

Refinement of LTL Formulas for Abstract Model Checking

María del Mar Gallardo, Pedro Merino, and Ernesto Pimentel

Dpto. de Lenguajes y Ciencias de la Computación
University of Málaga, 29071 Málaga, Spain
{gallardo,pedro,ernesto}@lcc.uma.es

Abstract. A crucial problem in abstract model checking is to find a tradeoff between constructing the “best” (the smallest) abstract model, approximating a given model, and preserving as much interesting properties over the original model as possible. In this paper, we present a method for dealing with this problem based on the definition of a new abstract satisfiability relation. This new relation allows us to analyze temporal properties with different degrees of precision, by means of a refinement process. The method subsumes the classic way of abstracting properties and the dual proposal of the authors. As a consequence, maintaining the same abstract model, we directly obtain the preservation of universal properties (as in the classic method) and the refutation of existential properties (as in the dual method). We also show the utility of this method by proving that the very important notions of completeness and precision in abstract model checking may be analyzed by using the new relation. In particular, we exploit the power of model checking to simultaneously refine both the model and the properties.¹

1 Introduction

Abstract model checking integrates the two well-known techniques of model checking [3,2] and abstract interpretation [8] to reduce the state space produced when verifying realistic and complex models, which permits the analysis of bigger systems using the available tools.

Abstract model checking involves two related activities. On the one hand, the first step is to apply abstract interpretation in order to reduce the model to be analyzed. Considering that models are represented as transition systems $M = \langle \Sigma, \rightarrow \rangle$, this task basically consists of abstracting data (states) and operations (transitions) as in the classic applications of abstract interpretation. Thus, the abstraction of states consists of defining a set of abstract states Σ^α and relating concrete states with their approximations in Σ^α . This may be done by defining Galois connections as in [19], abstraction functions as in [16,10,7,12] or surjective homomorphisms as in [4,6]. Regarding the transitions, we find in existing literature [6,10] two complementary definitions. One of them,

¹ This research is partially supported by the CICYT projects TIC2001-2705-C03-02 and TIC99-1083-C02-01.

$M^\alpha = \langle \Sigma^\alpha, \rightarrow^\alpha \rangle$, produces an over-approximation of M in the sense that each concrete transition in M is abstracted by one in M^α , while the other, here denoted by $M_{\exists}^\alpha = \langle \Sigma^\alpha, \rightarrow_{\exists}^\alpha \rangle$, is an under-approximation, this meaning that each abstract transition corresponds to a concrete one. Abstract transition relations \rightarrow^α and $\rightarrow_{\exists}^\alpha$ are respectively called *liberal* and *conservative* in [6] and *free* and *constrained* in [10]. An alternative method for abstracting the model is the abstraction of predicates that replaces selected predicates in the model by Boolean variables [1,22,23].

Once the model has been reduced, we must also abstract the original satisfiability relation \models , which evaluates temporal properties against concrete models, and define an abstract relation to reinterpret the meaning of properties against the abstract models. Given a generic temporal property f , the ideal objective of the abstraction process is “strong preservation” (that is, the preservation of both the truth and the falsehood) of the universal ($\forall f$) and existential ($\exists f$) properties between M and M^α . However, strong preservation of universal and existential properties is only possible if M and M^α are bi-similar [19], which entails a considerable constraint when the objective is to decrease the state space. Thus, it is accepted that a reasonable construction of abstract models may involve some loss of information when analyzing temporal properties.

Following the same ideas discussed above, we may consider different abstractions of \models , each preserving a different set of temporal formulas. For instance, the abstract satisfiability relation defined in [4,6,10] under-approximates properties in such a way that the abstract model satisfies less properties than the concrete one. We denote this relation as \models_c^α . This definition directly produces the weak preservation of universal (resp. existential) properties from M^α (resp. M_{\exists}^α) to M . Alternatively, the dual definition \models^α that we studied in [12] preserves the refutation of existential (resp. universal) properties in the same direction. Developing this dual approach, we complete abstract model checking with the same two objectives as standard model checking (satisfaction and refutation), and users can choose the method best suited to their personal preferences or to the formulation of property to be analyzed.

Proposals [6,10] both achieve weak preservation of universal and existential properties by integrating free and constrained abstract transition relations to construct the abstract model. Universal properties are proved using the free relation, while for existential properties, the constrained relation is used. However, the verification of properties over the abstract model may produce non-conclusive answers due to the imprecision and incompleteness of the abstract model. For instance, the non-satisfaction of a universal property over the abstract model may be produced by a *spurious trace*, that is, an abstract trace which does not correspond to any concrete one. Similar problems occur in the dual method.

Solutions for this problem follow different directions. On the one hand, the analysis of counter-examples serves to find out whether the trace is real or spurious [5,23,21]. Another possibility is to improve the abstract model by a refinement process as in [10]. In [15], authors prove that the strong preservation of properties in abstract model checking is strongly connected to the precision of the underlying abstract domain. In particular, they show how spurious counter-examples can be removed by refining the abstract domain. The theoretical basis

of this approach has been developed in a more general framework for achieving completeness in abstract interpretation [14]. In [6] the problem of the optimality in abstract model checking is studied by considering that the abstraction of the model also involves the abstraction of properties. In [9] authors present a new combination of the backward and forward analysis (techniques widely employed in abstract interpretation) for improving verification of universal safety properties.

In this paper, we present a new proposal for dealing with the problem of optimality in abstract model checking, in the context of linear time temporal logic (LTL) [20]. The idea is to define an abstract satisfiability relation \Vdash combining and extending both the classic \models_c^α and the dual \models^α methods. The benefits of this relation may be summarized as follows:

- Considering a unique abstract model M^α (the over-approximated version), we simultaneously achieve the preservation of the satisfaction of universal properties and the refutation of existential ones.
- The extended relation allows us to formalize the notion of precision of the abstract model with respect to the analysis of a given property.
- In addition, using relation \Vdash allows us to implicitly refine the model. The model checking tool exclusively produces the part of M^α required to analyze the property.
- The approach is suitable for refining properties depending on the actual precision of the abstract model.

The new satisfiability relation proposed is efficiently implementable. In fact, we have constructed a tool based on this proposal [13,24].

The paper is organized as follows. Section 2 is devoted to the construction of the abstract model. Section 3 presents and relates the classic method for abstracting properties and the over-approximated one, providing a framework with a common terminology. Section 4 presents the extended relation and its application for the verification of temporal formulas. In addition, we study its extension to include the analysis of refined formulas. Finally, in Section 5, we give conclusions and future work.

