

Nonstandard Axiomatic Semantics

Patrick Cousot

Courant Institute, New York University

Abstract. Similar to Skolem’s nonstandard models of Peano’s naturals, we show that axiomatic semantics based on Hoare logic has nonstandard models and so does not specify a unique, well-defined, and formal operational semantics of programming languages. We propose to enrich axiomatic semantics with additional proof obligations to solve this ambiguity problem. These proof obligations are always satisfied for standard trace models so that Hoare logic proofs are unchanged for these standard models.

Keywords: Axiomatic semantics, Hoare logic, Trace semantics, Nonstandard semantics, Semantics standardisation, Abstract Interpretation.

1 Introduction

In addition to Peano’s axiomatization [41] of the naturals, Dedekind’s categoricity [25] eliminates Skolem’s nonstandard models of arithmetic [42] thus ensuring that the definition of the naturals is unique and corresponds to the intended interpretation of the naturals, up to an isomorphism (section 5).

Since Hoare logic (section 3) is an abstraction (section 4) of a finite or infinite trace semantics indexed by the naturals [15,17] (section 2), an inevitable question is whether the logic is expressive enough to uniquely characterize this trace semantics, up to an isomorphism. This would show that Hoare logic is an operational characterization of program executions. On the contrary, we show that Lauer’s axiomatic semantics [36,32] based on Hoare logic [31] suffers from an ambiguity problem similar to that of Peano arithmetics. The rule for iteration (using an inductive invariant that must hold for all states of infinite executions) does not uniquely define the intended meaning of terminating and nonterminating loops (up to isomorphism) since many different infinite pasts and futures (section 6) are possible without changing the logic theory (section 7). We study this phenomenon (sections 8 and 9) and propose a remedy (section 11) inspired by termination proof methods (section 10) extended so as to allow for non-termination. The additional requirements of the axiomatic semantics rule out nonstandard semantics and are always satisfied by the standard semantics (section 12). It follows that Hoare logic proofs are unchanged under the hypothesis that Hoare logic should be interpreted with respect to the standard semantics.

2 Standard Trace Semantics of an Imperative Language

We consider the standard trace semantics [48] of a simple imperative language with iteration $\mathcal{L}$ (e.g. [16, Ch. 7]). We let $\ell, \ell', \dots \in \mathbb{L}$ be a set of labels/program points, $\mathbf{x}, \mathbf{y}, \dots \in \mathbb{X}$ be a set of variables, $v, 0, \text{true}, \dots \in \mathbb{V}$ be a set of values, $\rho, \rho', \dots \in \mathbb{E} \triangleq \mathbb{X} \rightarrow \mathbb{V}$ be a set of environments, and $\langle \ell, \rho \rangle, \dots \in \Sigma \triangleq \mathbb{L} \times \mathbb{E}$ be a set of states. Executions $\sigma, \sigma', \sigma_b, \dots \in \Sigma^{*\infty} \triangleq \Sigma^* \cup \Sigma^\infty$ are modeled by finite traces $\Sigma^* \triangleq \bigcup_{n \in \mathbb{N}} [0, n] \rightarrow \Sigma$ and infinite traces $\Sigma^\infty \triangleq \mathbb{N} \rightarrow \Sigma$. Trace concatenation is by juxtaposition of states and traces. A program semantics is a set of traces in $\wp(\Sigma^{*\infty})$.

The iteration statement $\mathbf{S} = \text{while}^\ell \mathbf{B} \text{ do } \ell' \mathbf{Sb} \ell''$ of $\mathcal{L}$ has unique labels $\ell \in \mathbb{L}$ at $\mathbf{S}$ and after the loop body $\mathbf{Sb}$, $\ell' \in \mathbb{L} \setminus \{\ell\}$ at $\mathbf{Sb}$, and $\ell'' \in \mathbb{L} \setminus \{\ell, \ell'\}$ after the iteration $\mathbf{S}$. The finite trace semantics $\llbracket \mathbf{S} \rrbracket^* \in \wp(\Sigma^*)$ is a set of finite traces starting at ℓ and finishing at ℓ'' . The infinite trace semantics $\llbracket \mathbf{S} \rrbracket^\infty \in \wp(\Sigma^\infty)$ is a set of infinite traces starting at ℓ that never terminate. These semantics may be empty (but not both). They are traditionally defined by structural induction (i.e. on the program syntax). Let $\llbracket \mathbf{Sb} \rrbracket^*$ (respectively $\llbracket \mathbf{Sb} \rrbracket^\infty$) be the standard finite (resp. infinite) trace semantics of the loop body $\mathbf{Sb}$, already defined by structural induction. The finite executions in $\llbracket \mathbf{Sb} \rrbracket^*$ have the form $\langle \ell', \rho \rangle \sigma_b \langle \ell, \rho' \rangle$ since they start at $\mathbf{Sb}$ that is ℓ' and terminate after $\mathbf{Sb}$ that is ℓ . These standard trace semantics are defined using least **lfp** and greatest **gfp** fixpoints of $\sqsubseteq$ -increasing functions $B^* \llbracket \mathbf{S} \rrbracket$ and $F^\infty \llbracket \mathbf{S} \rrbracket$ on the complete lattice $(\wp(\Sigma^{*\infty}), \sqsubseteq)$ [43], as follows (in these formulas, $\ell, \ell', \ell'' \in \mathbb{L}$ are labels of a statement $\mathbf{S}$, not local dummy variables of the formulas. So we write $\{\ell\}$ instead of the more rigorous $\{x \mid x = \ell\}$ where the dummy variable x is local to the formula and ℓ must be defined globally to be a label of a statement $\mathbf{S}$).

$$B^* \llbracket \mathbf{S} \rrbracket X \triangleq \{ \langle \ell, \rho \rangle \langle \ell'', \rho \rangle \mid \neg \mathcal{B}[\mathbf{B}] \rho \} \cup \{ \langle \ell, \rho \rangle \langle \ell', \rho \rangle \sigma_b \langle \ell, \rho' \rangle \sigma \mid \mathcal{B}[\mathbf{B}] \rho \wedge \langle \ell', \rho \rangle \sigma_b \langle \ell, \rho' \rangle \in \llbracket \mathbf{Sb} \rrbracket^* \wedge \langle \ell, \rho' \rangle \sigma \in X \} \quad (1)$$

$$F^\infty \llbracket \mathbf{S} \rrbracket X \triangleq \{ \langle \ell, \rho \rangle \langle \ell', \rho \rangle \sigma_b \langle \ell, \rho' \rangle \sigma \mid \mathcal{B}[\mathbf{B}] \rho \wedge \langle \ell', \rho \rangle \sigma_b \langle \ell, \rho' \rangle \in \llbracket \mathbf{Sb} \rrbracket^* \wedge \langle \ell, \rho' \rangle \sigma \in X \} \cup \{ \langle \ell, \rho \rangle \langle \ell', \rho \rangle \sigma_b \mid \mathcal{B}[\mathbf{B}] \rho \wedge \langle \ell', \rho \rangle \sigma_b \in \llbracket \mathbf{Sb} \rrbracket^\infty \} \quad (2)$$

$$\llbracket \mathbf{S} \rrbracket^* \triangleq \text{lfp}^\subseteq B^* \llbracket \mathbf{S} \rrbracket = \text{gfp}^\subseteq B^* \llbracket \mathbf{S} \rrbracket \quad \llbracket \mathbf{S} \rrbracket^\infty \triangleq \text{gfp}^\subseteq F^\infty \llbracket \mathbf{S} \rrbracket \quad \llbracket \mathbf{S} \rrbracket^{*\infty} \triangleq \llbracket \mathbf{S} \rrbracket^* \cup \llbracket \mathbf{S} \rrbracket^\infty.$$

(1) reflects the fact that $\text{while } \mathbf{B} \text{ do } \mathbf{Sb}$ is equivalent to $\text{if } \mathbf{B} \text{ then } (\mathbf{S}; \text{while } \mathbf{B} \text{ do } \mathbf{Sb})$. (2) reflects the fact that a loop does not terminate either because $\mathbf{B}$ is always true after zero or more terminating executions of its body or because after a number of iterations, the loop body does not terminate. The fixpoint equality $\text{lfp}^\subseteq B^* \llbracket \mathbf{S} \rrbracket = \text{gfp}^\subseteq B^* \llbracket \mathbf{S} \rrbracket$ follows from [15, Theorem 11].

