Strong Preservation by Model Deformation

Roberto Giacobazzi*, Isabella Mastroeni*, Đurica Nikolić*†
*Dipartimento di Informatica, University of Verona
†Microsoft Research - University of Trento Centre for Systems and Computational Biology
{roberto.giacobazzi, isabella.mastroeni, durica.nikolic}@univr.it

***Abstract*—Reliable and secure system design requires an increasing number of methods, algorithms, and tools for automatic program manipulation. Any program change corresponds to a transformation that affects the semantics at some given level of abstraction. We call these techniques model deformations. In this paper we propose a mathematical foundation for completeness-driven deformations of transition systems w.r.t. a given abstraction, and we introduce an algorithm for systematic deformation of Kripke structures for inducing strong preservation in abstract model checking. We prove that our model deformations are deeply related with *must* and *may* transitions in modal transition systems.**

I. Introduction

In this paper, we exploit a recently proposed idea of transforming models of computation for guaranteeing observational precision for a fixed semantics [1]. Our idea is to control the change of a model, later called *model deformation*, in order to induce strong preservation in model checking. Strong preservation means that properties hold in a concrete system iff they hold in the corresponding abstraction. The idea is that a desirable behavior (in our case, strong preservation) can be induced by modifying a system in such a way that an abstract model checker becomes precise for the modified system. It is well known that the lack of strong preservation corresponds to the presence of spurious counterexamples (abstract traces with no concrete counterpart) to the given specification. On the other hand, it is also well-known that completeness avoids spuriousness [2], [3], [4]. As observed in [1] we can induce completeness either acting at the abstraction level (e.g., see [2] for abstract interpretation and [4] for abstract model checking) or by modifying the underlying concrete system. In this paper we induce strong preservation by modifying the underlying concrete model. The comparison between the original model and the deformed one may provide an information about behaviors that are not preserved by the abstractions and set up an innovative framework for completeness-driven model design.

We propose both a mathematical foundation and an algorithm for the systematic design of completeness-driven deformations of Kripke structures. The mathematical foundation consists of specifying how the framework for transforming semantics, proposed in [1], can be instantiated in a particular context of partitions of the set of states of a transition system. The choice of partitions simplifies the framework, since they correspond to abstractions with very specific features, i.e., they preserve additivity and complementation. On the other hand, the choice of transition systems requires the model deformation, i.e., the modified transition relation, to preserve additivity which in general may not be preserved in the transformations introduced in [1]. We instantiate our results to modal transition systems. In particular, we prove that in the deformed models *must* and *may* transitions coincide, and they also coincide with the *must* or *may* transitions of the original transition system, depending on the model deformation method we use. We then introduce `CEGMOD`, a parameterized counterexample-guided model deformation algorithm for inducing strong preservation by model deformation. `CEGMOD` provides a systematic procedure for *minimally* deforming Kripke structures in order to guarantee strong preservation of safety properties.

II. Background

Basic notions. Let Σ be a (possibly infinite) set of states. A partition $\mathtt{P}$ of Σ is a set of non-empty subsets of Σ called *blocks*, that are pairwise disjoint and whose union is Σ. Let $\textsc{Part}(\Sigma)$ denote the set of partitions of Σ, then $\langle \textsc{Part}(\Sigma), \preccurlyeq \rangle$ is a complete lattice, where the partial order $\preccurlyeq$ (finer than) is defined as: $\mathtt{P}_1 \preccurlyeq \mathtt{P}_2$ iff $\forall B \in \mathtt{P}_1 . \exists B' \in \mathtt{P}_2 . B \subseteq B'$. For the sake of simplicity, let us denote any set $\{a, b, c\}$ by $[abc]$. Let $\rightarrow \subseteq \Sigma \times \Sigma$ be a transition relation, then we define its *pre* and *post* functions on $\wp(\Sigma)$ as: $post_{\rightarrow} \stackrel{\text{def}}{=} \lambda S . \{\sigma \in \Sigma \mid \exists \sigma' \in S . \sigma' \rightarrow \sigma\}$ and $pre_{\rightarrow} \stackrel{\text{def}}{=} \lambda S . \{\sigma \in \Sigma \mid \exists \sigma' \in S . \sigma \rightarrow \sigma'\}$.

Abstract interpretation. An *upper closure operator* (*uco*) $\rho : C \rightarrow C$ on a poset C is monotone, idempotent ($\forall x \in C . \rho(x) = \rho \circ \rho(x)$), and extensive ($\forall x \in C . x \leq_c \rho(x)$) map. Every *uco* ρ is uniquely determined by the set of its fixpoints $\rho(C)$, and if it is additive, then its right adjoint $\rho^+ \stackrel{\text{def}}{=} \bigvee \{y \mid \rho(y) \leq_c x\}$ exists. Let $\mathtt{P} \in \textsc{Part}(\Sigma)$ and let $\alpha_{\mathtt{P}} \stackrel{\text{def}}{=} \lambda S . \{B \in \mathtt{P} \mid B \cap S \neq \varnothing\}$ and $\gamma_{\mathtt{P}} \stackrel{\text{def}}{=} \lambda T . \bigcup_{B \in T} B$. Then we define $\mu_{\mathtt{P}} \stackrel{\text{def}}{=} \gamma_{\mathtt{P}} \circ \alpha_{\mathtt{P}} \in uco(\wp(\Sigma))$, the additive *uco* inducing $\mathtt{P}$. The function par maps an *uco* to the induced partition: $\mathsf{par}(\mu_{\mathtt{P}}) \stackrel{\text{def}}{=} \mathtt{P}$. Precision of an abstract

interpretation typically relies upon the structure of the abstract domain [2]. Depending on where we compare the concrete and the abstract computations we obtain two different notions of completeness: if the results are compared in the abstract domain, we deal with *backward completeness* ($\mathcal{B}$); if the results are compared in the concrete domain we deal with *forward completeness* ($\mathcal{F}$) [5], [2], [3]. This paper considers only $\mathcal{F}$-completeness, and we define it formally: given a monotone function $f : C \to C$ and an *uco* $\rho \in uco(C)$, we say that ρ is $\mathcal{F}$-complete for f if $\rho \circ f \circ \rho = f \circ \rho$. The problem of making abstract domains $\mathcal{F}$-complete has been solved in [2], [3].

Model checking and Kripke structures. Temporal logic *model checking* is a technique for verifying that a system satisfies its specification by (*i*) representing the system as a Kripke structure (k.s.) $\mathcal{K}$, (*ii*) writing the specification φ in a suitable temporal logic (*CTL*, *CTL**, *ACTL*, etc.), and (*iii*) algorithmically checking that $\mathcal{K}$ is a model of φ [6], [4]. *Abstract model checking* is one of the most important techniques dealing with the well-known state explosion problem. It relies on abstract Kripke structures (abstract k.s.) which are based on partitions of the state space of the original model [7]. Given a set AP of atomic propositions (of some language), a k.s. over AP is defined by a tuple $\mathcal{K} = (\Sigma, \to, l)$, where $\to \subseteq \Sigma \times \Sigma$ is a transition relation, while $l : \Sigma \to \wp(AP)$ is a state labelling function. Moreover, Kripke structures may contain a non-empty set $I \subseteq \Sigma$ of initial states. The notation $\sigma \vDash^{\mathcal{K}} \varphi$ means that a state $\sigma \in \Sigma$ satisfies in $\mathcal{K}$ a state formula φ, while $\mathcal{K} \vDash \varphi$ means that there exists $\sigma \in I$ such that $\sigma \vDash^{\mathcal{K}} \varphi$. $\mathcal{K}$ is finitely branching when $\forall \sigma \in \Sigma$, $post_{\to}(\{\sigma\})$ is a finite set. An abstract k.s. $\mathcal{A} = (\mathtt{P}, \to^{\sharp}, l^{\sharp})$ is defined over a set of abstract states, corresponding to the blocks of a state partition $\mathtt{P} \in \textsc{Part}(\Sigma)$, chosen as an abstract domain, where $\to^{\sharp} \subseteq \mathtt{P} \times \mathtt{P}$ and $l^{\sharp} : \mathtt{P} \to \wp(AP)$ are abstract transition relation and abstract state labelling function on $\mathtt{P}$ respectively. It is worth noting that $\to^{\sharp}$ and $l^{\sharp}$ can be instantiated to $\to^{\exists\exists}$ and $l_{\mathtt{P}}$, defined as: $\forall B_1, B_2 \in \mathtt{P},\ B_1 \to^{\exists\exists} B_2$ iff $\exists c_1 \in B_1 . \exists c_2 \in B_2 . c_1 \to c_2$, and $l_{\mathtt{P}} \stackrel{\text{def}}{=} \lambda B \in \mathtt{P}. \cup_{c \in B} l(c)$.

