The Completeness of a Hardware Inference System*

Zheng Zhu, C-J Seger

The Integrated System Design Laboratory
Department of Computer Science
The University of British Columbia
Vancouver, B.C. Canada V6T 1Z4

Abstract. Symbolic trajectory evaluation (STE) is a method for efficient circuit verification [1]. In [2] a set of inference rules was introduced for combining STE results. These inference rules were also proven sound. In this paper we show that, with one additional inference rule, the inference system is complete. Here, complete means that any formula $A \Rightarrow C$, that is valid in every model satisfying some collection Φ of STE assertions, can be derived from Φ by a finite applications of the inference rules. The completeness proof is based on the method of model construction—given Φ, a most general circuit model (in which every assertion in Φ holds) can be generated.

1 Introduction

In [1], Seger and Bryant introduced the underlying theory for symbolic trajectory evaluation. In general, symbolic trajectory evaluation—as implemented in the Voss system [3] for example—is an efficient and highly automated circuit verification method that has been applied to the verification of quite complex and large VLSI circuits. However, in order to successfully verify modern large and complex circuits, a method of breaking down the verification task into smaller, more manageable pieces, is needed. One step towards this goal is developing inference rules that allow the user to combine "smaller" verification results in a safe and sound manner. In [2], such a set of inference rules was introduced and a special purpose theorem prover aimed specifically at manipulating STE results was discussed. Although the inference rules were shown to be sound, they were not shown to be complete. In this paper we remedy this by first adding one more inference rule and then prove that the obtained inference system is complete. Intuitively, complete in this context means that the set of rules is powerful enough to derive all the logical consequences of a given set of trajectory assertions.

* This research was supported, in part, by operating grants OGPO 109688 and OGPO O46196 from the Natural Sciences and Engineering Research Council of Canada, fellowships from the Province of British Columbia Advanced Systems Institute, and by research contract 92-DJ-295 from the Semiconductor Research Corporation.

1.1 Trajectory Evaluation

If the state space of a system (for example a circuit) can be embedded in a complete lattice, the behavior of the system can be expressed as a *trajectory* $\sigma = \sigma^0 \cdots \sigma^n \cdots$, a sequence of values in the lattice determined by the initial state and the system functionality. We define a partial order between sequences of states by extending the partial order on the state space in a natural way.

The model we use of a system is simple and general. A *model structure* is a tuple $\mathcal{M} = [\langle \mathcal{S}, \sqsubseteq \rangle, Y]$, where $\langle \mathcal{S}, \sqsubseteq \rangle$ is a complete lattice ($\mathcal{S}$ being the state space and $\sqsubseteq$ a partial order on $\mathcal{S}$) and Y is a monotone successor function $Y{:}\mathcal{S} \rightarrow \mathcal{S}$. Intuitively, the ordering relation orders the state space according to "information content", and the monotonicity requirement guarantees that we cannot loose information about the future by adding information about the present. A sequence is a *trajectory* if and only if $Y(\sigma^i) \sqsubseteq \sigma^{i+1}$ for $i \geq 0$.

The key to the efficiency of trajectory evaluation is the restricted language that can be used to phrase questions about the model structure. The basic specification language we use is very simple, but expressive enough to capture many of the properties we would like to check.

A *predicate* over $\mathcal{S}$ is a mapping from $\mathcal{S}$ to the lattice $\{0, 1\}$. Informally, a predicate describes a potential state of the system: e.g., a predicate might be (A is x) which says that node A has the value x. A predicate is *simple* if it is monotonic and there is a unique weakest $s \in \mathcal{S}$ (measured by $\sqsubseteq$), called the *defining value*, for which $p(s) = 1$. A special simple predicate is the unc, or "unconstrained", that holds for every state in $\mathcal{S}$ and thus has $\perp$ as defining value. A *trajectory formula* is defined recursively as:

1. **Simple predicates**: Every simple predicate over $\mathcal{S}$ is a trajectory formula.
2. **Conjunction**: $(F_1 \wedge F_2)$ is a trajectory formula if F_1 and F_2 are trajectory formulas.
3. **Domain restriction**: $(e \rightarrow F)$ is a trajectory formula if F is a trajectory formula and e is either 1 or 0.
4. **Next time**: $(\mathbf{N}F)$ is a trajectory formula if F is a trajectory formula.

Note that, in general, symbolic trajectory evaluation, as described in [1], gains its power from extending trajectory formulas to a symbolic domain—and thus concisely encode a very large collection of formulas. However, it should be emphasized that the extension to a symbolic domain only increases the computational efficiency and not the expressiveness of the logic. Consequently, we will only deal with non-symbolic assertions in this paper.