2 Abstracting Concurrent Systems

Execution of a concurrent program may be defined by means of *labelled transition systems* such as $M = (A, \Sigma, \vec{\rightarrow}, s_0)$ where A is the set of *observable atomic actions*, Σ is the set of standard *states*, $\vec{\rightarrow} \subseteq \Sigma \times A \times \Sigma$ is a labelled transition relation and s_0 is the initial state. We write $s \xrightarrow{a} s'$ for $(s, a, s') \in \vec{\rightarrow}$. A *trace* x of M is a sequence $x = t_0 \xrightarrow{a_0} t_1 \xrightarrow{a_1} \dots$ of states, and it represents a (possibly infinite) computation from state t_0 , where $a_0 a_1 \dots$ is the sequence of actions executed. We denote by x^j the suffix path $t_j \xrightarrow{a_j} t_{j+1} \xrightarrow{a_{j+1}} \dots$ of a trace $x = t_0 \xrightarrow{a_0} \dots$. A *full-trace* $x = t_0 \xrightarrow{a_0} \dots$ is an infinite trace. We assume that terminating traces have a final state which is repeated forever. The set

$\mathcal{O}(M) = \{x \mid x = s_0 \xrightarrow{a_0} \dots \text{ is a full-trace}\}$ defines the trace semantics determined by the transition system M .

We may construct an abstract interpretation $M^\alpha = (A, \Sigma^\alpha, \bar{\rightarrow}_\alpha, s_0^\alpha)$ of a transition system $M = (A, \Sigma, \bar{\rightarrow}, s_0)$ by means of a triple $\mathcal{I}_\alpha = (\Sigma, (\Sigma^\alpha, \leq^\alpha), \alpha)$, where $\alpha : 2^\Sigma \rightarrow \Sigma^\alpha$ is the lower adjoint (the abstraction function) of a Galois insertion between $(2^\Sigma, \subseteq)$ and the lattice of abstract states $(\Sigma^\alpha, \leq^\alpha)$. Partial order \leq^α denotes the degree of precision of each abstract state, the smallest elements being the most precise ones. Function α associates each state s with its “best” approximation $\alpha(\{s\}) \in \Sigma^\alpha$ wrt \leq^α . In the sequel, we write $\alpha(s)$ for $\alpha(\{s\})$.

Given $x = t_0 \xrightarrow{a_0} \dots$, $\alpha(x)$ denotes the abstract trace $\alpha(t_0) \xrightarrow{a_0} \dots$. Abstract trace $\alpha(x)$ is the most precise approximation of x . However, note that possibly $\alpha(x)$ is not an element of $\mathcal{O}(M^\alpha)$. Finally, given $x^\alpha = t_0^\alpha \xrightarrow{a_0} \dots$ and $y^\alpha = r_0^\alpha \xrightarrow{a_0} \dots$, we write $x^\alpha \leq^\alpha y^\alpha$ when $\forall i \geq 0. t_i^\alpha \leq^\alpha r_i^\alpha$.

Definition 1. M^α is \mathcal{I}_α -correct wrt M , iff $\forall x \in \mathcal{O}(M)$ there exists $x^\alpha \in \mathcal{O}(M^\alpha)$ such that $\alpha(x) \leq^\alpha x^\alpha$.

\mathcal{I}_α -correctness means that the abstract system over-approximates the concrete one in two directions. First, the abstract trace x^α approximating x in $\mathcal{O}(M)$ is usually less precise than $\alpha(x)$, which means that abstract traces usually lose information with respect to concrete ones. Imprecision of abstract traces directly affects the analysis of properties. In addition, abstract models are often incomplete, that is, the construction of abstract transition systems may produce the so-called “spurious” traces, which can lead to obtaining false answers when analyzing temporal properties.

In the sequel, we always assume that M^α is \mathcal{I}_α -correct wrt the original model M .

3 Abstracting Temporal Logic: Relating Two Dual Approaches

Kripke structures are used to evaluate temporal formulas against models. In this section, we summarize the classic approach and the method based on over-approximation for abstracting Kripke structures. We also discuss the main preservation results that may be deduced from these two dual definitions. In order to easily integrate both approaches, we consider *weak* Kripke structures where the *negation* \neg is not dealt with as a connective, but as a way of constructing an atomic proposition.

3.1 Temporal Logic

Given $Prop$ a set of propositions, we construct the set $\mathcal{P} = Prop \cup \neg Prop$, where $\neg Prop = \{\neg p : p \in Prop\}$. Let \mathcal{F} be the set of LTL temporal formulas built inductively using the elements of \mathcal{P} , the standard Boolean operators, except \neg ,

and the *temporal operators*: *next* “ \bigcirc ”, *always* “ \square ”, *eventually* “ \diamond ”, and *until* “ U ”.

A labelled transition system $M = (A, \Sigma, \vec{\rightarrow}, s_0)$ may be extended to a *weak Kripke* structure $\mathcal{K}_{\mathcal{P}} = \langle M, \tau \rangle$ where $\tau : \Sigma \rightarrow 2^{\mathcal{P}}$ is a function that assigns truth values to the propositions of \mathcal{P} in each state.

$\mathcal{K}_{\mathcal{P}} = \langle M, \tau \rangle$ is a *Kripke* structure iff $\forall s \in \Sigma, \forall p \in Prop$ the *Principle of Excluded Middle* (PEM) (i. e., $p \in \tau(s) \vee \neg p \in \tau(s)$), and the *Principle of Non-Contradiction* (PNC) (i. e., $p \notin \tau(s) \vee \neg p \notin \tau(s)$) hold.

Note that negated propositions are explicitly included in \mathcal{P} . In the sequel, $p \in \mathcal{P}$ denotes both positive and negative atomic propositions. In addition, we assume that $\neg\neg p = p$. Structure $\mathcal{K}_{\mathcal{P}} = \langle M, \tau \rangle$ defines an interpretation of atomic propositions, which may be rewritten as a satisfiability relation \models^{τ} over traces $x = t_0 \xrightarrow{\alpha_0} \dots$ using the initial state t_0 to evaluate the propositions as:

$$x \models^{\tau} p \Leftrightarrow p \in \tau(t_0)$$

Now we can extend \models^{τ} to temporal formulas by defining the meaning of the temporal operators as follows.

Definition 2. *Let $\mathcal{K}_{\mathcal{P}} = \langle M, \tau \rangle$ be a weak Kripke structure. Given a trace x , and $f, g \in \mathcal{F}$, we define relation \models^{τ} inductively as follows:*

$$\begin{aligned} x \models^{\tau} f \vee g &\Leftrightarrow x \models^{\tau} f \text{ or } x \models^{\tau} g. \\ x \models^{\tau} f \wedge g &\Leftrightarrow x \models^{\tau} f \text{ and } x \models^{\tau} g. \\ x \models^{\tau} f \rightarrow g &\Leftrightarrow x \models^{\tau} f \text{ implies } x \models^{\tau} g. \\ x \models^{\tau} \bigcirc f &\Leftrightarrow x^1 \models^{\tau} f. \\ x \models^{\tau} \square f &\Leftrightarrow \forall k \geq 0. x^k \models^{\tau} f. \\ x \models^{\tau} \diamond f &\Leftrightarrow \exists k \geq 0. x^k \models^{\tau} f. \\ x \models^{\tau} f \text{ U } g &\Leftrightarrow \exists k \geq 0. (x^k \models^{\tau} g \text{ and } \forall j < k. (x^j \models^{\tau} f)). \end{aligned}$$

Finally, we extend \models^{τ} as follows.