Bisimulation equivalence. A relation $\mathtt{R} \subseteq \Sigma \times \Sigma$ is a *bisimulation* on $\mathcal{K}$ if for every $\sigma, \sigma' \in \Sigma$ such that $\sigma \,\mathtt{R}\, \sigma'$: (*i*) $l(\sigma) = l(\sigma')$; (*ii*) for any $\omega \in \Sigma$ such that $\sigma \to \omega$, there exists $\omega' \in \Sigma$ such that $\sigma' \to \omega'$ and $\omega \,\mathtt{R}\, \omega'$; (*iii*) $\sigma' \mathtt{R} \sigma$. There exists the largest bisimulation relation, it is an equivalence relation (called *bisimulation equivalence*) and $\mathtt{P}^{\mathcal{K}}_{\text{Bis}} \in \textsc{Part}(\Sigma)$ denotes the partition corresponding to this equivalence relation. A partition $\mathtt{P} \in \textsc{Part}(\Sigma)$ *induces* a bisimulation on $\mathcal{K}$ if $\mathtt{P} \preccurlyeq \mathtt{P}^{\mathcal{K}}_{\text{Bis}}$.

Weak and strong preservation. Given a specification language $\mathscr{L}$ of *state formulae*, a k.s. $\mathcal{K} = (\Sigma, \to, l)$ and its abstraction $\mathcal{A}$ induced by $\mathtt{P} \in \textsc{Part}(\Sigma)$, we say that $\mathcal{A}$ *weakly preserves* $\mathscr{L}$ if for any formula $\varphi \in \mathscr{L}$ holding on $\mathcal{A}$, φ also holds on $\mathcal{K}$, i.e., $\forall \varphi \in \mathscr{L} . \forall B \in \mathtt{P} . \forall \sigma \in \gamma_{\mathtt{P}}(B) . B \vDash^{\mathcal{A}} \varphi \Rightarrow \sigma \vDash^{\mathcal{K}} \varphi$. If also the converse holds, i.e., if $\forall \varphi \in \mathscr{L} . \forall B \in \mathtt{P} . \forall \sigma \in \gamma_{\mathtt{P}}(B) . B \vDash^{\mathcal{A}} \varphi \Leftrightarrow \sigma \vDash^{\mathcal{K}} \varphi$, we say that $\mathcal{A}$ *strongly preserves* $\mathscr{L}$. We define: $\mathtt{P}_l \stackrel{\text{def}}{=} \{[\sigma]_l \mid \sigma \in \Sigma\} \in \textsc{Part}(\Sigma)$, where for any $\sigma \in \Sigma$, $[\sigma]_l \stackrel{\text{def}}{=} \{\sigma' \in \Sigma \mid l(\sigma) = l(\sigma')\}$. It is well-known [7], [4], [6] that if $\mathtt{P} \in \textsc{Part}(\Sigma)$ satisfies $\mathtt{P} \preccurlyeq \mathtt{P}_l$, then we call it *appropriate* and the abstract k.s. $(\mathtt{P}, \to^{\exists\exists}, l_{\mathtt{P}})$ is strong preserving for $ACTL^*$. Moreover, if $\mathtt{P} = \mathtt{P}^{\mathcal{K}}_{\text{Bis}}$, then $(\mathtt{P}, \to^{\exists\exists}, l_{\mathtt{P}})$ is strong preserving for CTL^*.

Bisimulation vs. Completeness. Ranzato et al. [9] deal with generic temporal languages and they study relations between $\mathcal{F}$-completeness and strong preservation of state partitions for different temporal languages. In particular, it turns out that given a k.s. $\mathcal{K} = (\Sigma, \to, l)$ and an appropriate state partition $\mathtt{P} \in \textsc{Part}(\Sigma)$, we have that $\mathtt{P}$ induces a bisimulation on $\mathcal{K}$ iff $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete for $pre_{\to}$. A direct consequence of this result is that $\mathcal{F}$-completeness of μ_P for $pre_{\to}$ implies that the abstract k.s. $\mathcal{A} = (\mathtt{P}, \to^{\exists\exists}, l_{\mathtt{P}})$ is strongly preserving for *CTL*. We use $\mathtt{P} \leftrightarrow \mathcal{K}$ to denote that $\mathtt{P}$ is a bisimulation on $\mathcal{K}$.

III. Completeness-Driven Model Deformation

A model deformation is any operation modifying models, e.g., the semantics of a programming language. In this section we propose a mathematical foundation and a methodology for the systematic design of k.s. deformations. In particular, we fix a desired behavior of an abstraction of the original system, and we modify the concrete system making it satisfy the desired constraints.

We now turn our attention to *CTL* only and we introduce our general idea: suppose we are given a k.s. $\mathcal{K} = (\Sigma, \to, l)$ and an appropriate state partition $\mathtt{P} \preccurlyeq \mathtt{P}_l \in \textsc{Part}(\Sigma)$ and that we want $\mathtt{P} \leftrightarrow \mathcal{K}$ to hold. First of all, we check whether this request is already satisfied, and we can do it by controlling whether $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete for $pre_{\to}$ [9]. If the answer is positive, we can define an abstract k.s. which is strong preserving for *CTL*. But what happens if it is not the case? Then our task is to induce $\mathcal{F}$-completeness of $\mu_{\mathtt{P}}$ for $pre_{\to}$. There exist several approaches to this problem, and some well-known solutions are refinements of $\mu_{\mathtt{P}}$ [4], [3], [2] or simplifications of $\mu_{\mathtt{P}}$ [2], [10]. Our approach is a little bit different, and it follows the general idea of [1]: we show how we can modify $\mathcal{K}$, obtaining $\mathcal{K}'$, such that $\mathtt{P} \leftrightarrow \mathcal{K}'$, i.e., we change the relation $\to$ of $\mathcal{K}$ in such a way that $\mu_{\mathtt{P}}$ becomes $\mathcal{F}$-complete for $pre_{\to}$. We can move *upwards* or *downwards*, meaning that our resulting transition relation can be greater ($\nearrow_{\mathtt{P}} \supseteq \to$) or smaller ($\searrow_{\mathtt{P}} \subseteq \to$) than the initial one. Since the hint that guides us through these deformations is $\mathcal{F}$-completeness, we call them *completeness-driven*. Let us introduce the two approaches mentioned above.

Moving upwards. In this paragraph, we characterize a transition relation $\nearrow_{\mathtt{P}} \supseteq \to$ such that $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete

$$\mathbb{F}^{\uparrow}_{\rho}(f)(x)=\begin{cases}\rho f(x) & \text{if } x\in\rho\\ f(x) & \text{if } x\notin\rho\end{cases} \qquad \mathbb{F}^{\downarrow}_{\rho}(f)(x)=\begin{cases}\rho^{+} f(x) & \text{if } x\in\rho\\ f(x) & \text{if } x\notin\rho\end{cases}$$

Figure 1. "Up" and "Down" transformers

for $pre_{\nearrow_{\mathtt{P}}}$, although it might not be $\mathcal{F}$-complete for $pre_{\rightarrow}$. Note that, in general there exists a transformer $\mathbb{F}^{\uparrow}_{\rho}$ (see Fig. 1) that transforms a generic function $f : C \to C$ to the closest $g \sqsupseteq f$ for which $\rho \in uco(C)$ is $\mathcal{F}$-complete [1]. This operator cannot be directly applied to an additive function like $pre_{\rightarrow}$ since it loses monotonicity, but it highlights what (the $pre_{\rightarrow}$ images of the fixpoints of ρ) has to be changed in order to induce completeness and how. In order to better understand how this works we instantiate the whole problem to the specific context of partitions on transition systems. It is worth noting that $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete for a function $f : \wp(\Sigma) \to \wp(\Sigma)$ if f maps fixpoints of $\mu_{\mathtt{P}}$ into fixpoints of $\mu_{\mathtt{P}}$, i.e., (union of) blocks of $\mathtt{P}$ into (union of) blocks of $\mathtt{P}$. We want $pre_{\nearrow_{\mathtt{P}}}$ to satisfy this property and we induce it by changing all of the points in which $pre_{\rightarrow}$ falsifies it. More precisely, suppose that the $pre_{\rightarrow}$ image of a block $B \in \mathtt{P}$ contains a (possibly empty) set of complete blocks and a set of partial blocks of $\mathtt{P}$. Then we modify $pre_{\rightarrow}(B)$ by *completing its partial blocks*, which corresponds to applying $\mu_{\mathtt{P}}$ to $pre_{\rightarrow}(B)$.

