# 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 $(\mathit{in_1}\,\mathbf{is}\,1 \wedge \mathit{in_2}\,\mathbf{is}\,1) \Rightarrow \mathbf{N}(\mathit{out}\,\mathbf{is}\,1)$ 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. We denote by $\cap Q$ and $\cup Q$ the intersection and union of all members of any $Q \subseteq \mathcal{P}(C)$.

$M_R$ 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 $M_F(p)$. We also 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. It follows that $M_F$ preserves the empty set of constraints, i.e. $M_F(\emptyset) = \emptyset$, and is monotonic, i.e. $p \subseteq q \Rightarrow M_F(p) \subseteq M_F(q)$ for all $p, q \in \mathcal{P}(C)$. We adopt this functional model of $M$ and drop its subscript.

In practice, the predicates of $C$ correspond to signals in $M$ and divided into external, or "input" and "output", and 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 $\perp$ is therefore introduced, such that $\perp \sqsubseteq t$ and $\perp \neq t$ for all $t \in \mathbb{T}^m$. The extended $\mathbb{T}^m_\perp = \mathbb{T}^m \cup \{\perp\}$ then becomes a complete lattice. We denote the top element $\langle \mathrm{X}, \ldots, \mathrm{X} \rangle$ of $\mathbb{T}^m_\perp$ 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 $\mathbb{T}^m_\perp$ for any $m \in \mathbb{N}$ can be defined in a natural way by its concretisation function $\gamma \in \mathbb{T}^m_\perp \to \mathcal{P}(\mathbb{B}^m)$:

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

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

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

We adopt a different interpretation of Galois connections that is given by means of a binary relation between $\mathcal{P}(C)$ and $\hat{P}$. 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, this relation 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 the relation by: $\alpha(p) = \sqcap\{\hat{p} \in \hat{P} \mid p \ll \hat{p}\}$ and $\gamma(\hat{p}) = \cup\{p \in \mathcal{P}(C) \mid p \ll \hat{p}\}$. Conversely, the relation can be derived from $\alpha$ and $\gamma$ as: $p \ll \hat{p} \Leftrightarrow \alpha(p) \sqsubseteq \hat{p}$ and $p \ll \hat{p} \Leftrightarrow p \subseteq \gamma(\hat{p})$.

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 $\perp$, i.e. $\hat{M}(\perp) = \perp$; (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 $\alpha$ and concretisation $\gamma$ 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:

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

Where the last, most general matching is overlapped by the 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, we get two different results:

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

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 an abstact model $\hat{M}$ 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 functions $\Phi \in S \to \hat{P}$ and states $s \in S$, 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 $\bot$, both $F$ and $\mathcal{F}$ are monotonic, and 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, $\Phi_*$ is given by $\lim \Phi_n(s)$ where $\Phi_n$ is defined as follows:

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

$\hat{M}$ *satisfies* a 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 Refinement

### 2.1 Set-Theoretic refinement

Consider another fixed, but arbitrary, circuit model $N \in \mathcal{P}(D) \rightarrow \mathcal{P}(D)$, where $D$ is a non-empty and finite set of configurations which intersects the earlier set $C$. Exactly what configurations such as $C$ and $D$ are, were not important previously. To reason about refinement, however, we need to make a distinction between their external and internal elements. The rationale is that refinement relates the visible behaviour of circuits. Unnecessary to assume that internal states can be aligned.

Let the visible components visible elements of configurations in $D$ be identified by two projection mappings, $\mathrm{o} \in \mathcal{P}(D) \rightarrow \mathcal{P}(D)$ and $i \in \mathcal{P}(D) \rightarrow \mathcal{P}(D)$, identifying its "inputs" and "outputs", respectively. The set of all possible inputs in $N$ is given by the image $i[\mathcal{P}(D)] = \{i(p) \mid p \in \mathcal{P}(D)\}$ and denoted by $I$; outputs in $N$ are similarily given by $\mathrm{o}[\mathcal{P}(D)] = \{\mathrm{o}(p) \mid p \in \mathcal{P}(D)\}$. Note that, since models generally cannot control their input signals, transitions in $N$ from a state $d \in \mathcal{P}(D)$ are driven by its intersection with a chosen input $i \in I$, that is, $N(d \cap i)$ yeilds the next state.

A *trajectory* of $N$ is a non-empty sequence of configurations, $\tau \in D^+$, derived from the states generated by $N$ in response to a *driver* $\delta \in I^+$ when started in its most general state. Specifically, the trajectory $\tau$ induced by a $\delta$ is defined such that $\tau_0 = D$ and $\tau_{n+1} = N(\tau_n \cap \delta_n)$ for all $n \in \mathbb{N} : n < |\delta|$. We denote the trajectory of $N$ induced by a $\delta$ by $N_\delta$ and note that it is prefix-closed. The set of all possible drivers, and the trajectories they induce, describes every behaviour of $N$ for the arbitrary initial state.

With a slight abuse of notation, we extend $\mathrm{o}(\cdot)$ to sequences componentwise and overload it to accept configurations $D$. Furthermore, we assume that both inputs and outputs of $M$ are contained by $N$, i.e. $i[\mathcal{P}(C)] \subseteq i[\mathcal{P}(D)]$ and $\mathrm{o}[\mathcal{P}(C)] \subseteq \mathrm{o}[\mathcal{P}(D)]$. Finally, we say the circuit $M$ refines the circuit $N$, denoted by $M \leq_{\text{set}} N$, iff:

$$\forall \delta \in I^+ : \mathrm{o}(M_\delta) \subseteq \mathrm{o}(N_\delta)$$

Text.

