

## A HOARE LOGIC FOR THE COINDUCTIVE TRACE-BASED BIG-STEP SEMANTICS OF WHILE\*

KEIKO NAKATA AND TARMO UUSTALU

Institute of Cybernetics at Tallinn University of Technology, Akadeemia tee 21, 12618 Tallinn,  
Estonia  
*e-mail address:* {keiko,tarmo}@cs.ioc.ee

**ABSTRACT.** In search for a foundational framework for reasoning about observable behavior of programs that may not terminate, we have previously devised a trace-based big-step semantics for While. In this semantics, both traces and evaluation (relating initial states of program runs to traces they produce) are defined coinductively. On terminating runs, this semantics agrees with the standard inductive state-based semantics. Here we present a Hoare logic counterpart of our coinductive trace-based semantics and prove it sound and complete. Our logic subsumes the standard partial-correctness state-based Hoare logic as well as the total-correctness variation: they are embeddable. In the converse direction, projections can be constructed: a derivation of a Hoare triple in our trace-based logic can be translated into a derivation in the state-based logic of a translated, weaker Hoare triple. Since we work with a constructive underlying logic, the range of program properties we can reason about has a fine structure; in particular, we can distinguish between termination and nondivergence, e.g., unbounded classically total search fails to be terminating, but is nonetheless nondivergent. Our metatheory is entirely constructive as well, and we have formalized it in Coq.

### 1. INTRODUCTION

Standard big-step semantics and Hoare logics do not support reasoning about nonterminating runs of programs. Essentially, they ignore them. But of course nonterminating runs are important. Not only need we often program a partial function whose domain of definedness we cannot decide or is undecidable, e.g., an interpreter, but we also have to program functions that are inherently partial. In programming with interactive input/output, for example, diverging runs are often what we really want.

*2012 ACM CCS: [Theory of computation]:* Semantics and Reasoning—Program semantics—Operational semantics; Semantics and Reasoning—Program reasoning; Logic—Hoare logic.

*Key words and phrases:* big-step semantics, Hoare logic, nontermination, coinductive traces, formalization, Coq.

\* This article is a revised and expanded version of the ESOP 2010 conference paper [15].

The authors were supported by the EU FP6 IST integrated project no. 15905 (MOBIUS), the ERDF funded Estonian CoE project EXCS, the Estonian Ministry of Education and Research target-financed themes no. 0322709s06 and 0140007s12, and the Estonian Science Foundation grants no. 6940 and 9398.

In search for a foundational framework for reasoning about possibly nonterminating programs constructively (intuitionistically) and intrigued by attempts in this direction in the literature, we have previously devised a big-step semantics for While based on traces [17]. In this semantics, traces are possibly infinite sequences of states that a program run goes through. They are defined coinductively, as is the evaluation relation, relating initial states of program runs to traces they produce. On terminating runs, this nonstandard semantics agrees with the standard, inductive state-based big-step semantics.

In this paper, we put forward a Hoare logic to match this big-step semantics. In this new trace-based logic, program runs are reasoned about in terms of assertions on states and traces. More precisely, our Hoare triple  $\{U\} s \{P\}$  is given by a statement  $s$ , a state assertion  $U$  (a condition on the initial state of a run of  $s$ ) and a trace assertion  $P$  (a condition on the trace produced by the run). In the presentation we have chosen for this paper, assertions are nothing but predicates expressible in the meta-logic, i.e., we do not confine ourselves to a particular language of state and trace assertions. Nonetheless, we do not want to downplay the question of what makes a good assertion language for traces. We are after a set of connectives that allows for a concise formulation of a sound and complete Hoare logic over state and trace predicates and logical entailment as given by the constructive meta-logic. We adopt a solution that is reminiscent of interval temporal logic [14, 8], with a chop-connective. This gives us a set of connectives that is Spartan in terms of convenience of expression, but suffices for our meta-theoretical study. Our logic is intended foundational framework into which more specialized and more applied logics with more limited assertion languages can be embedded.

Besides being deterministic, the While language is also total as soon as we accept that traces of program runs can be infinite. This allows our logic to conservatively extend both the standard, state-based partial-correctness Hoare logic as well as the state-based total-correctness Hoare logic. On the level of derivability alone this can be proved semantically by going through the soundness and completeness results. But we go one step further: we show that derivations in these two state-based logics are directly transformable into derivations in our logic, yielding embeddings on the level of derivations, not just mere derivability. The transformations are relatively straightforward and do not require invention of new invariants or variants, demonstrating that our logic incurs no undue proof burden in comparison to the standard Hoare logics. In the converse direction, we can project derivations in our trace-based logic into derivations in the state-based logics: a derivation of a Hoare triple in the trace-based logic is translated into a derivation in the state-based logics with a translated, weaker postcondition.

However, the power of our logic goes beyond that of the state-based partial-correctness and total-correctness Hoare logics. The assertions have access to traces. As suggested by the similarity of our (open) assertion language to the that of interval temporal logic, this allows us to specify liveness properties of diverging runs. We will demonstrate this extra expressiveness of our logic by a series of examples. Also, interpreted into a constructive underlying logic, our assertion language becomes quite discerning. In particular we can distinguish between termination and nondivergence, e.g., unbounded classically total (constructively nonpartial) search fails to be terminating, but is nonetheless nondivergent.

We do not discuss this in the paper, but our logic can be adjusted to deal with exceptions and nondeterminism.

The paper is organized as follows. In Section 2, we present our trace-based big-step semantics. In Section 3, we proceed to the question of a corresponding Hoare logic. We

explain our design considerations and then present our Hoare logic and the soundness and completeness proofs. In Section 4, we show the embeddings of the state-based partial-correctness and total-correctness Hoare logics into our logic and the projections back. In Section 5, we consider examples. In Section 6, we discuss the related work, to conclude in Section 7.

We have formalized the development fully constructively in Coq version 8.1pl3 using the **Ssreflect** syntax extension library. The Coq development is available at <http://cs.io.c.ee/~keiko/code/abyss.tgz>.

Both the paper and the accompanying Coq code use coinductive types, corecursion and coinduction extensively. For an introduction, we can refer the reader to the exposition of Bertot and Castéran [1, Ch. 13]. In the paper, we have sought to abstract over the more bureaucratic aspects involved in the Coq formalization (e.g., working with Coq’s restricted guardedness condition on cofix definitions, i.e., definitions by corecursion, proofs by coinduction).

## 2. BIG-STEP SEMANTICS

We start with our big-step semantics. This is defined in terms of states and traces. The notion of a state is standard. A state  $\sigma \in \text{state}$  is an assignment of integer values to the variables. Traces  $\tau \in \text{trace}$  are defined coinductively by the rules<sup>1</sup>

$$\frac{}{\langle \sigma \rangle \in \text{trace}} \quad \frac{\tau \in \text{trace}}{\sigma :: \tau \in \text{trace}}$$

so a trace is a non-empty colist (possibly infinite sequence) of states. We also define (strong) bisimilarity of two traces,  $\tau \approx \tau'$ , coinductively by

$$\frac{}{\langle \sigma \rangle \approx \langle \sigma \rangle} \quad \frac{\tau \approx \tau'}{\sigma :: \tau \approx \sigma :: \tau'}$$

Bisimilarity is straightforwardly seen to be an equivalence. We think of bisimilar traces as equal, i.e., type-theoretically we treat traces as a setoid with bisimilarity as the equivalence relation.<sup>23</sup> Accordingly, we have to make sure that all functions and predicates we define on traces are setoid functions and predicates (i.e., insensitive to bisimilarity). We define the initial state  $hd \tau$  of a trace  $\tau$  by case distinction by  $hd \langle \sigma \rangle = \sigma, hd (\sigma :: \tau) = \sigma$ . The function  $hd$  is a setoid function. We also define finiteness of a trace (with a particular final state) and infiniteness of a trace inductively resp. coinductively by

$$\frac{}{\langle \sigma \rangle \downarrow \sigma} \quad \frac{\tau \downarrow \sigma'}{\sigma :: \tau \downarrow \sigma'} \quad \frac{\tau^\uparrow}{(\sigma :: \tau)^\uparrow}$$

Finiteness and infiniteness are setoid predicates. It should be noticed that infiniteness is defined positively, not as negation of finiteness. Constructively, it is not the case that  $\forall \tau. (\exists \sigma. \tau \downarrow \sigma) \vee \tau^\uparrow$ , which amounts to asserting that finiteness is decidable. In particular,  $\forall \tau. (\neg \exists \sigma. \tau \downarrow \sigma) \rightarrow \tau^\uparrow$  is constructively provable, but  $\forall \tau. \neg \tau^\uparrow \rightarrow \exists \sigma. \tau \downarrow \sigma$  is not.

<sup>1</sup>We mark coinductive definitions by double horizontal rules.

<sup>2</sup>Classically, strong bisimilarity is equality. But we work in an intensional type theory where strong bisimilarity of colists is weaker than equality (just as equality of two functions on all arguments is weaker than equality of these two functions).

<sup>3</sup>In particular, we “pattern-match” traces using bisimilarity: any trace  $\tau$  is bisimilar to either a trace of the form of  $\langle \sigma \rangle$  or one of the form  $\sigma :: \tau'$ .

$$\begin{array}{c}
\frac{}{\overline{\overline{(x := e, \sigma) \Rightarrow \sigma :: \langle \sigma[x \mapsto \llbracket e \rrbracket \sigma] \rangle}}} \quad \frac{}{\overline{\overline{(\text{skip}, \sigma) \Rightarrow \langle \sigma \rangle}}} \quad \frac{(s_0, \sigma) \Rightarrow \tau \quad (s_1, \tau) \overset{*}{\Rightarrow} \tau'}{\overline{\overline{(s_0; s_1, \sigma) \Rightarrow \tau'}}} \\
\frac{\sigma \models e \quad (s_t, \sigma :: \langle \sigma \rangle) \overset{*}{\Rightarrow} \tau}{\overline{\overline{(\text{if } e \text{ then } s_t \text{ else } s_f, \sigma) \Rightarrow \tau}}} \quad \frac{\sigma \not\models e \quad (s_f, \sigma :: \langle \sigma \rangle) \overset{*}{\Rightarrow} \tau}{\overline{\overline{(\text{if } e \text{ then } s_t \text{ else } s_f, \sigma) \Rightarrow \tau}}} \\
\frac{\sigma \models e \quad (s_t, \sigma :: \langle \sigma \rangle) \overset{*}{\Rightarrow} \tau \quad (\text{while } e \text{ do } s_t, \tau) \overset{*}{\Rightarrow} \tau'}{\overline{\overline{(\text{while } e \text{ do } s_t, \sigma) \Rightarrow \tau'}}} \quad \frac{\sigma \not\models e}{\overline{\overline{(\text{while } e \text{ do } s_t, \sigma) \Rightarrow \sigma :: \langle \sigma \rangle}}} \\
\frac{(s, \sigma) \Rightarrow \tau}{\overline{\overline{(s, \langle \sigma \rangle) \overset{*}{\Rightarrow} \tau}}} \quad \frac{(s, \tau) \overset{*}{\Rightarrow} \tau'}{\overline{\overline{(s, \sigma :: \tau) \overset{*}{\Rightarrow} \sigma :: \tau'}}}
\end{array}$$

Figure 1: Big-step semantics

The statements of the While language are given by the following grammar where  $x$  ranges over (integer) variables and  $e$  over (arithmetic) expressions built over variables.

$$s ::= x := e \mid \text{skip} \mid s_0; s_1 \mid \text{if } e \text{ then } s_t \text{ else } s_f \mid \text{while } e \text{ do } s_t$$

The integer value of an expression  $e$  in a state  $\sigma$  is denoted  $\llbracket e \rrbracket \sigma$ . We also interpret expressions as booleans;  $\sigma \models e$  stands for  $e$  being true in  $\sigma$ . Evaluation  $(s, \sigma) \Rightarrow \tau$ , expressing that running a statement  $s$  from a state  $\sigma$  produces a trace  $\tau$ , is defined coinductively by the rules in Figure 1. The rules for sequence and while implement the necessary sequencing with the help of extended evaluation  $(s, \tau) \overset{*}{\Rightarrow} \tau'$ , also defined coinductively, as the coinductive prefix closure of evaluation:  $(s, \tau) \overset{*}{\Rightarrow} \tau'$  expresses that running a statement  $s$  from the last state (if it exists) of an already accumulated trace  $\tau$  results in a total trace  $\tau'$ .

A remarkable feature of the definition of  $(s, \tau) \overset{*}{\Rightarrow} \tau'$  is that it does not hinge on deciding whether the trace  $\tau$  is finite or not, which is constructively impossible. A proof of  $(s, \tau) \overset{*}{\Rightarrow} \tau'$  simply traverses the already accumulated trace  $\tau$ : if the last element is hit, which is the case when  $\tau$  is finite, then the statement is run, otherwise the traversal goes on forever.

We look closer at the sequence rule. We want to conclude that  $(s_0; s_1, \sigma) \Rightarrow \tau'$  from the premise  $(s_0, \sigma) \Rightarrow \tau$ . Classically, either the run of  $s_0$  terminates, i.e.,  $\tau \downarrow \sigma'$  for some  $\sigma'$ , or it diverges, i.e.,  $\tau^\uparrow$ . In the first case, we would like to additionally use that  $\tau$  is a finite prefix of  $\tau'$  and that  $(s_1, \sigma') \Rightarrow \tau''$ , where  $\tau''$  is the rest of  $\tau'$ . In the second case, it should be case that  $\tau \approx \tau'$ . In both cases, the desirable condition is equivalent to  $(s_1, \tau) \overset{*}{\Rightarrow} \tau'$ , which is the second premise of our rule. The use of extended evaluation, defined as the coinductive (rather than inductive) prefix closure of evaluation, allows us to avoid the need to decide whether the run of  $s_0$  terminates or not.

Evaluation is a setoid predicate.

**Proposition 2.1** ([17]). *For any  $s, \sigma, \tau$  and  $\tau'$ , if  $(s, \sigma) \Rightarrow \tau$  and  $\tau \approx \tau'$ , then  $(s, \sigma) \Rightarrow \tau'$ .*

The trace produced from a state begins with this state.

**Proposition 2.2.** *For any  $s, \sigma$  and  $\tau$ , if  $(s, \sigma) \Rightarrow \tau$ , then  $hd \tau = \sigma$ .*

Moreover, for While, evaluation is deterministic (up to bisimilarity, as is appropriate for our notion of trace equality).

**Proposition 2.3** ([17]). *For any  $s, \sigma, \tau$  and  $\tau'$ , if  $(s, \sigma) \Rightarrow \tau$  and  $(s, \sigma) \Rightarrow \tau'$ , then  $\tau \approx \tau'$ .*

And it is also total.