The *depth* of a formula F, written $d(F)$, is defined recursively as:

1. $d(p) = 1$ if p is a simple predicate.
2. $d(F_1 \wedge F_2) = \max(d(F_1), d(F_2))$.
3. $d(e \rightarrow F) = d(F)$.

4. $d(\mathbf{N}F) = 1 + d(F)$.

The depth of a formula is simply the maximum number of nested next time operators plus one. A trajectory formula is said to be *instantaneous* if it does not contain any N operators, i.e., if the depth of the formula is 1. It is straightforward to show that any trajectory formula, A, can be written in the form:

$$A^0 \wedge \mathbf{N}A^1 \wedge \mathbf{N}^2A^2 \wedge \cdots \mathbf{N}^mA^m$$

where N^2A^2 is a shorthand for $N(NA^2)$), etc., all A^i's are instantaneous formulas, and m is a natural number.

The truth semantics of a trajectory formula is defined relative to a model structure and a trajectory. Whether a trajectory $\sigma = \sigma^0\tilde{\sigma}$ satisfies a formula F, written $\sigma \models_{\mathcal{M}} F$, is defined recursively as:

1. $\sigma^0\tilde{\sigma} \models_{\mathcal{M}} p$ iff $p(\sigma^0) = 1$.
2. $\sigma \models_{\mathcal{M}} (F_1 \wedge F_2)$ iff $\sigma \models_{\mathcal{M}} F_1$ and $\sigma \models_{\mathcal{M}} F_2$
3. (a) $\sigma \models_{\mathcal{M}} (0 \rightarrow F)$ always holds
   (b) $\sigma \models_{\mathcal{M}} (1 \rightarrow F)$ iff $\sigma \models_{\mathcal{M}} F$
4. $\sigma^0\tilde{\sigma} \models_{\mathcal{M}} \mathbf{N}F$ iff $\tilde{\sigma} \models_{\mathcal{M}} F$.

Given a model structure $\mathcal{M} = [\langle \mathcal{S}, \sqsubseteq \rangle, Y]$, let $\mathcal{S}^\omega$ denote the set of all infinite sequences of elements from $\mathcal{S}$. Before introducing the concept of defining sequences, it is convenient to introduce an infix "choice" function mapping $\{0,1\} \times \mathcal{S}^\omega$ to $\mathcal{S}^\omega$ and which is defined as:

$$e?\sigma = \begin{cases} \sigma & \text{if } e = 1 \\ \perp\perp \ldots & \text{otherwise} \end{cases}$$

Given a formula F, we can define the *defining sequence* of F, denoted by $\delta(F)$ as follows:

1. $\delta(p) = \overline{p}\ \perp\perp \ldots$ if p is a simple predicate with defining value $\overline{p}$.
2. $\delta(F_1 \wedge F_2) = \delta(F_1) \sqcup \delta(F_2)$.
3. $\delta(e \rightarrow F) = e?\delta(F)$.
4. $\delta(\mathbf{N}F) = \perp\delta(F)$.

The *defining trajectory* of F in the model structure $\mathcal{M} = [\langle \mathcal{S}, \sqsubseteq \rangle, Y]$, denoted by $\tau(F)$, is a sequence defined as [1]:

1. $\tau^0(F) = \delta^0(F)$
2. $\tau^{i+1}(F) = Y(\tau^i(F)) \sqcup \delta^{i+1}(F)$ for $i \geq 0$.

When it becomes necessary to indicate explicitly that it is the defining trajectory of the model structure $\mathcal{M} = [\langle \mathcal{S}, \sqsubseteq \rangle, Y]$, we use $\tau_{\mathcal{M}}(F)$ to denote the sequence.

The fundamental result of STE is the following theorem [1]:

Theorem 1. *Assume A and C are two trajectory formulas. Let $\tau(A)$ be the defining trajectory for formula A and $\delta(C)$ be the defining sequence for formula C. Then $\models A \Rightarrow C$ iff $\delta(C) \sqsubseteq \tau(A)$.*

Finally, for somewhat technical reasons, define an assertion $A \Rightarrow C$ to be *prime* if for every $i \geq 0$, $\delta(A^i) \sqcap \delta(C^i) = \bot$. Intuitively, by requiring the assertions to be prime, we avoid the trivial cases when we are both assuming and checking that a node has a particular value at the same time. Practically speaking, requiring assertions to be prime is no real restriction since non-prime assertions usually are indications of some error(s) in the assertions.