Theorem 1: Let $pre_{\nearrow_{\mathtt{P}}} : \wp(\Sigma) \to \wp(\Sigma)$ be an additive function such that for any fixpoint x of $\mu_{\mathtt{P}}$, $pre_{\nearrow_{\mathtt{P}}}(x) = \mu_{\mathtt{P}} \circ pre_{\rightarrow}(x)$. Then $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete for $pre_{\nearrow_{\mathtt{P}}}$.

Proof: Recall that $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete for $pre_{\nearrow_{\mathtt{P}}}$, if the latter maps fixpoints of $\mu_{\mathtt{P}}$ to fixpoints of $\mu_{\mathtt{P}}$. For every fixpoint of $\mu_{\mathtt{P}}$, x, we have that $pre_{\nearrow_{\mathtt{P}}}(x) = \mu_{\mathtt{P}} \circ pre_{\rightarrow}(x)$, is a fixpoint of $\mu_{\mathtt{P}}$, hence the result.

Theorem 1 *determines which* states should be added to the $pre_{\rightarrow}$ images of the blocks of $\mathtt{P}$, but it *does not specify how* we can modify $\rightarrow$ in order to obtain that. We can, though, precisely characterize $\mathcal{K}_{max} = (\Sigma, \nearrow_{\mathtt{P}_{max}}, l)$, the maximal deformation of $\mathcal{K}$, such that $\mathtt{P} \leftrightarrow \mathcal{K}_{max}$ by defining $\nearrow_{\mathtt{P}_{max}} \stackrel{\text{def}}{=} \{(\sigma_1, \sigma_2) \mid \sigma_1 \in pre_{\nearrow_{\mathtt{P}}}(\mu_{\mathtt{P}}(\sigma_2))\}$. On the other hand, we cannot characterize the minimal deformation of $\mathcal{K}$, because there can be more than one such deformation. We illustrate our method by an example. In our examples, circles (squares) represent concrete (abstract) states, double-circled (double-squared) states represent concrete (abstract) initial ones, and subscripts determine the labelling function.

Example 1: Consider $\mathcal{K} = (\Sigma, \rightarrow, l)$ in Fig. 2 (do not consider the dashed arcs), an appropriate partition $\mathtt{P} = \{[1,2], [3,5], 4\}$, and the abstract k.s. induced by $\mathtt{P}$, $\mathcal{A} = (\mathtt{P}, \rightarrow^{\exists\exists}, l_{\mathtt{P}})$. We apply $pre_{\rightarrow}$ to $\mu_{\mathtt{P}}$'s atoms[1]: $pre_{\rightarrow}(\{1,2\}) = \{1,2,3\} \notin \mu_{\mathtt{P}}$, $pre_{\rightarrow}(\{3,5\}) = \{1,2,4,5\} \notin \mu_{\mathtt{P}}$ and $pre_{\rightarrow}(\{4\}) = \{5\} \notin \mu_{\mathtt{P}}$, hence $\mu_{\mathtt{P}}$ is $\mathcal{F}$-incomplete for $pre_{\rightarrow}$. An extension $\nearrow_{\mathtt{P}} \supseteq \rightarrow$ should satisfy (Th. 1): $pre_{\nearrow_{\mathtt{P}}}(\{1,2\}) = \mu_{\mathtt{P}} \circ pre_{\rightarrow}(\{1,2\}) = \{1,2,3,5\}$, $pre_{\nearrow_{\mathtt{P}}}(\{3,5\}) = \{1,2,3,4,5\}$ and $pre_{\nearrow_{\mathtt{P}}}(\{4\}) = \{3,5\}$.

[1] Atoms in boolean lattices are the elements covering the bottom.

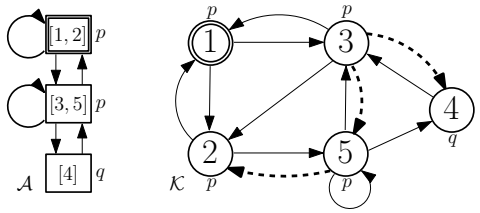


Figure 2. Example illustrating the "moving upwards" approach.

We enrich the $pre_{\rightarrow}$ image of $\{1,2\}$ with $\{5\}$, and the $pre_{\rightarrow}$ images of $\{3,5\}$ and $\{4\}$ with $\{3\}$. There are several ways of modifying $\rightarrow$: e.g., $\{5\}$ can be added to $pre_{\rightarrow}(\{1,2\})$ by enriching $\rightarrow$ with one of the following sets: $\{(5,1)\}$, $\{(5,2)\}$ or $\{(5,1),(5,2)\}$. If we set $\nearrow_{\mathtt{P}} = \rightarrow \cup \{(5,1),(5,2),(3,3),(3,5),(3,4)\}$, we induce $\mathcal{K}_{max}$. A minimal modification of $\mathcal{K}$ is obtained by enriching $\rightarrow$ with one of the sets: $\{(5,1), (3,3), (3,4)\}$, $\{(5,1),(3,5),(3,4)\}$, $\{(5,2),(3,3),(3,4)\}$ or $\{(5,2),(3,5),(3,4)\}$. The latter case is shown on the right of Fig. 2, where the dashed transitions are the added pairs, forming the resulting k.s. $\mathcal{K}' = (\Sigma, \nearrow_{\mathtt{P}}, l)$. Note that $\mathcal{A}$ is abstraction of both $\mathcal{K}'$ and $\mathcal{K}$.

The following theorem states that although the concrete k.s. is changed, the abstract one remains the same, as we noticed in the previous example.

Theorem 2: Given a state partition $\mathtt{P} \in \textsc{Part}(\Sigma)$ and Kripke structures $\mathcal{K} = (\Sigma, \rightarrow, l)$ and $\mathcal{K}' = (\Sigma, \nearrow_{\mathtt{P}}, l)$ such that $\mathtt{P} \leftrightarrow \mathcal{K}'$, it holds that $\rightarrow^{\exists\exists} = \nearrow_{\mathtt{P}}^{\exists\exists}$.

Proof: Let $B_1, B_2 \in \mathtt{P}$. Then, by definitions of $\nearrow_{\mathtt{P}}^{\exists\exists}$ and $\nearrow_{\mathtt{P}}$, $B_1 \nearrow_{\mathtt{P}}^{\exists\exists} B_2 \Leftrightarrow \exists\sigma_1 \in B_1.\exists\sigma_2 \in B_2.\sigma_1 \nearrow_{\mathtt{P}} \sigma_2 \Leftrightarrow \exists\omega_1 \in \mu_{\mathtt{P}}(\{\sigma_1\}) = B_1.\exists\omega_2 \in \mu_{\mathtt{P}}(\{\sigma_2\}) = B_2.\omega_1 \to \omega_2$, which entails $B_1 \rightarrow^{\exists\exists} B_2$.

Moving downwards. Let us now introduce the dual approach: we want to model a (possibly empty) transition relation $\searrow_{\mathtt{P}} \subseteq \rightarrow$ such that $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete for $pre_{\searrow_{\mathtt{P}}}$ although it is not $\mathcal{F}$-complete for $pre_{\rightarrow}$. In general, there exists a transformer $\mathbb{F}^{\downarrow}_{\rho}$ (Fig. 1) transforming a generic function $f : C \to C$ to the closest function $g \sqsubseteq f$ s.t. $\rho \in uco(C)$ is $\mathcal{F}$-complete [1]. Note that, in this case, the transformation is possible iff ρ is additive, which means that ρ^{+} exists. In the context of partitions on transition systems this transformation tells us that we have to transform $\rightarrow$ in order to make it satisfy the constraints fixed by transformer $\mathbb{F}^{\downarrow}$ for $pre_{\rightarrow}$ and $\mu_{\mathtt{P}}$: suppose that the $pre_{\rightarrow}$ image of a block $B \in \mathtt{P}$ contains a (possibly empty) set of complete blocks and a set of partial blocks of $\mathtt{P}$. Then we modify $pre_{\rightarrow}(B)$ by eliminating all the partial blocks from it. Since only complete blocks included in the initial value of $pre_{\rightarrow}(B)$ are kept, this deformation corresponds to $\mu^{+}_{\mathtt{P}}$ which, in this context, is determined as $\mu^{+}_{\mathtt{P}} = \lambda S.\{B \in \mathtt{P} \mid B \subseteq S\}$.

Theorem 3: Let $pre_{\searrow_{\mathtt{P}}} : \wp(\Sigma) \to \wp(\Sigma)$ be an additive function such that for every $\mu_{\mathtt{P}}$'s fixpoint x, $pre_{\searrow_{\mathtt{P}}}(x) = \mu^{+}_{\mathtt{P}} \circ pre_{\rightarrow}(x)$ holds. Then, $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete for $pre_{\searrow_{\mathtt{P}}}$.