**Proposition 2.4** ([17]). *For any  $s$  and  $\sigma$ , there exists  $\tau$  such that  $(s, \sigma) \Rightarrow \tau$ .*

In our definition, we have made a choice as regards to what grows the trace of a run. We have decided that assignments and testing of guards of if- and while-statements augment the trace by a state (but `skip` does not), e.g., we have  $(x := 17, \sigma) \Rightarrow \sigma :: \langle \sigma[x \mapsto 17] \rangle$ ,  $(\text{while false do skip}, \sigma) \Rightarrow \sigma :: \langle \sigma \rangle$  and  $(\text{while true do skip}, \sigma) \Rightarrow \sigma :: \sigma :: \sigma :: \dots$

This is good for several reasons. First, `skip` becomes the unit of sequential composition, i.e., the semantics does not distinguish  $s$ , `skip`;  $s$  and  $s$ ; `skip`. Second, we get a notion of small steps that fully agrees with a very natural coinductive trace-based small-step semantics arising as a straightforward variation of the textbook inductive state-based small-step semantics. The third and most important outcome is that any while-loop always progresses, because testing of the guard is a small step. For instance, in our semantics `while true do skip` can only derive  $(\text{while true do skip}, \sigma) \Rightarrow \sigma :: \sigma :: \sigma :: \dots$  (up to bisimilarity). As we discuss below, giving up insisting on progress in terms of growing the trace would introduce some semantic anomalies. It also ensures that evaluation is total—as we should expect. Given that it is also deterministic, we can thus equivalently turn our relational big-step semantics into a functional one: the unique trace for a given statement and initial state is definable by corecursion. (For details, see our previous paper [17].)

The coinductive trace-based semantics agrees with the inductive state-based semantics.

**Proposition 2.5** ([17]). *For any  $s$ ,  $\sigma$ ,  $\sigma'$ , existence of  $\tau$  such that  $(s, \sigma) \Rightarrow \tau$  and  $\tau \downarrow \sigma'$  is equivalent to  $(s, \sigma) \Rightarrow^{\text{ind}} \sigma'$ .*

We notice that the inductive state-based semantics cannot be made total constructively. It is unproblematic to complement the inductively defined terminating evaluation relation with a coinductively defined diverging evaluation relation, but this does not help, as we cannot decide the halting problem.

*Discussions on alternative designs.* We look at several seemingly not so different but problematic alternatives that we reject, thereby revealing some subtleties in designing coinductive big-step semantics and motivating our design choices.

Since progress of loops is not required for wellformedness of the definitions of  $\Rightarrow$  and  $\Rightarrow^*$ , one might be tempted to regard guard testing to be instantaneous and modify the rules for the while-loop to take the form

$$\frac{\sigma \models e \quad (s_t, \sigma) \Rightarrow \tau \quad (\text{while } e \text{ do } s_t, \tau) \xRightarrow{*} \tau'}{(\text{while } e \text{ do } s_t, \sigma) \Rightarrow \tau'} \quad \frac{\sigma \not\models e}{(\text{while } e \text{ do } s_t, \sigma) \Rightarrow \langle \sigma \rangle}$$

This leads to undesirable outcomes. We can derive  $(\text{while true do skip}, \sigma) \Rightarrow \langle \sigma \rangle$ , which means that the non-terminating `while true do skip` is considered semantically equivalent to the terminal (immediately terminating) `skip`. Worse, we can also derive  $(\text{while true do skip}; x := 17, \sigma) \Rightarrow \sigma :: \langle \sigma[x \mapsto 17] \rangle$ , which is even more inadequate: a sequence can continue to run after the non-termination of the first statement. Yet worse, inspecting the rules closer we discover we are also able to derive  $(\text{while true do skip}, \sigma) \Rightarrow \tau$  for any  $\tau$ . Mathematically, giving up insisting on progress in terms of growing the trace has also the consequence that the relational semantics cannot be turned into a functional one, although While should intuitively be total and deterministic. In a functional semantics,

evaluation must be a trace-valued function and in a constructive setting such a function must be productive.

Another option, where assignments and test of guards are properly taken to constitute steps, could be to define  $\overset{*}{\Rightarrow}$  by case distinction on the statement by rules such as

$$\frac{\tau \models^* e \quad (s_t, \text{duplast } \tau) \overset{*}{\Rightarrow} \tau' \quad (\text{while } e \text{ do } s_t, \tau') \overset{*}{\Rightarrow} \tau''}{(\text{while } e \text{ do } s_t, \tau) \overset{*}{\Rightarrow} \tau''} \quad \frac{\tau \not\models^* e}{(\text{while } e \text{ do } s_t, \tau) \overset{*}{\Rightarrow} \text{duplast } \tau}$$

Here, *duplast*  $\tau$ , defined corecursively, traverses  $\tau$  and duplicates its last state, if it is finite. Similarly,  $\tau \models^* e$  and  $\tau \not\models^* e$  traverse  $\tau$  and evaluate  $e$  in the last state, if it is finite:

$$\frac{\tau \models^* e}{\sigma :: \tau \models^* e} \quad \frac{\sigma \models e}{\langle \sigma \rangle \models^* e} \quad \frac{\tau \not\models^* e}{\sigma :: \tau \not\models^* e} \quad \frac{\sigma \not\models e}{\langle \sigma \rangle \not\models^* e}$$

(The rules for *skip* and sequence are very simple and appealing in this design.) The relation  $\Rightarrow$  would then be defined uniformly by the rule

$$\frac{(s, \langle \sigma \rangle) \overset{*}{\Rightarrow} \tau}{(s, \sigma) \Rightarrow \tau}$$

It turns out that we can still derive  $(\text{while true do skip}, \sigma) \Rightarrow \tau$  for any  $\tau$ . We can even derive  $(\text{while true do } x := x + 1, \sigma) \Rightarrow \tau$  for any  $\tau$ .

The third alternative (Leroy and Grall use this technique in [12]) is most close to ours. It introduces, instead of our  $\overset{*}{\Rightarrow}$  relation, an auxiliary relation *split*, defined coinductively by

$$\frac{}{\text{split } \langle \sigma \rangle \langle \sigma \rangle \sigma \langle \sigma \rangle} \quad \frac{\tau \approx \tau'}{\text{split } (\sigma :: \tau) \langle \sigma \rangle \sigma (\sigma :: \tau')} \quad \frac{\text{split } \tau \tau_0 \sigma' \tau_1}{\text{split } (\sigma :: \tau) (\sigma :: \tau_0) \sigma' \tau_1}$$

so that *split*  $\tau' \tau_0 \sigma' \tau_1$  expresses that the trace  $\tau'$  can be split into a concatenation of traces  $\tau_0$  and  $\tau_1$  glued together at a mid-state  $\sigma'$ . Then the evaluation relation is defined by replacing the uses of  $\overset{*}{\Rightarrow}$  with *split*, e.g., the rule for the sequence statement would be:

$$\frac{\text{split } \tau' \tau_0 \sigma' \tau_1 \quad (s_0, \sigma) \Rightarrow \tau_0 \quad (s_1, \sigma') \Rightarrow \tau_1}{(s_0; s_1, \sigma) \Rightarrow \tau'}$$

This third alternative does not cause any outright anomalies for While. But alarmingly  $s_1$  has to be run from some (underdetermined) state within a run of  $s_0; s_1$  even if the run of  $s_0$  does not terminate. In a richer language with abnormal terminations, we get a serious problem: no evaluation is derived for  $(\text{while true do skip}); \text{abort}$  although the *abort* statement should not be reached.

### 3. HOARE LOGIC

We now proceed to the Hoare logic and its soundness and completeness proof. We base our consequence rule on semantic entailment rather than derivability in some fixed proof system. This allows us to sidestep the problem of its unavoidable incompleteness due to the impossibility of complete axiomatization of any theory containing arithmetic. By identifying assertions with state and trace predicates (more precisely, predicates expressible in the meta-logic), we also avoid the risk of possible incompleteness due to a chosen narrower assertion language not being closed under weakest preconditions/strongest postconditions. To formulate the rules of the Hoare logic, we introduce a small set of assertion connectives,

$$\begin{array}{c}
\frac{}{\sigma \models \text{true}} \quad \frac{\neg(\sigma \models U)}{\sigma \models \neg U} \quad \frac{\sigma \models U \quad \sigma \models V}{\sigma \models U \wedge V} \quad \dots \\
\frac{}{\tau \models \text{true}} \quad \frac{\neg(\tau \models P)}{\tau \models \neg P} \quad \frac{\tau \models P \quad \tau \models Q}{\tau \models P \wedge Q} \quad \dots \\
\frac{\sigma \models U}{\langle \sigma \rangle \models \langle U \rangle} \quad \frac{\tau \models P \quad \tau' \models_{\tau} Q}{\tau' \models P ** Q} \quad \frac{\tau \models \langle \text{true} \rangle}{\tau \models P^{\dagger}} \quad \frac{\tau \models P \quad \tau' \models_{\tau} P^{\dagger}}{\tau' \models P^{\dagger}} \\
\frac{\sigma \models U}{\sigma :: (\sigma[x \mapsto \llbracket e \rrbracket \sigma]) \models U[x \mapsto e]} \quad \frac{\sigma \models U}{\sigma :: \langle \sigma \rangle \models \langle U \rangle^2} \\
\frac{\tau \models P \quad \tau \downarrow \sigma}{\sigma \models \text{Last } P} \quad \frac{\tau \downarrow \sigma}{\tau \models \text{finite}} \quad \frac{\tau^{\uparrow}}{\tau \models \text{infinite}} \\
\frac{hd \ \tau = \sigma \quad \tau \models Q}{\tau \models_{\langle \sigma \rangle} Q} \text{ [flw-nil]} \quad \frac{\tau' \models_{\tau} Q}{\sigma :: \tau' \models_{\sigma :: \tau} Q} \text{ [flw-delay]} \\
\frac{\forall \sigma. \sigma \models U \rightarrow \sigma \models V}{U \models V} \quad \frac{\forall \tau. \tau \models P \rightarrow \tau \models Q}{P \models Q}
\end{array}$$

Figure 2: Semantics of assertions

i.e., operations on predicates. To be able to express the strongest postcondition of any precondition, we need a few additional connectives.

**3.1. Assertions.** Our assertions are predicates over states and traces. A state predicate  $U$  is any predicate on states (in particular, it need not be decidable). From a trace predicate  $P$ , we require additionally that it is a setoid predicate, i.e., it must be unable to distinguish bisimilar traces.

Although we refrain from introducing a language of assertions, we introduce a number of connectives for our assertions, which are operations on predicates. All trace predicate connectives yield setoid predicates. The inference rules of the Hoare logic make use of these connectives. Indeed, it was an intriguing exercise for us to come up with connectives that would be small but expressive enough for practical specification purposes and at the same time allow us to prove the Hoare logic sound and complete in our constructive setting.

The definitions of these connectives are given in Figure 2.<sup>4</sup>

The two most primitive state (resp. trace) predicates are **true** and **false**, which are respectively true and false for any state (resp. trace). We can also use the standard connectives  $\neg, \wedge, \vee$  and quantifiers  $\forall, \exists$  to build state and trace predicates. The context disambiguates the overloaded notations for these state and trace predicates.

<sup>4</sup>We use the symbol  $\models$  to highlight application of a predicate to a state or trace. We are not defining a single satisfaction relation  $\models$  for some assertion language, but a number of individual state/trace predicates and operations on such predicates. Some of these operations are defined inductively, some coinductively, some definitions are not recursive at all.

For a state predicate  $U$ , the singleton  $\langle U \rangle$  is a trace predicate that is true of singleton traces given by a state satisfying  $U$ . In particular  $\langle \text{true} \rangle$  is true of any singleton trace.

For a state predicate  $U$ , the doubleton  $\langle U \rangle^2$  is true of a doubleton trace whose two states are identical and satisfy  $U$ .

For a state predicate  $U$ , the update  $U[x \mapsto e]$  is the strongest postcondition of the statement  $x := e$  for the precondition  $U$ . It is true of a doubleton trace whose first state  $\sigma$  satisfies  $U$  and second state is obtained from the first by modifying the value of  $x$  to become  $\llbracket e \rrbracket \sigma$ .

For trace predicates  $P$  and  $Q$ , the chop  $P ** Q$  is a trace predicate that is true, roughly speaking, of a trace  $\tau'$  that has a prefix  $\tau$  satisfying  $P$ , with the rest of  $\tau'$  satisfying  $Q$ . (To be more precise, the prefix and the rest overlap on a mid-state which is the last state of the prefix and the first state of the suffix.) But its definition is carefully crafted, so that  $Q$  is not checked, if  $\tau$  is infinite (in which case necessarily  $\tau \approx \tau'$ ), and this happens without case distinction on whether  $\tau$  is finite. This effect is achieved with the premise  $\tau' \models_{\tau} Q$ . The relation  $\tau' \models_{\tau} Q$  is defined coinductively. It traverses all of  $\tau$ , making sure that it is a prefix of  $\tau'$  (rule *flw-delay*), and, upon possible exhaustion of  $\tau$  in a finite number of steps, checks  $Q$  against the rest of  $\tau'$  (rule *flw-nil*). This way the problem of deciding whether  $\tau$  is finite is avoided, basically by postponing it, possibly infinitely.

Our chop operator is classically equivalent to the chop operator from interval temporal logic [14, 8] (cf. also the separating conjunction of separating logic). Indeed, classically,  $\tau' \models P ** Q$  holds iff

- either, for some finite prefix  $\tau$  of  $\tau'$ , we have  $\tau \models P$  and  $\tau'' \models Q$ , where  $\tau''$  is the rest of  $\tau'$ ,
- or  $\tau'$  is infinite and  $\tau' \models P$ .

This is how the semantics of chop is defined in interval temporal logic. But it involves an upfront decision of whether  $P$  will be satisfied by a finite or an infinite prefix of  $\tau'$ . Our definition is fine-tuned for constructive reasoning.

For a trace predicate  $P$ , its iteration  $P^{\dagger}$  is a trace predicate that is true of a trace which is a concatenation of a possibly infinite sequence of traces, each of which satisfies  $P$ . (This is modulo the overlap of the last and first states of consecutive traces in the sequence and the empty concatenation being a singleton trace.) It is reminiscent of the Kleene star operator. It is defined by coinduction and takes into account possibilities of both infiniteness of some single iteration and infinite repetition.

For a trace predicate  $P$ , *Last P* is a state predicate that is true of states that can be the last state of a finite trace satisfying  $P$ .

Trace predicates *finite* and *infinite* are true of finite and infinite traces, respectively.

For state predicates  $U$  and  $V$  (resp. trace predicates  $P$  and  $Q$ ),  $U \models V$  (resp.  $P \models Q$ ) denotes entailment.