1. Universal formulas: $M \models^{\tau} \forall f$ iff $\forall x \in \mathcal{O}(M). x \models^{\tau} f$.
2. Existential formulas: $M \models^{\tau} \exists f$ iff $\exists x \in \mathcal{O}(M). x \models^{\tau} f$.²

The set \mathcal{F} includes all temporal formulas in negation normal form, that is, negations only appear in atomic propositions. Note that we have not included a rule defining the satisfaction of a negated formula. Instead, we treat the evaluation of negated propositions independently of their corresponding non-negated ones. The reason for this will be explained below.

In the following three sections, we present and compare the two dual methods for abstracting the satisfiability relation \models . In what follows, in order to simplify notation, we will write $\models, \models^{\alpha}$ instead of $\models^{\tau}, \models^{\tau\alpha}$, respectively. Besides, to differentiate between the two different ways of abstracting the satisfiability relation considered in the paper, we will use \models_c^{α} when we refer to the classic relation.

² Note that symbols \forall and \exists are not LTL connectives. We use them to simplify the notation.

3.2 The Classic Method

Let $\mathcal{K}_{\mathcal{P}} = \langle M, \tau \rangle$ a weak Kripke structure. The classic abstraction of $\mathcal{K}_{\mathcal{P}}$, $\mathcal{K}_{\mathcal{P}}^{\alpha} = \langle M^{\alpha}, \tau_c^{\alpha} \rangle$, is given by

$$\tau_c^{\alpha}(s^{\alpha}) = \bigcap_{\{\alpha(s) \leq^{\alpha} s^{\alpha}\}} \tau(s) \quad (\text{Under}_c)$$

Note that the codomain of both τ and τ_c^{α} coincide. Usually, $\mathcal{K}_{\mathcal{P}}$ is a Kripke structure, that is, it satisfies the conditions PNC and PEM. However, note that the way of defining τ_c^{α} makes it possible that for a given abstract state s^{α} and a proposition $p \in \mathcal{P}$, neither $p \in \tau_c^{\alpha}(s^{\alpha})$ nor $\neg p \in \tau_c^{\alpha}(s^{\alpha})$ occur. This is why we skipped the rule for negated propositions from Definition 2. Condition Under_c has a number of interesting properties.

1. τ_c^{α} under-approximates τ . Given $s^{\alpha} \in \Sigma^{\alpha}$,

$$\forall s. (\alpha(s) \leq^{\alpha} s^{\alpha} \Rightarrow \tau(s) \supseteq \tau_c^{\alpha}(s^{\alpha})) \quad (A_c)$$

2. τ_c^{α} is the biggest set verifying the condition (A_c) . Given $s^{\alpha} \in \Sigma^{\alpha}$, and $\mathcal{Q} \subseteq \mathcal{P}$,

$$\text{if } \forall s. (\alpha(s) \leq^{\alpha} s^{\alpha} \Rightarrow \tau(s) \supseteq \mathcal{Q}) \text{ then } \tau_c^{\alpha}(s^{\alpha}) \supseteq \mathcal{Q} \quad (C_c)$$

3. τ_c^{α} is monotonic decreasing. Given $s_1^{\alpha}, s_2^{\alpha} \in \Sigma^{\alpha}$,

$$\text{if } s_1^{\alpha} \leq^{\alpha} s_2^{\alpha} \text{ and } \exists s. \alpha(s) \leq^{\alpha} s_1^{\alpha} \text{ then } \tau_c^{\alpha}(s_1^{\alpha}) \supseteq \tau_c^{\alpha}(s_2^{\alpha}) \quad (M_c)$$

4. A_c and C_c univocally determine τ_c^{α} ,

$$\text{Under}_c \iff A_c \wedge C_c \quad (E_c)$$

5. The extension to abstract traces preserves the satisfiability relation from the abstract to the concrete model, that is, given $f \in \mathcal{F}$

$$\alpha(x) \leq^{\alpha} x^{\alpha} \Rightarrow (x^{\alpha} \models_c^{\alpha} f \Rightarrow x \models f) \quad (P_c)$$

Condition P_c assures the weak conservation of universal properties from the abstract to the concrete model as stated in the following theorem.

Theorem 1. *Given $f \in \mathcal{F}$, if $M^{\alpha} \models_c^{\alpha} \forall f$ then $M \models \forall f$.*

This theorem is equivalent to the assertion $M \not\models \forall f \Rightarrow M^{\alpha} \not\models_c^{\alpha} \forall f$. However, assuming that $\neg f$ is in negation normal form, since $M^{\alpha} \not\models_c^{\alpha} \forall f \Rightarrow M^{\alpha} \models_c^{\alpha} \exists \neg f$ holds only if $\mathcal{K}_{\mathcal{P}}^{\alpha}$ is a Kripke structure, the theorem cannot be used to refute temporal formulas. In the following section, we will see that refutation is obtained by changing under-approximation by over-approximation.

3.3 The Over-Approximation Method

Considering the abstract weak Kripke structure $\mathcal{K}_{\mathcal{P}}^{\alpha} = \langle M^{\alpha}, \tau^{\alpha} \rangle$, where condition $Under_c$ is substituted by

$$\tau^{\alpha}(s^{\alpha}) = \bigcup_{\{\alpha(s) \leq^{\alpha} s^{\alpha}\}} \tau(s) \quad (Over)$$

we obtain a dual approach for abstracting the satisfiability relation \models . Note that with this definition, it is possible that for a given abstract state s^{α} and a proposition $p \in \mathcal{P}$, either $p \in \tau^{\alpha}(s^{\alpha})$ and $\neg p \in \tau^{\alpha}(s^{\alpha})$ occur.