1.2 Circuit Model Structure

In circuit verification, we restrict ourselves to a model structure of the form

$$(\langle I \times V, \sqsubseteq \rangle, Y)$$

where I and V are both complete lattices under partial orderings $\sqsubseteq_I$ and $\sqsubseteq_V$ respectively. Intuitively, I is the internal state space, and V is the space of visible nodes of a circuit, which is a space of cartesian products of $\{0, 1, \bot, \top\}$ and $\sqsubseteq_V$ is the cartesian (pair-wise) extension of the partial ordering $\preceq$: $\bot \preceq 0, 1 \preceq \top$. When it is clear by the context, we will use $\sqsubseteq$ to replace $\sqsubseteq_I$ and $\sqsubseteq_V$, and use $\bot$ to refer the bottom elements of V and I, as well as the bottom element of $I \times V$. Given an element $(i, v) \in I \times V$, we use $(i, v) \downarrow_I$ and $(i, v) \downarrow_V$ to denote i and v respectively.

Finally, we use the convention that the simple predicates used in circuit verification can only refer to the visible state. Consequently, for any circuit trajectory formula A, $\delta(A) = (\bot, v)$ for some $v \in V$. We call the trajectory formulas which specify properties of the values in V *the formulas defined on* V.

1.3 Conventions

Throughout this paper, the following notational conventions are adopted: Φ denotes a set of trajectory assertions. $A, A_1, \cdots, C, C_1, \cdots$ denote trajectory formulas. Also, for every trajectory formula A, we define:

$$A = A^0 \wedge \mathbf{N}A^1 \wedge \mathbf{N}^2 A^2 \wedge \cdots \mathbf{N}^m A^m$$

where all A^i are instantaneous formulas and $m = d(A) - 1$. Also, for $0 \leq i \leq m$, let

$$A^{\leq i} = A^0 \wedge \mathbf{N}A^1 \wedge \mathbf{N}^2 A^2 \wedge \cdots \mathbf{N}^i A^i$$

and

$$A^{\geq i} = A^i \wedge \mathbf{N}A^{i+1} \wedge \cdots \wedge \mathbf{N}^{m-i} A^m.$$

We use $\mathcal{M}$, $\mathcal{M}_\Phi$ to denote model structures. If an assertion φ holds in a model structure $\mathcal{M}$, we say $\mathcal{M}$ is *a model structure of* of φ. If every assertion in Φ holds in a model $\mathcal{M}$, we say that Φ is a model structure of Φ.

2 Constructing Circuit Models from Trajectory Assertions

In this section, we present a constructive method which generates a circuit model from a given set of trajectory assertions Φ such that every assertion of Φ holds true in the model [4].

2.1 Circuit Model Construction

Given an assertion $A \Rightarrow C$, *the set of all suffixes of* $A \Rightarrow C$, denoted by $P_{A,C}$, is:

$$P_{A,C} = \{(A^{\geq i}, C^{\geq i}) \mid 0 \leq i \leq depth(C) - 1 \}.$$

Let $A \Rightarrow C$ be a prime assertion. *The model structure constructed from* $A \Rightarrow C$, denoted by $\mathcal{M}_{A \Rightarrow C}$, is

$$\mathcal{M}_{A \Rightarrow C} = ((Q_{A,C} \times V, \sqsubseteq), Y_{A,C})$$

where

- $V = \{0, 1, X, \top\}^n$ is the visible circuit state partially ordered as before,
- $Q_{A,C}$ is the largest subset of $2^{P_{A,C}}$ such that for every $q \in Q_{A,C}$, $(A, C) \in q$,
- $\sqsubseteq$ is a binary relation of $Q_{A,C} \times V$: for arbitrary $q_1, q_2 \in Q_{A,C}$ and $v_1, v_2 \in V$, $(q_1, v_1) \sqsubseteq (q_2, v_2)$ if and only if $q_1 \subseteq q_2$ and $v_1 \sqsubseteq v_2$, and
- $Y_{A,C} : Q_{A,C} \times V \rightarrow Q_{A,C} \times V$ is the next-state function of the model. For arbitrary $q \in Q_{A,C}$ and $v \in V$, $Y_{A,C}(q, v) = (q', v')$ where

$$q' = \{ (a^{\geq 1}, c^{\geq 1}) \mid (a, c) \in q \text{ and } \delta(a^0) \downarrow_V \sqsubseteq_V v \} \cup \{ (A, C) \}$$

and

$$v' = \bigsqcup_{(a,c) \in q'} \delta(c^0) \downarrow_V \tag{1}$$

Let $\Phi = \{A_j \Rightarrow C_j \mid 0 \leq j \leq n-1 \}$ be a set of n prime assertions. Define

$$P_\Phi = \bigcup_{A \Rightarrow C \in \Phi} P_{A,C}$$

and Q_Φ as the largest subset of P_Φ such that every $q \in Q_\Phi$ contains Φ. *The model structure constructed from* Φ is

$$\mathcal{M}_\Phi = ((Q_\Phi \times V, \sqsubseteq), Y_\Phi)$$

where:

- $\sqsubseteq$ is a binary relation of $Q_\Phi \times V$ which is the pair-wise extension of $\subseteq$ and $\sqsubseteq_V$ to $Q_\Phi \times V$.