**Proposition 3.1.** *For any  $U$ ,  $\langle U \rangle$ ,  $U[x \mapsto e]$  and  $\langle U \rangle^2$  are setoid predicates. For any setoid predicates  $P$ ,  $Q$ ,  $P ** Q$  is a setoid predicate. For any setoid predicate  $P$ ,  $P^{\dagger}$  is a setoid predicate. Moreover, *finite* and *infinite* are setoid predicates.*

*Proof.* That  $\langle U \rangle$ ,  $U[x \mapsto e]$  and  $\langle U \rangle^2$  are setoid predicates follows from the definition. We prove that  $P ** Q$  and  $P^{\dagger}$  are setoid predicates when  $P$  and  $Q$  are by coinduction. That *finite* and *infinite* are setoid predicates is proved by induction and by coinduction respectively.  $\square$

**Proposition 3.2.** *For any  $U$  and  $V$ , if  $U \models V$ , then  $\langle U \rangle \models \langle V \rangle$ ,  $U[x \mapsto e] \models V[x \mapsto e]$  and  $\langle U \rangle^2 \models \langle V \rangle^2$ . For any setoid predicates  $P, P'$  and  $Q$ , if  $P \models P'$ , then  $P ** Q \models P' ** Q$*



and  $Q ** P \models Q ** P'$ . For any setoid predicates  $P$  and  $Q$ , if  $P \models Q$ , then  $P^\dagger \models Q^\dagger$  and  $Last P \models Last Q$ .

*Proof.* That  $\langle U \rangle$ ,  $U[x \mapsto e]$ ,  $\langle U \rangle^2$  are monotone follows from the definition. We prove that  $P ** Q$  and  $P^\dagger$  are monotone and that  $Last P$  is monotone by coinduction and by induction respectively.  $\square$

A number of logical consequences and equivalences hold about these connectives, to be proved in Lemma 3.6. We have the trivial equivalence:  $\langle true \rangle ** P \Leftrightarrow P \Leftrightarrow P ** \langle true \rangle$ . The chop operator is associative:  $(P ** Q) ** R \Leftrightarrow P ** (Q ** R)$ . The iterator operator  $P^\dagger$  repeats  $P$  either zero times or once followed by further repetitions:  $P^\dagger \Leftrightarrow \langle true \rangle \vee (P ** P^\dagger)$ . A trace is infinite if and only if  $false$  holds for any last state:  $infinite \Leftrightarrow true ** \langle false \rangle$ . If every trace satisfying  $P$  is infinite, i.e., if  $P \models infinite$ , then any trace satisfying  $P$  has no last state, i.e.,  $Last P \Leftrightarrow false$ . We have  $P ** \langle Last P \rangle \Leftrightarrow P$ , so that if a trace satisfies  $P$ , then its last state, if exists, satisfies  $Last P$ . The last state of a singleton trace  $\langle \sigma \rangle$  is  $\sigma$ , therefore we have  $Last \langle U \rangle \Leftrightarrow U$ . We also have  $Last (P ** Q) \models Last Q$ , but the converse does not hold. E.g.,  $Last true \not\models Last (false ** true)$ . Instead, we have  $Last (\langle Last P \rangle ** Q) \Leftrightarrow Last (P ** Q)$ . Finally we have  $Last (P ** \langle U \rangle) \Leftrightarrow Last P \wedge U$ . Namely, a state satisfies  $U$  and can be the last state of a finite trace satisfying  $P$  if and only if it can be the last state of a finite trace satisfying  $P ** \langle U \rangle$ .

We define the concatenation of traces  $\tau$  and  $\tau'$ ,  $\tau ++ \tau'$ , by replacing the last state of  $\tau$  by  $\tau'$ . Formally, it is defined by corecursion by

$$\langle \sigma \rangle ++ \tau = \tau \quad (\sigma :: \tau) ++ \tau' = \sigma :: (\tau ++ \tau')$$

We first observe three results, which are useful for later proofs.

**Lemma 3.3.** *For any  $\tau$ ,  $\tau \models_\tau \langle true \rangle$ .*

*Proof.* By coinduction and case analysis on  $\tau$ . The case of  $\tau \approx \langle \sigma \rangle$  follows from  $\langle \sigma \rangle \models \langle true \rangle$ . The case of  $\tau \approx \sigma :: \tau'$ : we get  $\tau' \models_{\tau'} \langle true \rangle$  from the coinduction hypothesis, from which we obtain  $\tau \models_\tau \langle true \rangle$ .  $\square$

**Lemma 3.4.** *For any  $U$  and  $\tau, \tau'$ , if  $\tau' \models_\tau \langle U \rangle$ , then  $\tau \approx \tau'$ .*

*Proof.* By coinduction and inversion on  $\tau' \models_\tau \langle U \rangle$ .  $\square$

**Lemma 3.5.** *For any  $U$  and  $\tau$ , if for any  $\sigma$ ,  $\tau \downarrow \sigma$  implies  $\sigma \models U$ , then  $\tau \models_\tau \langle U \rangle$ .*

*Proof.* By coinduction with case analysis on  $\tau$ .

The case of  $\tau \approx \langle \sigma \rangle$ : We have  $\tau \downarrow \sigma$ , hence  $\sigma \models U$  by our hypothesis. We conclude  $\tau \models_\tau \langle U \rangle$ .

The case of  $\tau \approx \sigma' :: \tau'$ : Since  $\tau' \downarrow \sigma$  implies  $\tau \downarrow \sigma$ , we have that, for any  $\sigma$ ,  $\tau' \downarrow \sigma$  implies  $\sigma \models U$ . We get  $\tau' \models_{\tau'} \langle U \rangle$  by the coinduction hypothesis, from where we conclude  $\tau \models_\tau \langle U \rangle$ .  $\square$

**Lemma 3.6.** *For any  $U, V$  and setoid predicates  $P, Q$  and  $R$ , we have*

- (1)  $\langle U \rangle ** \langle V \rangle^2 \Leftrightarrow \langle U \wedge V \rangle^2 \Leftrightarrow \langle U \rangle^2 ** \langle V \rangle$
- (2)  $\langle U \rangle ** \langle V \rangle \Leftrightarrow \langle U \wedge V \rangle$
- (3)  $\langle true \rangle ** P \Leftrightarrow P \Leftrightarrow P ** \langle true \rangle$
- (4)  $(P ** Q) ** R \Leftrightarrow P ** (Q ** R)$
- (5)  $P^\dagger \Leftrightarrow \langle true \rangle \vee (P ** P^\dagger)$
- (6)  $P^\dagger \Leftrightarrow P^\dagger ** P^\dagger$

- (7)  $\text{infinite} \Leftrightarrow \text{true} ** \langle \text{false} \rangle$
- (8) If  $P \models \text{infinite}$ , then  $\text{Last } P \Leftrightarrow \text{false}$ .
- (9)  $P \Leftrightarrow P ** \langle \text{Last } P \rangle$
- (10)  $\text{Last } \langle U \rangle \Leftrightarrow U$
- (11)  $\text{Last } (P ** Q) \models \text{Last } Q$
- (12)  $\text{Last } (\langle \text{Last } P \rangle ** Q) \Leftrightarrow \text{Last } (P ** Q)$
- (13)  $\text{Last } (P ** \langle U \rangle) \models U$
- (14)  $\text{Last } (\langle I \rangle ** (P ** \langle I \rangle^2)^\dagger) \models I$

*Proof.*

- (1) Follows from the definition.
- (2) Follows from the definition.
- (3)  $\langle \text{true} \rangle ** P \Leftrightarrow P$  follows from the definition.  $P \models P ** \langle \text{true} \rangle$  holds by Lemma 3.3. Suppose  $\tau' \models P ** \langle \text{true} \rangle$ . There exists  $\tau$  such that  $\tau \models P$  and  $\tau' \models_\tau \langle \text{true} \rangle$ . By Lemma 3.4  $\tau \approx \tau'$  holds, so we must have  $\tau' \models P$  since  $P$  is a setoid predicate. This proves  $P ** \langle \text{true} \rangle \models P$ .
- (4) Suppose  $\tau'' \models (P ** Q) ** R$ . There exist  $\tau$  and  $\tau'$  such that  $\tau \models P$  and  $\tau' \models_\tau Q$  and  $\tau'' \models_{\tau'} R$ . We prove, for any  $\tau_0, \tau_1$  and  $\tau_2$ ,  $\tau_1 \models_{\tau_0} Q$  and  $\tau_2 \models_{\tau_1} R$  imply  $\tau_2 \models_{\tau_0} Q ** R$  by coinduction and inversion on  $\tau_1 \models_{\tau_0} Q$ . This yields  $\tau'' \models_\tau Q ** R$  therefore  $\tau'' \models P ** (Q ** R)$ .

The converse is more subtle. Given  $\tau'' \models P ** (Q ** R)$ , we have to find a prefix  $\tau'$  of  $\tau''$  that satisfies  $P ** Q$  while  $\tau'' \models_{\tau'} R$ . To do so, we define a function  $\text{midp} : (\tau_1 \models_{\tau_0} P_0 ** Q_0) \rightarrow \text{trace}$  by corecursion (we take  $\tau_0$  and  $\tau_1$  to be implicit parameters of  $\text{midp}$ , inferred from the proof argument).<sup>5</sup>

$$\begin{aligned} & \text{midp } (\text{flw-nil } \sigma \tau_0 \ (\_ : \text{hd } \tau_0 = \sigma) \ (h : \tau_0 \models P_0 ** Q_0)) \\ & \quad = \text{let existT } \tau_1 \ (\_ : \tau_1 \models P_0 \wedge \tau_0 \models_{\tau_1} Q_0) = h \text{ in } \tau_1 \\ & \text{midp } (\text{flw-delay } \sigma \tau_0 \tau_1 \ (h : \tau_1 \models_{\tau_0} P_0 ** Q_0)) = \sigma :: \text{midp } h \end{aligned}$$

We then prove that, for any  $\tau_0, \tau_1$  and  $h : \tau_1 \models_{\tau_0} P_0 ** Q_0$ ,  $\text{midp } h \models_{\tau_0} P_0$  and  $\tau_1 \models_{\text{midp } h} Q_0$  hold by coinduction and inversion on  $h$ .

Now assume  $\tau'' \models P ** (Q ** R)$ . There exists  $\tau$  such that  $\tau \models P$  and  $h : \tau'' \models_\tau Q ** R$ . We have  $\text{midp } h \models P ** Q$ , since  $\text{midp } h \models_\tau Q$ . This together with  $\tau'' \models_{\text{midp } h} R$  proves  $\tau'' \models (P ** Q) ** R$ , as required.

- (5) Follows from the definition.
- (6) Suppose  $\tau \models P^\dagger$ . We have to prove  $\tau \models P^\dagger ** P^\dagger$ . From Lemma 3.3 and (5), we deduce  $\tau \models_\tau P^\dagger$ , which gives us  $\tau \models P^\dagger ** P^\dagger$ . Conversely, suppose  $\tau \models P^\dagger ** P^\dagger$ . There exists  $\tau'$  such that  $\tau' \models P^\dagger$  and  $\tau \models_{\tau'} P^\dagger$ . We close the case by proving the following two conditions by mutual coinduction<sup>6</sup>:
  - (a)  $\forall \tau \tau'. \tau \models P^\dagger \rightarrow \tau' \models_\tau P^\dagger \rightarrow \tau' \models P^\dagger$
  - (b)  $\forall \tau \tau' \tau''. \tau' \models_\tau P^\dagger \rightarrow \tau'' \models_{\tau'} P^\dagger \rightarrow \tau'' \models_\tau P^\dagger$ .
(a): We perform inversion on  $\tau \models P^\dagger$ . The case of  $\tau \models \langle \text{true} \rangle$ , i.e.,  $\tau \approx \langle \sigma \rangle$ : From  $\tau' \models_\tau P^\dagger$ , we conclude  $\tau' \models P^\dagger$ . The case of  $\tau'' \models P$  and  $\tau \models_{\tau''} P^\dagger$ : we get  $\tau' \models_{\tau''} P^\dagger$  by (b), from where we conclude  $\tau' \models P^\dagger$ .

<sup>5</sup>Where useful, we give hypothetical proofs names, like  $h$  below;  $\_$  is for an anonymous dummy argument.  $\text{existT}$  is the constructor of sigma-types in Coq.

<sup>6</sup>In Coq, we actually perform nested coinduction.

(b): We perform inversion on  $\tau' \models_{\tau} P^{\dagger}$ . The case of  $\tau \approx \langle \sigma \rangle$  and  $hd \tau' = \sigma$  and  $\tau' \models P^{\dagger}$  follows from (a). The case of  $\tau \approx \sigma :: \tau_0$  and  $\tau' \approx \sigma :: \tau'_0$  and  $\tau'_0 \models_{\tau_0} P^{\dagger}$ : We must have  $\tau'' \approx \sigma :: \tau''_0$  and  $\tau''_0 \models_{\tau''_0} P^{\dagger}$ . The coinduction hypothesis (b) gives us  $\tau_0 \models_{\tau''_0} P^{\dagger}$ , from which we conclude  $\tau \models_{\tau''} P^{\dagger}$ .

(7) We prove an auxiliary condition: for any  $\tau$ , *infinite*  $\tau$  iff  $\tau \models_{\tau} \langle \text{false} \rangle$  by coinduction. *infinite*  $\models \text{true} ** \langle \text{false} \rangle$  follows from the condition.  $\text{true} ** \langle \text{false} \rangle \models \text{infinite}$  follows from the condition and Lemma 3.4.

(8) Follows from *infinite*  $\wedge$  *finite*  $\models \text{false}$ .

(9) Suppose  $\tau \models P$ . By the definition of *Last P*, we have for any  $\sigma$ ,  $\tau \downarrow \sigma$  implies  $\sigma \models \text{Last } P$ . We then deduce  $\tau \models_{\tau} \langle \text{Last } P \rangle$  by Lemma 3.5, thus conclude  $\tau \models P ** \langle \text{Last } P \rangle$ .

Conversely, suppose that  $\tau' \models P ** \langle \text{Last } P \rangle$ , i.e.,  $\tau' \models P$  and  $\tau' \models_{\tau'} \langle \text{Last } P \rangle$  for some  $\tau$ . By Lemma 3.4,  $\tau \approx \tau'$  holds, so we must have  $\tau' \models P$  since  $P$  is a setoid predicate.

(10) Follows from the definition.

(11) Suppose  $\sigma \models \text{Last } (P ** Q)$ . There exist  $\tau$  and  $\tau'$  such that  $\tau' \downarrow \sigma$  and  $\tau \models P$  and  $\tau' \models_{\tau} Q$ . We have to prove  $\sigma \models \text{Last } Q$ . We do so by proving an auxiliary condition: for any  $\sigma_0$  and  $\tau_0$ , if  $\tau_0 \downarrow \sigma_0$ , then for any  $\tau_1$ ,  $\tau_0 \models_{\tau_1} Q$  implies  $\sigma_0 \models \text{Last } Q$  by induction on the derivation of  $\tau_0 \downarrow \sigma_0$ .

(12) Suppose  $\sigma \models \text{Last } (\langle \text{Last } P \rangle ** Q)$ . There exist  $\tau$  and  $\tau'$  such that  $\tau \models P$ ,  $\tau \downarrow hd \tau'$ ,  $\tau' \models Q$  and  $\tau' \downarrow \sigma$ . We then have that the concatenation of  $\tau$  and  $\tau'$  has the desired properties. Namely,  $\tau ++ \tau' \models P ** Q$  and  $(\tau ++ \tau') \downarrow \sigma$ . This proves  $\sigma \models \text{Last } (P ** Q)$  as we wanted.

