An abstract semantics of temporal logic specifications

Anatol Ursu
Technical University of Moldavia
Computer Science Dept.
bd. Stefan cel Mare 168
2004 Chisinau, Republic of Moldavia
email: ursu@server.utm.md
http://www.lix.polytechnique.fr/Labo/Anatole.Ursu
tel: +373 (2) 49-70-16 fax: +373 (2) 24-90-28

Abstract

In this paper we introduce an abstract semantics of the propositional temporal logic specifications which can be characterized by a class of deterministic Buchi automata, called abstract Buchi automata. A fixpoint characterization of the abstract semantics in the abstract Buchi automata domain is introduced and an abstract satisfiability analysis method is proposed. An abstract interpretation is proposed which allows to approximate correctly the concrete semantics of temporal logic specifications by abstract Buchi automata.

1 Introduction

The semantics of a temporal logic specification is a set of infinite sequences which can be defined by the least (greatest) fixpoint of a monotone function $\mathcal{F} : \wp(\Omega) \mapsto \wp(\Omega)$ defined on the semantic domain of the infinite sequences $\wp(\Omega)$, where Ω is an universe of infinite sequences. This set of infinite sequences can be characterized by a non-deterministic Buchi automaton.

In order to obtain a fixpoint characterization of a semantics using Buchi automata one has to take into account the fact that the Buchi automata are not closed under determinization [McN66] and that the deterministic Buchi automata are less expressive than the non-deterministic ones. This means that using Buchi automata one cannot perform equivalent transformations of the non-deterministic Buchi automata into the deterministic ones and vice versa without loss of information. In other words, the non-deterministic and the deterministic Buchi automata form different domains and for a correct analysis using Buchi automata it is necessary to stay always in the same domain of non-deterministic or deterministic Buchi automata.

The choice of the domain depends on the class of the properties which must be represented using Buchi automata. For the properties represented by the temporal logic specifications the Buchi automata must be non-deterministic. Nevertheless there exist temporal logic specifications which can be characterized by deterministic Buchi automata.

To switch from the domain of the non-deterministic Buchi automata to the domain of the deterministic Buchi automata is possible if the analysis to perform allows to approximate the non-deterministic Buchi automata by the less expressive deterministic ones. The advantage of such an approximation consists in the fact that the analysis in the deterministic Buchi automata domain is simpler and the deterministic automata can be easily used in the synthesis of the system.

In this paper we introduce an abstract semantics of the propositional temporal logic specifications which is characterized by a class of deterministic Buchi automata, called *abstract Buchi automata*. This allows to approximate a temporal logic specification by a *transition program*, which is a set of temporal logic formulas, the semantics of which are characterized by an abstract Buchi automaton. A fixpoint characterization of the abstract semantics in the abstract Buchi automata domain is introduced and an abstract satisfiability analysis method is proposed which is exponential on the number of propositional variables and not on the length of the temporal logic formula. An abstract interpretation [CC92, CC79] is proposed which allows to approximate correctly the concrete semantics of temporal logic specifications by abstract Buchi automata.

The paper is organized as follows. In section 2 a fixpoint characterization of the concrete semantics of the propositional temporal logic formulas is presented. In section 3 we introduce the abstract semantics of a propositional temporal logic formula which is characterized by the abstract Buchi automata. In section 4 the transition programs are introduced, the semantics of which is characterized by abstract Buchi automata. Transition programs are in fact approximations of temporal logic formulas. In section 5 the fixpoint characterization of the abstract semantics and an abstract satisfiability analysis method is proposed which extracts from a propositional temporal logic specification an abstract Buchi automaton. An abstract interpretation which allows to approximate correctly the semantics of the propositional temporal logic specifications is proposed in section 6.

2 The fixpoint characterization of the propositional temporal logic specifications semantics

In this article the notions of linear time propositional temporal logic are supposed to be known [MP95]. We restrict ourself to the future fragment of the propositional temporal logic, namely to the logic of temporal operators $\bigcirc$ (*next*), $\Diamond$ (*sometimes*), $\Box$ (*always*), $\mathcal{U}$ (*until*), $\mathcal{W}$ (*weak until (await)*).

The semantics of a temporal logic formula is a set of infinite sequences of states. Consider a set of propositional variables P and a propositional temporal logic formula φ such that $\mathcal{V}(\varphi) \subseteq P$, where $\mathcal{V}(\varphi)$ is the set of all propositional variables of the formula φ. To introduce the semantics of the propositional temporal logic formula φ consider the set Σ of $2^{|P|}$ different states and the total bijective function $Assign : \Sigma \mapsto 2^P$ which assign to each state $s \in \Sigma$ an interpretation of the variables of the set P. Obviously there exists such a function since there are $2^{|P|}$ different interpretations of the variables from P.

Let $\Omega(P) = \Sigma^\omega$ denote the set of all infinite sequences of states from the alphabet $\Sigma = 2^P$. Then the powerset $\wp(\Omega(P))$ is the *semantic domain of infinite sequences* of the formula φ, such that $\mathcal{V}(\varphi) \subseteq P$.

The semantics $\pi\,[\![\varphi]\!]$ of a formula φ is a subset of $\Omega(P)$ which satisfy the formula, $\pi\,[\![\varphi]\!] \subseteq \Omega(P)$. The semantics of all temporal operators except the $\bigcirc$ *next* operator can be characterized by the least *lfp* or by the greatest fixpoint *gfp* of a continuous function $\lambda x.F(x)$ defined on the set $\wp(\Omega(P))$ [BB86, BB89, Var88].

This characterization can be introduced as follows:

- for an atomic formula (proposition) q we have:

$$\pi\,[\![\varphi]\!] = \{\sigma \in \Omega(P) | \langle\sigma, 0\rangle \models \varphi\},$$
$$\pi\,[\![\neg\varphi]\!] = \{\sigma \in \Omega(P) | \langle\sigma, 0\rangle \not\models \varphi\},$$

  where the expression $\langle\sigma, i\rangle$ denotes the state in the i-th position in the sequence σ.

- for the propositional temporal logic formulas φ and ψ:

$$\pi\,[\![\varphi \vee \psi]\!] = \pi\,[\![\varphi]\!] \cup \pi\,[\![\psi]\!]\,,$$
$$\pi\,[\![\varphi \wedge \psi]\!] = \pi\,[\![\varphi]\!] \cap \pi\,[\![\psi]\!]\,.$$

- the semantics of the temporal operator $\bigcirc$:

$$\pi\,[\![\bigcirc\varphi]\!] = \{X |\, \mathit{sfx}(X) = \pi\,[\![\varphi]\!]\}.$$

  where $\mathit{sfx}(X) = \{\sigma^1 | \sigma^0 \in X\}$ is the constructor of all infinite suffices from the *next* positions of the infinite sequences X, $\sigma^0 = \sigma$, $\sigma^i = \langle\sigma, i\rangle\hat{}\sigma^{i+1}$, and $\hat{}$ is the concatenation operator.