- $Y_\Phi : Q_\Phi \times V \to Q_\Phi \times V$ is the next-state function of the model. For any $q \in Q_\Phi$ and $v \in V$, $Y_\Phi(q, v) = (q', v')$ where v' is as in (1) and

$$q' = \{ (a^{\geq 1}, c^{\geq 1}) \mid (a, c) \in q,\ \delta(a^0) \downarrow_V \sqsubseteq_V v \} \cup \{ (A_i, C_i) \mid i = 0, \cdots, n-1 \}$$

Given the above construction, the following properties can easily be shown:

Lemma 2. *Let Φ be a set of prime trajectory formulas and A, C be arbitrary trajectory formulas.*

1. *For every $\varphi \in \Phi$, $\mathcal{M}_\Phi \models \varphi$.*
2. *Let $\tau^i_{\mathcal{M}_\Phi}(A) = (q, d)$ and $A' \Rightarrow C' \in \Phi$, if for some k : $0 \leq k \leq i$, $(A'^{\geq k}, C'^{\geq k}) \in q$, then $\delta(\mathbf{N}^{i-k} A'^{\leq k}) \sqsubseteq \delta(A)$.*

2.2 An Example of Model Construction

To illustrate the above construction, we will apply the method to the assertion $A \Rightarrow C$, where:

$$A : (a \text{ is } 0) \wedge (b \text{ is } 0) \wedge \mathbf{N}(a \text{ is } 0) \wedge \mathbf{N}(b \text{ is } 0)$$
$$C : \mathbf{N}^2(c \text{ is } 1).$$

We assume the visible nodes are a, b, and c, and consequently that the state of the visible components is drawn from $\{0, 1, X, \top\}^3$.

To construct the model structure $\mathcal{M}_{A,C}$ as defined earlier, we first need to compute the set of all suffixes of $A \Rightarrow C$. In our case we get:

$$\{(A, C),\ ((a \text{ is } 0) \wedge (b \text{ is } 0), \mathbf{N}^1(c \text{ is } 1)),\ (\mathsf{unc}, (c \text{ is } 1))\}$$

Therefore, $Q_{A,C} = \{ Q_1, Q_2, Q_3, Q_4 \}$, where

$$Q_1 = \{(A, C),\ ((a \text{ is } 0) \wedge (b \text{ is } 0), \mathbf{N}^1(c \text{ is } 1)),\ (\mathsf{unc}, (c \text{ is } 1))\}$$
$$Q_2 = \{(A, C),\ ((a \text{ is } 0) \wedge (b \text{ is } 0), \mathbf{N}^1(c \text{ is } 1))\}$$
$$Q_3 = \{(A, C),\ (\mathsf{unc}, (c \text{ is } 1))\}$$
$$Q_4 = \{(A, C)\}.$$

Note that $Q_4 \subseteq Q_i$ and $Q_i \subseteq Q_1$ for all $Q_i \in Q_{A,C}$ and that Q_2 is neither a subset of nor a superset of Q_3. Thus, Q_4 is the bottom element and Q_1 is the top element in the partial order $(Q_{A,C}, \subseteq)$. It follows trivially that $(Q_{A,C}, \subseteq)$ is a complete lattice. Consequently,

$$S_{A,C} = Q_{A,C} \times \{0, 1, X, \top\}^3$$

is also a complete lattice, where $(a, b) \sqsubseteq (c, d)$ iff $a \subseteq c$ and $b \sqsubseteq_T d$.

Finally, the function $Y_{A,C}$ is defined as follows: For any $q \in Q_{A,C}$ and $d \in \{0,1,X,\top\}^3$, we have $Y_{A,C}(q,d) = (q'(q,d), d'(q,d))$, where

$$q'(q,d) = \begin{cases} Q_1 & \text{if } q \in \{Q_1, Q_2\} \text{ and } (0,0,X) \sqsubseteq_T d \\ Q_2 & \text{if } q \in \{Q_3, Q_4\} \text{ and } (0,0,X) \sqsubseteq_T d \\ Q_4 & \text{otherwise,} \end{cases}$$

and

$$d'(q,d) = \begin{cases} (X,X,1) & \text{if } q \in \{Q_1, Q_2\} \text{ and } (0,0,X) \sqsubseteq_T d \\ (X,X,X) & \text{otherwise.} \end{cases}$$