Proof: $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete for $pre_{\searrow_{\mathtt{P}}}$, if the latter maps fixpoints of $\mu_{\mathtt{P}}$ to fixpoints of $\mu_{\mathtt{P}}$. For every $\mu_{\mathtt{P}}$'s fixpoint

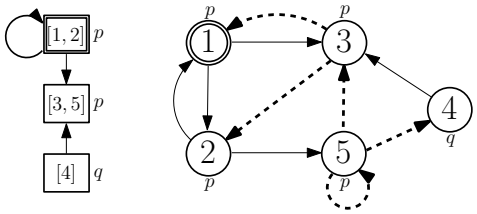


Figure 3. Example illustrating the "moving downwards" approach

x, $pre_{\searrow_{\mathtt{P}}}(x) = \mu_{\mathtt{P}}^{+} \circ pre_{\rightarrow}(x)$, which is a fixpoint of both $\mu_{\mathtt{P}}^{+}$ and $\mu_{\mathtt{P}}$, since they have the same fixpoints [1].

Theorem 3 *determines which* states should be removed from the $pre_{\rightarrow}$ images of the blocks of $\mathtt{P}$, but it *does not specify how* we modify $\rightarrow$ in order to do that. However, we can characterize both $\mathcal{K}_{max}$ and $\mathcal{K}_{min}$, i.e., maximal and minimal modification of $\mathcal{K}$ respectively: ${\searrow_{\mathtt{P}}}_{max} = \varnothing$ and ${\searrow_{\mathtt{P}}}_{min} = \{(\sigma_1, \sigma_2) \mid \sigma_1 \rightarrow \sigma_2 \wedge \sigma_1 \in pre_{\searrow_{\mathtt{P}}}(\mu_{\mathtt{P}}(\sigma_2))\}$.

Example 2: Consider $\mathcal{K}$ and $\mathtt{P}$ from Ex. 1. We recall that $\mu_{\mathtt{P}}$ is not $\mathcal{F}$-complete for $pre_{\rightarrow}$. A reduction $\searrow_{\mathtt{P}} \subseteq \rightarrow$ should satisfy (Th. 3): $pre_{\searrow_{\mathtt{P}}}(\{1,2\}) = \mu_{\mathtt{P}}^{+} \circ pre_{\rightarrow}(\{1,2\}) = \{1,2\}$, $pre_{\searrow_{\mathtt{P}}}(\{3,5\}) = \{1,2,4\}$ and $pre_{\searrow_{\mathtt{P}}}(\{4\}) = \varnothing$. Hence we should remove $\{3\}$from $pre_{\rightarrow}(\{1,2\})$ of $\{3\}$, while $pre_{\rightarrow}(\{3,5\})$ and $pre_{\rightarrow}(\{4\})$ should be deprived of $\{5\}$. We determine ${\searrow_{\mathtt{P}}}_{min} = \{(1,2),(1,3),(2,1),(2,5),(4,3)\}$. $\mathcal{K}_{min}$ is given on the right of Fig. 3 (the bold dashed arcs represent the pairs that we removed), and on the left we give the corresponding abstract k.s. $\mathcal{A}_{min} = (\mathtt{P}, {\searrow_{\mathtt{P}}}^{\exists\exists}_{min}, l_{\mathtt{P}})$, which is strong preserving for $\mu_{\mathtt{P}}$. Note that ${\searrow_{\mathtt{P}}}^{\exists\exists}_{min} \subset {\rightarrow}^{\exists\exists}$.

In this case the induced abstract k.s. changes, and we formalize this fact.

Theorem 4: Given a state partition $\mathtt{P} \in \textsc{Part}(\Sigma)$ and Kripke structures $\mathcal{K} = (\Sigma, \rightarrow, l)$ and $\mathcal{K}' = (\Sigma, {\searrow_{\mathtt{P}}}_{min}, l)$ such that $\mathtt{P} \leftrightarrow \mathcal{K}'$, it holds that ${\searrow_{\mathtt{P}}}^{\exists\exists}_{min} \subseteq {\rightarrow}^{\exists\exists}$.

Proof: If $B_1, B_2 \in \mathtt{P}$ then, by definitions of ${\nearrow_{\mathtt{P}}}^{\exists\exists}_{min}$ and ${\searrow_{\mathtt{P}}}_{min}$, we have $B_1 {\searrow_{\mathtt{P}}}^{\exists\exists}_{min} B_2 \Leftrightarrow \exists \sigma_1 \in B_1, \sigma_2 \in B_2 . \sigma_1 {\searrow_{\mathtt{P}}}_{min} \sigma_2 \Leftrightarrow \exists \sigma_1 \in B_1, \sigma_2 \in B_2 . \sigma_1 \rightarrow \sigma_2 \wedge \sigma_1 \in pre_{\searrow_{\mathtt{P}}}(\mu_{\mathtt{P}}(\sigma_2))$, which entails $B_1 \rightarrow^{\exists\exists} B_2$.

Correction. The main problem of the approach we have just introduced is that we can induce a (possibly empty) transition relation $\searrow_{\mathtt{P}}$ which might not be total. The majority of model checking algorithms require models defined on total relations. We propose a simple modification of the induced transition relation. Suppose that there exists a concrete state σ belonging to a block $B \in \mathtt{P}$ such that $post_{\searrow_{\mathtt{P}}}(\{\sigma\}) = \varnothing$, i.e., σ has no $\searrow_{\mathtt{P}}$ successors. We enrich $\searrow_{\mathtt{P}}$ by adding a pair (σ, σ). In order to be sure that this modification does not damage $\mathcal{F}$-completeness of $\mu_{\mathtt{P}}$ for $pre_{\searrow_{\mathtt{P}}}$, we apply the same modification to all states belonging to B. $\mathcal{F}$-completeness still holds, as we show in Theorem 5. Moreover it is easy to show that in this case the current value $pre_{\rightarrow}(B)$ is enriched with B.

Theorem 5: Let us define the *correctness relation* $\circlearrowleft^{\mathtt{P}} \subseteq \Sigma \times \Sigma$ as $\circlearrowleft^{\mathtt{P}} \stackrel{\text{def}}{=} \searrow_{\mathtt{P}} \cup \{(\sigma_1, \sigma_1) \mid \exists \sigma . post_{\searrow_{\mathtt{P}}}(\{\sigma\}) = \varnothing \wedge \sigma_1 \in \mu_{\mathtt{P}}(\{\sigma\})\}$. Then, $\mu_{\mathtt{P}}$ $\searrow_{\mathtt{P}}$ $\mathcal{F}$-complete for $pre_{\circlearrowleft^{\mathtt{P}}}$.

Proof: Let $B \in \mathtt{P}$ and $\sigma \in B$. If we enrich $\searrow_{\mathtt{P}}$ with $\langle \sigma, \sigma \rangle$, then we add $\langle \sigma_1, \sigma_1 \rangle$ for every $\sigma_1 \in \mu_{\mathtt{P}}(\{\sigma\})$, hence $pre_{\circlearrowleft^{\mathtt{P}}}(B) = pre_{\searrow_{\mathtt{P}}}(B) \cup B$, which is a fixpoint of $\mu_{\mathtt{P}}$.

Example 3: Consider the resulting model $\mathcal{K}_{min}$ of Ex. 2 presented on the right of Fig. 3. We can notice that states 3 and 5 do not have any $post_{\searrow_{\mathtt{P}}}$ successors. Therefore we induce $\circlearrowleft^{\mathtt{P}}$ by enriching $\searrow_{\mathtt{P}}$ with $\{(3,3),(5,5)\}$.

May-Must transitions. While developed independently, and from a different perspective, abstract Kripke structures bear some resemblance to the modal transition systems (MTS) [11], which contain two types of transition relations - *may* ($\rightarrow^{\exists\exists}$) and *must* ($\rightarrow^{\forall\exists}$). MTSs have been developed in the area of specification where *must* transitions specify what is required while *may* transitions specify what is admissible (they precisely correspond to the transitions $\rightarrow^{\exists\exists}$ that we have already used above). We define $\rightarrow^{\exists\exists}$ and $\rightarrow^{\forall\exists}$ transitions in our context of interest, i.e., partitions on transition systems.

Definition 1: Consider a k.s. $\mathcal{K} = (\Sigma, \rightarrow, l)$ and a partition $\mathtt{P} \in \textsc{Part}(\Sigma)$. We define an abstract modal k.s. as $\mathcal{M} \stackrel{\text{def}}{=} \langle \mathtt{P}, \rightarrow^{\exists\exists}, \rightarrow^{\forall\exists} \rangle$, where $\forall B_1, B_2 \in \mathtt{P}$, (i) $B_1 \rightarrow^{\exists\exists} B_2$ iff $\exists \sigma_1 \in B_1 . \exists \sigma_2 \in B_2 . \sigma_1 \rightarrow \sigma_2$; (ii) $B_1 \rightarrow^{\forall\exists} B_2$ iff $\forall \sigma_1 \in B_1 . \exists \sigma_2 \in B_2 . \sigma_1 \rightarrow \sigma_2$.