The following properties of τ^{α} are dual to the ones given in the preceding section:

1. τ^{α} over-approximates τ . Given $s^{\alpha} \in \Sigma^{\alpha}$,

$$\forall s. (\alpha(s) \leq^{\alpha} s^{\alpha} \Rightarrow \tau(s) \subseteq \tau^{\alpha}(s^{\alpha})) \quad (A)$$

2. τ^{α} is the smallest set verifying the condition (A). Given $s^{\alpha} \in \Sigma^{\alpha}$, and $\mathcal{Q} \subseteq \mathcal{P}$,

$$\text{if } \forall s. (\alpha(s) \leq^{\alpha} s^{\alpha} \Rightarrow \tau(s) \subseteq \mathcal{Q}) \text{ then } \tau^{\alpha}(s^{\alpha}) \subseteq \mathcal{Q} \quad (C)$$

3. τ^{α} is monotonic increasing. Given $s_1^{\alpha}, s_2^{\alpha} \in \Sigma^{\alpha}$,

$$\text{if } s_1^{\alpha} \leq^{\alpha} s_2^{\alpha} \text{ then } \tau^{\alpha}(s_1^{\alpha}) \subseteq \tau^{\alpha}(s_2^{\alpha}) \quad (M)$$

4. A and C univocally determine τ^{α} ,

$$Over \iff A \wedge C \quad (E)$$

5. The extension to abstract traces preserves the satisfiability relation from the concrete to the abstract model, that is, given $f \in \mathcal{F}$

$$\alpha(x) \leq^{\alpha} x^{\alpha} \Rightarrow (x \models f \Rightarrow x^{\alpha} \models^{\alpha} f) \quad (P)$$

Condition P assures the refutation of existential properties from the abstract to the concrete model.

Theorem 2. *Given $f \in \mathcal{F}$, if $M^{\alpha} \not\models^{\alpha} \exists f$ then $M \not\models \exists f$.*

As before, note that the previous theorem cannot be used to analyze the satisfaction of universal properties, since in general, assuming that formula $\neg f$ is in negation normal form, $M^{\alpha} \models^{\alpha} \exists f \not\Rightarrow M^{\alpha} \not\models^{\alpha} \forall \neg f$.

Example 1. Consider the model $M = (A, Int, \vec{\rightarrow}, s_0)$, containing an integer variable a and the abstraction function $\alpha_{np} : 2^{Int} \rightarrow NegPos^{\alpha}$ which transforms sets of integer values into one of the values of the lattice $NegPos^{\alpha} = \{\perp, neg, nopos, zero, noneg, pos, \top\}$, in the natural way. Figure 1 shows the partial order defined over $NegPos^{\alpha}$. Assume that $\mathcal{K}_{\mathcal{P}} = \langle M, \tau \rangle$ is a Kripke structure and consider propositions $a > 0, a \leq 0 \in \mathcal{P}$.

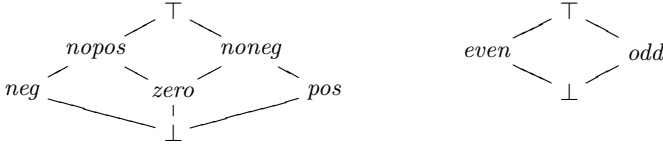


Fig. 1. Partial Orders over $NegPos^\alpha$ and $EvenOdd^\alpha$

Following the classic method, $a > 0 \in \tau_c^\alpha(a^\alpha) \Leftrightarrow a^\alpha = pos$ and $a \leq 0 \in \tau_c^\alpha(a^\alpha) \Leftrightarrow a^\alpha \in \{neg, zero, nopos\}$. However, for the over-approximation method, proposition $a > 0 \in \tau^\alpha(a^\alpha)$ iff $a^\alpha \in \{pos, noneg, \top\}$, and $a \leq 0 \notin \tau^\alpha(a^\alpha)$ iff $a^\alpha = pos$.

Therefore, checking $M^\alpha \models_c^\alpha \forall \square(a > 0)$ directly proves that variable a has always a positive value, and dually, checking $M^\alpha \not\models_c^\alpha \exists \diamond(a \leq 0)$ proves that a never takes a non-positive value. Note that both methods may prove the same property but using dual approaches.

3.4 Relating the Classic and the Over-Approximation Methods

The original model $\mathcal{K}_{\mathcal{P}}$ used to be a Kripke structure, that is, conditions PEM and PNC hold. However, we may weaken these conditions by only imposing *precision* on the propositions to be analyzed. We say that $p \in \mathcal{P}$ is *precise* in the original structure $\mathcal{K}_{\mathcal{P}} = \langle M, \tau \rangle$ iff $\forall s \in \Sigma$,

$$p \in \tau(s) \Leftrightarrow \neg p \notin \tau(s) \quad (Prec_p)$$

Condition $Prec_p$ is reasonable because we cannot obtain precision over the abstract model, if we do not have precision over the original one. In the rest of the paper, we always assume that the propositions to be analyzed are precise wrt the original structure. This means that $\mathcal{K}_{\mathcal{P}}$ may be a *weak* Kripke structure obtained from a previous process of abstraction.

Under these conditions, we obtain the following result relating the classic and the over-approximated methods.

Proposition 1. *Given $p \in \mathcal{P}$ and $f \in \mathcal{F}$,*

$$\begin{aligned} (a) \quad & \forall s^\alpha \in \Sigma^\alpha. (\neg p \notin \tau^\alpha(s^\alpha) \Leftrightarrow p \in \tau_c^\alpha(s^\alpha)) \\ (b) \quad & \forall x^\alpha \in \mathcal{O}(M^\alpha). (x^\alpha \not\models_c^\alpha \neg f \Leftrightarrow x^\alpha \models_c^\alpha f) \end{aligned}$$

Result (a) clearly shows the dual effect of the loss of information about a proposition p on both methods. Result (b) extends this effect to traces and temporal formulas.

Proposition 2. *Given $f \in \mathcal{F}$,*

$$\begin{aligned} (a) \quad & M^\alpha \models_c^\alpha \forall f \Rightarrow M^\alpha \models^\alpha \forall f \\ (b) \quad & M^\alpha \not\models_c^\alpha \exists f \Rightarrow M^\alpha \not\models_c^\alpha \exists f \\ (c) \quad & M^\alpha \models_c^\alpha \forall f \Leftrightarrow M^\alpha \not\models_c^\alpha \exists \neg f \end{aligned}$$

Results (a) and (b) hold due to the fact that the classic method under-approximates properties and the dual method over-approximates them. Result (c) is more interesting since it means that the two methods succeed under the same conditions. In other words, the complementary result of Theorem 1 is Theorem 2, and the opposite.

In addition, Proposition 2(c) also shows that both methods may be used for proving or refuting a given property. However, this is not true in general because it is not always possible/efficient to construct/verify the negation normal form of a temporal formula. For instance, the negation normal form of $\neg(pUq)$ makes use of the release operator “ \bigvee ” which is not implemented in most model checkers. Thus, in practice, we cannot always use the classic method for refuting properties, and dually the over-approximated method cannot be always used to prove universal properties.

From the other point of view, we can also say that the classic method cannot give a conclusive answer regarding the satisfaction of a universal property, iff the over-approximation method cannot refute its negation. The reasons for the failure are complementary in each method. Classic method fails because condition *PEM* does not hold, and the over-approximation method fails because *PNC* does not hold. This situation occurs when the abstract model is incomplete or when the abstract traces are imprecise with respect to the formula to be proved. Imprecision is usually obtained during the construction of the abstract models, and in general it cannot be avoided.

