Refinement for Symbolic Trajectory Evaluation

Authors

Chalmers

Abstract. Model refinement such that it preserves symbolic trajectory evalutions.

Keywords: STE · Refinement · ?

1 Introduction to STE

1.1 Original STE

Symbolic trajectory evaluation [5] (STE) is a high-performance model checking technique based on *symbolic simulation* extended with a temporal *next-time* operator to describe circuit behaviour over time. In its simplest form, STE tests the validity of an *assertion* of the form $A \Rightarrow C$, where both the *antecedent* A and *consequent* C are formulas in the following logic:

$$f ::= p \mid f \wedge f \mid P \rightarrow f \mid \mathbf{N}\, f$$

Here, p is a simple predicate over "values" in a circuit and P is a Boolean propositional formula, and the operators $\wedge$, $\rightarrow$ and $\mathbf{N}$ are conjunction, domain restriction and the next-time operator, respectively.

If the circuit contains Boolean signals, p is typically drawn from the following two predicates: $n\,\mathbf{is}\,1$ and $n\,\mathbf{is}\,0$, where n ranges over the signals (or nodes) in a circuit. For example, suppose we have a unit-delayed, two-input AND-gate, then it is reasonable to assume that the assertion $(in_1\,\mathbf{is}\,1 \wedge in_2\,\mathbf{is}\,1) \Rightarrow \mathbf{N}(out\,\mathbf{is}\,1)$ is true. Indeed, STE efficiently validates such statements for us.

While the truth semantics of an assertion in STE is defined as the satisfaction of its "defining" trajectory (bounded sequence of states) relative to a model structure of the circuit, what the STE algorithm computes is exactly the solution of a data-flow equation [1] in the classic format [4]. ...

1.2 Lattice-theoretic STE

Consider an arbitrary, but fixed, digital circuit M operating in discrete time. A *configuration* of M, denoted by C, is non-empty and finite set that represents a snapshot of M at a discrete point in time. If the circuit M has m boolean signals, then its set of configurations is typically represented as a sequence $\mathbb{B}^m$, where $\mathbb{B} = \{0, 1\}$ is the set of boolean values.

Circuit Model A simple conceptual model of M is a *transition relation*, $M_R \subseteq C \times C$, where $(c, c') \in M_R$ means that M can move from c to c' in one step[1]. The power set of C, denoted by $\mathcal{P}(C)$, can be viewed as a the set of *predicates* on configurations, where $\cap$, $\cup$, and $\subseteq$ correspond to conjunction, disjunction and implication, respectively. Furthermore, for any $Q \subseteq \mathcal{P}(C)$, we denote by $\cap Q$ and $\cup Q$ the intersection and union of all members of Q.

M_R induces a *predicate transformer* $M_F \in \mathcal{P}(C) \to \mathcal{P}(C)$ using the relational image operation:

$$M_F(p) = \{c' \in C \mid \exists c \in p : (c, c') \in M_R\}$$

It is intuitively obvious that if M is in one of the configurations in $p \in \mathcal{P}(C)$, then in one time step it must be in one of the configurations in $M_F(p)$. Furthermore, from its definition we see that M_F distributes over arbitrary unions:

$$M_F(\cup Q) = \cup\{M_F(q) \mid q \in Q\}$$

for all $Q \subseteq \mathcal{P}(C)$. Any M_F that satisfies this distributive property also defines a M_R through the equivalence $(c, c') \in M_R \Leftrightarrow c' \in M_F(\{c\})$, that is to say, there is no loss of information going from M_R to M_F or vice versa. We adopt this functional model of M and drop its subscript. It follows its distributivity that M also preserves the empty set of constraints, i.e. $M(\emptyset) = \emptyset$, and that M is monotonic, i.e. $p \subseteq q \Rightarrow M(p) \subseteq M(q)$ for all $p, q \in \mathcal{P}(C)$.

In practice, signals in M are typically divided into "input" signals and "output" or "internal" signals. While an input signal is typically controlled by the external environment, and thus unconstrained by M itself, non-input signals are determined by the circuit topology and functionality. For example, supposed M is the earlier example of a unit-delayed two-input AND gate, we could then define its model $\mathcal{M} \in \mathcal{P}(\mathbb{B}^3) \to \mathcal{P}(\mathbb{B}^3)$ as:

$$\mathcal{M}(p) = \{\langle b_1, b_2, i_1 \wedge i_2 \rangle \in \mathbb{B}^3 \mid \langle i_1, i_2, o \rangle \in p\}$$

Here i_1 and i_2 refer to the two inputs of the AND gate and o the ignored output; b_1 and b_2 are unconstrained inputs in the new configuration.