It turns out that $\mathcal{F}$-completeness of state partitions on transition systems implies equivalence of *must* and *may* transitions, as we show below.

Theorem 6: Consider a k.s. $\mathcal{K} = (\Sigma, \rightarrow, l)$ and a state partition $\mathtt{P} \in \textsc{Part}(\Sigma)$. If $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete for $pre_{\rightarrow}$, then $\rightarrow^{\exists\exists} = \rightarrow^{\forall\exists}$.

Proof: Since every *must* transition is also a *may* transition, we should show that under hypotheses of the theorem, every *may* transition is a *must* transition. Let $\mathtt{P} = \{B_1, \ldots, B_n\}$. By a slight abuse of notation we denote atoms of $\mu_{\mathtt{P}}$ by blocks of $\mathtt{P}$, i.e., we use B_i to denote $\gamma_{\mathtt{P}}(B_i)$, $\forall i \in [1..n]$. Suppose $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete for $pre_{\rightarrow}$, i.e., for any $\sigma \in \Sigma$. $pre_{\rightarrow} \circ \mu_{\mathtt{P}}(\{\sigma\}) = \mu_{\mathtt{P}} \circ pre_{\rightarrow} \circ \mu_{\mathtt{P}}(\{\sigma\})$. It means that the $pre_{\rightarrow}$ image of a fixpoint of $\mu_{\mathtt{P}}$ is a fixpoint of $\mu_{\mathtt{P}}$. Let us fix $\sigma \in \Sigma$, then it is clear that there exist $i_1, \ldots, i_m \in [1..n]$ such that $pre_{\rightarrow} \circ \mu_{\mathtt{P}}(\{x\}) = \cup_{k \in \{i_1, \ldots, i_m\}} B_k$. Suppose there exists $B_k, B_j \in \mathtt{P}$ such that $B_k \rightarrow^{\exists\exists} B_j$, i.e., $\exists \sigma \in B_k, \omega \in B_j . \sigma \rightarrow \omega$. Then, for every concrete state $\sigma_1 \in B_k$ there exists $\omega_1 \in B_j$ such that $\sigma_1 \rightarrow \omega_1$. Otherwise, there exists σ_1 such that $post_{\rightarrow}(\{\sigma_1\}) \cap B_j = \varnothing$ and therefore $\sigma_1 \notin pre_{\rightarrow}(B_j)$. On the other hand, $\sigma \in pre_{\rightarrow}(B_j)$, and therefore there exists a block of $\mathtt{P}$ (B_k) partially included in the $pre_{\rightarrow}$ image of B_j. This entails $\mathcal{F}$-incompleteness of $\mu_{\mathtt{P}}$, which contradicts assumptions of the theorem. Thus, $B_k \rightarrow^{\exists\exists} B_j \Rightarrow \forall \sigma \in B_k . \exists \omega \in B_j . \sigma \rightarrow \omega \Leftrightarrow B_k \rightarrow^{\forall\exists} B_j$.

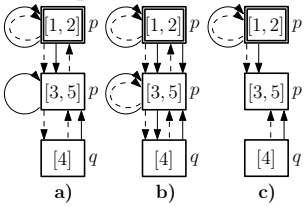
Note that, in general, every *must* is also a *may* transition, but not vice versa. Let $\neg$*must* be the *may* transitions which are not *must*. It turns out that all $\neg$*must* transitions of $\mathcal{K}$ become *must* in the system induced by $\nearrow_{\mathtt{P}}$, while they disappear from the system induced by $\searrow_{\mathtt{P}}$.

Theorem 7: Consider a k.s. $\mathcal{K} = (\Sigma, \rightarrow, l)$, a state partition $\mathtt{P} \in \textsc{Part}(\Sigma)$, and two induced relations $\nearrow_{\mathtt{P}} \supseteq \rightarrow$ and $\searrow_{\mathtt{P}} \subseteq \rightarrow$. We show that the following relations hold: ${\nearrow_{\mathtt{P}}}^{\exists\exists} = {\nearrow_{\mathtt{P}}}^{\forall\exists} = {\rightarrow}^{\exists\exists}$ and ${\searrow_{\mathtt{P}}}^{\exists\exists} = {\searrow_{\mathtt{P}}}^{\forall\exists} = {\rightarrow}^{\forall\exists}$.

Proof: By Th. 2 we have that ${\nearrow_{\mathtt{P}}}^{\exists\exists} = {\rightarrow}^{\exists\exists}$. Since $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete for $pre_{\nearrow_{\mathtt{P}}}$ (Th. 1), we conclude, by Th. 6, that ${\nearrow_{\mathtt{P}}}^{\exists\exists} = {\nearrow_{\mathtt{P}}}^{\forall\exists}$. Thus, ${\nearrow_{\mathtt{P}}}^{\exists\exists} = {\nearrow_{\mathtt{P}}}^{\forall\exists} = {\rightarrow}^{\exists\exists}$.

Consider blocks $B_k, B_j \in \mathtt{P}$. Then, $B_k {\searrow_{\mathtt{P}}}^{\exists\exists} B_j$ iff $\exists \sigma_k \in B_k . \exists \sigma_j \in B_j . \sigma_k \searrow_{\mathtt{P}} \sigma_j$, and by definition of $\searrow_{\mathtt{P}}$ it is possible iff $\forall \sigma_k \in B_k . \exists \sigma_j \in B_j . \sigma_k \rightarrow \sigma_j$ iff $B_k \rightarrow^{\forall\exists} B_j$. Hence, ${\searrow_{\mathtt{P}}}^{\exists\exists} = {\rightarrow}^{\forall\exists}$. On the other hand, since $\searrow_{\mathtt{P}}$ is such that $\mu_{\mathtt{P}}$ is $\mathcal{F}$-complete for $pre_{\searrow_{\mathtt{P}}}$, we have by Theorem 6 that ${\searrow_{\mathtt{P}}}^{\exists\exists} = {\searrow_{\mathtt{P}}}^{\forall\exists}$. Therefore, ${\searrow_{\mathtt{P}}}^{\exists\exists} = {\searrow_{\mathtt{P}}}^{\forall\exists} = {\rightarrow}^{\forall\exists}$.

The following figure shows three abstract modal k.s.: dashed and solid arcs represent *may* and *must* transitions respectively. System *a*) corresponds to the structures considered in Ex. 1 and 2, while *b*) and *c*) correspond to the structures induced in these examples.



It is worth noting that all $\neg$*must* transitions of system *a*), namely $[3,5] \rightarrow [1,2]$, $[3,5] \rightarrow [3,5]$, $[3,5] \rightarrow [4]$ become *must* in *b*) and disappear from *c*).

IV. CounterExample-Guided MOdel Deformation

In this section we introduce the **C**ounter**E**xample-**G**uided **MO**del **D**eformation (CEGMOD) algorithm as an implementation of completeness-driven model deformation. If we consider the problem of abstract model checking, it is well-known that given a concrete and an abstract k.s. $\mathcal{K}$ and $\mathcal{A}$, related by an appropriate state partition $\mathtt{P} \in \textsc{Part}(\Sigma)$ and a specification $\varphi \in ACTL^*$, if φ holds in $\mathcal{A}$, then it holds in $\mathcal{K}$ as well, i.e., $\mathcal{A} \models \varphi \Rightarrow \mathcal{K} \models \varphi$ [7], [4]. The converse does not hold in general, i.e., strong preservation is not guaranteed. It means that it is possible to generate an abstract *counterexample* (CE), namely an abstract path falsifying φ, which has no concrete counterpart. In this case the CE is called *spurious*. Existence of spurious CEs damages strong preservation, and therefore we aim to induce $\mathcal{K}'$, a deformation of $\mathcal{K}$, whose abstraction $\mathcal{A}'$ does not give rise to any spurious CEs to φ, namely, such that $\mathcal{A}' \models \varphi \Leftrightarrow \mathcal{K}' \models \varphi$.

If we focus on *safety properties* only, it turns out that our method guarantees that the resulting model is strong preserving for a given specification. We recall that the safety properties stipulate that *bad things do not happen* during execution [12]. We deal with the safety properties of the following form: $\mathrm{AG}\psi$, where ψ is a propositional formula, i.e., a formula not containing any temporal combinators. Let $\varphi = \mathrm{AG}\psi$, then a CE to φ is a finite loop-free path $\langle s_1, \ldots, s_n \rangle$, where $\forall i \in [1..n]$ s_i is a (concrete or abstract) state of the system that is being verified, and it holds that $s_i \not\models \varphi \Leftrightarrow i = n$. It is worth noting that some of the most well-known model checkers (e.g., SLAM, BLAST) deal with safety properties only. We now introduce the CEGMOD algorithm.