To illustrate the use of this derived model structure, consider computing the defining trajectory in this model structure for the formula:

$$A' = (a \text{ is } 0) \wedge (b \text{ is } 0) \wedge \mathbf{N}(a \text{ is } 0) \wedge \mathbf{N}(b \text{ is } 0) \wedge \mathbf{N}^2(a \text{ is } 0) \wedge \mathbf{N}^2(b \text{ is } 0).$$

First, by definition, $\delta(A')$ equals:

$$(Q_4,(0,0,X))\,(Q_4,(0,0,X))\,(Q_4,(0,0,X))\,(Q_4,(X,X,X))\,(Q_4,(X,X,X))\,\cdots$$

By the definition of τ,

$$\begin{aligned}
\tau^0(A') &= \delta^0(A') = (Q_4,(0,0,\boxed{X})) \\
\tau^1(A') &= Y(Q_4,(0,0,X)) \sqcup \delta^1(A') \\
&= (Q_2,(X,X,X)) \sqcup (Q_4,(0,0,X)) \\
&= (Q_2,(0,0,\boxed{X})) \\
\tau^2(A') &= Y(Q_2,(0,0,X)) \sqcup \delta^2(A') \\
&= (Q_1,(X,X,1)) \sqcup (Q_4,(0,0,X)) \\
&= (Q_1,(0,0,\boxed{1})) \\
\tau^3(A') &= Y(Q_1,(0,0,1)) \sqcup \delta^3_{A'} \\
&= (Q_1,(X,X,1)) \sqcup (Q_4,(X,X,X)) \\
&= (Q_1,(X,X,\boxed{1})) \\
\tau^4(A') &= Y(Q_1,(X,X,1)) \sqcup \delta^4_{A'} \\
&= (Q_4,(X,X,X)) \sqcup (Q_4,(X,X,X)) \\
&= (Q_4,(X,X,\boxed{X})) \\
\tau^5(A') &= Y(Q_4,(X,X,X)) \sqcup \delta^5_{A'} \\
&= (Q_4,(X,X,X)) \sqcup (Q_4,(X,X,X)) \\
&= (Q_4,(X,X,\boxed{X})) \\
&\vdots
\end{aligned}$$

The values surrounded by boxes represent the first 6 values on node "c".

3 A Simple Inference System

There are 7 inference rules in the system. They are presented in the form:

$$\Phi \frac{cond_1, cond_2, \cdots, cond_n}{A \Rightarrow C} \tag{2}$$

where Φ is a set of assertions which are premises of inferences. A, C are trajectory formulas. (2) reads: if the conditions $cond_1, \cdots, cond_n$ are true, then $A \Rightarrow C$ can be derived from Φ.

Rule 1. (Identity)

$$\{\}\frac{}{A \Rightarrow A}$$

Rule 2. (TimeShift)

$$\{A \Rightarrow C\}\frac{}{\mathbf{N}^t A \Rightarrow \mathbf{N}^t C} \qquad t \geq 0$$

Rule 3. (AntecedentStrengthen)

$$\{A \Rightarrow C\}\frac{\delta(A) \sqsubseteq \delta(A_1)}{A_1 \Rightarrow C}$$

Rule 4. (ConsequentWeaken)

$$\{A \Rightarrow C\}\frac{\delta(C_1) \sqsubseteq \delta(C)}{A \Rightarrow C_1}$$

Rule 5. (Conjunct)

$$\{A \Rightarrow C_1, \quad A \Rightarrow C_2\}\frac{}{A \Rightarrow C_1 \wedge C_2}$$

Rule 6. (Transitivity)

$$\{A_1 \Rightarrow C_1,\ A_2 \Rightarrow C_2\}\frac{\delta(A_2) \sqsubseteq \delta(C_1)}{A_1 \Rightarrow C_2}$$

Rule 7. (AntecedentTruncate)

$$\{A \Rightarrow C\}\frac{\forall i \leq depth(C).\ \delta(A)^i \sqcap \delta(C)^i = \bot}{A^{\leq t} \Rightarrow C^{\leq t+1}} \qquad t \geq 0$$

For proofs of the soundness of Rules 1–6, see [2].

The soundness of AntecedentTruncate can be shown as follows: Let $t \geq 0$ and $\mathcal{M}$ be a model structure of $A \Rightarrow C$, *i.e.*, $\delta(C) \sqsubseteq \tau_{\mathcal{M}}(A)$ (Theorem 1).