Conversely, suppose  $\sigma \models \text{Last } (P ** Q)$ . There exist  $\tau$  and  $\tau'$  such that  $\tau \models P$ ,  $\tau' \models_{\tau} Q$  and  $\tau' \downarrow \sigma$ . The finiteness of  $\tau'$  implies that of  $\tau$ , i.e., we have  $\tau \downarrow \sigma'$  for some  $\sigma'$ . We can therefore find the suffix  $\tau''$  of  $\tau'$  such that  $\tau' \approx \tau ++ \tau''$  and  $hd \tau'' = \sigma'$ . (Basically, we drop the first  $n$  elements from  $\tau'$  to obtain  $\tau''$ , where  $n$  is the length of  $\tau$ . Since  $\tau$  is finite, its length is defined.) Together  $\tau' \models_{\tau} Q$  and  $\tau \downarrow \sigma'$  proves  $\tau'' \models Q$ . This concludes  $\sigma \models \text{Last } (\langle \text{Last } P \rangle ** Q)$ , as we wanted.

(13) By the monotonicity of the last and chop operators, it suffices to prove  $\text{Last } (\text{true} ** \langle U \rangle) \models U$ . Suppose  $\sigma \models \text{Last } (\text{true} ** \langle U \rangle)$ . There exists  $\tau$  such that  $\tau \downarrow \sigma$  and  $\tau \models \text{true} ** \langle U \rangle$ . We now prove  $\forall \tau, \sigma. \tau \downarrow \sigma \rightarrow \forall \tau'. \tau \models_{\tau'} \langle U \rangle \rightarrow \sigma \models U$  by induction on the proof of  $\tau \downarrow \sigma$ , from where  $\sigma \models U$  follows.

(14) By (13) and the monotonicity of the chop and iterator operators, it suffices to prove  $\langle U \rangle ** (\text{true} ** \langle U \rangle^2)^{\dagger} \models \text{true} ** \langle U \rangle$ . Suppose  $\tau \models \langle U \rangle ** (\text{true} ** \langle U \rangle^2)^{\dagger}$ . There exists  $\tau'$  such that  $\tau' \models \langle U \rangle$  and  $\tau \models_{\tau'} (\text{true} ** \langle U \rangle^2)^{\dagger}$ , which give us  $\tau \models (\text{true} ** \langle U \rangle^2)^{\dagger}$  and  $hd \tau \models U$ . We want to prove  $\tau \models_{\tau} \langle U \rangle$ . We do so by proving the following conditions by mutual coinduction.

(a)  $\forall \tau. hd \tau \models U \rightarrow \tau \models (\text{true} ** \langle U \rangle^2)^{\dagger} \rightarrow \tau \models_{\tau} \langle U \rangle$

(b)  $\forall \tau, \tau', \tau''. \tau' \models_{\tau} \langle U \rangle^2 \rightarrow \tau'' \models_{\tau'} (\text{true} ** \langle U \rangle^2)^{\dagger} \rightarrow \tau'' \models_{\tau''} \langle U \rangle$ .

(a): We perform inversion on  $\tau \models (\text{true} ** \langle U \rangle^2)^{\dagger}$ . The case of  $\tau \models \langle \text{true} \rangle$ : We have  $\tau \approx \langle \sigma \rangle$ . This and  $hd \tau \models U$  prove  $\tau \models_{\tau} \langle U \rangle$ . The case of  $\tau' \models \text{true} ** \langle U \rangle^2$  and  $\tau \models_{\tau'} (\text{true} ** \langle U \rangle^2)^{\dagger}$ : The former gives us  $\tau' \models_{\tau''} \langle U \rangle^2$  for some  $\tau''$ . We therefore conclude  $\tau \models_{\tau} \langle U \rangle$  by (b).

(b): We perform inversion on  $\tau' \models_{\tau} \langle U \rangle^2$ : The case of  $\tau \approx \langle \sigma \rangle$  and  $hd \tau' = \sigma$  and  $\tau' \models \langle U \rangle^2$ : We have  $\tau' \approx \sigma :: \langle \sigma \rangle$ , therefore  $\tau'' \approx \sigma :: \tau''_0$  for some  $\tau''_0$  with  $hd \tau''_0 = \sigma$ , and  $\tau''_0 \models (\text{true} ** \langle U \rangle^2)^{\dagger}$ . From (a), we obtain  $\tau''_0 \models_{\tau''_0} \langle U \rangle$ , which yields  $\tau'' \models_{\tau''} \langle U \rangle$ ,

as required. The case of  $\tau' \approx \sigma :: \tau'_0$  and  $\tau \approx \sigma :: \tau_0$  and  $\tau'_0 \models_{\tau_0} \langle U \rangle^2$ : We have  $\tau'' \approx \sigma :: \tau''_0$  and  $\tau''_0 \models_{\tau'_0} (\text{true} ** \langle U \rangle^2)^\dagger$ . By (b), we get  $\tau''_0 \models_{\tau''_0} \langle U \rangle$ , which yields  $\tau'' \models_{\tau''} \langle U \rangle$ . □

**3.2. Inference rules.** The derivable judgements of the Hoare logic are given by the inductively interpreted inference rules in Figure 3. The proposition  $\{U\} s \{P\}$  states derivability of the judgement. The intent is that  $\{U\} s \{P\}$  should be derivable precisely when running a statement  $s$  from an initial state satisfying  $U$  is guaranteed to produce a trace satisfying  $P$ .

The rules for assignment and skip are self-explanatory.

The rule for sequence is defined in terms of the chop operator. The precondition  $V$  for the second statement  $s_1$  is given by those states in which a run of the first statement  $s_0$  may terminate. In particular, if  $\{U\} s_0 \{P\}$  and  $P \models \text{infinite}$ , i.e.,  $s_0$  is necessarily diverging for the precondition  $U$ , then we have  $\{U\} s_0 \{P ** \langle \text{false} \rangle\}$ . In this case, from the derivability of  $\{\text{false}\} s_1 \{Q\}$  for any  $Q$ , we get  $\{U\} s_0; s_1 \{P ** Q\}$  for any  $Q$ . But this makes sense, since  $P ** Q \Leftrightarrow P$  as soon as  $P \models \text{infinite}$ .

The rule for if-statement uses the doubleton operator in accordance with the operational semantics where we have chosen that testing the boolean guard grows the trace.

The rule for while-statement is inspired by the corresponding rule of the standard, state-based partial-correctness Hoare logic. It uses a loop invariant  $I$ . This is a state predicate that has to be true each time the boolean guard is about to be (re-)tested in a run of the loop. Accordingly, the precondition  $U$  should be stronger than  $I$ . Also,  $I$  must hold each time an iteration of  $s_t$  has finished, as enforced by having  $P ** \langle I \rangle$  as the postcondition of  $s_t$ . The postcondition  $\langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle$  of the loop consists of three parts.  $\langle U \rangle^2$  accounts for the first test of the guard;  $(P ** \langle I \rangle^2)^\dagger$  accounts for iterations of the loop body in alternation with re-tests of the guard (notice that that we are again using the doubleton operator);  $\langle \neg e \rangle$  accounts for the state in which the last test of the guard is finished.

We have chosen to introduce a separate rule for instantiating auxiliary variables. Alternatively, we might have stated the consequence rule in a more general form, as suggested by Kleymann [13]; yet the separation facilitates formalization in Coq.

The various logical consequences and equivalences about the connectives suggest also further alternative and equivalent formulations. For instance, we could replace the rule for the while-statement by

$$\frac{\{e \wedge I\} s_t \{P ** \langle I \rangle\}}{\{I\} \text{ while } e \text{ do } s_t \{\langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle\}}$$

if we strengthened the consequence rule to

$$\frac{U \models U' \quad \{U'\} s \{P'\} \quad \langle U \rangle ** P' \models P}{\{U\} s \{P\}}$$

With our chosen rule for while, this strengthened version of consequence is admissible:

**Lemma 3.7.** *For any  $U$ ,  $s$  and  $P$ , if  $\{U\} s \{P\}$ , then  $\{U\} s \{\langle U \rangle ** P\}$ .*

*Proof.* We prove the following more general statement by induction on the derivation of  $\{U\} s \{P\}$ : for any  $U$ ,  $s$  and  $P$ , if  $\{U\} s \{P\}$ , then for any  $V$ ,  $\{U \wedge V\} s \{\langle V \rangle ** P\}$ . □

$$\begin{array}{c}
\frac{}{\{U\} x := e \{U[x \mapsto e]\}} \quad \frac{}{\{U\} \text{skip} \{\langle U \rangle\}} \quad \frac{\{U\} s_0 \{P ** \langle V \rangle\} \quad \{V\} s_1 \{Q\}}{\{U\} s_0; s_1 \{P ** Q\}} \\
\frac{\{e \wedge U\} s_t \{P\} \quad \{\neg e \wedge U\} s_f \{P\}}{\{U\} \text{if } e \text{ then } s_t \text{ else } s_f \{\langle U \rangle^2 ** P\}} \\
\frac{U \models I \quad \{e \wedge I\} s_t \{P ** \langle I \rangle\}}{\{U\} \text{while } e \text{ do } s_t \{\langle U \rangle^2 ** (P ** \langle I \rangle)^\dagger ** \langle \neg e \rangle\}} \\
\frac{U \models U' \quad \{U'\} s \{P'\} \quad P' \models P}{\{U\} s \{P\}} \quad \frac{\forall z. \{U\} s \{P\}}{\{\exists z. U\} s \{\exists z. P\}}
\end{array}$$

Figure 3: Inference rules of Hoare logic

We do not attempt to argue that our formulation is the best choice; yet we found that the present formulation is viable from the points-of-view of both the meta-theory and applicability of the logic.

**3.3. Soundness.** The soundness result states that any derivable Hoare triple is semantically valid in the sense that, if the precondition holds of the initial state of an evaluation, then the postcondition is true of the trace produced.

**Proposition 3.8** (Soundness). *For any  $s$ ,  $U$  and  $P$ , if  $\{U\} s \{P\}$ , then, for all  $\sigma$  and  $\tau$ ,  $\sigma \models U$  and  $(s, \sigma) \Rightarrow \tau$  imply  $\tau \models P$ .*

*Proof.* By induction on the derivation of  $\{U\} s \{P\}$ . We show the main cases of sequence and while.

- $s = s_0; s_1$ : We are given as the induction hypothesis that, for any  $\sigma, \tau$ ,  $(s_0, \sigma) \Rightarrow \tau$  and  $\sigma \models U$  imply  $\tau \models P ** \langle V \rangle$ , and that, for any  $\sigma, \tau$ ,  $(s_1, \sigma) \Rightarrow \tau$  and  $\sigma \models V$  imply  $\tau \models Q$ . We have to prove  $\tau_1 \models P ** Q$ , given  $\sigma \models U$  and  $(s_0, \sigma) \Rightarrow \tau_0$  and  $(s_1, \tau_0) \xRightarrow{*} \tau_1$ . The induction hypothesis for  $s_0$  gives us  $\tau_0 \models P ** \langle V \rangle$ . By Lemma 3.4 and that  $P$  is a setoid predicate, we derive  $h_0 : \tau_0 \models P$  and  $\tau_0 \models_{\tau_0} \langle V \rangle$ . We prove by coinduction an auxiliary lemma: for any  $\tau, \tau'$ ,  $\tau \models_{\tau} \langle V \rangle$  and  $(s_1, \tau) \xRightarrow{*} \tau'$  give  $\tau' \models_{\tau} Q$ , using the induction hypothesis for  $s_1$ . The lemma gives us  $h_1 : \tau_1 \models_{\tau_0} Q$ . We can now close the case by  $h_0$  and  $h_1$ .
- $s = \text{while } e \text{ do } s_t$ : We are given as the induction hypothesis that for any  $\sigma$  and  $\tau$ ,  $\sigma \models e \wedge I$  and  $(\sigma, s_t) \Rightarrow \tau$  imply  $\tau \models P ** \langle I \rangle$ . We also have  $U \models I$ . We have to prove  $\tau \models \langle U \rangle^2 ** (P ** \langle I \rangle)^\dagger ** \langle \neg e \rangle$ , given  $\sigma \models U$  and  $(\text{while } e \text{ do } s_t, \sigma) \Rightarrow \tau$ . We prove that, for any  $\sigma$  and  $\tau$ ,  $(\text{while } e \text{ do } s_t, \sigma) \Rightarrow \tau$  implies  $\tau \models_{\tau} \langle \neg e \rangle$  by coinduction. It remains to prove  $\tau \models \langle U \rangle^2 ** (P ** \langle I \rangle)^\dagger$ . By inversion on  $(\text{while } e \text{ do } s_t, \sigma) \Rightarrow \tau$ , we learn that  $\tau \approx \sigma' :: \tau'$  for some  $\sigma'$  and  $\tau'$  such that  $hd \tau' = \sigma'$ . So, we close the case by proving the following conditions by mutual coinduction:
  - for any  $\sigma$  and  $\tau$ , if  $\sigma \models I$  and  $(\text{while } e \text{ do } s_t, \sigma) \Rightarrow \sigma :: \tau$ , then  $\tau \models (P ** \langle I \rangle)^\dagger$
  - for any  $\tau$  and  $\tau'$ , if  $\tau \models_{\tau} \langle I \rangle$  and  $(\text{while } e \text{ do } s_t, \tau) \xRightarrow{*} \tau'$ , then  $\tau' \models_{\tau} \langle I \rangle^2 ** (P ** \langle I \rangle)^\dagger$ .

□

$$\begin{aligned}
sp(x := e, U) &= U[x \mapsto e] \\
sp(\text{skip}, U) &= \langle U \rangle \\
sp(s_0; s_1, U) &= P ** sp(s_1, \text{Last } P) \text{ where } P = sp(s_0, U) \\
sp(\text{if } e \text{ then } s_t \text{ else } s_f, U) &= \langle U \rangle^2 ** (sp(s_t, e \wedge U) \vee sp(s_f, \neg e \wedge U)) \\
sp(\text{while } e \text{ do } s_t, U) &= \langle U \rangle^2 ** (sp(s_t, e \wedge I) ** \langle I \rangle^\dagger ** \langle \neg e \rangle) \\
&\text{where } I = \text{Inv}(e, s_t, U)
\end{aligned}$$

$$\frac{\sigma \models U}{\sigma \models \text{Inv}(e, s, U)} \quad \frac{V \models \text{Inv}(e, s, U) \quad \sigma \models \text{Last } (\langle \text{Inv}(e, s, U) \wedge e \rangle ** sp(s, V))}{\sigma \models \text{Inv}(e, s, U)}$$

Figure 4: Strongest postcondition

Thanks to Proposition 2.4 (totality of evaluation), as an immediate corollary of Proposition 3.8 (soundness) we learn that, if a state satisfies the precondition of a derivable Hoare triple, then there exists an evaluation producing a trace satisfying the postcondition.