In examples with only one variable like $\mathbf{x} := 0; \text{while}^\ell \text{true do } \ell' \mathbf{x} := \mathbf{x} + 1; \ell''$ we simplify the trace by abstracting states $\langle \ell', \rho \rangle$ by $\rho(\mathbf{x})$ when $\ell' \in \{\ell, \ell''\}$ and ignoring the other states in the trace e.g. $0 \ 1 \ 2 \ \dots$

By defining the computational ordering $X \sqsubseteq Y \triangleq ((X \cap \Sigma^*) \subseteq (Y \cap \Sigma^*)) \wedge ((X \cap \Sigma^\infty) \supseteq (Y \cap \Sigma^\infty))$, $(\wp(\Sigma^{*\infty}), \sqsubseteq)$ is a complete lattice, $B^* \llbracket \mathbf{S} \rrbracket$ and $F^\infty \llbracket \mathbf{S} \rrbracket$ are $\sqsubseteq$ -increasing, and we have $\llbracket \mathbf{S} \rrbracket^{*\infty} = \text{lfp}^\sqsubseteq \lambda X. B^* \llbracket \mathbf{S} \rrbracket (X \cap \Sigma^*) \cup F^\infty \llbracket \mathbf{S} \rrbracket (X \cap \Sigma^\infty)$ [20].

3 Hoare Logic

Hoare logic [31] uses triples $\{P\} S \{Q\}$ (originally written $P \{S\} Q$ in [31]) to specify that any execution of statement S from a state in P , which terminates (if ever), will terminate in a state satisfying Q . P and Q are assertions on (the values of) program variables. Auxiliary variables not appearing in S can be introduced to record the initial value of variables in P and express a relation between initial and final values in Q .

To avoid the inexpressivity problems of Hoare logic related to the choice of a logic to specify P (or Q) [11,12], we assume (unless specified otherwise) that predicates are defined in extension as the set of all states satisfying P (or Q). Hoare defined the semantics of a simple imperative programming language by the following rules, which are literally extracted from [31], except that predicates are defined in extension as sets of environments ρ mapping variables x to their value $\rho(x)$ satisfying these predicates. We define the semantic assignment $\rho[x \leftarrow v]$ as $\rho[x \leftarrow v](x) = v$ and $\rho[x \leftarrow v](y) = \rho(y)$ when $y \neq x$. We assume expressions f and B are well-formed first order logic formulas without quantifiers (hence subject to nonstandard interpretations) where free variables are program variables or auxiliary variables.

- Axiom of Assignment: $\{P\} x := f \{ \rho[x \leftarrow \llbracket f \rrbracket(\rho)] \mid \rho \in P \}$ where $\llbracket f \rrbracket(\rho)$ is the value of expression f when evaluated with values $\rho(y)$ of the variables y of f ;
- (Left and Right) Rules of Consequence:
 - If $\{P\} Q \{R\}$ and $R \subseteq S$ then $\{P\} Q \{S\}$ (where logical implication $R \subseteq S$ is written $R \supset S$ in [31]);
 - If $\{P\} Q \{R\}$ and $S \subseteq P$ then $\{S\} Q \{R\}$;
- Rule of Composition:

If $\{P\} Q_1 \{R_1\}$ and $\{R_1\} Q_2 \{R\}$ then $\{P\} Q_1; Q_2 \{R\}$
- Rule of Iteration:

If $\{P \cap B\} S \{P\}$ then $\{P\} \text{while } B \text{ do } S \{ \neg B \cap P \}$ where B (respectively $\neg B$) stands for $\{ \rho \mid \text{the evaluation of Boolean expression } B \text{ for the values } \rho(x) \text{ of the variables } x \text{ appearing in } B \text{ is true} \}$ (respectively false)."

The derived union and intersection rules, which from $\forall i \in \Delta . \{P_i\} S \{Q_i\}$ infer $\{ \bigcup_{i \in \Delta} P_i \} S \{ \bigcup_{i \in \Delta} Q_i \}$ and $\{ \bigcap_{i \in \Delta} P_i \} S \{ \bigcap_{i \in \Delta} Q_i \}$, can be proved by induction on the structure of programs [21].

Remark 1. Although Hoare logic is for partial correctness only, whose definition completely ignores infinite behaviors, something is nevertheless said about these infinite behaviors by the loop invariant when the precondition does not exclude infinite behaviors. For example, $P \triangleq x > 0$ in $\{P \cap x > 0\} x := x + 1; \{P\}$ for $\{P\} \text{while } x > 0 \text{ do } x := x + 1; \{false\}$ provides the reachability information that infinite iterations all have x strictly positive.

4 The Standard Interpretation of Hoare Logic is an Abstraction of the Standard Trace Semantics

Define, for all statements $S \in \mathcal{L}$ where ℓ is at S and ℓ' is after S , the abstraction α_S of a standard trace semantics T into the set of all Hoare triples valid for that semantics as follows [17].

$$\alpha_S(T) \triangleq \{ \{P\}S\{Q\} \mid P, Q \in \wp(\mathbb{E}) \wedge \forall \langle \ell, \rho \rangle \sigma \langle \ell', \rho' \rangle \in T \cap \Sigma^* . \rho \in P \Rightarrow \rho' \in Q \}$$

The abstraction α_S defines the standard interpretation of Hoare triples as partial correctness. α_S is the lower adjoint of a Galois connection [17]. Hoare logic theory $\mathcal{H}(\mathcal{L})$ for the language $\mathcal{L}$ is the set

$$\mathcal{H}(\mathcal{L}) \triangleq \bigcup \{ \alpha_S(\llbracket S \rrbracket^{*\infty}) \mid S \in \mathcal{L} \} \quad (3)$$

of all Hoare triples of all statements of the language $\mathcal{L}$ [17]. Hoare proof rules follow from this definition by considering the abstraction α_S of the structural fixpoint definition of the standard trace semantics $\llbracket S \rrbracket^{*\infty}$ [18].

5 Nonstandard Models of Arithmetic

Peano [41] defined the naturals $\mathbb{N}$ by $0 \in \mathbb{N}$ and $\forall n \in \mathbb{N} . \mathcal{S}n \in \mathbb{N}$ where $\mathcal{S}n = n + 1$ is the successor function, together with $\forall n \in \mathbb{N} . \mathcal{S}n \neq 0$ and injectivity $\forall n, m \in \mathbb{N} . n + 1 = m + 1 \Rightarrow n = m$. The intention was to define the naturals $\mathbb{N}$ as $0 < 1 < 2 < 3 < \dots$. Skolem [42] showed that there are nonstandard models of Peano's definition such as the model $S = 0 < 1 < 2 < 3 < \dots < \dots - 2 < -1 < 0' < 1' < 2' \dots$ which obviously satisfies Peano's axioms. Peano's arithmetic is not well-defined since, up to isomorphism, the axiomatization has more than one model.

Dedekind's idea [25] to ensure the uniqueness of the model (up to an isomorphism) relies on the chain principle (or second-order induction) that is essentially De Morgan's mathematical induction (proofs by recurrence) [24]. From $P(0)$ and $\forall n . P(n) \Rightarrow P(n + 1)$ conclude $\forall n \in \mathbb{N} . P(n)$ (which is also the contrapositive form of Fermat's earlier infinite descent [26,49], from $\forall n . \neg P(n + 1) \Rightarrow \neg P(n)$ and $P(0)$ conclude $\forall n \in \mathbb{N} . P(n)$). Dedekind's additional requirement is that the models of Peano's naturals are those for which proof by recurrence is valid (sound, correct). This eliminates the Skolem's model S since for $P = \{0, 1, 2, \dots\}$ (which is a definition in extension of a predicate where $P(n)$ denotes $n \in P$) the hypotheses of the recurrence principle are satisfied by P while the conclusion $S \subseteq P$ is not. So the soundness of the principle of proof by recurrence effectively limits Peano's definition to only those elements generated by iterating the successor function $\mathcal{S}$ starting from 0.