Example 2. In the context of Example 1, assume that $inc1, dec1 \in A$ are two actions that respectively increase and decrease the value of variable a by 1. The trace $x = 1 \xrightarrow{inc1} 2 \xrightarrow{inc1} 3 \xrightarrow{dec1} 2 \rightarrow \dots$ may be abstracted by $x^\alpha = pos \xrightarrow{inc1} pos \xrightarrow{inc1} pos \xrightarrow{dec1} noneg \rightarrow \dots$. Observe that $\alpha(x) \leq^\alpha x^\alpha$, but since the fourth state is approximated by *noneg* and not by *pos*, x^α is more imprecise than $\alpha(x)$. This is because the decrement by 1 of a positive value may produce the value *zero* or another positive number. Note that, in this case, imprecision can not be avoided. Thus, if $x^\alpha \in \mathcal{O}(M^\alpha)$ then $M^\alpha \not\models_c^\alpha \forall \square(a > 0)$ and $M^\alpha \models^\alpha \exists \diamond(a \leq 0)$.

However, the abstraction of states should be chosen in such a way that the propositions to be analyzed are not imprecise from the beginning. The notion of *consistency* introduces this idea.

Proposition $p \in \mathcal{P}$ is *consistent* wrt τ and \mathcal{I}_α iff $\forall s \in \Sigma$,

$$p \in \tau^\alpha(\alpha(s)) \Rightarrow p \in \tau(s)$$

Example 3. Consider again the Kripke structure used in Example 1 and the abstraction function $\alpha_{eo} : 2^{Int} \rightarrow EvenOdd^\alpha$ which abstracts integer values into the values of $EvenOdd^\alpha = \{\perp, even, odd, \top\}$ in the natural way. Figure 1 illustrates the partial order defined over this set. Proposition $a = 1$ is not *consistent* wrt τ and $\mathcal{I}_{\alpha_{eo}}$, since $a = 1 \in \tau^\alpha(\alpha_{eo}(3))$ but $a = 1 \notin \tau(3)$. In contrast, it is easy to prove that proposition $a \bmod 2 \neq 0$ is *consistent*.

When all atomic propositions in \mathcal{P} are consistent, we obtain a definition equivalent to the one given in [19]. The following proposition clarifies this equivalence.

Proposition 3.

$p \in \mathcal{P}$ is consistent wrt τ and \mathcal{I}_α iff $\forall s.(p \in \tau_c^\alpha(\alpha(s)) \vee \neg p \in \tau_c^\alpha(\alpha(s)))$

Proof. Necessary condition. Given $s \in \Sigma$, let us assume that there exist two states $s_1, s_2 \in \Sigma$ such that $\alpha(s_1) \leq^\alpha \alpha(s)$, $\alpha(s_2) \leq^\alpha \alpha(s)$, and $p \in \tau(s_1)$, $\neg p \in \tau(s_2)$. By Definition *Over*, this implies that $p, \neg p \in \tau^\alpha(\alpha(s))$, and applying the hypothesis, we have that $p, \neg p \in \tau(s)$. But this is not possible if p is precise in the original structure. Therefore, either $\forall s'.\alpha(s') \leq^\alpha \alpha(s), p \in \tau(s')$ or $\forall s'.\alpha(s') \leq^\alpha \alpha(s), \neg p \in \tau(s')$. That is, $p \in \tau_c^\alpha(\alpha(s)) \vee \neg p \in \tau_c^\alpha(\alpha(s))$.

Sufficient condition. Assume that $p \in \tau^\alpha(\alpha(s))$. Then, by definition, there exists $s_1 \in \Sigma$ such that $\alpha(s_1) \leq^\alpha \alpha(s)$ and $p \in \tau(s_1)$. Using the hypothesis, this implies that $\forall s'.\alpha(s') \leq^\alpha \alpha(s), p \in \tau(s')$. In particular, $p \in \tau(s)$. \square

This means the consistency of a proposition may be equivalently proved using the classic and the over-approximation methods. Proposition 3 also proves that only consistent propositions may be analyzed by the classic method. However, the over-approximation method may additionally use inconsistent propositions as studied in [12].

As commented in the introduction, imprecision and incompleteness in abstract model checking is usually analyzed by refining the abstract model or analyzing the counter-examples obtained. In Section 4.2, we present an alternative proposal. Since the over-approximation method introduces inconsistencies and the classic method eliminates them, we propose to refine formulas by progressively eliminating inconsistencies when using the over-approximation method or by introducing them when using the classic one. This refinement process makes use of an extended satisfiability relation which is developed below.

4 Refinement of Temporal Formulas

From the preceding discussion, and imitating the usual way of verifying in (standard) model checking, we may argue that the ideal approach for abstract model checking is the use of the classic method for proving the satisfaction of some key properties, and the over-approximation method to discard critical errors. Furthermore, in this section, we propose a common framework to integrate, and even to extend, the capabilities of these two basic methods. Note that the advantage of this combination is that the abstract model is not modified. This section also contains some methodological guidelines for the practical use of this framework.

Definition 3. Consider the sets $\mathcal{P}^\alpha = \{p^\alpha : p \in \mathcal{P}\}$ and $\overline{\mathcal{P}} = \mathcal{P} \cup \mathcal{P}^\alpha$. We construct the weak Kripke structure $\mathcal{K}_{\overline{\mathcal{P}}}^\alpha = \langle M^\alpha, \overline{\tau}^\alpha \rangle$, where $\overline{\tau}^\alpha : \Sigma^\alpha \rightarrow \overline{\mathcal{P}}$ is defined as:

$$\begin{aligned} p \in \overline{\tau}^\alpha(s^\alpha) &\Leftrightarrow \neg p \notin \tau^\alpha(s^\alpha) \\ p^\alpha \in \overline{\tau}^\alpha(s^\alpha) &\Leftrightarrow p \in \tau^\alpha(s^\alpha) \end{aligned}$$

Let \Vdash denote the satisfiability relation $\models^{\overline{\tau}^\alpha}$. The set \mathcal{P}^α contains the abstract versions of the elements of \mathcal{P} . Intuitively, $x^\alpha \Vdash p$ means that the abstract satisfaction of p does not involve loss of information, that is, it means that the abstraction process has not affected proposition p . In addition, $x^\alpha \Vdash p^\alpha$ means that the abstract satisfaction of p “may” involve loss of information. Clearly, $x^\alpha \Vdash p \Rightarrow x^\alpha \Vdash p^\alpha$, but the opposite may fail if the abstraction process has modified the meaning of p .