**Corollary 3.9** (Total-correctness soundness). *For any  $s, U$  and  $P$ , if  $\{U\} s \{P\}$ , then, for any  $\sigma$  such that  $\sigma \models U$ , there exists  $\tau$  such that  $(s, \sigma) \Rightarrow \tau$  and  $\tau \models P$ .*

**3.4. Completeness.** The completeness result states that any semantically valid Hoare triple is derivable. Following the standard approach (see, e.g., [20]) we define, for a given statement  $s$  and a given precondition  $U$ , a trace predicate  $sp(s, U)$ —the candidate strongest postcondition. Then we prove that  $sp(s, U)$  is a postcondition according to the logic (i.e.,  $\{U\} s \{sp(s, U)\}$  is derivable) and that  $sp(s, U)$  is semantically stronger than any other trace predicate that is a postcondition semantically. Completeness follows.

The trace predicate  $sp(s, U)$  is defined by recursion on  $s$  in Figure 4. The definition is mostly self-explanatory, as it mimics the inference rules of the logic, except that we need the loop-invariant  $\text{Inv}(e, s, U)$ .  $\text{Inv}(e, s, U)$  characterizes the set of states that are reachable by some run of while  $e$  do  $s_t$  from a state satisfying  $U$  and where the boolean guard is tested in that run.<sup>7</sup>

For any  $s$  and  $U$ , the predicate  $sp(s, U)$  is a monotone setoid predicate.

**Lemma 3.10.** *For any  $s, U, \tau, \tau'$ , if  $\tau \models sp(s, U)$  and  $\tau \approx \tau'$ , then  $\tau' \models sp(s, U)$ .*

*Proof.* By induction on the structure of  $s$ . □

**Lemma 3.11.** *For any  $s, U, U'$ , if  $U \models U'$ , then  $sp(s, U) \models sp(s, U')$ .*

*Proof.* By induction on the structure of  $s$ . □

The following lemma states that any trace which satisfies  $sp(s, U)$  has its first state satisfying  $U$ .

**Lemma 3.12.** *For any  $s, U, \tau$ , if  $\tau \models sp(s, U)$ , then  $hd \tau \models U$ .*

*Proof.* By induction on the structure of  $s$ . □

<sup>7</sup>Because of the induction-recursion involved in the simultaneous definition of  $\text{Inv}(e, s, U)$  and  $sp(s, U)$ , we have used impredicativity in our Coq development.

The next lemma states a crucial property of  $Inv(e, s, U)$ .

**Lemma 3.13.** *For any  $s, e, U$ ,  $sp(s, Inv(e, s, U) \wedge e) \Leftrightarrow sp(s, Inv(e, s, U) \wedge e) ** \langle Inv(e, s, U) \rangle$ .*

*Proof.* ( $\Rightarrow$ ): Suppose  $\tau \models sp(s, Inv(e, s, U) \wedge e)$ . It suffices to prove  $\tau \models_{\tau} \langle Inv(e, s, U) \rangle$ . We have  $hd \tau \models Inv(e, s, U) \wedge e$  by Lemma 3.12, and  $\tau \models sp(s, Inv(e, s, U))$  by Lemma 3.11 and  $Inv(e, s, U) \wedge e \models Inv(e, s, U)$ . These give us  $\tau \models \langle Inv(e, s, U) \wedge e \rangle ** sp(s, Inv(e, s, U))$ . By the definition of  $Inv$ , we have for any  $\sigma$ ,  $\tau \downarrow \sigma$  implies  $\sigma \models Inv(e, s, U)$ . Therefore we conclude  $\tau \models_{\tau} \langle Inv(e, s, U) \rangle$  by Lemma 3.5.

( $\Leftarrow$ ): Suppose  $\tau \models sp(s, Inv(e, s, U) \wedge e) ** \langle Inv(e, s, U) \rangle$ . We then have some  $\tau'$  such that  $\tau' \models sp(s, Inv(e, s, U) \wedge e)$  and  $\tau \models_{\tau'} \langle Inv(e, s, U) \rangle$ . The latter proves  $\tau \approx \tau'$  by Lemma 3.4. We conclude  $\tau \models sp(s, Inv(e, s, U) \wedge e)$  by Lemma 3.10.  $\square$

We are now ready to establish that  $sp(s, U)$  is a postcondition according to the Hoare logic.

**Lemma 3.14.** *For any  $s, U$ , we have  $\{U\} s \{sp(s, U)\}$ .*

*Proof.* By induction on  $s$ . We show the main cases of sequence and while.

- $s = s_0; s_1$ : We are given as the induction hypotheses that, for any  $U_0$ ,  $\{U_0\} s_0 \{sp(s_0, U_0)\}$  and  $\{U_0\} s_1 \{sp(s_1, U_0)\}$ . We have to prove  $\{U\} s_0; s_1 \{P ** sp(s_1, Last P)\}$  where  $P = sp(s_0, U)$ . By the induction hypothesis, we have  $\{U\} s_0 \{P\}$ , thus  $\{U\} s_0 \{P ** \langle Last P \rangle\}$  by (9) of Lemma 3.6 and the consequence rule. We therefore close the case with  $\{Last P\} s_1 \{sp(s_1, Last P)\}$  given by the induction hypothesis.
- $s = \text{while } e \text{ do } s_t$ : We are given as the induction hypothesis that  $\{U_0\} s_t \{sp(s_t, U_0)\}$ , for any  $U_0$ . We have to prove  $\{U\} \text{while } e \text{ do } s_t \{\langle U \rangle^2 ** (sp(s_t, e \wedge I) ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle\}$  where  $I = Inv(e, s_t, U)$ . It is sufficient to prove  $\{e \wedge I\} s_t \{(sp(s_t, e \wedge I) ** \langle I \rangle)\}$ , which follows from the induction hypothesis and Lemma 3.13.  $\square$

Following the standard route, it remains to prove the following condition: for any  $s, U, P$ , if for all  $\sigma, \tau$ ,  $\sigma \models U$  and  $(s, \sigma) \Rightarrow \tau$  imply  $\tau \models P$ , then  $sp(s, U) \models P$ .

This will be an immediate corollary from Lemma 3.12 and the following lemma, stating that any trace satisfying  $sp(s, U)$  is in fact produced by a run of  $s$ .

**Lemma 3.15.** *For any  $s, U, \tau$ , if  $\tau \models sp(s, U)$  then  $(s, hd \tau) \Rightarrow \tau$ .*

*Proof.* By induction on  $s$ . We show the main cases of sequence and while.

- $s = s_0; s_1$ : We are given as the induction hypotheses that, for any  $U', \tau', \tau' \models sp(s_0, U')$  (resp.  $\tau' \models sp(s_1, U')$ ) implies  $(s_0, hd \tau') \Rightarrow \tau'$  (resp.  $(s_1, hd \tau') \Rightarrow \tau'$ ). We have to prove  $(s_0; s_1, hd \tau) \Rightarrow \tau$ , given  $\tau \models sp(s_0; s_1, U)$ , which unfolds into  $\tau_0 \models sp(s_0, U)$  and  $\tau \models_{\tau_0} sp(s_1, Last (sp(s_0, U)))$ . By the induction hypothesis for  $s_0$ , we have  $(s_0, hd \tau_0) \Rightarrow \tau_0$ . Using the induction hypothesis for  $s_1$ , we prove by coinduction that, for any  $\tau_1, \tau_2$ ,  $\tau_2 \models_{\tau_1} sp(s_1, Last (sp(s_0, U)))$  implies  $(s_1, \tau_1) \stackrel{*}{\Rightarrow} \tau_2$ , thereby we close the case.
- $s = \text{while } e \text{ do } s_t$ : We are given as the induction hypothesis that, for any  $U', \tau', \tau' \models sp(s_t, U')$  implies  $(s_t, hd \tau') \Rightarrow \tau'$ . We have to prove  $(\text{while } e \text{ do } s_t, hd \tau) \Rightarrow \tau$ , given  $\tau \models \langle U \rangle^2 ** (sp(s_t, e \wedge I) ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle$  where  $I = Inv(e, s_t, U)$ . We do so by proving the following two conditions simultaneously by mutual coinduction:
  - for any  $\tau$ ,  $\tau \models (sp(s_t, e \wedge I) ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle$  implies  $(\text{while } e \text{ do } s_t, hd \tau) \Rightarrow hd \tau :: \tau$ ,
  - for any  $\tau$  and  $\tau'$ ,  $\tau' \models_{\tau} \langle I \rangle^2 ** (sp(s_t, e \wedge I) ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle$  implies  $(\text{while } e \text{ do } s_t, \tau) \stackrel{*}{\Rightarrow} \tau'$ .  $\square$

**Corollary 3.16.** *For any  $s, U$  and  $P$ , if for all  $\sigma$  and  $\tau$ ,  $\sigma \models U$  and  $(s, \sigma) \Rightarrow \tau$  imply  $\tau \models P$ , then  $sp(s, U) \models P$ .*

Completeness is proved as a corollary of Lemma 3.14 and Corollary 3.16.

**Proposition 3.17** (Completeness). *For any  $s, U$  and  $P$ , if for all  $\sigma$  and  $\tau$ ,  $\sigma \models U$  and  $(s, \sigma) \Rightarrow \tau$  imply  $\tau \models P$ , then  $\{U\} s \{P\}$ .*

*Proof.* Assume that for all  $\sigma, \tau$ ,  $\sigma \models U$  and  $(s, \sigma) \Rightarrow \tau$  imply  $\tau \models P$ . By Corollary 3.16, we have that  $sp(s, U) \models P$ . By Lemma 3.14, we have  $\{U\} s \{sp(s, U)\}$ . Applying consequence, we get  $\{U\} s \{P\}$ .  $\square$

Combining Propositions 2.3 (determinacy of evaluation) and 3.17 (completeness), we immediately get completeness for total correctness.

**Corollary 3.18** (Total-correctness completeness). *For any  $s, U$  and  $P$ , if, for all  $\sigma$  such that  $\sigma \models U$ , there is  $\tau$  such that  $(s, \sigma) \Rightarrow \tau$  and  $\tau \models P$ , then  $\{U\} s \{P\}$ .*

#### 4. RELATION TO THE STANDARD PARTIAL-CORRECTNESS AND TOTAL-CORRECTNESS HOARE LOGICS

It is easy to see, by going through the soundness and completeness results, that our trace-based Hoare logic is a conservative extension of the standard, state-based partial-correctness and total-correctness Hoare logics. But more can be said. The derivations in these two logics are directly transformable into derivations in our logic, preserving their structure, without invention of new invariants or variants. And in the converse direction, derivations in our logic are transformable into derivations into the standard logics in a way that removes from postconditions information about intermediate states. In this direction, the variant for a while-loop is obtained by bounding the length of traces satisfying the trace invariant of the loop.

Concerning total correctness, we use two variations of the while-rule. In the forward transformation, we use a version of the while-rule with a dedicated variant (a natural-valued *function* on states) whereas, in the backward transformation, we work with a version where the invariant (a state predicate) is made dependent on a natural number (i.e., becomes a *relation* between states and naturals; crucially, there is no functionality requirement: in the same state, the invariant can be satisfied by zero or one or several naturals). The two alternative while-rules for total correctness are:

$$\frac{\forall n : \text{nat}. \{e \wedge I \wedge t = n\} s_t \{I \wedge t < n\}}{\{I \wedge t = m\} \text{ while } e \text{ do } s_t \{I \wedge t \leq m \wedge \neg e\}} \text{ while-fun}$$

and

$$\frac{\forall n : \text{nat}. \{e \wedge J n\} s_t \{\exists k. k < n \wedge J k\}}{\{J m\} \text{ while } e \text{ do } s_t \{\exists k. k \leq m \wedge J k \wedge \neg e\}} \text{ while-rel}$$

There is a reason for this discrepancy, which reflects our compromise between pursuing a constructive approach and striving for purely syntactic translations. We will discuss it in Section 4.3 after having presented the transformations.

We have tried to fine-tune the inference rules in the different Hoare logics and the transformations between them for smoothness. There is some room for variations in them. The transformations are quite sensitive to the exact division of labor in the source and



target Hoare logics between the rules for the statement constructors and the consequence rule, but the effects of the possible variations are mostly inessential.

For reference, the inference rules of the state-based logics appear in the Appendix. Notice that also here we use predicates as assertions and entailment as consequence, so there is no dedicated assertion language or proof system for assertions.

**4.1. Embeddings of the standard Hoare logics into the trace-based logic.** We formalize our claim of embeddability of the standard Hoare logics in the following two propositions, whose direct proofs are algorithms for the transformations.

Proposition 4.1 states that, if  $\{U\} s \{Z\}$  is a derivable partial-correctness judgement, then  $\{U\} s \{\text{true} ** \langle Z \rangle\}$  is derivable in our logic. The trace predicate  $\text{true} ** \langle Z \rangle$  indicates that  $Z$  holds of any state that is reachable by traversing, in a finite number of steps, the whole trace  $\tau$  produced by running  $s$ . Classically, this amounts to  $Z$  being true of the last state of  $\tau$ , if  $\tau$  is finite and hence has one; if  $\tau$  is infinite, then nothing is required.

Proposition 4.2 states that, if  $\{U\} s \{Z\}$  is a derivable total-correctness judgement, then  $\{U\} s \{\text{finite} ** \langle Z \rangle\}$  is derivable in our logic (in fact, it states a little more). The trace predicate  $\text{finite} ** \langle Z \rangle$  expresses that the trace  $\tau$  produced by running  $s$  is finite and  $Z$  holds of the last state of  $\tau$ ; the finiteness of  $\tau$  guarantees the existence of this last state.

**Proposition 4.1.** *For any  $s, U$  and  $Z$ , if  $\{U\} s \{Z\}$  is derivable in the partial-correctness Hoare logic, then  $\{U\} s \{\text{true} ** \langle Z \rangle\}$  is derivable in the trace-based Hoare logic.*

*Proof.* By induction on the derivation of  $\{U\} s \{Z\}$ . We show the main cases of sequence and while.