Ternary lattices Manipulating subsets of $\mathbb{B}^m$ is however impractical for even moderately large m, which leads us to one of the key insights of STE. Namely, instead of manipulating subsets of $\mathbb{B}^m$ directly, one can use sequences of ternary values $\mathbb{T} = \mathbb{B} \cup \{\mathrm{X}\}$ to approximate them, whose sizes are only linear in m. Here the 1 and 0 from $\mathbb{B}$ denotes specific, defined values whereas X denotes an "unknown" value that could be either 1 or 0. This intuition induces a partial order $\sqsubseteq$ on $\mathbb{T}$, where $0 \sqsubseteq \mathrm{X}$ and $1 \sqsubseteq \mathrm{X}$[2]. For any $m \in \mathbb{N}$, this ordering on $\mathbb{T}$ is lifted component-wise to $\mathbb{T}^m$.

[1] Mention how this affects circuits with zero-delays?

[2] We use the reverse ordering of what is originally used in STE.

Note that $\mathbb{T}^m$ does not quite form a complete lattice because it lacks a bottom: both $0 \sqsubseteq \mathrm{X}$ and $1 \sqsubseteq \mathrm{X}$ but 0 and 1 are equally defined. A special bottom element $\bot$ is therefore introduced, such that $\bot \sqsubseteq t$ and $\bot \neq t$ for all $t \in \mathbb{T}^m$. The extended $\mathbb{T}^m_\bot = \mathbb{T}^m \cup \{\bot\}$ then becomes a complete lattice. We denote the top element $\langle \mathrm{X}, \ldots, \mathrm{X} \rangle$ of $\mathbb{T}^m_\bot$ by $\top$.

Generalising from any specific domain, let $(\hat{P}, \sqsubseteq)$ be a finite, complete lattice of *abstract predicates* in which the meet $\sqcap$ and join $\sqcup$ of any subset $Q \subseteq \hat{P}$ exists. Similar to the previous set operations for power sets, $\sqcap$, $\sqcup$ and $\sqsubseteq$ correspond to conjunction, disjunction and implication for abstract predicates, respectively. Furthermore, for any $Q \subseteq \hat{P}$, we denote by $\sqcap Q$ and $\sqcup Q$ the meet and join of all members of Q.

Abstract circuit model Let there be a Galois connection relating "concrete" predicates $\mathcal{P}(C)$ and abstract predicates $\hat{P}$. The usual definition of a Galois connection is in terms of an *abstraction* $\alpha \in \mathcal{P}(C) \to \hat{P}$ and a *concretisation* $\gamma \in \hat{P} \to \mathcal{P}(C)$ function, such that $\alpha(p) \sqsubseteq \hat{p} \Leftrightarrow p \subseteq \gamma(\hat{p})$ for all $p \in \mathcal{P}(C)$ and $\hat{p} \in \hat{P}$. For example, a Galois connection from $\mathcal{P}(\mathbb{B}^m)$ to $\mathbb{T}^m_\bot$ for any $m \in \mathbb{N}$ can be defined in a natural way by its concretisation function $\gamma \in \mathbb{T}^m_\bot \to \mathcal{P}(\mathbb{B}^m)$:

$$\gamma(\langle t_0, \ldots, t_{m-1} \rangle) = \{\langle b_0, \ldots, b_{m-1} \rangle \in \mathbb{B}^m \mid \forall i < m : t_i \neq \mathrm{X} \Rightarrow b_i = t_i\}$$
$$\gamma(\bot) = \emptyset$$

Listing each concrete predicate approximated by a given abstract predicate. The abstraction function $\alpha \in \mathcal{P}(\mathbb{B}^m) \to \mathbb{T}^m_\bot$ instead finds the most precise abstract predicate for a set of concrete predicates:

$$\alpha(p) = \sqcup\{\langle t_0, \ldots, t_{m-1} \rangle \in \mathbb{T}^m_\bot \mid \langle b_0, \ldots, b_{m-1} \rangle \in p, \forall i < m : b_i = t_i\}$$
$$\alpha(\emptyset) = \bot$$