- the fixpoint semantics of the temporal operators $\Diamond, \Box, \mathcal{U}, \mathcal{W}$:

$$\pi\,[\![\Diamond\varphi]\!] = \mathit{lfp}\ \ \lambda X.\pi\,[\![\varphi]\!] \cup \mathit{sfx}(X),$$
$$\pi\,[\![\Box\varphi]\!] = \mathit{gfp}\ \ \lambda X.\pi\,[\![\varphi]\!] \cap \mathit{sfx}(X),$$
$$\pi\,[\![\varphi\mathcal{U}\psi]\!] = \mathit{lfp}\ \ \lambda X.\pi\,[\![\varphi]\!] \cap \mathit{sfx}(X \cup \pi\,[\![\psi]\!]),$$
$$\pi\,[\![\varphi\mathcal{W}\psi]\!] = \pi\,[\![\Box\varphi]\!] \cup \pi\,[\![\varphi\mathcal{U}\psi]\!]\,.$$

Since the models of a temporal logic formula are characterized by a non-deterministic Buchi automaton [VW94, Var96] it is possible to imitate the fixpoint of a formula directly as a Buchi automaton [Dam92]. This means that one can restrict the semantic domain $\wp(\Omega(P))$, which is a complete lattice, to the semantic domain of infinite sequences accepted by Buchi automata (ω-sequences) and to characterize the semantics of a temporal logic specification φ by a fixpoint in the Buchi automata domain $\mathcal{AB}_{\Sigma(\varphi)}(\sqsubseteq, \emptyset, \mathcal{A}^{\omega}_{\Sigma(\varphi)}, \sqcap, \sqcup)$, where the ordering $\sqsubseteq$ is defined by the inclusion of the sets of infinite sequences accepted by the Buchi automata $\mathcal{A} \in \mathcal{AB}_{\Sigma(\varphi)}$. That is, if $\mathcal{A}_i,\ \mathcal{A}_j \in \mathcal{AB}_{\Sigma(\varphi)}$ then

$$\mathcal{A}_i \sqsubseteq \mathcal{A}_j \equiv L(\mathcal{A}_i) \subseteq L(\mathcal{A}_j).$$

The automaton $\mathcal{A}^{\omega}_{\Sigma(\varphi)}$ is the Buchi automaton, such that the language $L(\mathcal{A}^{\omega}_{\Sigma(\varphi)})$ accepted by this automaton is the set $\Sigma^{\omega}(\varphi)$ of all ω-sequences defined on the alphabet $\Sigma(\varphi) = 2^{Var(\varphi)}$,

$$L(\mathcal{A}^{\omega}_{\Sigma(\varphi)}) = \{\sigma \in \Sigma^{\omega}(\varphi)\}.$$

3 The abstract semantics of propositional temporal logic formulas

We propose an abstract semantics of propositional temporal logic specifications which can be characterized by deterministic Buchi automata.

First, let us approximate a set of infinite sequences X by a deterministic Buchi automaton as follows:

- Let $\Sigma = \{s_0, s_1, s_2, \dots\}$ be a finite set of states and $\Omega = \{\sigma_0, \sigma_1, \sigma_2, \dots\}$ - a set of infinite sequences defined on this set of states, and $X \subseteq \Omega$.
- Define the set of initial states as,

$$\mathcal{I} = \{s_i | \exists \sigma \in X \wedge \langle \sigma, 0 \rangle = s_i\}$$

- Define the relation R which corresponds to the set of transitions between the states of the infinite sequences X,

$$R = \{\langle s_i, s_j \rangle | \exists \sigma \in X \wedge \exists k \geq 0 \wedge \langle \sigma, k \rangle = s_i \wedge \langle \sigma, k+1 \rangle = s_j\}.$$

- Using the relation R determine for each state s_i the set of its successors,

$$Succ(s_i) = \{s_j | \langle s_i, s_j \rangle \in R\}.$$

- The set

$$\mathcal{C} = \{Succ(s_1), Succ(s_2), Succ(s_3), \dots\}$$

  is the finite set of all the sets of successors.
- Introduce the set

$$\mathcal{Q} = \mathcal{C} \cup \mathcal{I},$$

  as the set of nodes of an automaton.
- Since the nodes are sets of states there exists the following relation between two nodes $q_i, q_j \in \mathcal{Q}$ with respect to a state s_k,

$$\delta = \{\langle q_i, s_k, q_j \rangle | \exists s_l \cdot s_k \in q_i \wedge s_l \in q_j \ \wedge \langle s_k, s_l \rangle \in R \wedge Succ(s_k) = q_j\}.$$

- Introduce the abstraction function $\alpha = \langle \alpha_\Sigma, \alpha_\mathcal{Q}, \alpha_\delta, \alpha_I, \alpha_F \rangle$ which maps a set of infinite sequences $X \subseteq \Omega$ into a deterministic Buchi automaton:

$$\mathcal{A} = (\alpha_\Sigma(X), \alpha_\mathcal{Q}(X), \alpha_\delta(X), \alpha_I(X), \alpha_F(X)),$$

  where α_Σ is the alphabet, $\alpha_\mathcal{Q}$ is the set of nodes, α_δ is the transition relation between the nodes, α_I is the set of initial nodes, α_F is the set of final nodes of the automaton, such that $\alpha_\Sigma(X) = \Sigma$, $\alpha_\mathcal{Q}(X) = \mathcal{Q}$, $\alpha_\delta(X) = \delta$, $\alpha_I(X) = I$, $\alpha_F(X) = \mathcal{Q}$.

  We call this automaton the *abstract Buchi automaton* of the set of sequences X. It follows from the construction that an abstract Buchi automaton is deterministic.

The abstract semantics of a temporal formula φ can be introduced by the set of infinite sequences which is an upper approximation of the concrete semantics $\pi \llbracket \varphi \rrbracket$ of the formula and which is accepted by an abstract Buchi automaton.

Definition 1 *The abstract semantics of a propositional temporal logic formula φ is the following abstract Buchi automaton,*

$$\mathcal{A}_\varphi = (\alpha_\Sigma(\pi \llbracket \varphi \rrbracket), \alpha_\mathcal{Q}(\pi \llbracket \varphi \rrbracket), \alpha_\delta(\pi \llbracket \varphi \rrbracket), \alpha_I(\pi \llbracket \varphi \rrbracket), \alpha_F(\pi \llbracket \varphi \rrbracket)).$$

Example: Given the set of the infinite sequences $X = \pi\,[\![\varphi]\!]$,

$$\begin{array}{l} s_0 \to s_2 \to s_3 \to s_1 \to s_0 \ldots \\ s_0 \to s_3 \to s_1 \to s_0 \to s_2 \ldots \\ s_0 \to s_1 \to s_2 \to s_1 \to s_0 \ldots \\ \ldots \end{array}$$

which is the set of models of a formula φ. The sequences are defined on the set of states $\Sigma = \{s_0, s_1, s_2, s_3\}$, such that each sequence begins with the state s_0 and every occurrence of the state s_1 is followed by an occurrence of the state s_0.