1. Since $\delta(C) \sqsubseteq \tau_{\mathcal{M}}(A)$, $\delta(C^{\leq t}) \sqsubseteq \tau_{\mathcal{M}}(A^{\leq t})$. Therefore, $\mathcal{M} \models A^{\leq t} \Rightarrow C^{\leq t}$.
2. If $t = 0$, then $A^{\leq 0} = C^{\leq 0} = \mathsf{unc}$. By the definition of τ, $\delta(C)^0 \sqsubseteq \delta(A)^0$. Because $\delta_C^0 \sqcap \delta(A)^0 = \bot$, $C^0 = \mathsf{unc}$. Therefore, AntecedentTruncate is reduced to
$$\{A \Rightarrow C\}\frac{\forall i \leq depth(C).\ \delta(A)^i \sqcap \delta(C)^i = \bot}{\mathsf{unc} \Rightarrow \mathsf{unc}} \qquad t \geq 0$$
which is trivially true.
Assume $t > 0$. Because $\tau_{\mathcal{M}}^t(A)$ depends only on $A^{\leq t}$, $\tau_{\mathcal{M}}^t(A) = \tau_{\mathcal{M}}^t(A^{\leq t})$. Therefore,
$$\delta_C^t \sqsubseteq \tau_{\mathcal{M}}^t(A^{\leq t}) = Y(\tau_{\mathcal{M}}^{t-1}(A^{\leq t})) \sqcup \delta_A^t$$
Since $\delta_A^{t+1} \sqcap \delta_C^{t+1} = \bot$, $\delta_C^t \sqsubseteq Y(\tau_{\mathcal{M}}^{t-1}(A^{\leq t}))$. In both cases,
$$\mathcal{M} \models A^{\leq t} \Rightarrow \mathbf{N}^t C^t$$

Combining the results from 1 and 2 by Conjunct:

$$\mathcal{M} \models A^{\leq t} \Rightarrow C^{\leq t+1}$$ □

4 The Completeness of the Inference System

In this section, we prove the main result of this paper: the inference system given in Section 3 is powerful enough to derive all the logical consequences of Φ. A logical consequence of Φ is an assertion which is true in every model structure of Φ. We proved this claim by showing that if an assertion (φ) is true in the model structure $\mathcal{M}_\Phi$, then it can be derived by the inference system from Φ.

Theorem 3. *Let Φ be a set of trajectory formulas, A be any trajectory formula, C be any instantaneous trajectory formula, and $m \geq 0$. If $\mathcal{M}_\Phi \models A \Rightarrow N^m C$ then $\Phi \vdash A \Rightarrow N^m C$.*

A corollary of the theorem is the major conclusion of this paper:

Corollary 4. *Let Φ be a set of trajectory assertions and $\mathcal{M}_\Phi$ be the model structure of Φ. For any assertion $A \Rightarrow C$, $\mathcal{M}_\Phi \models A \Rightarrow C$ if and only if $\Phi \vdash A \Rightarrow C$.*

Proof. The proof that $\Phi \vdash A \Rightarrow C$ implies $\mathcal{M}_\Phi \models A \Rightarrow C$ follows the soundness proofs of the inference rules.

Assume $\mathcal{M}_\Phi \models A \Rightarrow C$. For every $t \geq 0$, $\mathcal{M}_\Phi \models A \Rightarrow \mathbf{N}^t C^t$. By Theorem 3, $\Phi \vdash A \Rightarrow \mathbf{N}^t C^t$ for every $t \geq 0$. Then by Conjunct (inference rule),

$$\Phi \vdash A \Rightarrow \bigwedge_{t \geq 0} \mathbf{N}^t C^t$$

which is equivalent as saying that $\Phi \vdash A \Rightarrow C$. □

We now prove Theorem 3. Although the theorem is equally applicable to general trajectory assertions, the following proof assumes that Φ contains only prime assertions.

Proof. Prove by induction on $m \geq 0$.

The base case is when $m = 0$. $\mathcal{M}_\Phi \models A \Rightarrow N^0 C$ implies that

$$\delta(C) \sqsubseteq \tau^0_{\mathcal{M}_\Phi}(A) = \delta^0_A \tag{3}$$

Then
(a) $\Phi \vdash A \Rightarrow A$ — By the Identity Axiom
(b) $\Phi \vdash A \Rightarrow C$ — By ConsequentWeaken and (a), (3)

Assume $m \geq 0$, and for every $i \leq m$, $\mathcal{M}_\Phi \models A \Rightarrow \mathbf{N}^i C$ implies $\Phi \vdash A \Rightarrow \mathbf{N}^i C$.