The def. used in [1] is similar but different, as it view a Galois connection from $\mathcal{P}(C)$ to $\hat{P}$ instead as binary relation[3]. Specifically, let $\ll \subseteq \mathcal{P}(C) \times \hat{P}$ be a binary relation, where $p \ll \hat{p}$ reads as "p can be approximated as $\hat{p}$", such that for all $Q \subseteq \mathcal{P}(C)$ and $\hat{Q} \subseteq \hat{P}$:

$$\forall p \in Q : \forall \hat{p} \in \hat{Q} : p \ll \hat{p} \Leftrightarrow \cup Q \ll \sqcap \hat{Q}$$

Intuitively, $\ll$ is an extension of the partial order $\subseteq$ of $\mathcal{P}(C)$ and $\sqsubseteq$ of $\hat{P}$ to an ordering between $\mathcal{P}(C)$ and $\hat{P}$. The original abstraction and concretisation functions can be derived from $\ll$ as: $\alpha(p) = \sqcap\{\hat{p} \in \hat{P} \mid p \ll \hat{p}\}$ and $\gamma(\hat{p}) = \cup\{p \in \mathcal{P}(C) \mid p \ll \hat{p}\}$. Conversely, the relation $\ll$ can be derived from α and γ as: $p \ll \hat{p} \Leftrightarrow \alpha(p) \sqsubseteq \hat{p}$ and $p \ll \hat{p} \Leftrightarrow p \subseteq \gamma(\hat{p})$.

[3] Used mainly to prove thesis in [1], included here since we use one for refinement later on. The relational view, I think, is used so that it can be made into a sim. relation.

An *abstract predicate transformer* $\hat{M} \in \hat{P} \to \hat{P}$ is an *abstract interpretation* [2] of $M \in \mathcal{P}(C) \to \mathcal{P}(C)$ iff: (1) $\hat{M}$ preserves $\bot$, i.e. $\hat{M}(\bot) = \bot$; (2) $\hat{M}$ is monotonic, i.e. $\hat{p} \sqsubseteq \hat{q} \Rightarrow \hat{M}(\hat{p}) \sqsubseteq \hat{M}(\hat{q})$ for all $\hat{p}, \hat{q} \in \hat{P}$; and (3) $\ll$ is a *simulation relation* from $\mathcal{P}(C)$ to $\hat{P}$, i.e. $p \ll \hat{p} \Rightarrow M(p) \ll \hat{M}(\hat{p})$ for all $p \in \mathcal{P}(C)$ and $\hat{p} \in \hat{P}$. That $\ll$ is a simulation relation can also be stated in terms of its abstraction α and concretisation γ functions: $\alpha(M(p)) \sqsubseteq \hat{M}(\alpha(p))$ for all $p \in \mathcal{P}(C)$, and $M(\gamma(\hat{p})) \subseteq \gamma(\hat{M}(\hat{p}))$ for all $\hat{p} \in \hat{P}$.

Note that $\hat{M}$ does not distribute over arbitrary join in general because information is potentially discarded when joining two lattices. As an example, let the following $\hat{M}$ abstract the earlier M for an unit-delayed AND gate:

$$\hat{M}(\langle p_1, p_2, p_3 \rangle) = \begin{cases} \langle \mathrm{X}, \mathrm{X}, 1 \rangle, & \text{if } p_1 = 1 \text{ and } p_2 = 1 \\ \langle \mathrm{X}, \mathrm{X}, 0 \rangle, & \text{if } p_1 = 0 \text{ or } p_2 = 0 \\ \langle \mathrm{X}, \mathrm{X}, \mathrm{X} \rangle, & \text{otherwise} \end{cases}$$
$$\hat{M}(\bot) = \bot$$

If we apply $\hat{M}$ to the join of $\langle 0, 1, \mathrm{X} \rangle$ and $\langle 1, 0, \mathrm{X} \rangle$, or if we apply $\hat{M}$ to them individually and then join the results, we get two different results:

$$\begin{aligned} \hat{M}(\langle 0, 1, \mathrm{X} \rangle \sqcup \langle 1, 0, \mathrm{X} \rangle) &= \hat{M}(\langle \mathrm{X}, \mathrm{X}, \mathrm{X} \rangle) &&= \langle \mathrm{X}, \mathrm{X}, \mathrm{X} \rangle \\ \hat{M}(\langle 0, 1, \mathrm{X} \rangle) \sqcup \hat{M}(\langle 1, 0, \mathrm{X} \rangle) &= \langle \mathrm{X}, \mathrm{X}, 0 \rangle \sqcup \langle \mathrm{X}, \mathrm{X}, 0 \rangle &&= \langle \mathrm{X}, \mathrm{X}, 0 \rangle \end{aligned}$$

The inequality $\sqcup\{\hat{M}(\hat{q}) \mid \hat{q} \in \hat{Q}\} \sqsubseteq \hat{M}(\sqcup \hat{Q})$ for all $\hat{Q} \sqsubseteq \hat{P}$ does however hold, since it is implied by the monotonicity of $\hat{M}$.