Observe that $\mathbb{N}$ is the smallest model of Peano's naturals since $\mathbb{N} \setminus \{n\}$, $n \in \mathbb{N}$ does not satisfy Peano's definition. This is Dedekind's idea of defining $\mathbb{N}$ as the smallest subset of an infinite set such that there exists a transformation (the

successor function $\mathcal{S}$) that is injective and a base element 0 not in the range of $\mathcal{S}$ [25, §71]. Otherwise stated, the naturals are the least fixpoint

$$\mathbb{N} = \text{lfp}^{\subseteq} F \quad (4)$$

of a transformer $F \in \wp(\mathbb{U}) \rightarrow \wp(\mathbb{U})$ defined as $F(X) \triangleq \{0\} \cup \{\mathcal{S}n \mid n \in X\}$ operating on the powerset of a universe $\mathbb{U}$ which can be chosen for example as finite sentences over an alphabet containing 0, 1, and $\mathcal{S}$ possibly $+$, $-$, $\times$, and any other required letters. Then the powerset $\langle \wp(\mathbb{U}), \subseteq \rangle$ is a complete lattice and F is $\subseteq$ -increasing (monotone) so that the $\subseteq$ -least fixpoint $\text{lfp}^{\subseteq} F$ of F exists by Tarski's theorem [43] and is unique by definition of “least”. This definition allows us to recover proof by recurrence thanks to Park's fixpoint induction [40] (itself a direct consequence of Tarski's fixpoint theorem [43]) as follows ($P \in \wp(\mathbb{U})$)

$$\begin{aligned} & \mathbb{N} \subseteq P \\ \Leftrightarrow & \text{lfp}^{\subseteq} F \subseteq P && \{\text{fixpoint definition } \mathbb{N} = \text{lfp}^{\subseteq} F\} \\ \Leftrightarrow & \exists I \in \wp(\mathbb{U}) . F(I) \subseteq I \wedge I \subseteq P && \{\text{fixpoint induction [40]}\} \\ \Leftrightarrow & \exists I \in \wp(\mathbb{U}) . \{0\} \cup \{\mathcal{S}n \mid n \in I\} \subseteq I \wedge I \subseteq P && \{\text{definition of } F\} \\ \Leftrightarrow & \exists I \in \wp(\mathbb{U}) . 0 \in I \wedge \forall n \in I . \mathcal{S}n \in I \wedge I \subseteq P && \{\text{definition of } \cup \text{ and } \subseteq\} \end{aligned}$$

This is exactly second-order induction also called recurrence principle (except that it is emphasized that P might have to be strengthened to an inductive argument I to be provable by recurrence, a situation which is so common in mathematics that it does not even need to be mentioned in the recurrence principle). We conclude that the fixpoint definition of the naturals $\mathbb{N}$ as “the smallest set such that $0 \in \mathbb{N}$ and $\forall n \in \mathbb{N} . \mathcal{S}n \in \mathbb{N}$ ” is equivalent to (and much simpler than) the mathematical definition requiring “ $0 \in \mathbb{N}$, $\forall n \in \mathbb{N} . \mathcal{S}n \in \mathbb{N}$, and the recurrence principle is valid in $\mathbb{N}$ ”.

Notice that there are different possible universes but the defined naturals in each case are isomorphic, this is Dedekind's categoricity theorem. Examples are Zermelo naturals $0 = \{\}$ and $n+1 = \{n\}$ in his axiom of infinity [50, Axiom VII] or Von Neumann naturals $0 = \{\}$ and $n+1 = n \cup \{n\}$ as part of the ordinals [46].

6 Nonstandard Traces of an Imperative Language

Let $\mathbb{N}^{+\delta}$ be the standard naturals $\mathbb{N}$ indexed by 0 followed by $\delta \in \mathbb{N}$ copies of $\mathbb{Z}$ indexed by $i \in [1, \delta]$ totally ordered by the lexicographic ordering on the index and the integers. Informally $\mathbb{N}^{+\delta} = 0_0 \ 1_0 \ 2_0 \ \dots \dots -2_1 \ -1_1 \ 0_1 \ 1_1 \ 2_1 \ \dots \ \dots \dots -2_\delta \ -1_\delta \ 0_\delta \ 1_\delta \ 2_\delta \ \dots$. So $\mathbb{N}^{+0} = \{n_0 \mid n \in \mathbb{N}\}$ is isomorphic to $\mathbb{N}$ while $\mathbb{N}^{+1} = \{n_0 \mid n \in \mathbb{N}\} \cup \{z_1 \mid z \in \mathbb{Z}\}$ is isomorphic to the Skolem model S in section 5.

Assuming that states $\langle \ell, \rho \rangle$ are abstracted to $\rho(\mathbf{x})$ and others ignored, the $\mathbb{N}^{+\delta}$, $\delta \in \mathbb{N}$ are trace models of the program **while true do $\mathbf{x}:=\mathbf{x}+1$** ; where $\mathbf{x} \in \mathbb{N}^{+\delta}$ that starts and never terminates. Another example suggested by a reviewer is “**y:=0; while (y==0) do $\mathbf{x}:=\mathbf{x}+1$** ;; and then have a trace with Skolem's nonstandard natural numbers as domain. After each of the standard

points $0, 1, \dots, n$, we will have $y = 0$; but on the nonstandard points $-3, -2, -1, 0', 1', 2', \dots$ one could choose y to be an arbitrary constant > 0 ."

Define $\mathbb{N}^{-\delta} \triangleq \{-m_{-i} \mid m_i \in \mathbb{N}^{+\delta}\}$. Informally $\mathbb{N}^{-\delta} = \dots -2_{-\delta} -1_{-\delta} 0_{-\delta} 1_{-\delta} 2_{-\delta} \dots \dots \dots -2_{-1} -1_{-1} 0_{-1} 1_{-1} 2_{-1} \dots \dots -2_0 -1_0 0_0$. The $\mathbb{N}^{-\delta}$, $\delta \in \mathbb{N}$ are trace models of program **while** $x \neq 0$ **do** $x = x + 1$; where $x \in \mathbb{N}^{-\delta}$. In addition define $\mathbb{N}^{-\delta+\delta'} \triangleq \mathbb{N}^{+\delta} \cup \mathbb{N}^{-\delta'}$ and $\mathbb{N}^{\pm\delta} \triangleq \mathbb{N}^{-\delta+\delta}$. For example $\mathbb{N}^{\pm 1} = \{\dots -2_{-1} -1_{-1} 0_{-1} 1_{-1} 2_{-1} \dots -2_0 -1_0 0_0 1_0 2_0 \dots -2_1 -1_1 0_1 1_1 2_1 \dots\}$.

We define nonstandard infinite traces as $\Sigma^\infty \triangleq \Sigma^* \cup (\mathbb{N}^{-\delta} \rightarrow \Sigma) \cup (\mathbb{N}^{+\delta} \rightarrow \Sigma) \cup (\mathbb{N}^{\pm\delta} \rightarrow \Sigma)$ as finite traces in Σ^* plus nonstandard backward traces in $\mathbb{N}^{-\delta} \rightarrow \Sigma$ that never start but terminate, nonstandard forward traces in $\mathbb{N}^{+\delta} \rightarrow \Sigma$ that start but never terminate, and nonstandard backward/forward traces in $\mathbb{N}^{\pm\delta} \rightarrow \Sigma$ that neither start nor terminate in the transfinite.

The parts of a trace indexed by $\mathbb{N}^{+0}$ or $\mathbb{N}^{-0}$ are called the standard parts of the trace and those indexed by $\mathbb{N}^{-\delta}$ or $\mathbb{N}^{+\delta}$, $\delta > 0$ are called the nonstandard parts.