**Theorem 1.** *If $M$ and $N$ are ciruit models such that $M \leq_{set} N$, and $A_{vis}$ is a visible trajectory assertion for $N$, then:*

$$N \models_{set} A_{vis} \Rightarrow M \models_{set} \alpha(A_{vis})$$

Text.

### 2.2 Lattice-Theoretic refinement

Let $\hat{N} \in \hat{Q} \rightarrow \hat{Q}$ be an abstract interpretation of $N$, where $\hat{Q}$ is an abstract predicate for which there is a Galois connection to $\mathcal{P}(D)$.

Let the output of an abstract predicate $\hat{Q}$ be identified by the idempotent mapping $\hat{o} \in \hat{Q} \to \hat{Q}$, such that $\hat{o}(\cdot)$ is monotonic, i.e. $\hat{p} \sqsubseteq \hat{q} \Rightarrow \hat{o}(\hat{p}) \sqsubseteq \hat{o}(\hat{q})$; and the greatest bound for an output, i.e. $\hat{o}(\hat{p}) \sqsubseteq \hat{o}(\hat{q}) \Leftrightarrow \hat{p} \sqsubseteq \hat{o}(\hat{q})$. In a similar vein, let the inputs of an abstract predicate $\hat{Q}$ be identified by the mapping $\hat{\imath} \in \hat{Q} \to \hat{Q}$.

In the case of an unit-delayed AND gate and its predicates $\langle p_0, p_1, p_2 \rangle \in \mathbb{T}^3_\perp$, for example, $\hat{\imath}(\cdot)$ and $\hat{o}(\cdot)$ can be defined naturally as projections which maps each non-input and non-output element to X, respectively:

$$\hat{\imath}(\langle \hat{p}_0, \hat{p}_1, \hat{p}_2 \rangle) = \langle \hat{p}_0, \hat{p}_1, \mathrm{X} \rangle \qquad \hat{o}(\langle \hat{p}_0, \hat{p}_1, \hat{p}_2 \rangle) = \langle \mathrm{X}, \mathrm{X}, p_2 \rangle$$
$$\hat{\imath}(\perp) = \perp \qquad \hat{o}(\perp) = \perp$$

It is easy to see that $\hat{o}(\cdot)$ is monotonic, and that it produces the greatest bound for any $\mathbb{T}^3$ with an output $\sqsubseteq \hat{p}_2$.

Let $\lll\ \subseteq \hat{P} \times \hat{Q}$ be a Galois connection, such that elements of subsets $\hat{P}^* \subseteq \hat{P}$ and $\hat{Q}^* \subseteq \hat{Q}$ are related iff their join and meet are related:

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

where $\hat{p} \lll \hat{q}$ reads as "$\hat{p}$ refines $\hat{q}$". Like the earlier Galois connection, this relation can also be thought of as an extension of the orderings inside $\hat{P}$ and $\hat{Q}$ to an ordering between them. The usual abstraction $\alpha \in \hat{P} \to \hat{Q}$ and concretisation $\gamma \in \hat{Q} \to \hat{P}$ functions can be derived from $\lll$ as follows:

$$\alpha(p) = \sqcap \{\hat{p} \in \hat{P} \mid \hat{q} \lll \hat{p}\} \qquad \gamma(\hat{p}) = \sqcup \{\hat{q} \in \hat{Q} \mid \hat{q} \lll \hat{p}\}$$

We note that $\gamma$ 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, $\alpha$ is monotone, preserves bottom and distributes over arbitrary join.

We say that a relation $\lll\ \subseteq \hat{P} \times \hat{Q}$ is a *visible simulation relation* between $\hat{M}$ and $\hat{N}$ iff the top of $\hat{P}$ refines the top of $\hat{Q}$, i.e. $(\top \in \hat{P}) \lll (\top \in \hat{Q})$, and $\hat{p} \lll \hat{q}$ implies (1) $\hat{o}(\hat{p}) \sqsubseteq \gamma(\hat{o}(\hat{q}))$ and (2) $\hat{M}(\hat{\imath} \sqcap \hat{p}) \lll \hat{N}(\alpha(\hat{\imath}) \sqcap \hat{q})$ for all inputs $\hat{\imath} \in \hat{\imath}(\hat{P})$.

Recall that a trajectory assertion for $\hat{N}$ is a quintuple $\hat{A} = (S, s_0, R, \pi_a, \pi_c)$, where $\pi_a \in S \to \hat{Q}$ and $\pi_c \in S \to \hat{Q}$ label each state $s \in S$ with an antecedent $\pi_a(s)$ and a consequent $\pi_c(s)$. If these antecedents only mention inputs, i.e. $\pi_a \in S \to \hat{\imath}(\hat{P})$, and consequents only mention outputs, i.e. $\pi_c \in S \to \hat{o}(\hat{P})$, then $\hat{A}$ is referred to as a *visible trajectory assertion* for $\hat{N}$, which we denote 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 $\alpha(\pi_a) = \lambda s \in S : \alpha(\pi_a(s))$. Note that, if $\hat{A}_{vis}$ is an assertion for $\hat{N}$, then $\alpha(\hat{A}_{vis})$ is an assertion for $\hat{M}$.

**Theorem 2.** *If $\hat{M}$ and $\hat{N}$ are abstract predicate transformers such that $\hat{M} \lll \hat{N}$, and $\hat{A}_{vis}$ is a visible trajectory assertion for $\hat{N}$, then:*

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

## A Appendices

Text.

## 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)