The abstract semantics $\mathcal{A}_\varphi$ of the formula φ can be obtained from the concrete semantics $\pi\,[\![\varphi]\!]$ as follows:

- The set of initial states,
$$\mathcal{I} = q_0 = \{s_0\},$$
forms the initial node q_0 of the abstract Buchi automaton.

- The transition relation R between the states of the infinite sequences $\pi\,[\![\varphi]\!]$ is the following:
$$\begin{array}{rcl} R & = & \{\langle s_1, s_0\rangle, \\ & & \langle s_0, s_0\rangle, \langle s_0, s_1\rangle, \langle s_0, s_2\rangle, \langle s_0, s_3\rangle, \\ & & \langle s_2, s_0\rangle, \langle s_2, s_1\rangle, \langle s_2, s_2\rangle, \langle s_2, s_3\rangle, \\ & & \langle s_3, s_0\rangle, \langle s_3, s_1\rangle, \langle s_3, s_2\rangle, \langle s_3, s_3\rangle\}. \end{array}$$

- From the relation R the sets of successors states s_0, s_1, s_2, s_3 are determined as follows,
$$\begin{array}{l} Succ(s_0) = \{s_0, s_1, s_2, s_3\}, \\ Succ(s_1) = \{s_0\}, \\ Succ(s_2) = \{s_0, s_1, s_2, s_3\}, \\ Succ(s_3) = \{s_0, s_1, s_2, s_3\}. \end{array}$$
This set of successors constitutes in fact the nodes q_0 and q_1 as follows,
$$\mathcal{C} = \{q_0 = \{s_0\}, q_1 = \{s_0, s_1, s_2, s_3\}\}.$$

- The set of nodes of the abstract Buchi automaton,
$$\mathcal{Q} = \mathcal{C} \cup \mathcal{I} = \{q_0 = \{s_0\}, q_1 = \{s_0, s_1, s_2, s_3\}\}.$$

- The transition relation between the nodes,
$$\delta = \{\langle q_0, s_0, q_1\rangle, \langle q_1, s_0, q_1\rangle, \langle q_1, s_2, q_1\rangle, \langle q_1, s_3, q_1\rangle, \langle q_1, s_1, q_0\rangle\}.$$

- The abstraction function α,
$$\alpha(\pi\,[\![\varphi]\!]) = \mathcal{A}_\varphi = (\Sigma, \mathcal{Q}, \delta, \mathcal{I}, \mathcal{Q}).$$

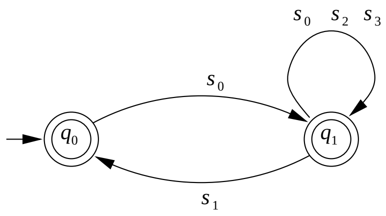

Figure 1: An abstract Buchi automaton

The abstraction function α allows to build the abstract Buchi automaton $\mathcal{A}_\varphi$ (Figure 1) which accepts the set of infinite sequences $L(\mathcal{A}_\varphi)$, where $L(\mathcal{A})$ denotes the set of all infinite sequences accepted by the automaton $L(\mathcal{A})$. From the construction above it follows that,

$$\pi [\![\varphi]\!] \subseteq L(\mathcal{A}_\varphi).$$

Since $L(\mathcal{A}_\varphi)$ is a set of infinite sequences, one can take $L(\mathcal{A}_\varphi)$ as a concretization function γ of the abstract Buchi automaton $\mathcal{A}_\varphi$,

$$\gamma(\mathcal{A}_\varphi) = L(\mathcal{A}_\varphi).$$

It follows that,

$$\pi [\![\varphi]\!] \subseteq \gamma(\mathcal{A}_\varphi),$$

or

$$\pi [\![\varphi]\!] \subseteq \gamma(\alpha(\pi [\![\varphi]\!])).$$

In the example above we have that $\pi [\![\varphi]\!]$ is the semantics of the formula,

$$\begin{aligned} \varphi \quad = \quad & s_0 \wedge \Box((s_1 \rightarrow \bigcirc s_0) \wedge (s_0 \vee s_1 \vee s_2 \vee s_3) \wedge \\ & \neg(s_0 \wedge s_1) \wedge \neg(s_0 \wedge s_2) \wedge \neg(s_0 \wedge s_3) \wedge \neg(s_1 \wedge s_2) \wedge \neg(s_1 \wedge s_3) \wedge \neg(s_2 \wedge s_3)). \end{aligned}$$

It is easy to see that for this formula the abstract and the concrete Buchi automata are identical (Figure 1). That is

$$\pi [\![\varphi]\!] = \gamma(\mathcal{A}_\varphi).$$

Thus,

$$\pi [\![\varphi]\!] = \gamma(\alpha(\pi [\![\varphi]\!])),$$

and the concretization of the abstract semantics of the formula φ is carried out without loss on information.

This observation allows to introduce a specific class of formulas the semantics of which can be characterized by abstract Buchi automata and which can be used for the approximation of the arbitrary temporal logic formulas.

For example, the abstract semantics of the formula,

$$\begin{aligned} \psi \quad = \quad & s_0 \wedge \Box((s_1 \rightarrow \bigcirc s_0) \wedge (s_1 \rightarrow \bigcirc \Diamond s_2) \wedge (s_0 \vee s_1 \vee s_2 \vee s_3) \wedge \\ & \neg(s_0 \wedge s_1) \wedge \neg(s_0 \wedge s_2) \wedge \neg(s_0 \wedge s_3) \wedge \neg(s_1 \wedge s_2) \wedge \neg(s_1 \wedge s_3) \wedge \neg(s_2 \wedge s_3)), \end{aligned}$$

is characterized by the same abstract Buchi automaton, Figure 1. The concretization of this automaton, however, does not correspond any more to the concrete semantics of the formula φ, since

$$\pi [\![\psi]\!] \subset \gamma(\alpha(\pi [\![\varphi]\!])).$$

Thus, the formula φ can be used as an upper approximation of the formula ψ. We introduce this class of approximation formulas as transition specifications.