7 Nonstandard Trace Semantics of an Imperative Language

The nonstandard trace semantics of a simple imperative language is an element of $\wp(\Sigma^\infty)$. It can be given fixpoint definitions. Consider for example the traces that terminate but never start for the iteration statement $S = \text{while } \ell \text{ do } \ell' \text{ Sb } \ell''$. In the following formulas, we assume that ℓ, ℓ' , and ℓ'' are the labels of S (i.e. not variables local to the formula). Let $\text{Seed}^{-\delta} \llbracket S \rrbracket X$ be the subset of nonstandard backward traces of X such that the standard and nonstandard parts all have at least one state of the form $\langle \ell, \rho \rangle$ such that $\mathcal{B} \llbracket B \rrbracket \rho$ holds. Define

$$B^{-\delta} \llbracket S \rrbracket X \triangleq \{\sigma_1 \langle \ell, \rho \rangle \langle \ell', \rho \rangle \sigma_2 \langle \ell, \rho' \rangle \langle \ell'', \rho' \rangle \in \text{Seed}^{-\delta} \llbracket S \rrbracket X \mid \quad (5.a)$$

$$\begin{aligned} & \mathcal{B} \llbracket B \rrbracket \rho \wedge \langle \ell', \rho \rangle \sigma_2 \langle \ell, \rho' \rangle \in \llbracket \text{Sb} \rrbracket^* \wedge \neg \mathcal{B} \llbracket B \rrbracket \rho' \} \cup \\ & \{\sigma_1 \langle \ell, \rho' \rangle \langle \ell', \rho' \rangle \sigma'_1 \langle \ell, \rho \rangle \langle \ell', \rho \rangle \sigma'_2 \langle \ell, \rho'' \rangle \sigma_2 \in \text{Seed}^{-\delta} \llbracket S \rrbracket X \mid \quad (5.b) \\ & \mathcal{B} \llbracket B \rrbracket \rho' \wedge \langle \ell', \rho' \rangle \sigma'_1 \langle \ell, \rho \rangle \in \llbracket \text{Sb} \rrbracket^* \wedge \mathcal{B} \llbracket B \rrbracket \rho \wedge \\ & \langle \ell', \rho \rangle \sigma'_2 \langle \ell, \rho'' \rangle \in \llbracket \text{Sb} \rrbracket^* \wedge \mathcal{B} \llbracket B \rrbracket \rho'' \} \end{aligned}$$

$$\llbracket S \rrbracket^{-\delta} \triangleq \text{gfp} \subseteq B^{-\delta} \llbracket S \rrbracket$$

The **gfp** is the intersection of the transfinite fixpoint iterations starting from all possible nonstandard backward traces $\mathbb{N}^{-\delta} \rightarrow \Sigma$. If some part of a trace has no seed state $\langle \ell, \rho \rangle$ with $\mathcal{B} \llbracket B \rrbracket \rho$ true, the trace is eliminated from the next fixpoint iterate so, at the limit by intersection of the fixpoint iterates, from the **gfp**. So after one fixpoint iteration all parts of all nonstandard backward traces have a seed state $\langle \ell, \rho \rangle$ where B holds. Then in the next fixpoint iteration, at least one seed state $\langle \ell, \rho \rangle$ in the trace is required

- (5.a) either to be the last in the standard part before termination, in which case it is eliminated if it is not preceded by an execution of the loop body **Sb** from ℓ to ℓ with B finally false;

(5.b) or, if not the last in the standard part to be preceded and followed by executions of the body **Sb** from ℓ to ℓ with **B** true.

It follows that each iteration ensures that the number of iterations around the seed is strictly increased. At the limit, the **gfp** which is the intersection of these transfinite fixpoint iterations must have all its standard and nonstandard parts built out of traces of a true test **B** followed by an execution of the loop body (5.b), except for the standard part which must end with **B** false (5.a). The difference with the standard semantics is that $B^* \llbracket \mathbf{S} \rrbracket$ in (1) requires successive loop iterations backward starting from the seed at the end of the trace while $B^{-\delta} \llbracket \mathbf{S} \rrbracket$ requires loop iterations before and after the seed that can be anywhere. For the standard part this will ultimately reach the end thanks to (5.b) at which time termination is required by (5.a).

The nonstandard forward trace semantics is defined similarly as $\llbracket \mathbf{S} \rrbracket^{+\delta} \triangleq \mathbf{gfp} \subseteq F^{+\delta} \llbracket \mathbf{S} \rrbracket$ where $F^{+\delta} \llbracket \mathbf{S} \rrbracket$ generalizes $F^\infty \llbracket \mathbf{S} \rrbracket$ to nonstandard traces.

$$F^{+\delta} \llbracket \mathbf{S} \rrbracket X \triangleq \{ \sigma_1 \langle \ell, \rho \rangle \sigma_2 \in \text{Seed}^{+\delta} \llbracket \mathbf{S} \rrbracket X \mid \mathcal{B} \llbracket \mathbf{B} \rrbracket \rho \wedge$$

$$((\sigma_1 = \epsilon) \vee (\exists \sigma'_1 \langle \ell, \rho' \rangle \langle \ell', \rho' \rangle \sigma''_1 = \sigma_1 \cdot \mathcal{B} \llbracket \mathbf{B} \rrbracket \rho' \wedge \langle \ell', \rho' \rangle \sigma''_1 \langle \ell, \rho \rangle \in \llbracket \mathbf{Sb} \rrbracket^*)) (6.a)$$

$$\wedge ((\exists \langle \ell', \rho \rangle \sigma'_2 \langle \ell, \rho'' \rangle \sigma''_2 = \sigma_2 \cdot \langle \ell', \rho \rangle \sigma'_2 \langle \ell, \rho'' \rangle \in \llbracket \mathbf{Sb} \rrbracket^* \wedge \mathcal{B} \llbracket \mathbf{B} \rrbracket \rho'') \vee (6.b)$$

$$(\exists \langle \ell', \rho \rangle \sigma_b = \sigma_2 \cdot \mathcal{B} \llbracket \mathbf{B} \rrbracket \rho \wedge \langle \ell', \rho \rangle \sigma_b \in \llbracket \mathbf{Sb} \rrbracket^\infty)) \} (6.c)$$

Let X be an iterate of $F^{+\delta} \llbracket \mathbf{S} \rrbracket$ (initially all nonstandard forward traces $\mathbb{N}^{+\delta} \rightarrow \Sigma$). In the next fixpoint iterate $F^{+\delta} \llbracket \mathbf{S} \rrbracket X$ appear only the traces $\sigma_1 \langle \ell, \rho \rangle \sigma_2$ which have at least one seed state in all their parts and at least one of these seeds $\langle \ell, \rho \rangle$ is, by (6.a) either the first state in the trace or else preceded by a terminating execution of the loop body **Sb** starting with **B** true and, by (6.b), followed by a terminating execution of the loop body **Sb** ending with **B** true, or by (6.c), followed by a nonterminating execution of the loop body **Sb**. (6.a) and (6.b) guarantee the existence of other seeds which at the next fixpoint iterate will be required to satisfy the same constraints. Passing to the limit by an intersection, after possibly transfinite fixpoint iterations, only infinite nonstandard loop traces will remain. In particular, it follows that $\llbracket \mathbf{S} \rrbracket^{+0} \triangleq \mathbf{gfp} \subseteq F^{+0} \llbracket \mathbf{S} \rrbracket = \mathbf{gfp} \subseteq F^\infty \llbracket \mathbf{S} \rrbracket \triangleq \llbracket \mathbf{S} \rrbracket^\infty$ in the standard case.

We also define

$$\llbracket \mathbf{S} \rrbracket^{-\delta+\delta'} \triangleq \{ \sigma \langle \ell, \rho \rangle \sigma' \mid \sigma \langle \ell, \rho \rangle \in \mathbf{gfp} \subseteq B^{-\delta} \llbracket \mathbf{S} \rrbracket \wedge \langle \ell, \rho \rangle \sigma' \in \mathbf{gfp} \subseteq F^{+\delta'} \llbracket \mathbf{S} \rrbracket \}$$

which has both backward and forward infinite nonstandard traces.

The traditional trace semantics is $\llbracket \mathbf{S} \rrbracket^{*\infty} \triangleq \llbracket \mathbf{S} \rrbracket^* \cup \llbracket \mathbf{S} \rrbracket^\infty = \llbracket \mathbf{S} \rrbracket^* \cup \llbracket \mathbf{S} \rrbracket^{+0}$ where $\llbracket \mathbf{S} \rrbracket^*$ is the classic finite semantics and $\llbracket \mathbf{S} \rrbracket^\infty = \llbracket \mathbf{S} \rrbracket^{+0}$ the standard infinite traces.

There are several possible choices for the nonstandard semantics, for example, $\llbracket \mathbf{S} \rrbracket^* \cup \llbracket \mathbf{S} \rrbracket^{-\delta}$ ignoring nontermination as in natural semantics [33] (but keeping the traces that terminate but never start), $\llbracket \mathbf{S} \rrbracket^* \cup \llbracket \mathbf{S} \rrbracket^{-\delta} \cup \llbracket \mathbf{S} \rrbracket^{+\delta}$ or $\llbracket \mathbf{S} \rrbracket^\infty \triangleq \llbracket \mathbf{S} \rrbracket^* \cup \llbracket \mathbf{S} \rrbracket^{-\delta} \cup \llbracket \mathbf{S} \rrbracket^{+\delta} \cup \llbracket \mathbf{S} \rrbracket^{-\delta+\delta}$. Then $\llbracket \text{while true do skip} \rrbracket^\infty = \llbracket \text{while true do skip} \rrbracket^{+\delta} \cup \llbracket \text{while true do skip} \rrbracket^{-\delta+\delta}$ since it either starts and never terminates or never starts and never terminates.

