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 $M \in \mathcal{P}(\mathbb{B}^3) \to \mathcal{P}(\mathbb{B}^3)$ as:

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

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 unit-delayed AND gate:

$$\begin{array}{ll} \hat{M}(\langle 1,1,p_2\rangle) = \langle \mathrm{X},\mathrm{X},1\rangle & \hat{M}(\langle 0,0,p_2\rangle) = \langle \mathrm{X},\mathrm{X},0\rangle \\ \hat{M}(\langle 0,\mathrm{X},p_2\rangle) = \langle \mathrm{X},\mathrm{X},0\rangle & \hat{M}(\langle \mathrm{X},0,p_2\rangle) = \langle \mathrm{X},\mathrm{X},\mathrm{X}\rangle \\ \hat{M}(\langle p_0,p_1,p_2\rangle) = \langle \mathrm{X},\mathrm{X},\mathrm{X}\rangle & \end{array}$$

Where the last, most general matching is overlapped by the more concrete ones. 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:

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

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 *external* and *internal* elements of $\hat{P}$ and $\hat{Q}$. Specifically, we will argue that $\hat{M}$ *refines* $\hat{N}$ if every visible behaviour of $\hat{M}$ can be matched by $\hat{N}$ while assuming nothing about initial configurations.

2.1 Visible simulation as refinement

Let the visible output of an abstract predicate $\hat{P}$ be identified by the mapping $\text{out} \in \hat{P} \to \hat{P}$, such that $\text{out}(\cdot)$ is monotonic, i.e. $\hat{p} \sqsubseteq \hat{q} \Rightarrow \text{out}(\hat{p}) \sqsubseteq \text{out}(\hat{q})$; and produces the greatest bound for an output, i.e. $\text{out}(\hat{p}) \sqsubseteq \text{out}(\hat{q}) \Leftrightarrow \hat{p} \sqsubseteq \text{out}(\hat{q})$. In the case of $\mathbb{T}^3_\perp$, for example, $\text{out}(\cdot)$ can be defined naturally as a projection which maps each non-output element to X:

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

It is easy to see that this definition is idempotent and monotonic, and that it produces the greatest bound for any $\mathbb{T}^3$ with an output $\sqsubseteq \hat{p}_2$. With a slight abuse of notation, we overload $\text{out}(\cdot)$ to accept predicates from $\hat{Q}$.

Let $\lll \subseteq \hat{P} \times \hat{Q}$, where $\hat{p} \lll \hat{q}$ reads as "$\hat{p}$ refines $\hat{q}$", 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.

Let the binary relation $\lll$ be a simulation between the visible parts of $\hat{P}$ and $\hat{Q}$, such that $\hat{p} \lll \hat{q}$ implies (1) $\text{out}(\hat{p}) \sqsubseteq \gamma(\text{out}(\hat{q}))$ and (2) $\hat{M}(\hat{\imath} \sqcap \hat{p}) \lll \hat{N}(\alpha(\hat{\imath}) \sqcap \hat{q})$ for all inputs $\hat{\imath} \in \text{in}(\hat{P})$. We extend this simulation relation to abstract model

transformers, and say that $\hat{M}$ refines $\hat{N}$, denoted $\hat{M} \lll \hat{N}$, if the top element of $\hat{P}$ refines the top element of $\hat{Q}$, i.e. $(\top \in \hat{P}) \lll (\top \in \hat{Q})$. Intuitively, if $\hat{M}$ refines $\hat{N}$, then every visible behaviour of $\hat{M}$ has a corresponding behaviour in $\hat{N}$ that ...

A trajectory assertion $\hat{A} = (S, s_0, R, \pi_a, \pi_c)$ for $\hat{N}$ where antecedents only mention inputs, $\pi_a \in S \to \text{in}(\hat{P})$, and consequents only mention outputs, $\pi_c \in S \to \text{out}(\hat{P})$, is referred to as a *visible trajectory assertion* and denoted by $\hat{A}_{vis}$. Define $\alpha(\hat{A}) = (S, s_0, R, \alpha(\pi_a), \alpha(\pi_c))$, where $\alpha(\pi_a) = \lambda s \in S : \alpha(\pi_a(s))$ and $\alpha(\pi_a) = \lambda s \in S : \alpha(\pi_a(s))$.