4 The transition specifications and the transition programs

Definition 2 *A transition specification Spec is a propositional temporal logic formula of the form*

$$\Theta \wedge \tau_1 \wedge \tau_2 \wedge \ldots \tau_m \wedge \mathcal{R},$$

where

- Θ *is an initial condition which is a formula of the form*

$$\neg Pr \; \mathcal{U} \; \theta,$$

- $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_m\}$ *is a set of transition formulas,*

$$\tau \doteq \Box(a \rightarrow \bigcirc(\neg Pr \; \mathcal{U} \; c)),$$

- $\mathcal{R}$ *is a restriction, which is an invariant formula of the form* $\Box r$*, where* r *is an assertion.*

Pr, θ, a, c, r *are propositional formulas defined on a set of propositional variables* $\mathcal{E} = \{x_1, x_2, \ldots, x_n\}$*, called event variables, and* $Pr = \bigvee_{i=1}^{n} x_i$.

We will distinguish an antecedent a and a consequent subformula c in a transition formula. The function Pr is introduced for the modular composition of transion specifications defined on different subsets of event variables. The assertion r can be used, for example, to specify deterministic transition programs, each transition of which is identified by an individual event variable. In this case r will represent the expression

$$(\bigvee_{1 \leq i \leq n} x_i) \wedge (\bigwedge_{1 \leq i < j \leq n} \neg(x_i \wedge x_j)),$$

called the *single event condition.*

For the transition specifications more convenient will be the form with each transition formula in the conjunction written as a separate formula.

Definition 3 *A transition program is the set of the formulas of a transition specification. A transition program is a structured representation of a transition specification.*

An example of a transition program:

$$\neg Pr\ \mathcal{U}\ (x_1 \vee x_4) \qquad \text{-the initial condition;}$$

$$\left.\begin{array}{l} \Box((x_1 \vee x_3) \rightarrow \bigcirc(\neg Pr\ \mathcal{U}\ (x_2 \vee x_3))), \\ \Box((x_2 \vee x_4) \rightarrow \bigcirc(\neg Pr\ \mathcal{U}\ (x_1 \vee x_4))), \\ \Box((x_1 \vee x_4) \rightarrow \bigcirc(\neg Pr\ \mathcal{U}\ (x_1 \vee x_3))), \end{array}\right\} \qquad \text{-the transition formulas;}$$

$$\left.\begin{array}{l} \Box((x_1 \vee x_2 \vee x_3 \vee x_4) \wedge \neg(x_1 \wedge x_2) \wedge \\ \neg(x_1 \wedge x_3) \wedge \neg(x_1 \wedge x_4) \wedge \neg(x_2 \wedge x_3) \wedge \\ \neg(x_2 \wedge x_4) \wedge \neg(x_3 \wedge x_4)) \end{array}\right\} \qquad \text{-the single event condition.}$$

The fact that transition programs approximate temporal logic formulas can be formulated by the following theorem:

Theorem 1 *Given a propositional temporal logic formula $\varphi(x_1, x_2, \ldots, x_n)$. If the formula*

$$\psi(x_1, x_2, \ldots, x_n) = \varphi(x_1, x_2, \ldots, x_n) \wedge ((\bigvee_{1 \leq i \leq n} x_i) \wedge (\bigwedge_{1 \leq i < j \leq n} \neg(x_i \wedge x_j))$$

is satisfiable then there exists a transition program Spec, the semantics of which $\pi \llbracket Spec \rrbracket$ is identical to the abstract semantics $\alpha(\pi \llbracket \psi \rrbracket)$ of the foirmula $\psi(x_1, x_2, \ldots, x_n)$, $\alpha(\pi \llbracket \psi \rrbracket) \equiv \pi \llbracket Spec \rrbracket$.

Proof: For each satisfiable propositional temporal logic formula ψ there exists a set of infinite sequences $\pi \llbracket \psi \rrbracket$. Since the expression

$$(\bigvee_{1 \leq i \leq n} x_i) \wedge (\bigwedge_{1 \leq i < j \leq n} \neg(x_i \wedge x_j))$$

represents the fact that only one propositional variable can be true at a time, every position $\langle \sigma, i \rangle$ of an infinite sequence σ corresponds to a state $s_k = \langle \sigma, i \rangle$ which assign the true value to exactly one variable. Without loss of generality select such an enumeration of the states, for which the number of a state s_k corresponds to the number of the propositional variable x_k which is true in this state, i.e. $s_k \models x_k$.

For this set of sequences there exists an abstract Buchi automaton $\mathcal{A}_\psi = (\Sigma(\varphi), \mathcal{Q}, \delta, \mathcal{I}, F)$. This automaton corresponds to a transition program *Spec*, so that $\pi \llbracket Spec \rrbracket = L(\mathcal{A}_\psi)$. This transition program can be extracted directly from the abstract Buchi automaton as follows:

- For the initial node of the automaton $\mathcal{I} = \{q_0\}$ the set of variables $\mathcal{V}_{q_0}$ is determined,

  $$\mathcal{V}_{q_0} = \{s_i | \exists q_j \wedge \langle q_0, s_i, q_j \rangle \in \delta\}$$

  and the initial condition formula is built up,

  $$\neg Pr\ \mathcal{U}\ (s_1 \vee s_2 \vee \cdots \vee s_k),$$

  where $s_1, s_2, \ldots, s_k \in \mathcal{V}_{q_0}$.

- For each node $q_l \in \mathcal{Q}$ determine

– the set of variables corresponding to the entering arcs $\mathcal{V}_{q_l(in)}$,

$$\mathcal{V}_{q_l(in)} = \{s_i | \exists q_j \wedge \langle q_j, s_i, q_l \rangle \in \delta\},$$

– the set of variables corresponding to the leaving arcs $\mathcal{V}_{q_l(out)}$,

$$\mathcal{V}_{q_l(out)} = \{s_i | \exists q_j \wedge \langle q_l, s_i, q_j \rangle \in \delta\},$$

and build up the transition formula,

$$\Box((\bigvee_{s_i \in \mathcal{V}_{q_l(in)}} s_i) \rightarrow \bigcirc(\neg Pr\ \mathcal{U}\ (\bigvee_{s_i \in \mathcal{V}_{q_l(out)}} s_i))).$$

- Introduce the following restriction formula:

$$\Box((\bigvee_{1 \le i \le n} s_i) \wedge (\bigwedge_{1 \le i < j \le n} \neg(s_i \wedge s_j))).$$

Since each state s_i can be replaced by a variable x_i, the following transition program is obtained:

$$\begin{array}{rcl} Spec & = & \neg Pr\ \mathcal{U}\ (\bigvee_{x_i \in \mathcal{V}_{q_0}} x_i) \\ \bigwedge_{q_l \in \mathcal{Q}} & & \Box((\bigvee_{x_i \in \mathcal{V}_{q_l(in)}} x_i) \rightarrow \bigcirc(\neg Pr\ \mathcal{U}(\bigvee_{x_i \in \mathcal{V}_{q_l(out)}} x_i))) \\ \wedge & & \Box((\bigvee_{1 \le i \le n} x_i) \wedge (\bigwedge_{1 \le i < j \le n} \neg(x_i \wedge x_j))). \end{array}$$

■

Corollary 1 *The semantics of a propositional temporal logic formula including the single event condition can by approximated by the Buchi automaton of a transition program. That is, a propositional temporal logic formula with the single event condition can be approximated by a transition program.*

The proof of the Theorem 1 shows how to obtain a transition program, the semantics of which is identical to the abstract semantics of a temporal logic formula with the single event condition. The condition that only one propositional variable can be true at a time has been used to obtain an abstract Buchi automaton, each transition of which is labeled by a propositional variable. In this case the antecedents and the consequents of the transition formulas of the transition program are disjunctions of propositional variables.