Let $\overline{\mathcal{F}}$ denote the set of temporal formulas which can be constructed using the atomic propositions of $\overline{\mathcal{P}}$. We may construct the abstract versions of the temporal formulas in \mathcal{F} by inductively substituting each sub-formula by its abstraction as follows: $(f \vee g)^\alpha = f^\alpha \vee g^\alpha$, $(f \wedge g)^\alpha = f^\alpha \wedge g^\alpha$, $(f \rightarrow g)^\alpha = f^\alpha \rightarrow g^\alpha$, $(\diamond f)^\alpha = \diamond f^\alpha$, $(\Box f)^\alpha = \Box f^\alpha$, $(\bigcirc f)^\alpha = \bigcirc f^\alpha$, and $(fUg)^\alpha = f^\alpha U g^\alpha$.

We may extend \Vdash to temporal formulas as in Definition 2.

Proposition 1 allows us to assert that $x^\alpha \Vdash p \Leftrightarrow x^\alpha \models_c^\alpha p$ which means that relation \models_c^α may be expressed in terms of \Vdash . Extending this assertion and the one given by Definition 3 to temporal formulas, we obtain that

$$\begin{aligned} x^\alpha \Vdash f^\alpha &\Leftrightarrow x^\alpha \models_c^\alpha f \\ x^\alpha \Vdash f &\Leftrightarrow x^\alpha \models_c^\alpha f \end{aligned}$$

which mean that \Vdash may be used to model both the classic relation \models_c^α , and \models^α .

4.1 Imprecision and Incompleteness

The extended relation \Vdash may help us know, and even modify, the degree of precision when analyzing temporal properties. In addition, this technique may serve to refine the model by using model checking power to discard traces which are not interesting for any reason (for instance, because it is known that they are “spurious”.) The next definition states the notion of precision.

Definition 4. We say that formula $f \in \mathcal{F}$ does not lose precision wrt $\mathcal{O}(M^\alpha)$ iff $\forall x^\alpha \in \mathcal{O}(M^\alpha)$, if $x^\alpha \models_c^\alpha f$ then $x^\alpha \not\models_c^\alpha \neg f$.

Note that the notion of precision is associated with a given property. Thus, even though abstract models are, in general, imprecise, they may be useful for analyzing some properties precisely. By Proposition 1, Definition 4 could have also been given as $x^\alpha \models_c^\alpha f$ instead of $x^\alpha \not\models_c^\alpha \neg f$ which allows us to formulate, in the following proposition, the notion of precision in terms of \Vdash . This means that we may use the model checker to automatically analyze the precision of the abstract model with respect to a given property. The proposition also proves that, when the formula f is precise wrt the abstract model, the two methods may be used to respectively check the satisfaction of $\forall f$ and the refutation of $\exists f$.

Proposition 4. *Given $f \in \mathcal{F}$, if $M^\alpha \Vdash \forall(f^\alpha \rightarrow f)$ then*

- (a) $M^\alpha \models_c^\alpha \forall f \Leftrightarrow M^\alpha \models^\alpha \forall f$
- (b) $M^\alpha \not\models_c^\alpha \exists f \Leftrightarrow M^\alpha \not\models^\alpha \exists f$

Proof. (a) By Proposition 2(a), $M^\alpha \models_c^\alpha \forall f \Rightarrow M^\alpha \models^\alpha \forall f$. Conversely, if $x^\alpha \models^\alpha f$ then $x^\alpha \Vdash f^\alpha$, and by hypothesis, this means that $x^\alpha \Vdash f$. Therefore, $M^\alpha \models^\alpha \forall f \Rightarrow M^\alpha \models_c^\alpha \forall f$. (b) By Proposition 2(b), $M^\alpha \not\models_c^\alpha \exists f \Rightarrow M^\alpha \not\models^\alpha \exists f$. Conversely, if $x^\alpha \models^\alpha f$ then $x^\alpha \Vdash f^\alpha$. By hypothesis, this means that $x^\alpha \Vdash f$ or, equivalently, $x^\alpha \models_c^\alpha f$ which is not possible. Therefore, $M^\alpha \not\models_c^\alpha \exists f \Rightarrow M^\alpha \not\models^\alpha \exists f$. \square

Proposition 5. *Let $f, g \in \mathcal{F}$.*

- (a) $M \models \forall g$ and $M^\alpha \Vdash \forall(g^\alpha \rightarrow f) \Rightarrow M \models \forall f$
- (b) $M \not\models \exists g$ and $M^\alpha \Vdash \forall(f^\alpha \rightarrow g) \Rightarrow M \not\models \exists f$

Proof. Given $x \in \mathcal{O}(M)$, by the \mathcal{I}_α -correctness, $x^\alpha \in \mathcal{O}(M^\alpha)$ exists such that $\alpha(x) \leq^\alpha x^\alpha$. (a) By hypothesis, $x \models g$ which implies (Condition P) that $x^\alpha \models^\alpha g$, that is, $x^\alpha \Vdash g^\alpha$. Using the hypothesis this means that $x^\alpha \Vdash f$, or equivalently, $x^\alpha \models_c^\alpha f$. Finally, by Condition P_c , we obtain that $x \models f$. Thus, (a) holds. (b) By hypothesis, $x \not\models g$ and $x^\alpha \Vdash f^\alpha \rightarrow g$. Therefore, using condition P_c , $x^\alpha \not\models f^\alpha$, that is, $x^\alpha \not\models_c^\alpha f$. And, by condition P , we obtain that $x \not\models f$. \square

The utility of Proposition 5 is clear. We may utilize previously checked formulas to refine the model. Thus, for instance, if we know that the original model satisfies a given property, we may discard all the abstract traces which do not satisfy it. As before, the crucial point here is that the task of refinement is automatically realized by the model checker.

A very important application of this proposition is the elimination of the so-called non-progress cycles. Abstraction may add abstract traces that do not correspond to any original one. For instance, it is usual that the abstraction process transforms terminating loops into non-deterministic loops including non-progress cycles. Many liveness properties may be violated by abstract traces containing these cycles. But these traces are really “spurious”, thus, if we avoid them, we augment the precision of our analysis. Assume that *progress* represents the absence of non-progress cycles from a given state and that it is known that $M \models \forall \square \text{progress}$; then, in order to prove that a formula $f \in \mathcal{F}$ holds, we may discard the abstract traces that do not satisfy $\square \text{progress}$ by using the formula $(\square \text{progress}^\alpha) \rightarrow f$ to prove f . In addition, usually property *progress*³ is not affected by the abstraction process, that is, it coincides with *progress* ^{α} .