- $s = s_0; s_1$ : We are given as the induction hypotheses  $\{U\} s_0 \{\text{true} ** \langle V \rangle\}$  and  $\{V\} s_1 \{\text{true} ** \langle Z \rangle\}$ . We have to prove  $\{U\} s_0; s_1 \{\text{true} ** \langle Z \rangle\}$ , which is derived by

$$\frac{\frac{\begin{array}{c} \vdots \\ \text{IH}_0 \end{array} \quad \frac{\{U\} s_0 \{\text{true} ** \langle V \rangle\}}{\{U\} s_0; s_1 \{\text{true} ** \text{true} ** \langle Z \rangle\}} \quad \begin{array}{c} \vdots \\ \text{IH}_1 \end{array} \quad \{V\} s_1 \{\text{true} ** \langle Z \rangle\}}{\{U\} s_0; s_1 \{\text{true} ** \text{true} ** \langle Z \rangle\}}}{\{U\} s_0; s_1 \{\text{true} ** \langle Z \rangle\}} \quad (1)$$

(1) We have

$$\text{true} ** \text{true} \Leftrightarrow \text{true}$$

- $s = \text{while } e \text{ do } s_t$ : We are given as the induction hypothesis  $\{e \wedge I\} s_t \{\text{true} ** \langle I \rangle\}$ . We have to prove  $\{I\} \text{while } e \text{ do } s_t \{\text{true} ** \langle I \wedge \neg e \rangle\}$ , which is derived by

$$\frac{\frac{\begin{array}{c} \vdots \\ \text{IH}_t \end{array} \quad \{e \wedge I\} s_t \{\text{true} ** \langle I \rangle\}}{\{I\} \text{while } e \text{ do } s_t \{\langle I \rangle^2 ** (\text{true} ** \langle I \rangle)^{\dagger} ** \langle \neg e \rangle\}}}{\{I\} \text{while } e \text{ do } s_t \{\text{true} ** \langle I \wedge \neg e \rangle\}} \quad (1)$$

(1) We have

$$\begin{aligned} & \langle I \rangle^2 ** (\text{true} ** \langle I \rangle)^{\dagger} ** \langle \neg e \rangle \\ & \models \text{true} ** \langle I \rangle ** \langle \neg e \rangle \\ & \Leftrightarrow \text{true} ** \langle I \wedge \neg e \rangle \quad (\text{by Lemma 3.6 (2)}) \end{aligned}$$

□

For the embedding of total-correctness derivations, we prove a slightly stronger statement to have the induction go through.

**Proposition 4.2.** *For any  $s, U$  and  $Z$ , if  $\{U\} s \{Z\}$  is derivable in the total-correctness Hoare logic with while-fun, then for any  $W$ ,  $\{U \wedge W\} s \{\langle W \rangle ** \text{finite} ** \langle Z \rangle\}$  is derivable in the trace-based Hoare logic.*

*Proof.* By induction on the derivation of  $\{U\} s \{Z\}$ . We show the main cases of sequence and while.

- $s = s_0; s_1$ : We are given as the induction hypotheses that, for any  $W_0$ ,  $\{U \wedge W_0\} s_0 \{\langle W_0 \rangle ** \text{finite} ** \langle V \rangle\}$  and, for any  $W_1$ ,  $\{V \wedge W_1\} s_1 \{\langle W_1 \rangle ** \text{finite} ** \langle Z \rangle\}$ . We have to prove that, for any  $W$ ,  $\{U \wedge W\} s_0; s_1 \{\langle W \rangle ** \text{finite} ** \langle Z \rangle\}$ . This is done by the derivation

$$\frac{\frac{\frac{\vdots \text{IH}_0 W}{\{U \wedge W\} s_0 \{\langle W \rangle ** \text{finite} ** \langle V \rangle\}} \quad \frac{\frac{\vdots \text{IH}_1 V}{\{V \wedge V\} s_1 \{\langle V \rangle ** \text{finite} ** \langle Z \rangle\}}}{\{V\} s_1 \{\langle V \rangle ** \text{finite} ** \langle Z \rangle\}}}{\{U \wedge W\} s_0; s_1 \{\langle W \rangle ** \text{finite} ** \langle V \rangle ** \text{finite} ** \langle Z \rangle\}}}{\{U \wedge W\} s_0; s_1 \{\langle W \rangle ** \text{finite} ** \langle Z \rangle\}} \quad (1)$$

(1) We have

$$\text{finite} ** \langle V \rangle ** \text{finite} \models \text{finite} ** \text{finite} \Leftrightarrow \text{finite}$$

- $s = \text{while } e \text{ do } s_t$ : We are given as the induction hypothesis that, for all  $n : nat$  and  $W_t$ ,  $\{e \wedge I \wedge t = n \wedge W_t\} s_t \{\langle W_t \rangle ** \text{finite} ** \langle I \wedge t < n \rangle\}$ . We have to prove that, for any  $W$ ,  $\{I \wedge t = m \wedge W\} \text{while } e \text{ do } s_t \{\langle W \rangle ** \text{finite} ** \langle I \wedge \neg e \wedge t \leq m \rangle\}$ . This is accomplished by the derivation

$$\frac{\frac{\frac{\frac{\vdots \forall n. \text{IH}_t n (t = n \wedge t \leq m)}{\forall n. \{e \wedge I \wedge t = n \wedge t = n \wedge t \leq m\} s_t \{\langle t = n \wedge t \leq m \rangle ** \text{finite} ** \langle I \wedge t < n \rangle\}}}{\forall n. \{e \wedge I \wedge t = n \wedge t \leq m\} s_t \{\langle t = n \rangle ** \text{finite} ** \langle I \wedge t < n \wedge t \leq m \rangle\}}}{\{\exists n. e \wedge I \wedge t = n \wedge t \leq m\} s_t \{\exists n. \langle t = n \rangle ** \text{finite} ** \langle I \wedge t < n \wedge t \leq m \rangle\}}}{\{e \wedge I \wedge t \leq m\} s_t \{\langle \exists n. \langle t = n \rangle ** \text{finite} ** \langle t < n \rangle \rangle ** \langle I \wedge t \leq m \rangle\}}}{\frac{\{I \wedge t = m \wedge W\} \text{while } e \text{ do } s_t \quad \{\langle I \wedge t = m \wedge W \rangle^2 ** (\langle \exists n. \langle t = n \rangle ** \text{finite} ** \langle t < n \rangle \rangle ** \langle I \wedge t \leq m \rangle)^{\dagger} ** \langle \neg e \rangle\}}{\{I \wedge t = m \wedge W\} \text{while } e \text{ do } s_t \{\langle W \rangle ** \text{finite} ** \langle I \wedge \neg e \wedge t \leq m \rangle\}} \quad (1)}$$

- (1) The repetition gives rise to a non-empty colist of values of  $t$ , which is strictly decreasing and must hence be of finite length. The concatenation of a finite colist of finite traces is finite.
- (2) If  $t = n$  and  $t \leq m$  in the first state of a trace, then  $n \leq m$  (everywhere), so in the last state  $t < n$  gives  $t < m$ , which can be weakened to  $t \leq m$ .  $\square$

**Corollary 4.3.** *For any  $s, U$  and  $Z$ , if  $\{U\} s \{Z\}$  is derivable in the total-correctness Hoare logic with while-fun, then  $\{U\} s \{\text{finite} ** \langle Z \rangle\}$  is derivable in the trace-based Hoare logic.*

*Proof.* Immediate from Proposition 4.2 by choosing  $W = \text{true}$ .  $\square$

**4.2. Projections of the trace-based logic into standard Hoare logics.** Given that derivations in the standard Hoare logics can be transformed into derivations in the trace-based Hoare logic, it is natural to wonder, if it is also possible to translate derivations in the converse direction. In this direction we would expect some loss (or displacement) of information. Reducing a condition on traces into a condition on those last states that happen to exist or into a condition that also requires their existence must lose or displace the constraints on the intermediate states.

We now proceed to demonstrating that meaningful transformations (“projections”) from the trace-based logic to the standard logics are indeed possible.

We will show that, if  $\{U\} s \{P\}$  is derivable in the trace-based Hoare logic, then so are  $\{U\} s \{Last\ P\}$  in the partial-correctness Hoare logic and  $\{U \wedge [P]m\} s \{Last\ P\}$  in the total-correctness Hoare logic. We will shortly define  $[P]m$  formally, but intuitively, the total-correctness judgement states that  $s$  terminates in a state satisfying  $Last\ P$  from any initial state  $\sigma$  such that  $U$  holds in  $\sigma$  and any  $\sigma$ -headed trace satisfying  $P$  has length at most  $m$ . Since  $m$  is universally quantified on the top level, we can actually derive  $\{U \wedge \exists m. [P]m\} s \{Last\ P\}$ , stating that  $s$  terminates in a state satisfying  $Last\ P$  from any initial state  $\sigma$  such that  $U$  holds and the length of  $\sigma$ -headed traces satisfying  $P$  is bounded.

**Proposition 4.4.** *For any  $s, U$  and  $P$ , if  $\{U\} s \{P\}$  in the trace-based Hoare logic, then for any  $W$ ,  $\{U \wedge W\} s \{Last\ (\langle W \rangle ** P)\}$  is derivable in the partial-correctness Hoare logic.*

*Proof.* By induction on the derivation of  $\{U\} s \{P\}$ . We show the main cases of sequence and while.

- $s = s_0; s_1$ : We are given as the induction hypotheses that  $\{U \wedge W_0\} s_0 \{Last\ (\langle W_0 \rangle ** P ** \langle V \rangle)\}$ , for any  $W_0$ , and  $\{V \wedge W_1\} s_1 \{Last\ (\langle W_1 \rangle ** Q)\}$ , for any  $W_1$ . We have to show that, for any  $W$ ,  $\{U \wedge W\} s_0; s_1 \{Last\ (\langle W \rangle ** P ** Q)\}$ . Let  $V' = Last\ (\langle W \rangle ** P ** \langle V \rangle)$ . We have the derivation

$$\frac{\frac{\frac{\vdots \text{IH}_0\ W \quad \frac{\{V \wedge V'\} s_1 \{Last\ (\langle V' \rangle ** Q)\}}{\{V'\} s_1 \{Last\ (\langle V' \rangle ** Q)\}}}{\{U \wedge W\} s_0 \{V'\}} \quad \vdots \text{IH}_1\ V'}{\{U \wedge W\} s_0; s_1 \{Last\ (\langle V' \rangle ** Q)\}}}{\{U \wedge W\} s_0; s_1 \{Last\ (\langle W \rangle ** P ** Q)\}} \quad (1)$$

- (1) Recalling that  $Last$  is monotone, we have

$$\begin{aligned} & Last\ (\langle V' \rangle ** Q) \\ &= Last\ (\langle Last\ (\langle W \rangle ** P ** \langle V \rangle) \rangle ** Q) \\ &\Leftrightarrow Last\ (\langle W \rangle ** P ** \langle V \rangle ** Q) \quad (\text{by Lemma 3.6 (12)}) \\ &\models Last\ (\langle W \rangle ** P ** Q) \end{aligned}$$

- (2) By Lemma 3.6 (10) and (11),  $V' = Last\ (\langle W \rangle ** P ** \langle V \rangle) \models V$ , therefore  $V' \models V \wedge V'$ .

- $s = \text{while } e \text{ do } s_t$ : We are given as the induction hypothesis that, for any  $W_t$ ,  $\{I \wedge e \wedge W_t\} s_t \{Last\ (\langle W_t \rangle ** P ** \langle I \rangle)\}$ . We now have to prove that, for any  $W$ ,  $\{U \wedge W\} \text{while } e \text{ do } s_t \{Last\ (\langle W \rangle ** \langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle)\}$ , given  $U \models I$ . Let

$I' = \text{Last} (\langle W \rangle ** \langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger)$ . We close the case by the derivation

$$\frac{\frac{\frac{\vdots \text{IH}_t I'}{\{e \wedge I \wedge I'\} s_t \{ \text{Last} (\langle I' \rangle ** P ** \langle I \rangle) \}}}{\{e \wedge I'\} s_t \{I'\}}}{\{I'\} \text{ while } e \text{ do } s_t \{I' \wedge \neg e\}}}{\{U \wedge W\} \text{ while } e \text{ do } s_t \{ \text{Last} (\langle W \rangle ** \langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle) \}} \quad (1)$$

(1) We have

$$\begin{aligned} & U \wedge W \\ \Leftrightarrow & \text{Last} (\langle W \rangle ** \langle U \rangle^2 ** \langle \text{true} \rangle) \quad (\text{by Lemma 3.6 (1) and (10)}) \\ \models & \text{Last} (\langle W \rangle ** \langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger) \quad (\text{by Lemma 3.6 (5)}) \\ = & I' \end{aligned}$$

(2) We have

$$\begin{aligned} & \text{Last} (\langle I' \rangle ** P ** \langle I \rangle) \\ = & \text{Last} (\langle \text{Last} (\langle W \rangle ** \langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger) \rangle ** P ** \langle I \rangle) \\ \Leftrightarrow & \text{Last} (\langle W \rangle ** \langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** P ** \langle I \rangle) \quad (\text{by Lemma 3.6 (12)}) \\ \Leftrightarrow & \text{Last} (\langle W \rangle ** \langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** P ** \langle I \rangle^2) \\ \models & \text{Last} (\langle W \rangle ** \langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger) \\ = & I' \end{aligned} \quad \square$$

**Corollary 4.5.** *For any  $s$ ,  $U$  and  $P$ , if  $\{U\} s \{P\}$ , then  $\{U\} s \{\text{Last } P\}$  is derivable in the partial correctness Hoare logic.*

*Proof.* Immediate from Proposition 4.4 and Lemma 3.6 (3) by instantiating  $W = \text{true}$ .  $\square$

To define the assertion translation for the projection of the trace-based Hoare logic into the total-correctness Hoare logic, we introduce two new connectives. The inductively defined trace predicate  $\text{len } n$  is true of finite traces with length at most  $n$  (we take the singleton trace to have length 0). Given a trace predicate  $P$ , the state predicate  $\lceil P \rceil n$  is defined to be true of a state  $\sigma$ , if every trace headed by  $\sigma$  and satisfying  $P$  also satisfies  $\text{len } n$ .

$$\frac{\frac{}{\langle \sigma \rangle \models \text{len } n} \quad \frac{\tau \models \text{len } n}{\sigma :: \tau \models \text{len } (n+1)}}{\frac{\forall \tau. \text{hd } \tau = \sigma \wedge \tau \models P \rightarrow \tau \models \text{len } n}{\sigma \models \lceil P \rceil n}}$$

**Lemma 4.6.** *For any  $n$ ,  $P$  and  $Q$ , if  $Q \models P$  then  $\lceil P \rceil n \models \lceil Q \rceil n$ .*

For setoid predicates  $P$  and  $Q$ , we say  $Q$  extends  $P$ , written  $P \triangleleft Q$ , if, whenever  $\tau \models P$ , there exists  $\tau'$  such that  $\tau' \models_\tau Q$ , i.e.,  $\forall \tau. \tau \models P \rightarrow \exists \tau'. \tau' \models_\tau Q$ .