8 The Nonstandard Interpretation of the Axiomatic Semantics

Define, for all statements $S \in \mathcal{L}$, the abstraction of a nonstandard trace semantics T into the set of all Hoare triples valid for that semantics (again $\ell = \text{at}(S)$ and $\ell'' = \text{after}(S)$ are global variables in the formula denoting program points).

$$\begin{aligned} \tilde{\alpha}_S(T) \triangleq \{ \{P\}S\{Q\} \mid & P, Q \in \wp(\mathbb{E}) \wedge \\ & ((\forall \langle \ell, \rho \rangle \sigma \langle \ell'', \rho'' \rangle \in T \cap \mathbb{N}^* \rightarrow \Sigma . \rho \in P \Rightarrow \rho'' \in Q) \\ & \vee (\forall \sigma \langle \ell, \rho \rangle \sigma' \langle \ell'', \rho'' \rangle \in T \cap \mathbb{N}^{-\delta} \rightarrow \Sigma . \\ & (\langle \ell, \rho \rangle \sigma' \langle \ell'', \rho'' \rangle \in (\mathbb{N}^{-0} \rightarrow \Sigma) \wedge \rho \in P) \Rightarrow \rho'' \in Q) \} \end{aligned}$$

This is the standard requirement for finite traces. For backward nonstandard traces that never start but terminate, their nonstandard part is ignored. If their standard part has a state $\langle \ell, \rho \rangle$ satisfying P then the final state must satisfy the postcondition Q . Notice that if auxiliary variables record the values of variables at ℓ in P when ℓ is reached at any past loop iteration and these auxiliary variables appear in Q then Q must hold between these values of the auxiliary variables when ℓ was reached and the final values of the program variables at ℓ'' .

The nonstandard Hoare logic theory $\tilde{\mathcal{H}}(\mathcal{L})$ for the language $\mathcal{L}$ is the set

$$\tilde{\mathcal{H}}(\mathcal{L}) \triangleq \bigcup \{ \tilde{\alpha}_S(\llbracket S \rrbracket^\infty) \mid S \in \mathcal{L} \} \quad (7)$$

9 Ambiguity of the Axiomatic Semantics

The standard $\mathcal{H}(\mathcal{L})$ and nonstandard $\tilde{\mathcal{H}}(\mathcal{L})$ theories are the same.

Theorem 1. $\tilde{\mathcal{H}}(\mathcal{L}) = \mathcal{H}(\mathcal{L})$

Proof. Obviously $\mathcal{H}(\mathcal{L}) \subseteq \tilde{\mathcal{H}}(\mathcal{L})$. The converse follows from the fact that if $\sigma \langle \ell, \rho \rangle \sigma' \langle \ell'', \rho'' \rangle \in \llbracket S \rrbracket^\infty \cap \mathbb{N}^{-\delta} \rightarrow \Sigma$ and $\langle \ell, \rho \rangle \sigma' \langle \ell'', \rho'' \rangle \in (\mathbb{N}^{-0} \rightarrow \Sigma)$ then $\langle \ell, \rho \rangle \sigma' \langle \ell'', \rho'' \rangle \in \llbracket S \rrbracket^*$ together with $\rho \in P$ implies $\rho'' \in Q$ and therefore $\{P\}S\{Q\}$ holds by definition of $\alpha_S(\llbracket S \rrbracket^*)$. $\square$

Since the theories are the same by theorem 1, their proof rules are the same by structural induction on the language $\mathcal{L}$ and Aczel's correspondence between theories and proof rules [2] for iteration. It follows that the popular statement that “Axiomatic semantics is a formal approach in computer science that defines a program's meaning by its effects on logical assertions (pre- and post-conditions), rather than how it executes” is wrong since the defined meaning is not unique up to isomorphism, unlike trace semantics do. This is the same problem as for Peano's original definition of the naturals.

Example 1. Consider the program `while x<0 do x:=x+1;` on integers whose standard traces in $\mathbb{N}^{*\infty} \rightarrow \Sigma = \mathbb{N}^* \rightarrow \Sigma \cup \mathbb{N}^{+0} \rightarrow \Sigma$ are isomorphic to $\{-n - n + 1 \dots 0 \mid n \in \mathbb{N}\}$ and $\{n \mid n \in \mathbb{N}\}$ since the program executions starting

with a negative value $-n$ of variable x will iteratively increment it by one until reaching 0 and then exiting the loop or else the program executions will start with a value n of x which is greater than or equal to 0, in which case the loop is not entered and immediately exited.

The method to show that Hoare logic includes any of these traces is for each one to define P to be the set of states along that trace and to prove that P is a loop inductive invariant according to Hoare's rule of iteration.

For example, consider the trace $-n \ -n+1 \ \dots \ 0$ for a given $n \in \mathbb{N}$ where $\rho \in \Sigma$ such that $\rho(x) = k$ is encoded simply by k . Let $P = \{\rho \mid \rho(x) \in [-n, 0]\}$ be the set of states along that trace. We have $P \cap \{\rho \mid \rho(x) < 0\} = \{\rho \in P \mid \rho(x) < 0\} = \{\rho \mid \rho(x) \in [-n, -1]\}$ and $\{\{\rho \mid \rho(x) \in [-n, -1]\}x:=x+1; \{\rho \mid \rho(x) \in [-n+1, 0]\}\}$ by the assignment rule. Moreover $\{\rho \mid \rho(x) \in [-n+1, 0]\} \subseteq P$ so that $\{P \cap \{\rho \mid \rho(x) < 0\}x:=x+1; \{P\}$ holds by the consequence rule. By the iteration rule, $\{P\} \text{ while } x < 0 \text{ do } x:=x+1; \{\neg(x < 0) \cap P\}$ proving that the trace $-n \ -n+1 \ \dots \ 0$ is feasible for that program. Similarly for any $n \in \mathbb{N}$ we can prove $\{\{n\}\} \text{ while } x < 0 \text{ do } x:=x+1; \{\{n\}\}$ (considering only the value n of x), proving the feasibility of the traces $\{n \mid n \in \mathbb{N}\}$.

Now consider the traces in $\mathbb{N}^{-0} \rightarrow \Sigma$ so that the program semantics now include the trace $\dots \ -2 \ -1 \ 0$ that never starts and stops at 0. The invariant is now $P = \{\rho \mid \rho(x) \in \mathbb{Z} \wedge \rho(x) \leq 0\}$ which also satisfies Hoare's verification conditions for iteration so that this trace is also proved to be feasible.

It follows that program **while** $x < 0$ **do** $x:=x+1$; has the standard semantics $\{-n \ -n+1 \ \dots \ 0 \mid n \in \mathbb{N}\} \cup \{n \mid n \in \mathbb{N}\}$ in $\mathbb{N}^{*\infty} \rightarrow \Sigma$ and the nonstandard semantics $\{\dots \ -2 \ -1 \ 0\} \cup \{-n \ -n+1 \ \dots \ 0 \mid n \in \mathbb{N}\} \cup \{n \mid n \in \mathbb{N}\}$ in $\mathbb{N}^{-0} \rightarrow \Sigma$. This shows that for some programs, Hoare logic cannot distinguish between the standard trace semantics on $\mathbb{N}^{*\infty} \rightarrow \Sigma$ and the nonstandard one on $\mathbb{N}^* \rightarrow \Sigma \cup \mathbb{N}^{-0} \rightarrow \Sigma$.