4.2 Intermediate Precision

Relation \Vdash induces a partial order relation \Rightarrow over the set of formulas $\overline{\mathcal{F}}$ as $f_1 \Rightarrow f_2 \Leftrightarrow \forall x^\alpha \in \mathcal{O}(M^\alpha). x^\alpha \Vdash f_1 \Rightarrow x^\alpha \Vdash f_2$. Clearly, it holds that for

³ Non-progress cycles are usually analyzed without inspecting the whole state space.

all $f \in \mathcal{F}$, $f \Rightarrow f^\alpha$ and this relation means that the classic method gives an abstract interpretation of f more precise than the one provided by the over-approximation method.

Let us assume that we may construct an intermediate Kripke structure $\mathcal{K}_{\mathcal{P}}^i = \langle M^\alpha, \tau_i^\alpha \rangle$, such that for all $s^\alpha \in \Sigma^\alpha$, $\tau_c^\alpha(s^\alpha) \subseteq \tau_i^\alpha(s^\alpha) \subseteq \tau^\alpha(s^\alpha)$. Using τ_i^α and Definition 2, we may define an intermediate satisfiability relation \models_i^α . Now we extend the relation \Vdash to include the precision given by the abstract Kripke structure $\mathcal{K}_{\mathcal{P}}^i$ as follows.

Consider the sets $\mathcal{P}^i = \{p^i : p \in \mathcal{P}\}$ and $\overline{\mathcal{P}} = \mathcal{P} \cup \mathcal{P}^\alpha \cup \mathcal{P}^i$. We construct the weak Kripke structure $\mathcal{K}_{\overline{\mathcal{P}}}^\alpha = \langle M^\alpha, \overline{\tau}^\alpha \rangle$, where $\overline{\tau}^\alpha : \Sigma^\alpha \rightarrow \overline{\mathcal{P}}$ is defined as:

$$\begin{aligned} p \in \overline{\tau}^\alpha(s^\alpha) &\Leftrightarrow p \in \tau_c^\alpha(s^\alpha), \\ p^i \in \overline{\tau}^\alpha(s^\alpha) &\Leftrightarrow p \in \tau_i^\alpha(s^\alpha) \\ p^\alpha \in \overline{\tau}^\alpha(s^\alpha) &\Leftrightarrow p \in \tau^\alpha(s^\alpha) \end{aligned}$$

Now consider the extension of $\overline{\mathcal{F}}$ including the abstract propositions with intermediate precision. In addition, given a temporal formula f , we may construct the formula f^i by inductively substituting each sub-formula by its intermediate version as was done previously for formulas f^α . The extension of \Vdash including the new formulas satisfies that

$$\begin{aligned} x^\alpha \Vdash f^\alpha &\Leftrightarrow x^\alpha \models^\alpha f \\ x^\alpha \Vdash f^i &\Leftrightarrow x^\alpha \models_i^\alpha f \\ x^\alpha \Vdash f &\Leftrightarrow x^\alpha \models_c^\alpha f \end{aligned}$$

In addition, for all $f \in \mathcal{F}$, $f \Rightarrow f^i \Rightarrow f^\alpha$.

Example 4. Following Example 1, assume that we define $a > 0$ with a intermediate precision τ_i^α between the precisions given by τ_c^α and τ^α as follows: $\tau_i^\alpha(a^\alpha) = \tau^\alpha(a^\alpha)$, $\forall a^\alpha \neq \top$, and $\tau_i^\alpha(\top) = \tau^\alpha(\top) - \{a > 0\}$. Using the notation introduced above, we have that $a > 0 \Rightarrow (a > 0)^i \Rightarrow (a > 0)^\alpha$.

4.3 Methodological Guidelines

As in standard model checking, there are two complementary approaches for ensuring the correctness of a model. In the ‘‘satisfaction-oriented method’’, the user must specify the desired property to be held over the model. Denote this property with $f \in \mathcal{F}$. If $M^\alpha \Vdash \forall f$, then by Theorem 1 property f holds on all traces of M . However, if $M^\alpha \not\Vdash \forall f$, the user may continue the analysis with the formula $\forall f^\alpha$. If $M^\alpha \not\Vdash \forall f^\alpha$, the generous way of defining f^α makes an error on M very probable, except for spurious traces. However, if $M^\alpha \Vdash \forall f^\alpha$, the user knows that ‘‘from the abstract point of view’’, the model satisfies $\forall f$, although this information may be too imprecise. Thus, the user may refine it with an intermediate formula f^i , verifying $f \Rightarrow f^i \Rightarrow f^\alpha$. If $M^\alpha \Vdash \forall f^i$, then the user knows that property f holds on all traces of M until the precision defined by f^i , and this may be enough for her/him. Note that we could incrementally refine the formula until the desired precision is achieved. The probability of obtaining

a real error when $M^\alpha \not\models \forall f^i$ decreases when the precision of f^i increases. As the set of possible intermediate p^i for a given p is finite, part of this task can be done with a semi-automatic tool.

Example 5. In the context of Examples 1 and 4, consider the property “If a is null and it is incremented by 1, then it takes a positive value”, which may be formulated as $f \equiv a = 0 \wedge inc \rightarrow \diamond(a > 0)$. If $M^\alpha \not\models \forall f$, then the user knows that there exists a trace where a has been incremented by 1 but it never takes the abstract value pos . Assume that $M^\alpha \models \forall f^\alpha$, but since this result is too imprecise we construct the more precise version of f^α , $f^i \equiv (a = 0) \wedge inc \rightarrow \diamond(a > 0)^i$, where $(a > 0)^i$ represents $(a > 0)^\alpha \wedge (a \neq \top)$. Note that $f \Rightarrow f^i \Rightarrow f^\alpha$. Now, if $M^\alpha \models \forall f^i$, we know that after incrementing by 1, variable a has taken some of the abstract values pos or $noneg$, but it has never taken the value neg . This may be due to the imprecision when abstracting operations as illustrated in Example 2. If the increment inc was implemented by adding 2 and then subtracting 1, the result may produce the imprecise value $noneg$. In this case, the information given by $M^\alpha \models \forall f^i$ may be sufficient for the user.

The “refutation-oriented” method could be alternatively used. In this case, the user must specify the erroneous behavior as a temporal formula to be refused. Denote this formula with f . If $M^\alpha \not\models \exists f^\alpha$, by Theorem 2, user knows that no trace of M matches f . However, if $M^\alpha \models \exists f^\alpha$, the user may continue the analysis with the formula $\exists f$. Since the classic method carries out a very precise analysis, if $M^\alpha \models \exists f$, then it is very probable, except for spurious traces, that an erroneous trace exists in M . Otherwise, if $M^\alpha \not\models \exists f$, the user could improve the analysis with a less precise formula f^i such that $f \Rightarrow f^i \Rightarrow f^\alpha$. If $M^\alpha \not\models \exists f^i$, the user knows that no trace satisfies f until the imprecision given by f^i . Formula f^i may be refined until the desired imprecision is obtained. The probability of obtaining a real error when $M^\alpha \models \exists f^i$ decreases when f^i decreases.