**Lemma 4.7.** *For any  $P, P', Q$  and  $Q'$ , if  $P' \models P$  and  $Q \models Q'$  then  $P \triangleleft Q \rightarrow P' \triangleleft Q'$ .*

*Proof.* Follows from the definition.  $\square$

**Lemma 4.8.** *For any  $P, Q$  and  $R$ ,  $(P ** Q) \triangleleft R \rightarrow (\langle \text{Last } P \rangle ** Q) \triangleleft R$ .*

*Proof.* Suppose  $\tau_0 \models \langle \text{Last } P \rangle ** Q$ . There exists  $\tau_1$  such that  $\tau_1 \models P$  and  $\tau_1 \downarrow \text{hd } \tau_0$ . We then have  $\tau_1 ++ \tau_0 \models P ** Q$ . The hypothesis gives us that there exists  $\tau_2$  such that  $\tau_2 \models_{\tau_1 ++ \tau_0} Q$ . We then build a trace  $\tau_3$  such that  $\tau_3 \models_{\tau_0} Q$  from  $\tau_2$  by dropping the first  $n$  elements, where  $n$  is the length of  $\tau_1$ . ( $n$  is welldefined because  $\tau_1$  is finite.)  $\square$

**Lemma 4.9.** *For any  $P$  and  $Q$ ,  $\langle \text{Last } P \rangle \triangleleft Q \rightarrow P \triangleleft Q$ .*

*Proof.* Suppose  $\tau \models P$ . From the hypothesis, we know that, for any  $\sigma$ , if  $\tau \downarrow \sigma$ , then there exists  $\tau'$  such that  $\text{hd } \tau' = \sigma$  and  $\tau' \models Q$ . Using this, we build a trace  $\tau'$  such that  $\tau' \models_{\tau} Q$ , by traversing  $\tau$  and invoking the hypothesis when the last state of  $\tau$  is hit.  $\square$

**Proposition 4.10.** *For any  $s, U$  and  $P$ , if  $\{U\} s \{P\}$  in the trace-based Hoare logic, then for any  $m : \text{nat}, W$  and  $R$  such that  $\langle W \rangle ** P \triangleleft R$ ,  $\{U \wedge W \wedge [P ** R]m\} s \{\text{Last } (\langle W \rangle ** P) \wedge [R]m\}$  is derivable in the total-correctness Hoare logic with while-rel.*

*Proof.* By induction on the derivation of  $\{U\} s \{P\}$ . We show the main cases of sequence and while.

- $s = s_0; s_1$ : We are given as the induction hypotheses that, for any  $m_0, W_0$  and  $R_0$  such that  $(\langle W_0 \rangle ** P ** \langle V \rangle) \triangleleft R_0$ , we have that

$$\{U \wedge W_0 \wedge [P ** \langle V \rangle ** R_0]m_0\} s_0 \{\text{Last } (\langle W_0 \rangle ** P ** \langle V \rangle) \wedge [R_0]m_0\} \quad (\text{IH}_0)$$

and, for any  $m_1, W_1$  and  $R_1$  such that  $(\langle W_1 \rangle ** Q) \triangleleft R_1$ , we have that

$$\{V \wedge W_1 \wedge [Q ** R_1]m_1\} s_1 \{\text{Last } (\langle W_1 \rangle ** Q) \wedge [R_1]m_1\}. \quad (\text{IH}_1)$$

We have to prove that, for any  $m, W$  and  $R$  such that

$$(\langle W \rangle ** P ** Q) \triangleleft R, \quad (\text{H})$$

we have that

$$\{U \wedge W \wedge [P ** Q ** R]m\} s_0; s_1 \{\text{Last } (\langle W \rangle ** P ** Q) \wedge [R]m\}.$$

Let  $V' = \text{Last } (\langle W \rangle ** P ** \langle V \rangle)$ . We close the case by the derivation

$$\frac{\frac{\frac{\vdots (1) \text{ IH}_0 m, W, Q ** R}{\{U \wedge W \wedge [P ** \langle V \rangle ** Q ** R]m\} s_0} \quad \{V' \wedge [Q ** R]m\}}{\{U \wedge W \wedge [P ** Q ** R]m\} s_0} \quad (3) \quad \frac{\frac{\vdots (2) \text{ IH}_1 m, V', R}{\{V \wedge V' \wedge [Q ** R]m\} s_1} \quad \{\text{Last } (\langle V' \rangle ** Q) \wedge [R]m\}}{\{V' \wedge [Q ** R]m\} s_1} \quad (4)}{\{V' \wedge [Q ** R]m\} s_1} \quad \{\text{Last } (\langle V' \rangle ** Q) \wedge [R]m\}}{\frac{\{U \wedge W \wedge [P ** Q ** R]m\} s_0; s_1 \{\text{Last } (\langle V' \rangle ** Q) \wedge [R]m\}}{\{U \wedge W \wedge [P ** Q ** R]m\} s_0; s_1 \{\text{Last } (\langle W \rangle ** P ** Q) \wedge [R]m\}} \quad (5)}$$

- (1) To invoke the induction hypothesis, we have to prove  $\langle W \rangle ** P ** \langle V \rangle \triangleleft Q ** R$ . By Corollary 3.9 and the hypothesis  $\{V\} s_1 \{Q\}$ , we have that, for any  $\sigma$  such that  $\sigma \models V$ , there exists  $\tau$  such that  $\text{hd } \tau = \sigma$  and  $\tau \models Q$ . This gives us  $\langle V \rangle \triangleleft Q$ . By Lemmata 3.6 (11) and 4.7, we have  $\langle \text{Last } (\langle W \rangle ** P ** \langle V \rangle) \rangle \triangleleft Q$ , therefore  $\langle W \rangle ** P ** \langle V \rangle \triangleleft Q$  by Lemma 4.9. Now, for any  $\tau$ , suppose  $\tau \models \langle W \rangle ** P ** \langle V \rangle$ . There exists  $\tau'$  such that  $\tau' \models_{\tau} Q$ . We deduce  $\tau' \models \langle W \rangle ** P ** \langle V \rangle ** Q$ . From the hypothesis H, we know that there exists  $\tau''$  such that  $\tau'' \models_{\tau'} R$ , therefore  $\tau'' \models_{\tau} Q ** R$ , as required.

- (2) To invoke the induction hypothesis, we have to prove  $\langle V' \rangle ** Q \triangleleft R$ , namely  $\langle Last (\langle W \rangle ** P ** \langle V \rangle) \rangle ** Q \triangleleft R$ . By Lemma 4.8, it suffices to show  $\langle W \rangle ** P ** \langle V \rangle ** Q \triangleleft R$ , which follows from the hypothesis (H),  $\langle W \rangle ** P ** \langle V \rangle ** Q \models \langle W \rangle ** P ** Q$  and Lemma 4.7.
- (3)  $[P ** Q ** R]m \models [P ** \langle V \rangle ** Q ** R]m$  follows from Lemma 4.6 and  $P ** \langle V \rangle ** Q ** R \models P ** Q ** R$ .
- (4)  $V' \models V \wedge V'$  holds because  $V' = Last (\langle W \rangle ** P ** \langle V \rangle) \models Last \langle V \rangle \Leftrightarrow V$  from Lemmata 3.6 (10) and (11).
- (5) We have

$$\begin{aligned}
& Last (\langle V' \rangle ** Q) \\
&= Last (\langle Last (\langle W \rangle ** P ** \langle V \rangle) \rangle ** Q) \\
&\Leftrightarrow Last (\langle W \rangle ** P ** \langle V \rangle ** Q) \quad (\text{by Lemma 3.6 (12)}) \\
&\models Last (\langle W \rangle ** P ** Q)
\end{aligned}$$

- $s = \text{while } e \text{ do } s_t$ . We are given as the induction hypothesis that, for any  $m_t, W_t$  and  $R_t$  such that  $\langle W_t \rangle ** P ** \langle I \rangle \triangleleft R_t$ ,

$$\{e \wedge I \wedge W_t \wedge [P ** \langle I \rangle ** R_t]m_t\} s_t \{Last (\langle W_t \rangle ** P ** \langle I \rangle) \wedge [R_t]m_t\}. \quad (\text{IH})$$

We also know  $U \models I$  and  $\{e \wedge I\} s \{P ** \langle I \rangle\}$ , the latter of which gives us

$$\{I\} \text{ while } e \text{ do } s \{ \langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle \}. \quad (4.1)$$

We have to prove that, for any  $m, W$  and  $R$  such that

$$\langle W \rangle ** \langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle \triangleleft R, \quad (\text{H})$$

we have that

$$\begin{aligned}
& \{U \wedge W \wedge [ \langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R ]m\} \\
& \text{ while } e \text{ do } s_t \{ Last (\langle W \rangle ** \langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle) \wedge [R]m \}.
\end{aligned}$$

Let

$$J_0 = Last (\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger),$$

and

$$J_1 n = [ (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R ]n,$$

and

$$Q = \langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R.$$

We first prove that, for all  $n$ ,

$$\{e \wedge J_0 \wedge J_1 n\} s_t \{ \exists k. k < n \wedge J_0 \wedge J_1 k \} \quad (4.2)$$

holds by case analysis on  $n$ .

- $n = 0$ . We prove  $e \wedge J_0 \wedge J_1 0 \models \text{false}$ . Then, by the derivability of  $\{\text{false}\} s \{U\}$  for any  $s$  and  $U$  in the total correctness Hoare logic, we obtain (4.2). Suppose  $\sigma_0 \models e \wedge J_0 \wedge J_1 0$  holds. From  $\sigma_0 \models J_0$ , there exists a trace  $\tau_0$  such that  $\tau_0 \models \langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger$  and  $\tau_0 \downarrow \sigma_0$ . By Lemma 3.6 (14),  $\sigma_0 \models J_0$  and  $U \models I$  give us  $\sigma_0 \models I$ . Applying Corollary 3.9 to (4.1), we know that there exists a trace  $\tau_1$  such that  $hd \tau_1 = \sigma_0$  and  $\sigma_0 :: \tau_1 \models \langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle$ . Hence, we have  $\tau_1 \models \langle I \rangle ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle$ . We deduce

$$\tau_0 ++ \tau_1 \models \langle W \rangle ** \langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle,$$

where  $++$  was defined by corecursion by

$$\langle \sigma \rangle ++ \tau = \tau \quad \sigma :: \tau ++ \tau' = \sigma :: (\tau ++ \tau').$$

The assumption H gives us a trace  $\tau_2$  such that  $\tau_2 \models_{\tau_0 ++ \tau_1} R$ . Since  $\tau_0$  is finite, we find the suffix  $\tau_3$  of  $\tau_2$  such that  $\tau_3 \models (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R$  by dropping the first  $n$  elements from  $\tau_2$ , where  $n$  is the length of  $\tau_0$ . We also have  $hd \tau_3 = \sigma_0$  by construction. The trace  $\tau_3$  cannot be a singleton  $\langle \sigma_0 \rangle$ , which would imply  $\sigma_0 \models e$  and  $\sigma_0 \models \neg e$ . Contradiction by  $\sigma_0 \models J_1 0$ .

–  $n \neq 0$ . We invoke the induction hypothesis IH on  $n$ ,  $J_0$  and  $Q$ . To do so, we have to prove

$$\langle J_0 \rangle ** P ** \langle I \rangle \triangleleft Q. \quad (4.3)$$

By Lemma 4.8, it suffices to prove  $(\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** P ** \langle I \rangle) \triangleleft Q$ . From Corollary 3.9 and (4.1), we deduce  $\langle I \rangle \triangleleft \langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle$ , hence  $\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** P ** \langle I \rangle \triangleleft \langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle$  by Lemma 4.9. This together with the hypothesis (H) proves (4.3).

We have now proved

$$\{e \wedge I \wedge J_0 \wedge [P ** \langle I \rangle ** Q]n\} \text{ st } \{Last (\langle J_0 \rangle ** P ** \langle I \rangle) \wedge [Q]n\}.$$

We close the case by invoking the consequence rule with

$$\begin{aligned} & e \wedge J_0 \wedge J_1 n \\ &= e \wedge J_0 \wedge [(P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R]n \\ &\models e \wedge J_0 \wedge [P ** \langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R]n \\ &\quad (\text{by Lemma 3.6 (5)}) \\ &\Leftrightarrow e \wedge J_0 \wedge [P ** \langle I \rangle ** \langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R]n \\ &\quad (\text{as } \langle I \rangle^2 \Leftrightarrow \langle I \rangle ** \langle I \rangle^2) \\ &= e \wedge Last (\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger) \wedge [P ** \langle I \rangle ** Q]n \\ &\Leftrightarrow e \wedge I \wedge Last (\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger) \wedge [P ** \langle I \rangle ** Q]n \\ &\quad (\text{by Lemma 3.6 (14) and } U \models I) \\ &= e \wedge I \wedge J_0 \wedge [P ** \langle I \rangle ** Q]n, \end{aligned}$$

and

$$\begin{aligned} & Last (\langle J_0 \rangle ** P ** \langle I \rangle) \wedge [Q]n \\ &= Last (\langle Last (\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger) ** P ** \langle I \rangle) \wedge [Q]n \\ &\models Last (\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** P ** \langle I \rangle) \wedge [Q]n \\ &\quad (\text{by Lemma 3.6 (12)}) \\ &\Leftrightarrow Last (\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** P ** \langle I \rangle) \wedge [Q]n \quad (1) \\ &\models Last (\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger) \wedge [Q]n \quad (\text{by Lemma 3.6 (5)}) \\ &= J_0 \wedge [\langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R]n \\ &\models J_0 \wedge [(P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R](n-1) \quad (2) \\ &= J_0 \wedge J_1 (n-1) \\ &\models \exists k. k < n \wedge J_0 \wedge J_1 k \end{aligned}$$

- (1) Given a trace  $\tau$  and a state  $\sigma$  satisfying  $\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** P ** \langle I \rangle$  and  $\tau \downarrow \sigma$ , we have  $\text{duplast } \tau \models \langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** P ** \langle I \rangle^2$  and  $\text{duplast } \tau \downarrow \sigma$ .
- (2) From Lemma 3.6 (14) and  $U \models I$ , we know, for any  $\sigma$ , if  $\sigma \models J_0$ , namely  $\sigma \models \text{Last}(\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger)$ , then  $\sigma \models I$ . Suppose  $\sigma \models J_0 \wedge [\langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R]n$ . It suffices to prove that, for any  $\tau$ , if  $\text{hd } \tau = \sigma$  and  $\tau \models (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R$ , then  $\tau \models \text{len}(n-1)$ . We have  $\sigma :: \tau \models \langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R$ , which together with  $\sigma \models [\langle I \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R]n$  yields  $\sigma :: \tau \models \text{len } n$ . This gives us  $\tau \models \text{len}(n-1)$ .