Assume $\mathcal{M}_\Phi \models A \Rightarrow C$ and let $\tau^{m+1}_{\mathcal{M}_\Phi}(A) = (q, d)$ where q is a set of suffixes of the assertions in Φ: without losing generality, assume there exists $l \leq n$, such that

$$q = \{ (A_i^{\geq k_i}, C_i^{\geq k_i}) \mid 0 \leq i \leq l,\ 1 \leq k_i \leq m \}$$

By the definition of $\mathcal{M}_\Phi$,

$$\delta(C) \sqsubseteq d' = \delta(A)^{m+1} \sqcup \bigsqcup_{i \leq l} \delta(C_i^{k_i}) \tag{4}$$

By Lemma 2, $\tau^{m+1}_{\mathcal{M}_\Phi}(A) = (q, d)$ implies that for every $i : 0 \leq i \leq l$,

$$\delta(\mathbf{N}^{m-k_i+1} A_i^{\leq k_i}) \sqsubseteq \delta(A)$$

Therefore, $\mathcal{M}_\Phi \models A \Rightarrow \mathbf{N}^{m-k_i+1} A_i^{\leq k_i}$. This means that for every $j : 0 \leq j \leq k_i - 1$, $\mathcal{M}_\Phi \models A \Rightarrow \mathbf{N}^{m-k_i+1}\, \mathbf{N}^j A_i^j$. By the induction hypothesis,

$$\Phi \vdash A \Rightarrow \mathbf{N}^{m+j-k_i+1} A^j \qquad 0 \leq j \leq k_i - 1 \tag{5}$$

By Conjunct, (5) implies

$$\Phi \vdash A \Rightarrow \mathbf{N}^{m-k_i+1} A_i^{\leq k_i} \tag{6}$$

What follows derives that $\Phi \vdash A \Rightarrow \mathbf{N}^{m+1} C$.

Step 1. Because $A_i \Rightarrow C_i \in \Phi$ for every $i : 0 \leq i \leq l$, by TimeShift,

$$\Phi \vdash \mathbf{N}^{m-k_i+1} A_i \Rightarrow \mathbf{N}^{m-k_i+1} C_i \qquad 0 \leq i \leq l \tag{7}$$

Step 2. By ConsequentWeaken, (7) implies

$$\Phi \vdash \mathbf{N}^{m-k_i+1} A_i \Rightarrow \mathbf{N}^{m-k_i+1} \mathbf{N}^{k_i} C_i^{k_i} \qquad 0 \leq i \leq l$$

Therefore,

$$\Phi \vdash \mathbf{N}^{m-k_i+1} A_i \Rightarrow \mathbf{N}^{m+1} C_i^{k_i} \qquad 0 \leq i \leq l \tag{8}$$

Step 3. By AntecedentTruncate, (8) implies

$$\Phi \vdash \mathbf{N}^{m-k_i+1} A_i^{\leq k_i} \Rightarrow \mathbf{N}^{m+1} C_i^{k_i} \qquad 0 \leq i \leq l \tag{9}$$

Step 4. By Transitivity, (6) and (9) imply

$$\Phi \vdash A \Rightarrow \mathbf{N}^{m+1} C_i^{k_i} \qquad 0 \leq i \leq l \tag{10}$$

Step 5. By Conjunct, (10) implies

$$\Phi \vdash A \Rightarrow \bigwedge_{0 \leq i \leq l} \mathbf{N}^{m+1} C_i^{k_i} \tag{11}$$

Step 6. By Identity and ConsequentWeaken,

$$\Phi \vdash A \Rightarrow \mathbf{N}^{m+1} A^{m+1} \tag{12}$$

By Conjunct, (11) and (12) imply

$$\Phi \vdash A \Rightarrow \mathbf{N}^{m+1} A^{m+1} \wedge \bigwedge_{0 \leq i \leq l} \mathbf{N}^{m+1} C_i^{k_i} \tag{13}$$

Step 7. Let

$$B = \mathbf{N}^{m+1} A^{m+1} \wedge \bigwedge_{0 \leq i \leq l} \mathbf{N}^{m+1} C_i^{k_i}$$

By the definition of δ,

$$\delta(B)^{m+1} = \delta(A^{m+1}) \sqcup \bigsqcup_{0 \leq i \leq l} C_i^{k_i}$$

Therefore, by (4) and ConsequentWeaken, (13) implies $\Phi \vdash A \Rightarrow \mathbf{N}^{m+1} C$. □

5 Model Construction Revisited

When constructing a circuit model from a give set of assertions, we assumed that every assertion in the given set is prime and non-symbolic. However, the method can be extended to non-prime and symbolic assertions as well.

First, a symbolic trajectory assertion can be viewed as a compact representation of a set of assertions. For example, an inverter specification:

$$(a \text{ is } x) \Rightarrow \mathbf{N}(b \text{ is } \neg x)$$

is equivalent to two assertions:

$$(a \text{ is } 0) \Rightarrow \mathbf{N}(b \text{ is } 1) \quad \text{and} \quad (a \text{ is } 1) \Rightarrow \mathbf{N}(b \text{ is } 0)$$

Therefore, a model of a symbolic assertion can be constructed by means of constructing the model of the corresponding set of assertions. The following theorem relates models of general trajectory assertions to models of prime assertions.