5 Conclusions

Previous works dealing with abstract model checking focus on satisfaction of temporal formulas and treat the problems of imprecision and incompleteness by means of the model refinement [4,6,10,5,15]. In this work, we have shown that these problems may be also solved from a dual perspective, that is, we may extract information from an incomplete abstract model by refining the formulas to be analyzed against it. In this discussion, the key point is the combination of the classic and the over-approximated methods for abstracting properties.

Note that we have distinguished between incompleteness and imprecision, although, as it has been extensively studied in [14], both notions reflect the same situation from the abstract interpretation perspective: the abstract model does not reproduce the concrete one exactly. However, from the abstract model checking point of view, these two types of incompleteness have to be differently treated since, while imprecise traces cannot be removed from the abstract model (they approximate some concrete traces), it would be desirable to eliminate spurious traces. Of course, an imprecise trace may be also transformed into a

set of more precise abstract traces, possibly including some spurious traces. But this transformation involves a modification of the abstract model.

The results presented in this paper are complementary (dual) to the above mentioned in three key aspects: a) the integration of the refutation of temporal formulas in abstract model checking; b) the study of imprecision and incompleteness problems by means of the refinement of formulas; c) the combination of the classic and the over-approximation methods in a uniform framework. We think that all these proposals are compatible with the other works for improving abstract model checking. In particular, if no information is obtained by the refinement of formulas, we can switch to model refinement or, even, in the worst case, to modifying the abstraction.

One future work is to develop a similar framework using the under-approximated model (M_{\exists}^{α}) (extending the works in [10,6]) for debugging the model. Note that in this case the counter-examples found are real.

Of course, the ideal aim is to integrate all existing techniques in the same tool. In this direction, we have already implemented the proposals presented in the paper. Our current tool, α Spin[13], extends the model checker Spin [17, 18] with syntactic transformation for PROMELA (its modelling language) and LTL (its version of temporal logic). The transformation of PROMELA allows us to obtain a new model that exhibits the behaviours of M^{α} over-approximating M (presented in [11]). The partial/full transformation of LTL produces a new formula which is verified against M^{α} . Using standard Spin verification with these transformed specifications corresponds to implementing the satisfaction relations \models^{α} , \models_c^{α} and \Vdash . The transformation cost is negligible, whereas the experiences with real examples show considerable benefits for verification. The tool can be downloaded free from [24].

References

1. S. Bensalem, Y. Lakhnech, S. Owre, Computing abstractions of infinite state systems compositionally and automatically. In *Computer-Aided Verification CAV'98*, LNCS-1427, pp. 319–331, (1998).
2. E.M. Clarke, O. Grumberg, D. Peled. *Model Checking* (The MIT Press, 2000).
3. E.M. Clarke, E. A. Emerson, A.P. Sistla. Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications. *ACM Trans. on Programming Languages and Systems*, **8**(2) (1986) 244–263.
4. E.M. Clarke, O. Grumberg, D.E. Long. Model Checking and Abstraction. *ACM Transaction on Languages and Systems*, **16**(5) (1994) 1512–1245.
5. E.M. Clarke, O. Grumberg, S. Jha, Y. Lu, H. Veith. Counterexample-guided abstraction refinement. In *Computer-Aided Verification CAV'00*, LNCS-1855, pp. 154–169, (2000).
6. R. Cleveland, P. Iyer, D. Yankelevich. Optimality in Abstractions of Model Checking. In *Proceedings of Static Analysis Symposium* LNCS-983, pp. 51–63, (1995)
7. J. Corbett, M. Dwyer, J. Hatcliff, L. Shawn, C. Pasareanu, H. Zheng. Bandera: Extracting Finite-State Models from Java Source Code. In *Proc. 22nd Int. Conf. On Software Engineering*, pp. 439–448, (2000).

8. P. Cousot, R. Cousot, Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints. In *Conf. Record of the 4th ACM Symp. on Princ. of Prog. Languages*, pp. 238-252, (1977).
9. P. Cousot, R. Cousot. Refining Model Checking by Abstract Interpretation. *Automated Software Engineering* **6** (1999) 69-95.
10. D. Dams, R. Gerth, O. Grumberg. Abstract Interpretation of Reactive Systems. *ACM Trans. on Programming Languages and Systems*, **19**(2) (1997) 253-291.
11. M.M. Gallardo, P. Merino. A Framework for Automatic Construction of Abstract PROMELA Models. In *Theoretical and Practical Aspects of SPIN Model Checking*, LNCS-1680, pp. 184-199, (1999).
12. M.M. Gallardo, P. Merino, E. Pimentel. Verifying Abstract LTL Properties on Concurrent Systems. In *Proc. of the 6th World Conference on Integrated Design & Process Technology*, (2002).
13. M. M. Gallardo, J. Martínez, P. Merino, E. Pimentel. α SPIN: Extending SPIN with Abstraction. In *Proc. of the 9th International SPIN Workshop on Model Checking of Software*, LNCS 2318, pp. 254-258, (2002).
14. R. Giacobazzi, F. Ranzato, F. Scozzari. Making abstract interpretation complete. In *Journal of ACM*, **47**(2) (2000), 361-416.
15. R. Giacobazzi, E. Quintarelli. Incompleteness, Counterexamples and Refinement in Abstract Model-Checking. In *The 8th International Static Analysis Symposium SAS'01*, LNCS 2126, pp. 356-373, (2001).
16. S. Graf. Verification of a distributed Cache Memory by using abstractions. In *Computer Aided Verification CAV'94*, LNCS-818, pp. 207-219, (1994).
17. G.J. Holzmann. *Design and Validation of Computer Protocols*. Prentice-Hall, 1991.
18. G.J. Holzmann. The Model Checker SPIN. *IEEE Transactions on Software Engineering* **23**(5) (1997) 279-295.
19. C. Loiseaux , S. Graf, J. Sifakis, A. Boujjani, S. Bensalem. Property Preserving Abstractions for the Verification of Concurrent Systems. *Formal Methods in System Design* **6** (1995) 1-35.
20. Z. Manna , A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems - Specification*. Springer-Verlag, New York, (1992).
21. C.S. Pasareanu, M.B. Dwyer, W. Visser. Finding Feasible Counter-examples when Model Checking Abstracted Java Programs. In *Proc. of 7th Int. Conf. Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2001*, LNCS-2031, pp. 284-298, (2001).
22. H. Saïdi, N. Shankar. Abstract and model check while you prove. In *Computer-Aided Verification CAV'99* LNCS-1633, pp. 443-454, (1999).
23. H. Saïdi. Model Checking guided abstraction and analysis. In *Seventh International Static Analysis Symposium (SAS'00)*, LNCS-1824, pp. 377-396, (2000).
24. α Spin project. University of Málaga.
<http://www.lcc.uma.es/~gisum/fmse/tools> .