Assertions and satisfaction A *trajectory assertion* for $\hat{M}$ is a quintuple $\hat{A} = (S, s_0, R, \pi_a, \pi_c)$, where S is a finite set of *states*, $s_0 \in S$ is an *initial state*, $R \subseteq S \times S$ is a *transition relation*, $\pi_a \in S \to \hat{P}$ and $\pi_c \in S \to \hat{P}$ label each state s with an *antecedent* $\pi_a(s)$ and a *consequent* $\pi_c(s)$. Furthermore, we assume that $(s, s_0) \notin S$ for all $s \in S$ without any loss of generality.

For all $\Phi \in S \to \hat{P}$ and $s \in S$, define $F \in S \to (\hat{P} \to \hat{P})$ and $\mathcal{F} \in (S \to \hat{P}) \to (S \to \hat{P})$ as follows:

$$F(s)(\hat{p}) = \hat{M}(\pi_a(s) \sqcap \hat{p}) \tag{1}$$
$$\mathcal{F}(\Phi)(s) = \textbf{if } (s = s_0) \textbf{ then } \top \textbf{ else } \sqcup \{F(s')(\Phi(s')) \mid (s', s) \in R\} \tag{2}$$

F preserves $\bot$ and both F and $\mathcal{F}$ are monotonic, where two $\Phi, \Phi' \in S \to \hat{P}$ are ordered as $\Phi \sqsubseteq \Phi' \Leftrightarrow \forall s \in S : \Phi(s) \sqsubseteq \Phi'(s)$. Let $\Phi_* \in S \to \hat{P}$ be the least fixpoint of the equation $\Phi = \mathcal{F}(\Phi)$ [3]. Since both S and $\hat{P}$ are finite, Φ_* is given by $\lim \Phi_n(s)$ where Φ_n is defined as follows:

$$\Phi_n = \textbf{if } (n = 0) \textbf{ then } (\lambda s \in S : \bot) \textbf{ else } \mathcal{F}(\Phi_{n-1}) \tag{3}$$

We say that the abstract circuit $\hat{M}$ *satisfies* a lattice-based, abstract trajectory assertion $\hat{A}$, denoted by $\hat{M} \models \hat{A}$, iff:

$$\forall s \in S : \Phi_*(s) \sqcap \pi_\alpha(s) \sqsubseteq \pi_c(s) \tag{4}$$

$\hat{M} \models \hat{A}$ implies that a concretisation of $\hat{A}$ can also be satisfied by the original, set-based model M [1].

2 System refinement

Consider another fixed, but arbitrary, circuit model $N \in \mathcal{P}(C') \to \mathcal{P}(C')$ and let $\hat{N} \in \hat{Q} \to \hat{Q}$ be an abstract interpretation of it; $\hat{Q}$ is an abstract predicate with a Galois connection to $\mathcal{P}(C')$. In the previous sections, exactly what abstract predicates like $\hat{Q}$ are, were not important. In order to reason about refinement, however, we need to make a distinction between their *visible* and internal elements. Specifically, we wish to say that $\hat{M}$ refines $\hat{N}$ if every visible behaviour of $\hat{M}$ is allowed by $\hat{N}$ while assuming nothing about initial configurations.

Let the visible elements of an abstract predicate $\hat{P}$ be those given by two monotone, idempotent mappings, $\mathrm{in} \in \hat{P} \to \hat{P}$ and $\mathrm{out} \in \hat{P} \to \hat{P}$, which identify its "inputs" and "outputs", respectively. That is, an *input* of $\hat{P}$ is a predicate $\hat{\imath} \in \hat{P}$ such that $\hat{\imath} = \mathrm{in}(\hat{\imath})$. Similarly, an *output* is a $\hat{\mathrm{o}} \in \hat{P}$ such that $\hat{\mathrm{o}} = \mathrm{out}(\hat{\mathrm{o}})$. For example:

$$\mathrm{in}(\langle \hat{p}_0, \hat{p}_1, \hat{p}_2 \rangle) = \langle \hat{p}_0, \hat{p}_1, \mathrm{X} \rangle \qquad \mathrm{out}(\langle \hat{p}_0, \hat{p}_1, \hat{p}_2 \rangle) = \langle \mathrm{X}, \mathrm{X}, p_2 \rangle$$

With a slight abuse of notation, we overload both $\mathrm{in}(\cdot)$ and $\mathrm{out}(\cdot)$ to accept predicates from $\hat{Q}$.