Theorem 5. *Let $A \Rightarrow C$ be an arbitrary trajectory assertion defined on V. There exists a unique trajectory formula C' such that*

1. *$\delta_A^i \sqcap \delta(C')^i = \bot$, and $\delta(C')^i \sqsubseteq \delta(C)^i$ for every $i \geq 0$.*
2. *for any trajectory assertion $A'' \Rightarrow C''$, $\mathcal{M}_{A \Rightarrow C} \models A'' \Rightarrow C''$ if and only if $\mathcal{M}_{A \Rightarrow C'} \models A'' \Rightarrow C''$.*

It can be shown that for any given formulas A and C defined on V, such a C' can be computed. Therefore, given a non-prime assertion $A \Rightarrow C$, its constructed model structure is defined as that of $A \Rightarrow C'$. Note that, in general, such C' may not exist in arbitrary lattices. Interested readers are referred to [4].

Finally, a consequence of Theorem 3 is that given a set of trajectory assertions Φ, $\mathcal{M}_\Phi$ is the most-abstract model structure of Φ.

Theorem 6. *Let $\mathcal{M}$ be an arbitrary model of Φ. If $A \Rightarrow C$ is an assertion of $\mathcal{M}_\Phi$ defined on the domain of Φ, then $A \Rightarrow C$ is also an assertion of $\mathcal{M}$.*

Proof. We prove the theorem by contradiction: assume there exists model structure $\mathcal{M}$ of Φ and an assertion $A \Rightarrow C$ of $\mathcal{M}_\Phi$ such that

$$\mathcal{M}_\Phi \models A \Rightarrow C \qquad \text{but} \qquad \mathcal{M} \not\models A \Rightarrow C$$

By Corollary 4, $M_\Phi \models A \Rightarrow C$ implies $\Phi \vdash A \Rightarrow C$. By the soundness of the inference system, $A \Rightarrow C$ is an assertion of every model structure of Φ. In particular, it is an assertion of $\mathcal{M}$, *i.e.*, $\mathcal{M} \models A \Rightarrow C$. This contradicts to our assumption that $\mathcal{M} \not\models A \Rightarrow C$. This proves that $\mathcal{M} \models A \Rightarrow C$. □

6 Conclusions

There are two main results in this paper: the construction of a most general model structure from a set of trajectory assertions, and the soundness and completeness proofs for the inference system. Although the completeness result is more of theoretical, than practical value, it is nevertheless useful in that no further "basic" inference rules are needed. Consequently, a very safe—but still very practical—theorem prover for composing trajectory evaluation results can easily be constructed by simply implementing these few inference rules as an abstract data type and exporting only these construction functions, as pioneered by LCF[5] and extensively used by offsprings of LCF.

The method of constructing the most-general circuit model from a given set of trajectory assertions Φ is interesting in its own right. For example, it makes it possible to "simulate" a specification. This is often very useful in avoiding simple errors in the specifications. Another interesting possibility is to use the construction to create a most general model structure for some part of the system that has not yet been designed. This model structure could then be composed with parts that has been designed to allow verification of the complete system. However, for either of these applications to be practical, the construction must be implemented to work over a symbolic domain efficiently. We have in fact done so and are currently in the process of building a small prototype system to determine the practicality of the approach.

Acknowledgments We are indebted to the reviewers who have made many helpful and constructive remarks on an earlier draft of this paper.

References

1. SEGER, C.-J., AND BRYANT, R. Formal verification of digital circuits by symbolic evaluation of partially-ordered trajectories. Tech. Rep. Technical Report 93-8, The Computer Science Department, The University of British Columbia, The Computer Science Department, The University of B.C. Vancouver B.C. V6T 1Z4, 1993.
2. HAZELHURST, S., AND SEGER, C.-J. A simple theorem prover based on symbolic trajectory evaluation and obdds. Tech. Rep. Technical Report 93-41, The Computer Science Department, The University of British Columbia, The Computer Science Department, The University of B.C. Vancouver B.C. V6T 1Z4, 1993. (An abridged version of this work appears in this proceedings).
3. SEGER, C.-J. Voss – a formal hardware verification system, user's guide. Tech. Rep. Technical Report 93-45, The Computer Science Department, The University of British Columbia, The Computer Science Department, The University of B.C. Vancouver B.C. V6T 1Z4, 1993.
4. ZHU, Z. Construction of circuit models from trajectory specifications. In progress, 1994.
5. GORDON, M., MILNER, A., AND C., W. *Edinburgh LCF*, vol. 78 of *Lecture Notes in Computer Science*. Springer-Verlag, 1979.