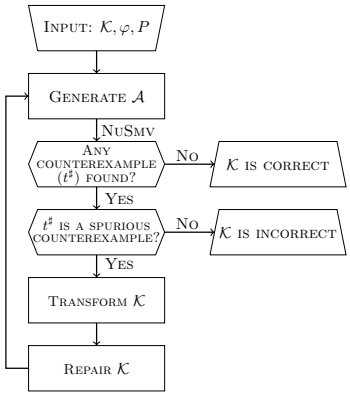


Figure 4.

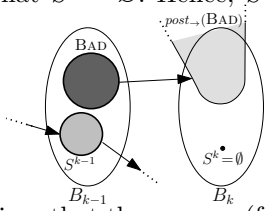
General idea. We illustrate the general idea of CEGMOD by a flowchart depicted in Fig 4, and then we give a detailed explanation of each step. Suppose we are given a k.s. $\mathcal{K} = (\Sigma, \rightarrow, l)$ and an appropriate partition P. We want to understand whether $\mathcal{K}$ satisfies a safety specification φ. In every iteration CEGMOD generates $\mathcal{A} = (\mathtt{P}, \rightarrow^{\exists\exists}, l_{\mathtt{P}})$, an abstraction of the current model $\mathcal{K}$, and we pass it, together with φ to a model checker (e.g., NuSMV), which tries to find an abstract CE to φ. If there is none, CEGMOD terminates stating that $\mathcal{K}$ is correct, and this is justified by the fact that P is appropriate [4]. Otherwise, an abstract CE $t^\sharp$ is found, and we check whether it is spurious. If the answer is negative, CEGMOD terminates stating that $\mathcal{K}$ is incorrect. Otherwise, we characterize $\rightarrow_1$, a deformation of $\rightarrow$, and determine $\mathcal{K}_1 = (\Sigma, \rightarrow_1, l)$ whose abstraction $\mathcal{A}_1$ does not give rise to $t^\sharp$, i.e., the model checker applied to $\mathcal{A}_1$ does not report $t^\sharp$ as a CE The induced transition relation $\rightarrow_1$ might not be total, and in that case our algorithm enriches it in order to obtain a relation satisfying both requirements: it is total and it does not give rise to $t^\sharp$. This relation is used in the next iteration of CEGMOD. Let us explain some of its steps in detail.

Spurious conterexample. Suppose that a model checker analyzes an abstract model and it detects a CE $t^\sharp$. Since we deal with safety properties only, we are sure that $t^\sharp$ is represented by a (finite) sequence of abstract states (blocks of P) without loops. Let $\varphi = \mathrm{AG}\psi$ and suppose that $t^\sharp = \langle B_1, \ldots, B_n \rangle$, where $\forall i \in [1..n] . B_i \in \mathtt{P}$. Recall that $B_i \not\models \psi$ iff $i = n$, since $t^\sharp$ represents a CE to particular kind of safety specifications we are interested in. By a slight abuse of notation we denote atoms of $\mu_{\mathtt{P}}$ by blocks of P, i.e., we use B_i to denote $\gamma_{\mathtt{P}}(B_i)$, $\forall i \in [1..n]$. In order to check whether $t^\sharp$ is spurious, we follow the idea of Clarke et al.[4] (SplitPATH): starting from the initial states I, they follow the (abstract) path $t^\sharp$ step by step, and try to detect concrete steps corresponding to each abstract one. At the i^{th} step they determine the set of reachable concrete states belonging to B_i. If at a certain point we can still make an abstract step, but the current set of reachable concrete states is empty,

it means that $t^\sharp$ is spurious. More precisely, given the set of initial states, I, we define $S^1 \stackrel{\text{def}}{=} I \cap B_1$ and for each $i \in [2..n]$, $S^{i+1} \stackrel{\text{def}}{=} post_{\rightarrow}(S^i) \cap B_{i+1}$. It is known [4, Lemma 4.10] that $t^\sharp$ is spurious iff there exists an $i \in [1..n]$ such that $S^i = \varnothing$. For the rest of this section we suppose that $t^\sharp$ is spurious, and that `SplitPATH` returns index k, representing the smallest index such that $S^k = \varnothing$.

Transforming model. In this paragraph we explain our strategy for removing spurious counterexamples. The `SplitPATH` method determines the exact point of $t^\sharp$ in which spuriousness occurs, i.e., B_k. It is known [3] that B_k represents a point of $\mathcal{F}$-incompleteness for $\mu_{\mathtt{P}}$ and $pre_{\rightarrow}$, namely $\mu_{\mathtt{P}} \circ pre_{\rightarrow}(B_k) \neq pre_{\rightarrow}(B_k)$.

The figure on the left represents a graphical explanation of this fact. $\varnothing \neq S^{k-1} \subset B_{k-1}$ is the set of states of B_{k-1} which can be reached from I following $t^\sharp$, but from which we cannot go any further (following $t^\sharp$), since we know that $S^k = \varnothing$. Hence, $S^{k-1} \cap pre_{\rightarrow}(B_k) = \varnothing$. We define $\textsc{Bad} \stackrel{\text{def}}{=} pre_{\rightarrow}(B_k) \cap B_{k-1} \neq \varnothing$, a set containing all concrete states of B_{k-1} which can reach B_k via $\rightarrow$, and we call them *bad states*, because they made us believe that there was a (feasible) transition between B_{k-1} and B_k. The definition of Bad guarantees that there exists at least a block of `P` partially contained in the $pre_{\rightarrow}$ image of B_k, i.e., B_{k-1}, and we have $\mathcal{F}$-incompleteness.



We aim to modify $\rightarrow$ in order to induce a k.s. whose abstraction does not give rise to $t^\sharp$. Since spuriousness of $t^\sharp$ damages $\mathcal{F}$-completeness, one possible solution could be to modify $\rightarrow$ in order to make $\mu_{\mathtt{P}}$ $\mathcal{F}$-complete for $pre_{\rightarrow}$, i.e., to generate $\nearrow$ or $\searrow$. Unfortunately, this deformation could add/remove too many transitions, which do not concern $t^\sharp$. At this point, since we want our transformation to be counterexample-guided, we propose to apply the *downwards* method (Section III) only to the first source of incompleteness met in the CE $t^\sharp$, i.e., the block B_k. This way, we erase the CE since we "break" the abstract path. Hence, the deformation we propose is: $\searrow_{t^\sharp} \; = \rightarrow \setminus \{(\sigma, \omega) \mid \sigma \in \textsc{Bad} \wedge \omega \in B^k\}$, namely, we remove all transitions from the states in Bad to those in B_k. This way we are sure that the corresponding abstract k.s. does not contain the transition $B_{k-1} \searrow^{\exists\exists} B_k$, and therefore $t^\sharp$ could not be captured by the model checker.

Repairing model. The deformation we have just proposed might introduce an undesired effect: the induced relation $\searrow_{t^\sharp}$ may not be total. Since `CEGMOD` removes only transitions from the states in $\textsc{Bad} \subset B_{k-1}$ to the states in B_k, it is clear that potential states without any outgoing transition have to be in Bad. In Section III we dealt with this problem, and we introduced the correction relation $\circlearrowleft^{\mathtt{P}}$. Theorem 5 states that given a *non-total* relation $\searrow_{t^\sharp}$, the correction relation makes $\searrow_{t^\sharp}$ total and preserves its $\mathcal{F}$-completeness. In this particular case, the correction relation becomes: $\circlearrowleft_{t^\sharp} \stackrel{\text{def}}{=} \searrow_{t^\sharp} \cup C$, where $C = \{(\sigma, \sigma) \mid \sigma \in B_{k-1}\}$ if there exists $\sigma \in B_{k-1}$ s.t. $post_{\searrow_{t^\sharp}}(\{\sigma\}) = \varnothing\}$, and $C = \varnothing$ otherwise.

Complexity. The total worst-time complexity of `CEGMOD` depends on the model structure, complexity of the model checker, and the order in which it detects CEs. In this paragraph we determine the worst-time complexity of each iteration of `CEGMOD`. Suppose that the model checker detects an abstract CE $t^\sharp = \langle B_1, \ldots, B_n \rangle$, that we use a set implementation that permits us to perform the intersection in linear time and that *post* and *pre* have time complexity c_p. Then `SplitPATH` runs in $O(|\Sigma| \times c_p + |\Sigma|) = O(|\Sigma|)$ time, the transforming model phase works in $O(|B_k| \times c_p + \max\{B_{k-1}, B_k\} + |\textsc{Bad}|) = O(|\Sigma|)$ time and the repairing model phase works in $O(|B_{k-1}| \times c_p) = O(|\Sigma|)$ time. Thus, the worst-time complexity of each iteration is $O(|\Sigma|)$, i.e., linear.