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

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

Corollary 1. *If $\hat{N} = \hat{N}_1 \times \hat{N}_2$ is an abstract predicate transformer defined as the product of two transformers $\hat{N}_1$ and $\hat{N}_2$, and $\hat{M}$ is another transformer such that $\hat{M} \lll \hat{N}_0$, then for any trajectory assertion $\hat{A}_{vis}$ restricted to visible elements of $\hat{N}$ we must have that:*

$$\hat{N}_1 \times \hat{N}_2 \models \hat{A}_{vis} \Rightarrow \hat{M} \times \hat{N}_2 \models \hat{A}_{vis}$$

A Appendices

A.1 Theorem 1

We first prove a few intermediate lemmas.

Lemma 1. $(\bot \in \hat{P}) \lll (\bot \in \hat{Q})$.

A Galois connection always relates the two bottoms. Because $\hat{M}$ and $\hat{N}$ preserves bottom, any path that touches bottom must also stay in bottom, regardless of inputs. Thus, $\bot \in \hat{P}$ refines $\bot \in \hat{Q}$. ■

Lemma 2. *If $\hat{p}_1 \lll \hat{q}_1$ and $\hat{p}_2 \lll \hat{q}_2$, then $(\hat{p}_1 \sqcup \hat{p}_2) \lll (\hat{q}_1 \sqcup \hat{q}_2)$.*

To prove our lemma, we must show that three properties are fulfilled: $\hat{p}_1 \sqcup \hat{p}_2$ must be a valid refinment for $\hat{q}_1 \sqcup \hat{q}_2$, the output of $\hat{p}_1 \sqcup \hat{p}_2$ must be more specific than that of $\hat{q}_1 \sqcup \hat{q}_2$, and that each next step for $\hat{p}_1 \sqcup \hat{p}_2$ is refined by the corresponding step in $\hat{q}_1 \sqcup \hat{q}_2$. We prove these properties in order.

Using the definition of $\lll$, we see that the assumption $\hat{p}_1 \lll \hat{q}_1$ implies that $\sqcup\{\hat{p}_1\} \lll \sqcap\{\hat{q}_1\}$. As $\hat{q}_1 \sqcap (\hat{q}_1 \sqcup \hat{q}_2) = \hat{q}_1$, we also have $\sqcup\{\hat{p}_1\} \lll \sqcap\{\hat{q}_1, \hat{q}_1 \sqcup \hat{q}_2\}$ and thus $\hat{p}_1 \lll \hat{q}_1 \sqcup \hat{q}_2$. A similar argument can be used to show $\hat{p}_2 \lll \hat{q}_1 \sqcup \hat{q}_2$. Since both $\hat{p}_1$ and $\hat{p}_2$ are refined by $\hat{q}_1 \sqcup \hat{q}_2$, we must have that $\sqcup\{\hat{p}_1, \hat{p}_2\} \lll \sqcap\{\hat{q}_1 \sqcup \hat{q}_2\}$ and thus $(\hat{p}_1 \sqcup \hat{p}_2) \lll (\hat{q}_1 \sqcup \hat{q}_2)$.

We show that the output of $\hat{p}_1 \sqcup \hat{p}_2$ is more specific than that of $\hat{q}_1 \sqcup \hat{q}_2$ in two steps: First, we show that $\gamma(\text{out}(\hat{q}_2)) \sqsubseteq \gamma(\text{out}(\hat{q}_1 \sqcup \hat{q}_2))$ is an upper bound for $\gamma(\text{out}(\hat{q}_1))$ and $\gamma(\text{out}(\hat{q}_2))$. Secondly, that $\text{out}(\hat{p}_1 \sqcup \hat{p}_2)$ is less than or equal to any other upper bound on the output of $\hat{p}_1$ and $\hat{p}_2$.