For the converse question, fix the semantics to be either on the standard $\Sigma^{*\infty}$ or on the nonstandard $\Sigma^* \cup \Sigma^{-0}$. The question is whether there is a triple $\{P\} S \{Q\}$ allowing us to know which semantics is considered. This cannot be on finite executions and must be on infinite or transfinite executions, the only ones making a difference. This difference can only appear in the loop invariant in $\wp(\mathbb{E})$ for infinite trace. If this loop invariant is expressed in first-order logic then, by the compactness theorem of first-order logic, it is true for Σ isomorphic to $\mathbb{N}$ if and only if it is true for Σ isomorphic to nonstandard naturals so no distinction is possible in this case. More generally, if the invariant holds for a transfinite trace, it must hold for its standard part, hence for the standard semantics (in which case it is an over-approximation, possibly referring to nonstandard naturals, up to an isomorphism with Σ).

Example 2. Consider now the program **while** **true** **do** $x:=x+1$; which traces in $\Sigma^\infty = \Sigma^{+0}$ are $\{n \ n+1 \ n+2 \ \dots \mid n \in \mathbb{Z}\}$ and traces on $\Sigma^{-0} \cup \Sigma^{+0}$ also contain the trace $\dots \ -2 \ -1 \ 0 \ 1 \ 2 \ \dots$. Again for all of these traces the set of their states is a valid invariant in Hoare logic so that Hoare logic cannot distinguish between the standard semantics on $\Sigma^{*\infty}$ and the nonstandard semantics on Σ^{*-0+0} .

10 A Reminder on Termination Proof Methods

Turing/Floyd termination proof method [44,27] consists in exhibiting a variant function mapping program variables into a well-founded set and showing that its value strictly decreases on each loop iteration (or after a finite number of iterations).

Knuth's termination proof method [35, Section 3a] consists in adding an integer counter to each loop which is initialized, e.g. to 0, before entering the loop and incremented by a strictly positive value, e.g. by 1, within the loop body (the counter is assumed to be a mathematical integer so the incrementation cannot overflow). If, thanks to the invariants statically relating the counter to the other variables in the loop, the counter value can be proved to be upper-bounded by $M \in \mathbb{N}$ then termination in finitely many steps is proved. The method is sound but incomplete, e.g. for unbounded non-determinism requiring transfinite ordinals [23].

This method was used in [30, section 8.3] to prove termination of programs by linear inequalities/polyhedral static analysis [22].

In the same way as Manna and Pnueli extended Hoare logic to total correctness, i.e. partial correctness with termination [37,4], it is easy to extend Hoare logic with Knuth's termination proof method [34, section 2] (t , α , and M are auxiliary variables in $\mathbb{N}$ that do not appear in P , B , and Sb).

$$\frac{P \wedge B \Rightarrow t < M, \quad \{P \wedge B \wedge t = \alpha\} Sb \{P \wedge t = \alpha + 1\}}{\{P \wedge t = 0\} \text{ while } B \text{ do } Sb \{P \wedge \neg B\}} \quad (8)$$

11 Standardisation of the Axiomatic Semantics

We now extend the axiomatic semantics so as to exclude nonstandard interpretations and show in section 12 that this does not change Hoare's proof method under the hypothesis that the logic is interpreted in the standard trace semantics of section 2. In fact this hypothesis is implicit in most uses of Hoare logic.

For naturals, considering the least model eliminates nonstandard models. A similar idea would be to consider the strongest loop invariant. But this does not help, e.g. `x:=0; while true do skip;` has strongest invariant $x = 0$ which does not exclude forward nonstandard traces. Anyway, Hoare logic cannot check that a given invariant is the strongest possible one.

11.1 Preventing Nonstandard Infinite Forward Traces

To prevent the nonstandard forward traces of section 6, we extend Knuth's method by taking the counter to be an ordinal which is initialized to 0 before the loop, and the incrementation within the loop is the ordinal addition of 1 to the least upper bound of all previous values of the counter. It follows that for a standard infinite forward iteration the counter is bounded by ω where ω is the smallest infinite limit ordinal. However, for a nonstandard infinite forward iteration the counter should be incremented by one at each step in the nonstandard

part which is impossible since there is no order isomorphism between integers and ordinals. So adding this further proof obligation in Hoare's logic for loops will guarantee that the counter for that loop is upper bounded by ω which will definitely exclude nonstandard forward infinite traces.

Knuth's auxiliary counter is equivalent to the existence of a variant function K recording the value of the counter over time. Let $\mathbb{O}$ be the class of Von Neumann ordinals [46], $\oplus$ be ordinal addition, and ω be the first infinite limit ordinal. The following requirement limits loop iterations to the standard first ω ones.

$$\begin{aligned} \exists K \in \mathbb{N}^{+\delta} \rightarrow \mathbb{O} \text{ increasing} \quad . \forall n \in \mathbb{N}^{+\delta} . P \wedge B \Rightarrow K(n) < \omega \wedge \\ \forall \alpha \in \mathbb{O} . \{P \wedge B \wedge K(n) = \alpha\} \text{ Sb } \{P \wedge K(n+1) = \alpha \oplus 1\} \end{aligned} \quad (9)$$

In (8), t is the number of iterations so far in the loop. It is replaced in (9) by $K(n)$ which maps the position $n \in \mathbb{N}^{+\delta}$ in the trace to the number of iterations so far in the loop. So n should be initialized to zero in $\{P \wedge n = 0\}$ while B do Sb $\{P \wedge \neg B\}$ and incremented after each test or assignment e.g. $\{P\} x := f \{ \rho[x \leftarrow \llbracket f \rrbracket(\rho), n \leftarrow \rho(n) + 1] \mid \rho \in P \}$ where the incrementation of $\rho(n)$ on $\mathbb{N}^{+\delta}$ is nonstandard.

K must be increasing (monotone) and there exists no order-isomorphism between nonstandard integer parts and ordinals since ordinals are well-founded and integers admit infinite strictly decreasing chains. So for $\delta > 0$, K does not exist for nonstandard parts and the requirement can only be satisfied for $\delta = 0$ that is $K \in \mathbb{N}^{+0} \rightarrow \omega + 1$. For the classic traces of section 2 where $\delta = 0$ and $\mathbb{N}^{+0} = \mathbb{N}$, one can choose $K(n) = n$ (which is t in (8)) so that $\bigcup_{n \in \mathbb{N}^{+0}} K(n) \leq \omega$.

11.2 Preventing Nonstandard Infinite Backward or Backward/Forward traces

The idea is that traversing standard traces backward should terminate, which is the inverse of the requirement (9) that forward traversal of forward traces should terminate when program termination is required (that is $\bigcup_{n \in \mathbb{N}^{+0}} K(n) < \omega$). It follows that we can exclude transfinite nonstandard backward traces by an additional proof obligation applying the inverse of Knuth's proof method.

$$\begin{aligned} \exists K \in \mathbb{N}^{-\delta} \rightarrow \mathbb{O} \text{ decreasing (antitone)} \quad . \forall n \in \mathbb{N}^{-\delta} . \forall \alpha \in \mathbb{O} \quad (10) \\ \{P \wedge B \wedge K(n) = \alpha \oplus 1\} \text{ Sb } \{P \wedge K(n+1) = \alpha\} \wedge \\ (P \wedge K(n) = 0) \Rightarrow \neg B \wedge \bigcup_{n \in \mathbb{N}^{-0}} K(n) < \omega \end{aligned}$$

For example, for a backward nonstandard trace with domain $\mathbb{N}^{-1}$, we would have

$$\begin{array}{cccccccccccc} B & B & B & B & B & & B & B & \neg B & & \\ \dots & -2_{-1} & -1_{-1} & 0_{-1} & 1_{-1} & 2_{-1} & \dots & \dots & -2_0 & -1_0 & 0_0 & \leftarrow n \in \mathbb{N}^{-1} \\ & & & & & & & \dots & 2 & 1 & 0 & \leftarrow K(n) \in \mathbb{O} \end{array}$$

$\underbrace{\hspace{15em}}_{K \text{ not definable for } \delta > 0}$
 $\underbrace{\hspace{10em}}_{K \text{ not bounded}}$

so that (10) does not hold. Requirement (10) will also exclude nonstandard infinite backward/forward traces. For the standard trace semantics there is no infinite backward trace so the requirement (10) only applies to finite traces, in which case, for deterministic programs, $K(n)$ is the number of remaining steps after n previous execution steps. For unbounded nondeterminism, the bound ω must be replaced by a larger ordinal depending on the cardinality of Σ [13, section 10, p. 292]

12 Standard Hoare Logics

We have enriched the axiomatic semantics with two additional proof obligations eliminating nonstandard infinite traces. Should these additional requirements be included in proofs by Hoare logic? The answer is no provided Hoare logic is explicitly specified to be interpreted in the standard trace semantics. In that case the additional verification conditions of section 11 are always satisfied. This hypothesis is usually not specified in the definition of axiomatic semantics by Hoare logic, and so almost always remains completely implicit. This is the case when Hoare logic and its variants are designed by abstraction of the standard trace semantics [17,45].