The Theorem 1 can be generalized for arbitrary temporal logic formulas (without the single event condition).

Theorem 2 *For every propositional temporal logic formula $\varphi(x_1, x_2, \ldots, x_n)$ there exists a transition program Spec, such that the semantics $\pi [\![Spec]\!]$ of this transition program is identical to the abstract semantics $\alpha(\pi [\![\varphi]\!])$ of the formula $\varphi(x_1, x_2, \ldots, x_n)$, $\alpha(\pi [\![\varphi]\!]) \equiv \pi [\![Spec]\!]$.*

Proof Sketch: To imitate the proof of the Theorem 1 for a propositional temporal logic formula $\varphi(x_1, x_2, \ldots, x_n)$ without the single event condition consider all the conjunctions of propositional variables and of negated propositional variables from $Var(\varphi)$ and codify each state by $s_k = \langle \sigma, i \rangle$ by a conjunction y_k, such that $s_k \models y_k$. Obviously there are 2^n such conjunctions. Then the following transition program can be obtained, the semantics of which is characterized by the abstract Buchi automaton of the formula φ,

$$\begin{array}{rcl} Spec & = & \neg Pr\ \mathcal{U}\ (\bigvee_{y_i \in \mathcal{V}_{q_0}} y_i) \\ & \bigwedge_{q_l \in \mathcal{Q}} & \Box((\bigvee_{y_i \in \mathcal{V}_{q_l(in)}} y_i) \to \bigcirc(\neg Pr\ \mathcal{U}(\bigvee_{y_i \in \mathcal{V}_{q_l(out)}} y_i))) \\ & \wedge & \Box((\bigvee_{1 \le i \le 2^n} y_i) \wedge (\bigwedge_{1 \le i < j \le 2^n} \neg(y_i \wedge y_j))). \end{array}$$

■

Corollary 2 *The semantics of a propositional temporal logic formula can by approximated by the Buchi automaton of a transition program. That is, a propositional temporal logic formula can be approximated by a transition program.*

5 The fixpoint characterization of the abstract semantics of temporal logic specifications

The abstract semantics of a temporal logic specification φ is characterized by a deterministic Buchi automaton $\mathcal{A} = (\Sigma(\varphi), \mathcal{Q}, \delta, \mathcal{I}, F)$, defined on the alphabet $\Sigma(\varphi) = 2^{Var(\varphi)}$, which is the set of states defined as interpretations of the propositional variables of the specification φ. By the construction of the abstract semantics this automaton has the following properties: the automaton is deterministic; each node of the automaton is uniquely identified by a set of states; each transition of the automaton is labeled by a state; the set of labels of the transitions leaving a node is equal to the set of states identifying the node; there exists one initial node; all the nodes are terminal, $F = \mathcal{Q}$.

These properties of the abstract Buchi automata defined on the alphabet $\Sigma(\varphi) = 2^{Var(\varphi)}$, allow to consider the following abstract semantic domain, which is a complete finite lattice of abstract Buchi automata,

$$\mathcal{DB}_{\Sigma(\varphi)}(\sqsubseteq^\natural, \mathcal{A}_\bot, \mathcal{A}_\top, \sqcap, \sqcup),$$

where the ordering $\sqsubseteq^\natural$ is defined by the inclusions, $\mathcal{A}_i \sqsubseteq^\natural \mathcal{A}_j \equiv \mathcal{I}_i \subseteq \mathcal{I}_j \wedge \mathcal{Q}_i \subseteq \mathcal{Q}_j \wedge \delta_i \subseteq \delta_j$, for $\mathcal{A}_i = (\Sigma(\varphi), \mathcal{Q}_i, \delta_i, \mathcal{I}_i, \mathcal{Q}_i), \mathcal{A}_j = (\Sigma(\varphi), \mathcal{Q}_j, \delta_j, \mathcal{I}_j, \mathcal{Q}_j) \in \mathcal{DB}_{\Sigma(\varphi)}$; $\mathcal{A}_\bot = (\Sigma(\varphi), \emptyset, \emptyset, \emptyset, \emptyset)$, $\mathcal{A}_\top = (\Sigma(\varphi), \wp(\Sigma(\varphi)), \mathbf{1}, \{\Sigma(\varphi)\}, \wp(\Sigma(\varphi)))$, where $\mathbf{1}$ denotes the unitary transition relation, which represents all the possible transitions between the nodes of the automaton.

The sets of infinite sequences accepted by the abstract Buchi automata $\mathcal{A}_\bot$ and $\mathcal{A}_\top$ are the empty set and the set of all sequences on the alphabet $\Sigma(\varphi)$, that is $L(\mathcal{A}_\bot = \emptyset$ and $L(\mathcal{A}^\omega_{\Sigma(\varphi)}) = \{\sigma \in \Sigma^\omega(\varphi)\}$ respectively, where $\Sigma^\omega(\varphi)$ is the set of all ω-sequences defined on the alphabet $\Sigma(\varphi)$.

The ordering $\sqsubseteq^\natural$ on abstract Buchi automata domain induces the inclusion of the sets of infinite sequences accepted by these automata, as stated by the following theorem.

Theorem 3 *If* $\mathcal{I}_i \subseteq \mathcal{I}_j \wedge \mathcal{Q}_i \subseteq \mathcal{Q}_j \wedge \delta_i \subseteq \delta_j$, *then* $L(\mathcal{A}_i) \subseteq L(\mathcal{A}_j)$ *where* $\mathcal{A}_i = (\Sigma(\varphi), \mathcal{Q}_i, \delta_i, \mathcal{I}_i, \mathcal{Q}_i)$, $\mathcal{A}_j = (\Sigma(\varphi), \mathcal{Q}_j, \delta_j, \mathcal{I}_j, \mathcal{Q}_j) \in \mathcal{DB}_{\Sigma(\varphi)}$

Proof Hints:

1. If $\mathcal{I}_i \subseteq \mathcal{I}_j$ and $\mathcal{Q}_i = \mathcal{Q}_j$ and $\delta_i = \delta_j$ then $prefix(L(\mathcal{A}_i)) \subseteq prefix(L(\mathcal{A}_j))$, where $prefix(L(\mathcal{A}))$ is the set of all finite prefixes of the infinite sequences from $L(\mathcal{A})$. This means that $L(\mathcal{A}_i) \subseteq L(\mathcal{A}_j)$.
2. If $\mathcal{I}_i = \mathcal{I}_j$ and $\mathcal{Q}_i \subseteq \mathcal{Q}_j$ and $\delta_i \subseteq \delta_j$ then from the properties of abstract Buchi automata follows that $L(\mathcal{A}_i) \subseteq L(\mathcal{A}_j)$.