1. By definition of $\sqcup$, we know that $\hat{q}_1 \sqsubseteq \hat{q}_1 \sqcup \hat{q}_2$ and $\hat{q}_1 \sqsubseteq \hat{q}_1 \sqcup \hat{q}_2$. It then follows from the monotonicity of $\text{out}(\cdot)$ and γ that $\gamma(\text{out}(\hat{q}_1)) \sqsubseteq \gamma(\text{out}(\hat{q}_1 \sqcup \hat{q}_2))$ and $\gamma(\text{out}(\hat{q}_2)) \sqsubseteq \gamma(\text{out}(\hat{q}_1 \sqcup \hat{q}_2))$.
2. For any $\hat{x}$, if $\text{out}(\hat{p}_1) \sqsubseteq \text{out}(\hat{x})$ and $\text{out}(\hat{p}_2) \sqsubseteq \text{out}(\hat{x})$, then by property 3 of $\text{out}(\cdot)$ we have that $\hat{p}_1 \sqsubseteq \text{out}(\hat{x})$ and $\hat{p}_2 \sqsubseteq \text{out}(\hat{x})$. It then follows by definition of $\sqcup$ that $\hat{p}_1 \sqcup \hat{p}_2 \sqsubseteq \text{out}(\hat{x})$. Using the other direction of property 3 for $\text{out}(\cdot)$, we have that $\text{out}(\hat{p}_1 \sqcup \hat{p}_2) \sqsubseteq \text{out}(\hat{x})$.

Combining these two steps with the assumptions that $\text{out}(\hat{p}_1) \sqsubseteq \gamma(\text{out}(\hat{q}_1))$ and $\text{out}(\hat{p}_2) \sqsubseteq \gamma(\text{out}(\hat{q}_2))$, it follows immediately that $\text{out}(\hat{p}_1 \sqcup \hat{p}_2) \sqsubseteq \gamma(\text{out}(\hat{q}_1 \sqcup \hat{q}_2))$.

Lastly, we show that any step taken from $\hat{p}_1 \sqcup \hat{p}_2$ in response to some input $\hat{\imath} \in \text{in}(\hat{P})$ is refined by the corresponding step taken from $\hat{q}_1 \sqcup \hat{q}_2$. More specifically, we show that $\hat{M}(\hat{\imath} \sqcap (\hat{p}_1 \sqcup \hat{p}_2)) \lll \hat{N}(\alpha(\hat{\imath}) \sqcap (\hat{q}_1 \sqcup \hat{q}_2))$ for all inputs $\hat{\imath} \in \text{in}(\hat{P})$. However, the reasoning we use to prove this last property is quite similar to the first one. Consider an arbitrary $\hat{\imath} \in \text{in}(\hat{P})$ and note that $\hat{N}(\alpha(\hat{\imath}) \sqcap \hat{q})$ is a monotone transformation for any $\hat{q} \in \hat{Q}$. $\hat{N}(\alpha(\hat{\imath}) \sqcap \hat{q}_1)$ must therefore be the meet of $\hat{N}(\alpha(\hat{\imath}) \sqcap \hat{q}_1)$ and $\hat{N}(\alpha(\hat{\imath}) \sqcap (\hat{q}_1 \sqcup \hat{q}_2))$, thus $\hat{M}(\hat{\imath} \sqcap \hat{p}_1) \lll \hat{N}(\alpha(\hat{\imath}) \sqcap (\hat{q}_1 \sqcup \hat{q}_2))$. A similar argument can be used to show that $\hat{M}(\hat{\imath} \sqcap \hat{p}_2) \lll \hat{N}(\alpha(\hat{\imath}) \sqcap (\hat{q}_1 \sqcup \hat{q}_2))$. As $\hat{M}(\hat{\imath} \sqcap \hat{p})$ is also a monotone transformation for any $\hat{p} \in \hat{P}$, we must have that $\hat{M}(\hat{\imath} \sqcap (\hat{p}_1 \sqcup \hat{p}_2)) \lll \hat{N}(\alpha(\hat{\imath}) \sqcap (\hat{q}_1 \sqcup \hat{q}_2))$ for any $\hat{\imath} \in \text{in}(\hat{P})$. ■