But there are exceptions. For example, in Event-B the traces are explicitly specified to be the standard ones [1, Ch. 14.3] and machines are specified by defining the states and invariants satisfying proof obligations which are essentially Hoare proof rules [19]. So specifications in Event-B, although based on an axiomatic semantics, cannot have nonstandard interpretations.

13 Static Analysis by Abstraction of the Axiomatic Semantics

The axiomatic semantics has been used to prove the soundness of static analyses (e.g. [3]) or typing algorithms (e.g. [38,7,5]). However, it is not very well adapted for static analyzers based on program transformations for which Hoare logic extensions must be used [28,6]. For example Astrée's loop unrolling [8] is better explained by Burstall's proof method [10,14] or by cyclic proofs in Hoare logic [9]. Of course all these logics have nonstandard interpretations that can be excluded by the standardisation methods proposed in section 11.

14 Abstract Hoare logics

Abstract Hoare logics have been introduced in [21] with the idea that state properties in $\wp(\mathbb{E})$ in Hoare triples, can be replaced by abstract properties chosen in an abstract domain $\langle L, \sqsubseteq \rangle$ whose meaning is given by a Galois connection $\langle \wp(\mathbb{E}), \sqsubseteq \rangle \xleftrightarrow[\alpha]{\gamma} \langle L, \sqsubseteq \rangle$ [16, Ch. 11]. If γ is increasing but not join-preserving then the abstract Hoare rules may be sound while the intersection rule is incorrect [21, p. 219]. Even though the concrete Hoare logic may be standard, the abstraction

may introduce nonstandard executions. The standardisation methods of section 11 may not be applicable when the abstract domain $\langle L, \sqsubseteq \rangle$ is not expressive enough.

15 Related work

Another nonstandard transfinite trace semantics in $\wp(\mathbb{O} \rightarrow \Sigma)$ where $\mathbb{O}$ is the class of ordinals suggested in [15, footnote 17, page 53] and fully developed in [29] yields another nonstandard semantic interpretation of axiomatic semantics (it is different from the nonstandard semantics of section 6 since $\mathbb{O}$ is not a Skolem model of Peano’s naturals in which any $n \neq 0$ must have a predecessor, which is not the case for limit ordinals). Such transfinite forward traces can also be eliminated by Knuth’s ordinal method of section 11.1.

16 Conclusion and Future Work

Hoare postulated the logic named after him and relied implicitly on a model of computation which defines possible computations of programs. When later proved sound and complete, the same implicit model of computation was used [11,12]. The design of logic by abstract interpretation of a trace semantics [18,17] makes the same hypothesis. So nonstandard computation models were never considered.

A solution to avoid nonstandard models of computation is to make explicit the assumption that the axiomatic semantics should be interpreted by standard computation models only, as explicitly specified in Event-B. Another solution is to add additional requirements (9) and (10) to restrict the axiomatic semantics to the standard computation model. Remarkably these requirements (9) and (10) need not be checked in proofs by Hoare logic since they are always satisfied in the standard model.

We anticipate that reverse Hoare logics [47,39] can also be given nonstandard interpretations and restricted to the standard interpretation using Knuth’s ordinal methods (9) and (10).

Acknowledgements. I thank the organizers of Helmut Seidel Fest (TU München, Garching) where I first presented this work on July 16, 2026 and the reviewers for their suggestions. I used Claude (Opus 5) to check for misprints and grammatical errors and search for the bibliography’s DOIs.

References

1. Abrial, J.: Modeling in Event-B: System and Software Engineering. Cambridge Univ. Press (2010)
2. Aczel, P.: An introduction to inductive definitions. In: Barwise, J. (ed.) Handbook of Mathematical Logic, chap. 7, pp. 739–782. North–Holland, Amsterdam (1977)

3. Appel, A.W.: VeriSmall: Verified Smallfoot shape analysis. In: Jouannaud, J., Shao, Z. (eds.) *Certified Programs and Proofs - First Int. Conference, CPP 2011, Kenting, Taiwan, December 7-9, 2011. Proceedings. LNCS*, vol. 7086, pp. 231–246. Springer (2011). https://doi.org/10.1007/978-3-642-25379-9_18
4. Apt, K.R., Plotkin, G.D.: Countable nondeterminism and random assignment. *J. ACM* **33**(4), 724–767 (1986). <https://doi.org/10.1145/6490.6494>
5. Barthe, G., D’Argenio, P.R., Rezk, T.: Secure information flow by self-composition. In: 17th IEEE Computer Security Foundations Workshop, (CSFW-17 2004), 28-30 June 2004, Pacific Grove, CA, USA. pp. 100–114 (2004). <https://doi.org/10.1109/CSFW.2004.17>
6. Benton, N.: Simple relational correctness proofs for static analyses and program transformations. In: Jones, N.D., Leroy, X. (eds.) *Proc. 31st ACM Symp. Principles of Programming Languages, POPL 2004, Venice, Italy, January 14-16, 2004*. pp. 14–25 (2004). <https://doi.org/10.1145/964001.964003>
7. Beringer, L., Hofmann, M.: Secure information flow and program logics. In: 20th IEEE Computer Security Foundations Symposium, CSF 2007, 6-8 July 2007, Venice, Italy. pp. 233–248 (2007). <https://doi.org/10.1109/CSF.2007.30>
8. Blanchet, B., Cousot, P., Cousot, R., Feret, J., Mauborgne, L., Miné, A., Monniaux, D., Rival, X.: A static analyzer for large safety-critical software. In: Cytron, R., Gupta, R. (eds.) *Proc. ACM SIGPLAN 2003 Conf. on Programming Language Design and Implementation 2003*, San Diego, California, USA, June 9-11, 2003. pp. 196–207 (2003). <https://doi.org/10.1145/781131.781153>
9. Brotherston, J., Le, Q.L., Desai, G., Oda, Y.: Cyclic proofs in Hoare logic and its reverse. In: Kupke, C., Milius, S. (eds.) *Proc. 41st Conf. on the Mathematical Foundations of Programming Semantics, MFPS XLI, Univ. of Strathclyde, Glasgow, UK, June 16-21, 2025. Electronic Notes in Theoretical Informatics and Computer Science*, vol. 5. EpiSciences (Apr 2025). <https://doi.org/10.46298/ENTICS.16696>
10. Burstall, R.M.: Program proving as hand simulation with a little induction. In: Rosenfeld, J.L. (ed.) *Information Processing, Proc. 6th IFIP Congress 1974*, Stockholm, Sweden, August 5-10, 1974. pp. 308–312. North-Holland (1974)
11. Cook, S.A.: Soundness and completeness of an axiom system for program verification. *SIAM J. Comput.* **7**(1), 70–90 (1978). <https://doi.org/10.1137/0207005>
12. Cook, S.A.: Corrigendum: Soundness and completeness of an axiom system for program verification. *SIAM J. Comput.* **10**(3), 612 (1981). <https://doi.org/10.1137/0210045>
13. Cousot, P., Cousot, R.: ‘à la Floyd’ induction principles for proving inevitability properties of programs. In: Nivat, M., Reynolds, J. (eds.) *Algebraic methods in semantics*. pp. 277–312. Cambridge Univ. Press (Dec 1985)
14. Cousot, P.: A Hoare-style axiomatization of Burstall’s intermittent assertion method for non-deterministic programs. Tech. rep., Univ. Paul Verlaine, Metz, France (September 1983), <https://cs.nyu.edu/~pmc309/publications.www/RR-LRIM-83-04-sep-1983.PDF>
15. Cousot, P.: Constructive design of a hierarchy of semantics of a transition system by abstract interpretation. *Theor. Comput. Sci.* **277**(1-2), 47–103 (2002). [https://doi.org/10.1016/S0304-3975\(00\)00313-3](https://doi.org/10.1016/S0304-3975(00)00313-3)
16. Cousot, P.: *Principles of Abstract Interpretation*. MIT Press (2021)
17. Cousot, P.: Calculational design of [in]correctness transformational program logics by abstract interpretation. *Proc. ACM Program. Lang.* **8**(POPL), 175–208 (2024). <https://doi.org/10.1145/3632849>