Correctness. At this point an iteration of `CEGMOD` is completed, and we construct new concrete and abstract k.s. using relations $\circlearrowleft_{t^\sharp}$ and $\circlearrowleft^{\exists\exists}_{t^\sharp}$ respectively. The next iteration begins with model checking of the abstract k.s., and then we repeat all the steps explained above. Theorem 8 proves the correctness of `CEGMOD`, and Ex. 4 illustrates one of its iterations.

Theorem 8: Let $\mathcal{A}_i$ be the abstract structure considered in the i^{th} iteration of `CEGMOD`, $(i \in \mathbb{N})$, and suppose that $t^\sharp$ is a spurious counterexample returned by a model checker applied to $\mathcal{A}_i$. The following statements hold:

1) for any $j > i$, model checking of $\mathcal{A}_j$ does not report $t^\sharp$ as a counterexample;
2) the i^{th} iteration of `CEGMOD` does not add any new spurious abstract paths;
3) the i^{th} iteration of `CEGMOD` does not introduce any new real counterexamples.

Proof: Let $\varphi = \text{AG}\psi$ be a safety specification we want to verify, where ψ does not contain any temporal combinators, and suppose $t^\sharp = \langle B_1, \ldots, B_n \rangle$, where $\forall i \in [1..n]. B_i \in \mathtt{P}$. Recall that $B_i \not\models \psi$ iff $i = n$, since $t^\sharp$ represents a CE to particular kind of safety specifications we are interested in. Moreover, suppose that in the i^{th} iteration concrete and abstract models have the following forms: $\mathcal{K}_i = (\Sigma, \rightarrow, l)$ and $\mathcal{A}_i = (\mathtt{P}, \rightarrow^{\exists\exists}, l_{\mathtt{P}})$.

1) Since $t^\sharp$ is spurious, the `SplitPATH` method returns an index $k \in [2, n)$, representing the smallest index such that $S^k = \varnothing$. Recall that $S^{k-1} \subset B_{k-1}$ represents the set of states of B_{k-1} which can be reached from the set of initial states, I, following $t^\sharp$, but from which we cannot go any further (following $t^\sharp$) - since $S^k = \varnothing$. We define $\textsc{Bad} \subset B_{k-1}$, containing all the concrete states of B_{k-1} which can reach B_k via $\rightarrow$, and we call them *bad states*,

because they made us believe that there was a (feasible) transition between B_{k-1} and B_k. CEGMOD modifies $\rightarrow$ in the following way: $\searrow_{t^\sharp} = \rightarrow \setminus \{(\sigma,\omega) | \sigma \in \text{BAD} \wedge \omega \in B^k\}$, and then it repairs $\searrow_{t^\sharp}$ if it is necessary: $\circlearrowleft_{t^\sharp} \stackrel{\text{def}}{=} \searrow_{t^\sharp} \cup \{(\sigma,\sigma) \mid \sigma \in B_{k-1} \wedge \exists \sigma \in B_{k-1}.post_{\searrow_{t^\sharp}}(\{\sigma\}) = \varnothing\}$, namely, we remove all the transitions from states in $\text{BAD} \subset B_{k-1}$ to states in B_k, and if there is any state in B_{k-1} whose $post_{\searrow_{t^\sharp}}$ is $\varnothing$, the relation is "corrected". This way we are sure that the corresponding abstract k.s. does not contain transition $B_{k-1} \circlearrowleft_{t^\sharp}^{\exists\exists} B_k$, since there is no concrete state in $B_{k-1} \setminus \text{BAD}$ reaching B_k via $\rightarrow$, and any potential correction would introduce only this abstract transition: $B_{k-1} \circlearrowleft_{t^\sharp}^{\exists\exists} B_{k-1}$. We now generate $\mathcal{K}_{i+1}$ and $\mathcal{A}_{i+1}$ using relations $\circlearrowleft_{t^\sharp}$ and $\circlearrowleft_{t^\sharp}^{\exists\exists}$. It is obvious that any abstract path of $\mathcal{A}_{i+1}$ cannot contain B_{k-1} and B_k as two adjacent states (in this order), and therefore any abstract CE obtained by model checking $\mathcal{A}_{i+1}$ cannot contain it (since abstract counterexamples represent particular paths of abstract systems that are being checked). It implies that in the $(i+1)^{th}$ iteration we cannot obtain $t^\sharp$ as an abstract CE. Moreover, since CEGMOD only removes abstract transitions, and does not add them (except from an abstract state to itself), it is not possible to obtain any abstract transition relation $\circlearrowleft'^{\exists\exists}_{t^\sharp}$ such that $B_{k-1} \circlearrowleft'^{\exists\exists}_{t^\sharp} B_k$. Therefore, for any $j > i$, $t^\sharp$ will not appear either as an abstract path or as an abstract CE of $\mathcal{A}_j$.

2) Recall that CEGMOD modifies $\rightarrow$ in the following way:

$$\searrow_{t^\sharp} = \rightarrow \setminus \{(\sigma,\omega) \mid \sigma \in \text{BAD} \wedge \omega \in B^k\}$$
$$\circlearrowleft_{t^\sharp} \stackrel{\text{def}}{=} \searrow_{t^\sharp} \cup \{(\sigma,\sigma) \in B_{k-1}^2 | \exists \sigma \in B_{k-1}.post_{\searrow_{t^\sharp}}(\{\sigma\}) = \varnothing\}.$$

In the repairing phase, CEGMOD may add some concrete transitions, and if this phase is required, the transitions CEGMOD would add arc (σ,σ), for any $\sigma \in B_{k-1}$. In this case we would add even the abstract transition $B_{k-1} \circlearrowleft'^{\exists\exists}_{t^\sharp} B_{k-1}$, but it is not spurious, since it is a *must* transition (for every concrete state $\sigma \in B_{k-1}$, there is a concrete state (σ) in B_{k-1} such that $\sigma \circlearrowleft'_{t^\sharp} \sigma$.

3) Recall that the counterexamples of a safety specifications are (finite) paths of (abstract) states. In our case, we have that $\forall i \in [1..n).B_i \vDash \psi$, and $B_n \nvDash \psi$. The concrete transitions that may be added during the i^{th} iteration are (σ,σ), for all $\sigma \in B_{k-1}$. It means that we add a may transition $B_{k-1} \circlearrowleft'^{\exists\exists}_{t^\sharp} B_{k-1}$, and it cannot damage a verification of φ since this transition forms a loop (counterexamples to the particular kind of safety specifications we are interested in do not contain loops), and since it goes from an abstract state satisfying ψ to an abstract state satisfying ψ (since $k-1 < n$), we have that $B_{k-1} \vDash \psi$, i.e., this transitions do not permit us to reach a bad state.

Example 4: Consider k.s. $\mathcal{K} = ([1..12], \rightarrow, l)$ and $\mathcal{A} = (\mathtt{P}, \rightarrow^{\exists\exists}, l_{\mathtt{P}})$, where $\mathcal{A}$ is induced by $\mathcal{K}$ and a partition $\mathtt{P} = \{[1,2,3],[4,5,6],[7,8,9],[10,11,12]\} = \{B_1,B_2,B_3,B_4\}$ in Fig. 5. Let $t^\sharp = \langle B_1,B_2,B_3,B_4\rangle$ be a CE produced by a model checker. SplitPATH determines $S^1 = \{1,2,3\}$, $S^2 = \{4,5,6\}$, $S^3 = \{9\}$, $S^4 = \varnothing$, and it returns $k = 4$. We consider blocks $B_3 = \{7,8,9\}$ and $B_4 = \{10,11,12\}$, and we notice that $\text{BAD} = \{7\}$. CEGMOD removes the transition $7 \rightarrow 12$ (the dotted one) from $\mathcal{K}$ and the corresponding one, $B_3 \rightarrow^{\exists\exists} B_4$ from $\mathcal{A}$. Then CEGMOD realizes that there is no outgoing transition from 7, and it adjusts the induced partial relation by enriching it with $7 \rightarrow 7$. We induced a new system $\mathcal{K}'$ whose abstraction does not give rise to $t^\sharp$: in fact, in the induced system it is not possible to reach B_4 from B_3, i.e., $B_3 \not\rightarrow^{\exists\exists} B_4$.