The following lemmas regard the fix-points and functions used to determine satisfaction of a trajectory assertion in $\hat{M}$ and $\hat{N}$. To clarify which definitions are used with $\hat{N}$ and which are used with $\hat{N}$, we first duplicate some of these definitions. Specifically, let G, $\mathcal{G}$ and Ψ be equivalent to the earlier operations F, $\mathcal{F}$ and Φ for $\hat{M}$, respectively, but defined in terms of $\hat{N}$:

$$
\begin{aligned}
G(s)(\hat{q}) &= \hat{N}(\alpha(\pi_a(s)) \sqcap \hat{q}) \\
\mathcal{G}(\Psi)(s) &= \textbf{if } (s = s_0) \textbf{ then } \top \textbf{ else } \sqcup \{G(s')(\Psi(s')) \mid (s', s) \in R\} \\
\Psi_n &= \textbf{if } (n = 0) \textbf{ then } (\lambda s \in S : \bot) \textbf{ else } \mathcal{G}(\Psi_{n-1})
\end{aligned}
$$

Lemma 3. *If $\hat{p} \lll \hat{q}$, then $F(s)(\hat{p}) \lll G(s)(\hat{q})$ for all $s \in S$*

Expanding definitions, we see that $F(s)(\hat{p}) = \hat{M}(\pi_a(s) \sqcap \hat{p})$ and $G(s)(\hat{q}) = \hat{N}(\alpha(\pi_a(s)) \sqcap \hat{q})$ for some $\pi_a(s) \in \text{in}(\hat{P})$. That $F(s)(\hat{p}) \lll G(s)(\hat{q})$ then follows from the second property of $\hat{p} \lll \hat{q}$.

Lemma 4. *If $\hat{M} \lll \hat{N}$ and $\hat{p}(s) \lll \hat{q}(s)$ for all $s \in S' \subseteq S$, then it must be that $\sqcup\{F(s)(\hat{p}(s)) \mid s \in S'\} \lll \sqcup\{G(s)(\hat{q}(s)) \mid s \in S'\}$.*

Using induction on the size of S', we show that the join of sets with such pairwise refined predicates also produces two predicates that refine each other. In the base case, $|S'| = 0$, that $(\top \in \hat{P}) \lll (\top \in \hat{Q})$ (union of $\emptyset$ is $\top$) follows from the assumption that $\hat{M} \lll \hat{N}$. For the inductive step, $|S'| = n$, let $\hat{f} = \sqcup\{F(s)(\hat{p}(s)) \mid s \in S'\}$, let $\hat{g} = \sqcup\{G(s)(\hat{q}(s)) \mid s \in S'\}$, and assume that $\hat{f} \lll \hat{g}$.

We extend S' with s, and show that $F(s)(\hat{p}(s)) \sqcup \hat{f} \lll G(s)(\hat{q}(s)) \sqcup \hat{g}$. From the assumption, and lemma 3, it follows that $F(s)(\hat{p}(s) \lll G(s)(\hat{q}(s))$. Lemma 2 then states that the union of two such refining pairs also refine each other. ■

Lemma 5. *If $\hat{M} \lll \hat{N}$, then $\Phi_*(s) \lll \Psi_*(s)$ for all $s \in S$*

Since $\Phi_*(s) = \lim \Phi_n(s)$ and $\Psi_*(s) = \lim \Psi_n(s)$, it suffices to prove that $\Phi_n(s) \lll \Psi_n(s)$ for all $s \in S$ and $n \in \mathbb{N}$. We do so by induction on n. The base case, where $\Phi_0(s) = \bot \in \hat{P}$ and $\Psi_0(s) = \bot \in \hat{Q}$, follows from lemma 1. For the inductive step, assume that $\Phi_n(s) \lll \Psi_n(s)$ for all $s \in S$. For $s = s_0$, we have that $\Phi_{n+1}(s_0) = \top \in \hat{P}$ and $\Psi_{n+1}(s_0) = \top \in \hat{Q}$, which follows from the assumption $\hat{M} \lll \hat{N}$. For any $s \neq s_0$, we have that $\Phi_{n+1}(s) = \mathcal{F}(\Phi_n)(s) = \sqcup\{F(s')(\Phi_n(s')) \mid (s', s) \in R\}$ and $\Psi_{n+1}(s) = \mathcal{G}(\Psi_n)(s) = \sqcup\{G(s')(\Psi_n(s')) \mid (s', s) \in R\}$. By the induction hypothesis and lemma 4: $\Phi_{n+1}(s) \lll \Psi_{n+1}(s)$ for all $s \in S$. ■

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)