■

A fixpoint characterization of the abstract semantics of the temporal logic specification φ can be obtained by a monotone function $\mathcal{F}^\natural : \mathcal{DB}_{\Sigma(\varphi)} \mapsto \mathcal{DB}_{\Sigma(\varphi)}$ which allows to calculate iteratively the least or the greatest fixpoint on the abstract domain.

This function can be obtained as an incremental satisfiability analysis procedure [MAZ83, MW84, Jan90, MP91, KMMP93] which allows to obtain incrementally the Buchi automaton of the specification. The incremental procedures of these methods are in fact such monotone functions.

Intuitively the abstract Buchi automaton $\mathcal{A}_\varphi = (\Sigma(\varphi), \mathcal{Q}, \delta, \mathcal{I}, \mathcal{Q})$ which characterizes the abstract semantics of a propositional temporal logic formula φ, must be obtained as a limit of the sequence:

$$\mathcal{A}_0 \sqsubseteq^\natural \mathcal{A}_1 \sqsubseteq^\natural \mathcal{A}_2 \sqsubseteq^\natural \cdots \sqsubseteq^\natural \mathcal{A}_n = \mathcal{A}_\varphi,$$

such that $\mathcal{A}_0 = (\Sigma(\varphi), \{\mathcal{I}\}, \emptyset, \mathcal{I}, \{\mathcal{I}\})$, $\mathcal{A}_i = (\Sigma, \mathcal{Q}_i, \delta_i, \mathcal{I}, \mathcal{Q}_i)$, $\mathcal{Q}_i = \mathcal{Q}_{i-1} \cup Succ(\varphi, \mathcal{Q}_{i-1})$, $\delta_i \subseteq \delta$, where δ_i is the projection of δ on the set of nodes $\mathcal{Q}_i = Succ(\varphi, \mathcal{Q}_{i-1})$. The function $Succ(\varphi, \mathcal{Q})$ generates the successors of the set of nodes $\mathcal{Q}$ of the Buchi automaton $\mathcal{A}$ using the specification φ. This limit is the fixpoint of the equation $X = \mathcal{F}^\natural(X)$ of the abstract satisfiability analysis method.

In the method presented below the decision procedure corresponds to a monotone function $\mathcal{F}^\natural$ which extracts from the specifications a new abstract Buchi automaton extending a previous abstract Buchi automaton by a set of nodes and updating of the transition relation δ.

5.1 The abstract satisfiability analysis

We propose in this section an incremental abstract satisfiability analysis method which extracts from a temporal logic specification an abstract Buchi automaton. The method uses the terminology of the method of Janssen [Jan90].

Let us start by a short introduction of the method presented in [Jan90]. The method uses an expansion function E which represents a reduced propositional temporal logic formula φ as a disjunction of conjunctions of the form $f \wedge \bigcirc g$,

$$E(\varphi) = \bigvee_{i \in I} (f_i \wedge \bigcirc g_i),$$

where f is a conjunction of propositional variables and negated propositional variables, g stands for a conjunction of certain subformulas of the formula φ and I is the set of conjunctions $f \wedge \bigcirc g$ in the expansion expression of the formula φ.

The function constructs an expansion graph which is the model graph of the temporal logic formula. The expansion graph is built up as follows:

1. Initially the graph consists of one node labeled by the set of formulas $\{f_i^0 \wedge \bigcirc g_i^0 | i \in I_0\}$, such that $E(\varphi) = \bigvee_{i \in I_0}(f_i^0 \wedge \bigcirc g_i^0)$.
2. For each conjunction $f_i^0 \wedge \bigcirc g_i^0$:
   (a) The expansion $E(g_i^0) = \bigvee_{i \in I_1}(f_i^1 \wedge \bigcirc g_i^1)$ is carried out and a new node labeled by the set of formulas $\{f_i^1 \wedge \bigcirc g_i^1 | i \in I_1\}$ is created.
   (b) If there already exists a node labeled by the set of formulas $\{f_i^1 \wedge \bigcirc g_i^1 | i \in I_1\}$ the nodes are collapsed into a single node.
   (c) The original and the created nodes are connected by an arc labeled by the expression $(\bigcirc g_i^0, \bigcirc g_k^1)_{f_i^0}$.
3. For each new node of the expansion graph the expansion of the formulas is repeated until a fixpoint is obtained.
4. Eliminate all the nodes without outputs and all the adjacent to these nodes arcs. The obtained graph is the model graph of the formula φ.

This model graph is not a Buchi automaton since special acceptance conditions are required to accept a model of the temporal logic formula. A model of the temporal formula φ corresponds to an infinite path in the model graph which begins at the node labeled by the set of formulas $\{f_i^0 \wedge \bigcirc g_i^0 | i \in I_0\}$, where $\varphi = \bigvee_{i \in I_0}(f_i^0 \wedge \bigcirc g_i^0)$, and which satisfies all the eventualities of the formula. The path consists of a sequence of arcs labeled by the expressions of the form $(\bigcirc g_k^i, \bigcirc g_r^{i+1})_{f_k^i}$, where f_k^i is a label, which is a conjunction of propositional variables and negated propositional variables. A label denotes a state. The eventualities are satisfied if all the eventualities of the nodes of the model graph are satisfied. The eventualities of the nodes of the model graph are satisfied if there exists an infinite path beginning at this node and such that the set of the propositional variables of the eventualities of the node is a subset of the set of variables appearing as arc labels on the path.

To introduce the abstract satisfiability method we introduce a modified expansion function $E^\natural$ which transforms a propositional temporal formula into a Temporal Normal Disjunction Form.

Definition 4 *A Temporal Disjunction Normal Form (TDNF) of a propositional temporal logic formula φ is the decomposition of the formula in the form*

$$\varphi = \bigvee_{i \in I}(f_i \wedge \bigcirc G_i),$$

such that

$$\left(\bigwedge_{1 \leq i < j \leq I} \neg(f_i \wedge f_j) \right) = true,$$

where f_i is the conjunction of propositional variables and of negated propositional variables of the formula φ such that $\bigvee_{i \in I} f_i$ is a Disjunction Normal Form (DNF), G_i is a disjunction of the conjunctions of subformulas of the formula φ.