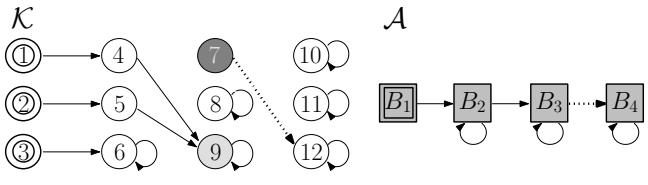


Figure 5.

The following theorem states that the CEGMOD induces strong preservation.

Theorem 9: Given a k.s. $\mathcal{K}$, a specification $\varphi = \text{AG}\psi$, where ψ is a propositional formula and a partition P. Let $\mathcal{K}'$ be the deformation of $\mathcal{K}$ induced by CEGMOD and $\mathcal{A}'$ its abstraction induced by P. Then, $\mathcal{A}'$ is strong preserving for φ, i.e., $\mathcal{A}' \vDash \varphi \Leftrightarrow \mathcal{K}' \vDash \varphi$.

Proof: Suppose that we are at the i^{th} iteration of CEGMOD, for some $i \in \mathbb{N}$, and suppose that the current concrete and abstract models are $\mathcal{K}_i$ and $\mathcal{A}_i$ respectively. We distinguish the following cases:

Case 1. Suppose the model checker verifies $\mathcal{A}_i$ w.r.t. φ and it states that it is correct. In this case, CEGMOD returns $\mathcal{K}_i$. Then, weak preservation guarantees that even the concrete model, i.e., $\mathcal{K}_i$ is correct, and therefore we can conclude that $\mathcal{K}_i \vDash \varphi \Leftrightarrow \mathcal{A}_i \vDash \varphi$.

Case 2. Suppose the model checker verifies $\mathcal{A}_i$ w.r.t. φ and it states that it is incorrect, reporting a real counterexample $t^\sharp$. In this case CEGMOD returns an incorrect model $\mathcal{K}_i$ and we can state that $\mathcal{K}_i \vDash \varphi \Leftrightarrow \mathcal{A}_i \vDash \varphi$. It is worth noting that $t^\sharp$ is a counterexample of the original model as well, and it is guaranteed by the third claim of Theorem 8, namely, CEGMOD does not add real counterexamples.

Case 3. Suppose the model checker verifies $\mathcal{A}_i$ w.r.t. φ and it states that it is incorrect, reporting a spurious counterexample $t^\sharp$. In this case CEGMOD does not return any model, but it modifies current model by eliminating some spurious abstract paths and obtaining concrete and abstract models $\mathcal{K}_{i+1}$ and $\mathcal{A}_{i+1}$. Moreover, CEGMOD does not add any new spurious abstract path, and it is guaranteed by the second claim of Theorem 8. Then the CEGMOD loop is iterated again, on new induced models. In the worst case, CEGMOD removes all possible transitions from the original system, and the repairing phase adds

transitions from every concrete state to itself. Anyway, the execution of CEGMOD eventually terminates. In this case CEGMOD states that the obtained model is correct iff all initial states satisfies φ.

It is worth noting that the k.s. returned by CEGMOD depends on the order in which the model checker detects the counterexamples. If CEGMOD returns a model $\mathcal{K}'$ stating that it is correct, weak preservation guarantees the correctness of this result. Otherwise, if CEGMOD states that $\mathcal{K}'$ is incorrect, we do not guarantee that $\mathcal{K}'$ contains no other spurious counterexamples to φ, but since a real one has been detected, we are sure that both the induced model and the original one are incorrect.

V. Related Work and Conclusion

In this paper, we introduce an orthogonal approach to the one of [9]. Starting from a partition $\mathtt{P}_{\text{INIT}} \in \text{Part}(\Sigma)$ and a k.s. $\mathcal{K}_{\text{INIT}}$, they show how it is possible to obtain $\mathtt{P}_{\text{FIN}} \preccurlyeq \mathtt{P}_{\text{INIT}}$ such that $\mathtt{P}_{\text{FIN}} \leftrightarrow \mathcal{K}_{\text{INIT}}$, i.e., they fix $\mathcal{K}_{\text{INIT}}$ and modify $\mathtt{P}_{\text{INIT}}$. On the contrary, we obtain the model $\mathcal{K}_{\text{FIN}}$ from the original model $\mathcal{K}_{\text{INIT}}$ such that $\mathtt{P}_{\text{INIT}} \leftrightarrow \mathcal{K}_{\text{FIN}}$, i.e., we fix $\mathtt{P}_{\text{INIT}}$ and modify $\mathcal{K}_{\text{INIT}}$. As in [9] we deal with state formulae only, and it would be interesting to investigate how to deform models to induce strong preservation for *LTL*. In [1] the authors introduce the theoretical foundation of the operators making a predicate transformer (semantics) complete w.r.t. a given abstraction; we use the same idea and we characterize the corresponding deformations in the particular context of partitions and transition systems.

Moreover, we highlight some properties of the systems deformed by our *upwards* and *downwards* methods in terms of *must* and *may* transitions: we induce the abstract structures whose *must* and *may* transitions coincide, and they also coincide with the *must* (*may*) transitions of the original structures, in the case of the *upwards* (*downwards*) deformation. These results relate our work with [11], [8] and provide an interesting insight into the relation between strong preservation and *must* and *may* transitions.

Finally, in Section IV, we introduce the CEGMOD algorithm for safety properties, based on the theoretical results presented in Section III. Although it seems similar to the CEGAR paradigm [4], these two approaches represent two orthogonal perspectives to the problem of strong preservation in abstract model checking. Starting from a k.s. $\mathcal{K}$, a state partition P and a specification $\varphi \in ACTL^*$, CEGAR induces $\mathtt{P}' \preccurlyeq \mathtt{P}$ such that the abstract structure corresponding to $\mathcal{K}$ w.r.t. $\mathtt{P}'$ is strong preserving for φ. On the contrary, CEGMOD modifies $\mathcal{K}$ and induces $\mathcal{K}'$ whose abstraction related to P is strong preserving for φ. In the worst case, CEGMOD removes all possible transitions from the original system, and the repairing phase adds transitions from every concrete state to itself. Anyway, the execution of CEGMOD eventually always terminates. A similar approach is in CEGIS [13], which also deals with modifying models, but with a different aim. Indeed, CEGIS synthesizes models that always satisfy a given starting specification. This is not the case with CEGMOD, where strong preservation is the only requirement and modified models may not satisfy the original specification.

References

[1] R. Giacobazzi and I. Mastroeni, "Transforming abstract interpretations by abstract interpretation," in *Proc. of SAS'08*, ser. LNCS, vol. 5079. Springer, pp. 1–17.

[2] R. Giacobazzi, F. Ranzato, and F. Scozzari., "Making abstract interpretation complete," *Journal of the ACM*, vol. 47, no. 2, pp. 361–416, March 2000.

[3] R. Giacobazzi and E. Quintarelli, "Incompleteness, counterexamples and refinements in abstract model-checking," in *Proc. of the 8th SAS*, ser. LNCS, vol. 2126. Springer, 2001, pp. 356–373.

[4] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith, "Counterexample-guided abstraction refinement for symbolic model checking," *J. of the ACM*, vol. 50, no. 5, pp. 752–794, 2003.

[5] P. Cousot and R. Cousot, "Systematic design of program analysis frameworks," in *Conf. Rec. of the 6th POPL*. ACM, 1979, pp. 269–282.

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

[7] E. M. Clarke, O. Grumberg, and D. E. Long, "Model checking and abstraction," *ACM Trans. Program. Lang. Syst.*, vol. 16, no. 5, pp. 1512–1542, 1994.

[8] D. Dams, "Abstract interpretation and partition refinement for model checking," Ph.D. dissertation, Eindhoven Univ. of Technology, Eindhoven, The Netherlands, 1996.

[9] F. Ranzato and F. Tapparo, "Generalized strong preservation by abstract interpretation," *Journal of Logic and Computation*, vol. 17, no. 1, pp. 157–197, 2007.

[10] R. Giacobazzi and F. Ranzato, "Example-guided abstraction simplification," in *Proc. of the 37th ICALP: Part II*, ser. ICALP'10. Springer, 2010, pp. 211–222.

[11] K. G. Larsen and B. Thomsen, "A modal process logic," in *Proc. of the 3rd Annual Symposium on Logic in Computer Science*. IEEE, 1988, pp. 203–210.

[12] B. Alpern and F. B. Schneider, "Recognizing safety and liveness," *Distributed Comp.*, vol. 2, pp. 117–126, 1987.

[13] A. Solar-Lezama, G. Arnold, L. Tancau, R. Bodík, V. A. Saraswat, and S. A. Seshia, "Sketching stencils," in *PLDI'07*. ACM, 2007, pp. 167–178.