal{O}$ by hypothesis and we obtain, following the definition of $\sqsubseteq$, that

$$\gamma\left(\bigsqcup_{0\leq k\leq i+1} (post^{\#})^k(\alpha(I))\right) \subseteq \gamma(((post^{\#})^* \circ \alpha)(I)) \subseteq \mathcal{O} \ .$$

Since $\mathcal{O}$ is a $\succeq$-dc-set, $(X \setminus \mathcal{O})\!\uparrow \cap \mathcal{O} = \emptyset$. Hence,

$$\gamma\left(\bigsqcup_{0\leq k\leq i+1} (post^{\#})^k(\alpha(I))\right) \cap (pre^*(\mathsf{bad}) \cap (X \setminus \mathcal{O}))\!\uparrow = \emptyset \ . \tag{20}$$

Notice that

$$\forall U \in UCS(X), \forall A \subset X\colon (U \cap A)\!\uparrow \cup (U \cap (X \setminus A))\!\uparrow = U \ . \tag{21}$$

We finally conclude from (21) where $U = pre^*(\mathsf{bad})$ and $A = \mathcal{O}$, (19) and (20) that

$$\gamma\left(\bigsqcup_{0\leq k\leq i+1} (post^{\#})^k(\alpha(I))\right) \cap pre^*(\mathsf{bad}) = \emptyset \ .$$

$\square$