The abstract satisfiability analysis method proposed below uses the abstract expansion function $E^{\natural}(\varphi)$,

$$E^{\natural}(\varphi) = \bigvee_{i \in I} (f_i \wedge \bigcirc G_i).$$

The property of this function that $\left(\bigwedge_{1 \leq i < j \leq I} \neg (f_i \wedge f_j)\right) = true$ is important since it allows to obtain a deterministic expansion graph corresponding to an abstract Buchi automaton of the temporal logic formula φ.

5.1.1 The abstract satisfiability analysis method

The abstract satisfiability analysis method uses the expansion function $E^{\natural}(\varphi)$ to extract the deterministic Buchi automaton of a propositional temporal logic formula φ as a fixpoint of the following expansion graph construction procedure:

1. Initially the expansion graph consists of one node marked *new* and labeled by the sets of formulas:
   (a) The set of formulas $\{f_i^0 \wedge \bigcirc G_i^0 | i \in I_0\}$, such that $E^{\natural}(\varphi) = \bigvee_{i \in I_0} (f_i^0 \wedge \bigcirc G_i^0)$, which are the conjunctions of the TNDF.
   (b) The set of formulas $\{f_i^0 | i \in I_0\}$, which are the conjunctions of the form DNF.
2. For each *new* node labeled by the sets of formulas $\{f_i^k \wedge \bigcirc G_i^k | i \in I_k\}$ and $\{f_i^k | \in I_k\}$:
   (a) Delete the mark *new*.
   (b) For each conjunction $f_i^k \wedge \bigcirc G_i^k$:
      i. The expansion $E^{\natural}(G_i^k) = \bigvee_{i \in I_{k+1}} (f_i^{k+1} \wedge \bigcirc G_i^{k+1})$ is carried out and a *new* node labeled by two sets of formulas is created:
         A. The set of formulas $\{f_i^{k+1} \wedge \bigcirc g_i^{k+1} | \in I_{k+1}\}$ of the form TDNF.
         B. The set of formulas $\{f_i^{k+1} | i \in I_{k+1}\}$ of the form DNF.
      ii. The original and the created nodes a connected by an arc labeled by the expression $(\bigcirc g_i^k, \bigcirc g_j^{k+1})_{f_i^k}$.
      iii. If there already exists a node to which points an arc labeled by an expression $(\bigcirc x, \bigcirc y)_{f_i^k}$, where x and y are temporal formulas, the *new* and the existing nodes are collapsed. To combine the *new* and the existing nodes the disjunction of the two forms TDNF is normalized to obtain a new TDNF. If the new TDNF form differs from the old TDNF form the node is marked as *new*.
3. For each new node of the expansion graph the expansion of the formulas is repeated until a fixpoint is obtained.
4. Eliminate all the nodes without outputs and all the adjacent to these nodes arcs. The obtained graph is the model graph of the formula φ. This model graph is the abstract Buchi automaton of the formula φ.

Theorem 4 *The maximal number of the nodes of the model graph generated by the abstract satisfiability analysis method is bounded by 2^n, where n is the number of the propositional variables of the analyzed formula φ.*

Proof: According to the decision procedure of this method each node of the expansion graph is uniquely identified by the arcs which point to these nodes, namely by the formulas f on the labels $(\bigcirc x, \bigcirc y)_f$ of these arcs. Since a formula f is a conjunction of propositional variables and negated propositional variables there are 2^n of such formulas. Thus the maximal number of nodes of the expansion graph is 2^n.

■

From this theorem it follows that the complexity of the abstract satisfiability analysis method is exponential on the number of propositional variables and not on the length of the formula.

The abstract satisfiability analysis method is in fact a decision procedure based on a monotone function $\mathcal{F}^\natural : \mathcal{DB}_{\Sigma(\varphi)} \mapsto \mathcal{DB}_{\Sigma(\varphi)}$, which allows to determine the abstract Buchi automaton $\mathcal{A}_\varphi$ of a temporal logic formula φ as a least fixpoint $\mathcal{A}_\varphi = \mathit{lfp}\, \mathcal{F}^\natural$ in the abstract semantic domain $\mathcal{DB}_{\Sigma(\varphi)}(\sqsubseteq^\natural, \mathcal{A}_\bot, \mathcal{A}_\top, \sqcap, \sqcup)$.

6 Semantic approximation of temporal logic formulas using Abstract Interpretation

To find a correct semantic approximation of a temporal logic formula φ with $\mathcal{V}(\varphi) \in P$, where P is a set of propositional variables, we use a Galois connection [CC92].

Consider the following Galois connection:

$$(\wp(\Omega(P)), \subseteq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} (\mathcal{DB}_\Sigma, \sqsubseteq^\natural),$$

which, by definition, is a pair of maps $\alpha \in \wp(\Omega(P)) \mapsto \mathcal{DB}_\Sigma$ and $\gamma \in \mathcal{DB}_\Sigma \mapsto \wp(\Omega(P))$ between the set $\wp(\Omega(P))$ of subsets of the set $\Omega(P)$ of infinite sequences, ordered by the inclusion $\subseteq$, and the set $\mathcal{DB}_\Sigma$ of the abstract Buchi automata defined on the alphabet $\Sigma = 2^P$, ordered by the relation $\sqsubseteq^\natural$, such that,

$$\forall x \in \wp(\Omega(P)) : \forall y \in \mathcal{DB}_\Sigma : \alpha(x) \sqsubseteq^\natural \mathcal{DB}_\Sigma \Longleftrightarrow \gamma(y) \subseteq \wp(\Omega(P)).$$

The function α is in fact the abstraction function $\alpha = \langle \alpha_\Sigma, \alpha_\mathcal{Q}, \alpha_\delta, \alpha_I, \alpha_F \rangle$ introduced in section 3. It maps a set of infinite sequences $X \subseteq \Omega(P)$ into an abstract Buchi automaton $\mathcal{A}_X$:

$$\mathcal{A}_X = \alpha(X) = (\alpha_\Sigma(X), \alpha_\mathcal{Q}(X), \alpha_\delta(X), \alpha_I(X), \alpha_F(X)).$$

The function γ can be defined by the function L, such that $L(\mathcal{A}_X)$ is the set of infinite sequences accepted by the Buchi automaton $\mathcal{A}_X$,

$$\gamma(\mathcal{A}_X) = L(\mathcal{A}_X).$$

Theorem 5 *The concrete semantic domain $\wp(\Omega(P))$ and the abstract semantic domain $\mathcal{DB}_\Sigma$ with the abstraction function $\alpha(X) = (\alpha_\Sigma(X),\ \alpha_\mathcal{Q}(X),\ \alpha_\delta(X),\ \alpha_I(X),\ \alpha_F(X))$ and the concretization function $\gamma(\mathcal{A}_X) = L(\mathcal{A}_X)$ constitute a Galois connection.*

Proof: To show that the functions α and γ form a Galois connection it is necessary to prove that the properties of a Galois connection are satisfied as formulated in [CC92]

- The functions α and γ are monotone:

– the function α is monotone since for each pair of sets of $X_1,\ X_2 \subseteq \Omega(P)$ of infinite sequences if $X_1 \subseteq X_2$, then $\mathcal{A}_{X_1} \sqsubseteq^\natural \mathcal{A}_{X_2}$ by the definition of the approximation of a set sequences by an abstract Buchi automaton and hence $\alpha(X_1) \sqsubseteq^\natural \alpha(X_2)$.