18. Cousot, P.: On the design of program logics. In: Dietsch, D., Rybalchenko, A., Schäfer, M., Wies, T. (eds.) *On the Pursuit of Insight and Elegance - Essays Dedicated to Andreas Podelski on the Occasion of His 65th Birthday*. pp. 92–106. LNCS, Springer (2026). https://doi.org/10.1007/978-3-032-13711-1_6
19. Cousot, P.: The Essence of Event-B from an Abstract Interpretation Perspective. *Abrial Gedenkschrift*, Springer (2026, to appear))
20. Cousot, P., Cousot, R.: Inductive definitions, semantics and abstract interpretation. In: Sethi, R. (ed.) *Conf. Rec. 19th ACM Symp. Principles of Programming Languages*, Albuquerque, New Mexico, USA, January 19–22, 1992. pp. 83–94 (1992). <https://doi.org/10.1145/143165.143184>
21. Cousot, P., Cousot, R., Logozzo, F., Barnett, M.: An abstract interpretation framework for refactoring with application to extract methods with contracts. In: Leavens, G.T., Dwyer, M.B. (eds.) *Proc. 27th ACM Conf. on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA 2012, part of SPLASH 2012*, Tucson, AZ, USA, October 21–25, 2012. pp. 213–232 (2012). <https://doi.org/10.1145/2384616.2384633>
22. Cousot, P., Halbwachs, N.: Automatic discovery of linear restraints among variables of a program. In: Aho, A.V., Zilles, S.N., Szymanski, T.G. (eds.) *Conf. Rec. Fifth Annual ACM Symp. Principles of Programming Languages*, Tucson, Arizona, USA, January 1978. pp. 84–96 (1978). <https://doi.org/10.1145/512760.512770>, <https://doi.org/10.1145/512760.512770>
23. Cousot, R.: *Fondements des méthodes de preuve d’invariance et de fatalité de programmes parallèles* (in French). Thèse d’État ès sciences mathématiques, Nancy, Institut national polytechnique de Lorraine (15 November 1985), <https://pcousot.github.io/publications/RadhiaCousotTheseEsSciences.PDF>
24. De Morgan, A.: Mathematical induction. *The Penny Cyclopaedia of the Society for the Diffusion of Useful Knowledge* **12** (1838)
25. Dedekind, R.: *Was sind und was sollen die Zahlen?* Friedr. Vieweg & Sohn, Braunschweig, 4th edn. (1888)
26. de Fermat, P.: *Relation des nouvelles découvertes en la science des nombres*. letter to Carcavi, in *Œuvres de Fermat*, ed. P. Tannery and C. Henry, vol. II, Paris: Gauthier-Villars, 1894, pp. 431–436. (Aug 1659)
27. Floyd, R.W.: Assigning meaning to programs. In: Schwartz, J. (ed.) *Proc. Symp. in Applied Math.* vol. 19, pp. 19–32. Amer. Math. Soc. (1967). https://doi.org/10.1007/978-94-011-1793-7_4
28. Gerhart, S.L.: Correctness-preserving program transformations. In: Graham, R.M., Harrison, M.A., Reynolds, J.C. (eds.) *Conf. Rec. Second ACM Symp. Principles of Programming Languages*, Palo Alto, California, USA, January 1975. pp. 54–66 (1975). <https://doi.org/10.1145/512976.512983>
29. Giacobazzi, R., Mastroeni, I.: Non-standard semantics for program slicing. *High. Order Symb. Comput.* **16**(4), 297–339 (2003). <https://doi.org/10.1023/A:1025872819613>
30. Halbwachs, N.: *Détermination automatique de relations linéaires vérifiées par les variables d’un programme* (in French). Thèse de 3^{ème} cycle, Université Scientifique et Médicale de Grenoble (12 March 1979), <https://theses.hal.science/tel-00288805>
31. Hoare, C.A.R.: An axiomatic basis for computer programming. *Commun. ACM* **12**(10), 576–580 (1969). <https://doi.org/10.1145/363235.363259>
32. Hoare, C.A.R., Lauer, P.E.: Consistent and complementary formal theories of the semantics of programming languages. *Acta Informatica* **3**, 135–153 (1974). <https://doi.org/10.1007/BF00264034>

33. Kahn, G.: Natural semantics. In: Brandenburg, F., Vidal-Naquet, G., Wirsing, M. (eds.) STACS 87, 4th Annual Symp. Theoretical Aspects of Computer Science, Passau, Germany, February 19-21, 1987, Proceedings. LNCS, vol. 247, pp. 22–39. Springer (1987). <https://doi.org/10.1007/BFB0039592>
34. Katz, S., Manna, Z.: A closer look at termination. *Acta Informatica* **5**, 333–352 (1975). <https://doi.org/10.1007/BF00264565>
35. Knuth, D.E.: Structured programming with go to statements. *ACM Comput. Surv.* **6**(4), 261–301 (1974). <https://doi.org/10.1145/356635.356640>
36. Lauer, P.E.: Consistent formal theories of the semantics of programming languages. Ph.D. thesis, Queen’s Univ. Belfast, UK (1971)
37. Manna, Z., Pnueli, A.: Axiomatic approach to total correctness of programs. *Acta Informatica* **3**, 243–263 (1974). <https://doi.org/10.1007/BF00288637>
38. Nanevski, A., Morrisett, J.G., Birkedal, L.: Hoare type theory, polymorphism and separation. *J. Funct. Program.* **18**(5-6), 865–911 (2008). <https://doi.org/10.1017/S0956796808006953>
39. O’Hearn, P.W.: Incorrectness logic. *Proc. ACM Program. Lang.* **4**(POPL), 10:1–10:32 (2020). <https://doi.org/10.1145/3371078>
40. Park, D.: Fixpoint induction and proofs of program properties. In: Meltzer, B., Michie, D. (eds.) *Machine Intelligence 5*, pp. 59–78. Edinburgh Univ. Press (1970)
41. Peano, G.: *Arithmetices principia, nova methodo exposita*. Fratres Bocca, Torino (1889)
42. Skolem, T.: Über die Nicht-charakterisierbarkeit der Zahlenreihe mittels endlich oder abzählbar unendlich vieler Aussagen mit ausschließlich Zahlenvariablen. *Fundamenta Mathematicae* **23**, 150–161 (1934). <https://doi.org/10.4064/fm-23-1-150-161>
43. Tarski, A.: A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics* **5**(2), 285–309 (1955). <https://doi.org/10.2140/pjm.1955.5.285>
44. Turing, A.: Checking a large routine. In: Report of a Conf. on High Speed Automatic Calculating Machines. pp. 67–69. Univ. of Cambridge Mathematical Laboratory, Cambridge, England (1949 [1950]), <https://turingarchive.kings.cam.ac.uk/publications-lectures-and-talks-amtb/amt-b-8>
45. Verscht, L., Kaminski, B.L.: A taxonomy of Hoare-like logics: Towards a holistic view using predicate transformers and kleene algebras with top and tests. *Proc. ACM Program. Lang.* **9**(POPL), 1782–1811 (2025). <https://doi.org/10.1145/3704896>
46. Von Neumann, J.: Zur Einführung der transfiniten Zahlen. *Acta Scientiarum Mathematicarum* (Univ. of Szeged) **1**(4), 199–208 (1923)
47. de Vries, E., Koutavas, V.: Reverse Hoare logic. In: SEFM. LNCS, vol. 7041, pp. 155–171. Springer (2011). https://doi.org/10.1007/978-3-642-24690-6_12
48. Wegner, P.: Operational semantics of programming languages. In: Proceedings of ACM Conf. on Proving Assertions About Programs, Las Cruces, New Mexico, USA, January 6-7, 1972. pp. 128–141 (1972). <https://doi.org/10.1145/800235.807081>
49. Wirth, C.: Descente infinie + deduction. *Log. J. IGPL* **12**(1), 1–96 (2004). <https://doi.org/10.1093/JIGPAL/12.1.1>
50. Zermelo, E.: Untersuchungen über die Grundlagen der Mengenlehre I. *Mathematische Annalen* **65**, 261–281 (1908), <http://dx.doi.org/10.1007/BF01449999>