Let $\lll \subseteq \hat{P} \times \hat{Q}$ be a Galois connection such that for all $\hat{P}' \subseteq \hat{P}$ and $\hat{Q}' \subseteq \hat{Q}$:

$$\forall \hat{p} \in \hat{P}' : \forall \hat{q} \in \hat{Q}' : \hat{p} \lll \hat{q} \Leftrightarrow \sqcup \hat{P}' \lll \sqcap \hat{Q}'$$

Like the earlier Galois connection, $\lll$ can intuitively be thought of as an extension of the orderings inside $\hat{P}$ and $\hat{Q}$ to an ordering between them. The abstraction and concretisation functions, $\alpha \in \hat{P} \to \hat{Q}$ and $\gamma \in \hat{Q} \to \hat{P}$, can also be derived from $\lll$ as before: $\alpha(p) = \sqcap\{\hat{p} \in \hat{P} \mid \hat{q} \lll \hat{p}\}$ and $\gamma(\hat{p}) = \sqcup\{\hat{q} \in \hat{Q} \mid \hat{q} \lll \hat{p}\}$. We note that γ is monotone, preserves top and distributes over arbitrary meet, i.e. $\gamma(\sqcap \hat{Q}) = \sqcap\{\gamma(\hat{q}) \in \hat{P} \mid \hat{q} \in \hat{Q}\}$. Similarly, α is monotone, preserves bottom and distributes over arbitrary join.

The binary relation $\lll$ is a *visible simulation* between $\hat{P}$ and $\hat{Q}$ if $\hat{p} \lll \hat{q}$ implies (1) $\mathrm{out}(\hat{p}) \sqsubseteq \gamma(\mathrm{out}(\hat{q}))$ and (2) $\hat{M}(\hat{\imath} \sqcap \hat{p}) \lll \hat{N}(\alpha(\hat{\imath}) \sqcap \hat{q})$ for all inputs $\hat{\imath} \in \hat{P}$. The abstract model $\hat{M}$ refines $\hat{N}$, denoted by $\hat{M} \lll \hat{N}$, if the top element of $\hat{P}$ visibly simulates the top element of $\hat{Q}$, i.e. $(\top \in \hat{P}) \lll (\top \in \hat{Q})$.

A trajectory assertion $\hat{A} = (S, s_0, R, \pi_a, \pi_c)$ for $\hat{N}$ where antecedents only mention inputs, $\pi_a \in S \to \mathrm{in}(\hat{P})$, and consequents only mention outputs, $\pi_c \in S \to \mathrm{out}(\hat{P})$, is referred to as a *visible trajectory assertion*.

We state our first result:

Theorem 1. *If $\hat{M}$ and $\hat{N}$ are abstract interpretations such that $\hat{M} \lll \hat{N}$, and $\hat{A}_{vis}$ is a trajectory assertion restricted to visible elements of $\hat{N}$, then:*

$$\hat{N} \models \hat{A}_{vis} \Rightarrow \hat{M} \models \hat{A}_{vis}$$

Because $\top$ represents every possible state in its predicate type, that $\hat{M} \lll \hat{N}$ thus implies that every state in $\hat{M}$ is simulated by every state in $\hat{N}$. However, using $\top$ also means we simply demand that their outputs are ordered, since both will output X until the inputs have flushed trough them.

A Appendices

References

1. Chou, C.T.: The mathematical foundation of symbolic trajectory evaluation. In: International Conference on Computer Aided Verification. pp. 196–207. Springer (1999)
2. Cousot, P.: Abstract interpretation. ACM Computing Surveys (CSUR) **28**(2), 324–328 (1996)
3. Davey, B.A., Priestley, H.A.: Introduction to lattices and order. Cambridge university press (2002)
4. Muchnick, S., et al.: Advanced compiler design implementation. Morgan kaufmann (1997)
5. Seger, C.J.H., Bryant, R.E.: Formal verification by symbolic evaluation of partially-ordered trajectories. Formal Methods in System Design **6**(2), 147–189 (1995)