– the function γ is monotone since for each pair of abstract Buchi automata $\mathcal{A}_1,\ \mathcal{A}_2 \in \mathcal{DB}_\Sigma$, if $\mathcal{A}_1 \sqsubseteq^\natural \mathcal{A}_2$, then $L(\mathcal{A}_1) \subseteq L(\mathcal{A}_2)$ by the definition of the ordering $\sqsubseteq^\natural$, and hence $\gamma(\mathcal{A}_1) \subseteq \gamma(\mathcal{A}_2)$.

- $\gamma \circ \alpha$ is extensive:
$$\forall X \in \wp(\Omega(P)) : X \subseteq \gamma \circ \alpha(X).$$
This condition is satisfied due to the property of an abstract Buchi automaton $\mathcal{A}_X = \alpha(X)$ of a set of infinite sequences X, which accepts a set of infinite sequences Y greater or equal to the set X, $Y = \gamma(\alpha(X)) = \gamma(\mathcal{A}_X) = L(\mathcal{A}_X) \supseteq X$.

- $\alpha \circ \gamma$ is reductive:
$$\forall \mathcal{A} \in \mathcal{DB}_\Sigma : \alpha \circ \gamma(\mathcal{A}) \subseteq \mathcal{A}.$$
This property is satisfied since the set of infinite sequences $L(\mathcal{A})$, accepted by an abstract Buchi automaton $\mathcal{A}$, corresponds to the models Y of a transition program *Spec*, $\pi\, [\![Spec]\!] = Y$. From the Theorem 2 follows that the concrete and the abstract semantics of a transition program is characterized by the same Buchi automaton which is an abstract Buchi automaton $\mathcal{A}$, $\alpha(Y) = \alpha(L(\mathcal{A})) = \alpha(\gamma(\mathcal{A})) = \mathcal{A}$.

■

7 Conclusions

The satisfiability analysis of temporal logic specifications deals with two major problems, the exponential blowup of the decision procedures and the fact that Buchi automata which are used to characterize the models of temporal logic specifications are not closed under determinization. Both these problems have a serious impact on the application of the temporal logic specification in the systems design. The second problem means that using Buchi automata one cannot perform equivalent transformations of the non-deterministic Buchi automata into deterministic ones and vice versa without loss of information and that for a correct analysis it is necessary to stay always in the same domain of non-deterministic or deterministic Buchi automata. We proposed an abstract semantics and an abstract interpretation that allows to approximate the concrete semantics of temporal logic specifications by a class of deterministic Buchi automata which accept an upper approximation of the ω-languages accepted by non-deterministic Buchi automata. The decision procedure in the abstract semantic domain has the exponential complexity on the number of the propositional variables and not on the length of the temporal logic formula.

References

[BB86] Behnam Banieqbal and Howard Barringer. A Study of an Extended Temporal Logic and a Temporal Fixed Point Calculus. Technical report, University of Manchester, Manchester, 1986.

[BB89] Behnam Banieqbal and Howard Barringer. Temporal logic with fixed points. In B. Banieqbal, H. Barringer, and A. Pnueli, editors, *Proceedings of the Conference on Temporal Logic in Specification*, volume 398 of *LNCS*, pages 62–74, Berlin, April 1989. Springer-Verlag.

[CC79] Patrick Cousot and Radhia Cousot. Systematic design of program analysis frameworks. In *Conference Record of the Sixth Annual ACM Symposium on Principles of Programming Languages*, pages 269–282, San Antonio, Texas, January 1979.

[CC92] Patrick Cousot and Radhia Cousot. Abstract Interpretation and applications to logic programs. *Journal of Logic Programming*, 13(2-3):103–180, July 1992.

[Dam92] Mads Dam. Fixed points of Büchi automata. In Rudrapatna Shyamasundar, editor, *Proceedings of Foundations of Software Technology and Theoretical Computer Science*, volume 652 of *LNCS*, pages 39–50, Berlin, Germany, December 1992. Springer.

[Jan90] Geert Janssen. Hardware verification using temporal logic: A practical view. In *Formal VLSI Correctness Verification, VLSI Design Methods-II.* Elsevier Science Publishers B. V., (North-Holland), ifip edition, 1990.

[KMMP93] Yonit Kesten, Zohar Manna, Hugh McGuire, and Amir Pnueli. A Decision Algorithm for Full Propositional Temporal Logic. In *5th Conference on Computer Aided Verification*, volume 697 of *LNCS*, pages 97–109. Springer-Verlag, 1993.

[MAZ83] Mordechai Ben-Ari, Amir Pnueli, and Zohar Manna. The Temporal Logic of Branching Time. *Acta Informatica*, 20(3):207–226, 1983.

[McN66] Robert McNaughton. Testing and generating infinite sequences by a finite automaton. *Information and Control*, 9(5):521–530, October 1966.

[MP91] Zohar Manna and Amir Pnueli. *Temporal Logic of Reactive and Concurrent Systems: Specification.* Springer-Verlag, New York, 1991.

[MP95] Zohar Manna and Amir Pnueli. *Temporal Verification of Reactive Systems: Safety.* Springer-Verlag, New York, 1995.

[MW84] Zohar Manna and Pierre Wolper. Synthesis of Communicating Processes from Temporal Logic Specifications. *ACM Transactions on Programming Languages and Systems*, 6(1):68–93, January 1984.

[Var88] Moshe Y. Vardi. A temporal fixpoint calculus. In ACM-SIGPLAN ACM-SIGACT, editor, *Conference Record of the 15th Annual ACM Symposium on Principels of Programming Languages (POPL '88)*, pages 250–259, San Diego, CA, USA, January 1988. ACM Press.

[Var96] Moshe Y. Vardi. An automata-theoretic approach to linear temporal logic. In *Logics for Concurrency*, volume 1043 of *LNCS*, pages 238–266, 1996.

[VW94] Moshe Y. Vardi and Pierre Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1–37, 15 November 1994.