By the rule for the while-statement by taking  $Jn$  to be  $J_0 \wedge J_1 n$ , we deduce

$$\{J_0 \wedge J_1 m\} \text{ while } e \text{ do } s_t \{ \exists k. k \leq m \wedge J_0 \wedge J_1 k \wedge \neg e \}$$

We have

$$\begin{aligned} & U \wedge W \wedge [\langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R]m \\ & \models \text{Last}(\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger) \wedge [(P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R]m \\ & = J_0 \wedge J_1 m \end{aligned}$$

and

$$\begin{aligned} & \exists k. k \leq m \wedge J_0 \wedge J_1 k \wedge \neg e \\ & \models J_0 \wedge J_1 m \wedge \neg e \quad (\text{by monotonicity of } J_1) \\ & = \text{Last}(\langle U \wedge W \rangle^2 ** (P ** \langle I \rangle^2)^\dagger) \wedge [(P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R]m \wedge \neg e \\ & \models \text{Last}(\langle W \rangle ** \langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle) \wedge [R]m \end{aligned}$$

So, by consequence, we have

$$\begin{aligned} & \{U \wedge W \wedge [\langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle ** R]m\} \\ & \text{ while } e \text{ do } s_t \{ \text{Last}(\langle W \rangle ** \langle U \rangle^2 ** (P ** \langle I \rangle^2)^\dagger ** \langle \neg e \rangle) \wedge [R]m \} \end{aligned}$$

as we wanted.  $\square$

**Corollary 4.11.** *For any  $s$ ,  $U$  and  $P$ , if  $\{U\} s \{P\}$  in the trace-based Hoare logic, then  $\{U \wedge \exists m. [P]m\} s \{\text{Last } P\}$  is derivable in the total-correctness Hoare logic with while-rel.*

*Proof.* From Proposition 4.10 by taking  $W = \text{true}$ ,  $R = \langle \text{true} \rangle$  and then invoking the consequence rule and the admissible rule

$$\frac{\forall m. \{U\} s \{Z\}}{\{\exists m. U\} s \{\exists m. Z\}}$$

of the total-correctness Hoare logic.  $\square$

**4.3. Discussion.** Let us now discuss the transformations from and to the total-correctness Hoare logic. We used two different rules for while. In the forward direction (the embedding), we used while-fun, whereas in the backward direction (the projection), we used while-rel.

With while-rel, the total-correctness Hoare logic is complete, constructively. In particular, while-fun is weaker and straightforwardly derivable from while-rel by instantiating  $Jn = (I \wedge t = n)$ .



In the converse direction, attempts to derive while-fun from while-rel constructively meet several problems. It is natural to define the invariant  $I$  as  $I = \exists m. J m$ . The variant  $t$  should then be defined as

$$t = \begin{cases} \text{the least } k \text{ such that } J k & \text{if } \exists m. J m \\ 0 \text{ (or any other natural)} & \text{if } \neg \exists m. J m \end{cases}$$

This is a sensible classical definition. But constructively, there are two issues with it. First, in the first case, we must be able to find the least  $k$  such that  $J k$ . We have a bound for this search, which is given by the witness  $m_0$  of  $\exists m. J m$ . But in order to search, we must be able to decide where  $J k$  or  $\neg J k$  for each  $k < m_0$ , which requires  $J$  to be decidable. Second, in order to choose between the two cases, we must be able to decide whether  $\exists m. J m$  or  $\neg \exists m. J m$ , which is generally impossible.

These issues show that while-fun is not fine-tuned for constructive reasoning. To formulate a more sensitive rule, we may notice that we only ever need  $t$  in contexts where  $I = \exists m. J m$  can be assumed to hold. So instead of  $I \wedge t = n$  we could use  $\Sigma p : I. t p = n$  making the variant  $t$  depend on the proof of the invariant. The corresponding modified rule is

$$\frac{\forall n : \text{nat} \{e \wedge \Sigma p : I. t p = n\} \ s_t \ \{\Sigma p : I. t p < n\}}{\{\Sigma p : I. t p = m\} \ \text{while } e \ \text{do } s_t \ \{\Sigma p : I. t p \leq m \wedge \neg e\}} \text{ while-fun}'$$

Ideally, we would like to use while-rel (or some interderivable rule) both in the transformations to and from the trace-based Hoare logic. But in the forward direction we have been unable to do it. We conjecture the reason to be that the state-based total-correctness triples and trace-based (partial-correctness) triples express different types of propositions about the underlying evaluation relations: total-correctness triples are about existence of a final state satisfying the postcondition, whereas partial-correctness triples are about all traces produced satisfying the postcondition. This discrepancy shows itself also in our backward transformation: we need to invoke the totality of the trace-based evaluation relation.

## 5. EXAMPLES

Propositions 4.1 and 4.2 show that our trace-based logic is expressive enough to perform the same type of analyses that the state-based partial or total correctness Hoare logics can perform. However, the expressiveness of our logic goes beyond that of the partial and the total correctness Hoare logics. In this section, we demonstrate this by a series of examples. We adopt the usual notational convention that any occurrence of a variable in a state predicate represents the value of the variable in the state, e.g., a state predicate  $x + y = 7$  abbreviates  $\lambda \sigma. \sigma x + \sigma y = 7$ .

**5.1. Unbounded total search.** Since we work in a constructive underlying logic, we can distinguish between termination of a run, *finite*, and nondivergence,  $\neg$ *infinite*. For instance, any unbounded nonpartial search fails to be terminating but is nonetheless nondivergent.

This example is inspired by Markov's principle:  $(\neg \forall n. \neg B n) \rightarrow \exists n. B n$  for any decidable predicate  $B$  on natural numbers, i.e., a predicate satisfying  $\forall n. B n \vee \neg B n$ . Markov's principle is a classical tautology, but is not valid constructively. This implies we cannot constructively prove a statement  $s$  that searches a natural number  $n$  satisfying  $B$  by successively checking whether  $B 0, B 1, B 2, \dots$  to be terminating. In other words, we cannot constructively derive a total correctness judgement for  $s$ . The assumption  $\neg \forall n. \neg B n$  only

guarantees that  $B$  is not false everywhere, therefore the search cannot diverge; indeed, we can constructively prove that  $s$  is nondivergent in our logic.

We assume given a decidable predicate  $B$  on natural numbers and an axiom  $B\_noncontradictory$ :  $\neg\forall n. \neg B n$  stating that  $B$  is not false everywhere. Therefore running the statement

$$Search \equiv x := 0; \text{while } \neg B x \text{ do } x := x + 1$$

cannot diverge: this would contradict  $B\_noncontradictory$ . In Proposition 5.3 we prove that any trace produced by running  $Search$  is nondivergent and  $B x$  holds of the last state.

We define a family of trace predicates  $cofinally n$  coinductively as follows:

$$\frac{\sigma \ x = n \quad B \ n}{\sigma :: \langle \sigma \rangle \models cofinally \ n} \quad \frac{\sigma \ x = n \quad \neg B \ n \quad \tau \models cofinally \ (n + 1)}{\sigma :: \sigma :: \tau \models cofinally \ n}$$

$cofinally n$  is a setoid predicate.

A crucial observation is that, in the presence of  $B\_noncontradictory$ ,  $cofinally 0$  is stronger than nondivergent:

**Lemma 5.1.**  $cofinally 0 \models \neg infinite$ .

*Proof.* It is sufficient to prove that, for any  $\tau$ ,  $\tau \models cofinally 0$  and  $\tau \models infinite$  are contradictory. Suppose there is a trace  $\tau$  such that  $\tau \models cofinally 0$  and  $\tau \models infinite$ . Then by induction on  $n$  we can show that, for any  $n$ , there is a trace  $\tau'$  such that  $\tau' \models cofinally n$  and  $\tau' \models infinite$ . But whenever this holds for some  $\tau'$  and  $n$ , then  $\neg B n$ . Hence we also have  $\forall n. \neg B n$ . But this contradicts  $B\_noncontradictory$ .  $\square$

It is straightforward to prove that, if  $\tau$  satisfies  $cofinally 0$ , then its last state (if exists) satisfies  $B$  at  $x$ . (We refer to the program variable  $x$  in the statement  $Search$ .)

**Lemma 5.2.**  $cofinally 0 \models \text{true} ** \langle B \ x \rangle$ .

*Proof.* We prove the following condition by coinduction, from which the lemma follows: for any  $n, \tau$ , if  $\tau \models cofinally n$ , then  $\tau \models_{\tau} \langle B \ x \rangle$ .  $\square$

**Proposition 5.3.**  $\{\text{true}\} Search \{(\text{true} ** \langle B \ x \rangle) \wedge \neg infinite\}$ .

*Proof.* A sketch of the derivation is given in Figure 5. (At several places we have applied the consequence rule silently.)

- (1) We use Lemmata 5.1 and 5.2.
- (2)  $x$  is incremented by one in every iteration until  $B$  holds at  $x$ .  $\langle x = 0 \rangle^2 ** ((\neg B \ x)[x \mapsto x + 1]) ** \langle \text{true} \rangle^{\dagger} ** \langle B \ x \rangle \models cofinally 0$  follows from the definition of  $cofinally$ . (It is proved by coinduction.)
- (3) We take  $\text{true}$  as the loop invariant.  $\square$

**5.2. Liveness.** As the similarity of our assertion language to the interval temporal logic suggests, we can specify and prove liveness properties. In Proposition 5.5, we prove that the statement

$$x := 0; \text{while true do } x := x + 1$$

eventually sets the value of  $x$  to  $n$  for any  $n : nat$  at some point.

The example is simple but sufficient to demonstrate core techniques used to prove liveness properties of more practical examples. For instance, imagine that assignment to  $x$  involves a system call, with the assigned value as the argument. It is straightforward to

$$\begin{array}{c}
\frac{\{\neg B x\} x := x + 1 \{(\neg B x)[x \mapsto x + 1]\}}{\{x = 0\} \text{ while } \neg B x \text{ do } x := x + 1} \quad (3) \\
\frac{\{x = 0\}^2 ** ((\neg B x)[x \mapsto x + 1] ** \langle \text{true} \rangle^2)^\dagger ** \langle B x \rangle}{\{ \text{true} \} x := 0 \{ \text{true}[x \mapsto 0] \} \{x = 0\} \text{ while } \neg B x \text{ do } x := x + 1 \{ \text{cofinally } 0 \}} \quad (2) \\
\frac{\{ \text{true} \} x := 0; \text{ while } \neg B x \text{ do } x := x + 1 \{ \text{true}[x \mapsto 0] ** \text{cofinally } 0 \}}{\{ \text{true} \} x := 0; \text{ while } \neg B x \text{ do } x := x + 1 \{ (\text{true} ** \langle B x \rangle) \wedge \neg \text{infinite} \}} \quad (1)
\end{array}$$

Figure 5: Derivation of  $\{ \text{true} \} \text{ Search } \{ (\text{true} ** \langle B x \rangle) \wedge \neg \text{infinite} \}$ 

$$\begin{array}{c}
\frac{\{ \text{true} \} x := x + 1 \{ \text{true}[x \mapsto x + 1] \}}{\{x = 0\} \text{ while true do } x := x + 1} \quad (3) \\
\frac{\{x = 0\}^2 ** (\text{true}[x \mapsto x + 1] ** \langle \text{true} \rangle^2)^\dagger ** \langle \text{false} \rangle}{\{ \text{true} \} x := 0 \{ \text{true}[x \mapsto 0] \} \{x = 0\} \text{ while true do } x := x + 1 \{ \text{eventually } n \}} \quad (2) \\
\frac{\{ \text{true} \} x := 0; \text{ while true do } x := x + 1 \{ \text{true}[x \mapsto 0] ** \text{eventually } n \}}{\{ \text{true} \} x := 0; \text{ while true do } x := x + 1 \{ \text{finite} ** \langle x = n \rangle ** \text{true} \}} \quad (1)
\end{array}$$

Figure 6: Derivation of  $\{ \text{true} \} x := 0; \text{ while true do } x := x + 1 \{ \text{finite} ** \langle x = n \rangle ** \text{true} \}$ 

enrich traces to record such special events, and we can then apply the same proof technique to prove the statement eventually performs the system call with  $n$  as the argument for any  $n$ .

For every  $n$ , we define inductively a trace predicate *eventually*  $n$  stating that a state  $\sigma$  in which the value of  $x$  is  $n$  is eventually reached by finitely traversing  $\tau$ :

$$\frac{\sigma x = n}{\langle \sigma \rangle \models \text{eventually } n} \quad \frac{\sigma x = n}{\sigma :: \tau \models \text{eventually } n} \quad \frac{\tau \models \text{eventually } n}{\sigma :: \tau \models \text{eventually } n}$$

If  $\tau$  satisfies *eventually*  $n$ , then it has a finite prefix followed by a state that maps  $x$  to  $n$ .

**Lemma 5.4.** *For any  $n, \tau$ , if  $\tau \models \text{eventually } n$  then  $\tau \models \text{finite} ** \langle x = n \rangle ** \text{true}$ .*

*Proof.* By induction on the derivation of *eventually*  $n$ . □

**Proposition 5.5.** *For any  $n : \text{nat}$ ,  $\{ \text{true} \} x := 0; \text{ while true do } x := x + 1 \{ \text{finite} ** \langle x = n \rangle ** \text{true} \}$ .*

*Proof.* A sketch of the derivation is given in Figure 6.

- (1) We use Lemma 5.4.
- (2)  $x$  is incremented by one in every iteration, starting from zero. Each iteration is finite: the length of the trace of each iteration is invariably two.  $x$  must eventually become  $n$  after a finite number of iterations for any  $n$ .
- (3) We take *true* as the invariant. □

**5.3. Weak trace equivalence.** The last example is inspired by a notion of weak trace equivalence: two traces are weakly equivalent if they are bisimilar by identifying a finite number of consecutive identical states with a single state. It is conceivable that (strong) bisimilarity is too strong for some applications and one needs weak bisimilarity. For instance, we may want to prove that the observable behavior, such as the colist of i/o events of a potentially diverging run, is bisimilar to a particular colist of i/o events. Then we must be able to collapse a finite number of non-observable internal steps. We definitely should not collapse an infinite number of internal steps, otherwise we would end up concluding that a statement performing an i/o operation after a diverging run, e.g., `while true do skip; print "hello"`, is observably equivalent to a statement immediately performing the same i/o operation, e.g., `print "hello"`.

In this subsection, we prove that the trace produced by running the statement

$$\text{while true do } (y := x; (\text{while } y \neq 0 \text{ do } y := y - 1); x := x + 1)$$

is weakly bisimilar to the ascending sequence of natural numbers  $0 :: 1 :: 2 :: 3 :: \dots$ , by projecting the value of  $x$ , assuming that  $x$  is initially 0. The statement differs from that of the previous subsection in that it “stutters” for a finite but unbounded number of steps, i.e., `while  $y \neq 0$  do  $y := y - 1$` , before the next assignment to  $x$  happens.

This exercise is instructive in that we need to formalize weak trace equivalence in our constructive underlying logic. We do so by supplying an inductive predicate  $\tau \overset{*}{\rightsquigarrow} \tau'$  stating that  $\tau'$  is obtained from  $\tau$  by dropping finitely many elements from the beginning, until the first state with a different value of  $x$  is encountered, and a coinductive trace predicate  $up\ n$  stating that the given trace is weakly bisimilar to the ascending sequence of natural numbers starting at  $n$ , by projecting the value of  $x$ . Formally:

$$\frac{\sigma\ x = hd\ \tau\ x\ \ \ \tau\ \overset{*}{\rightsquigarrow}\ \tau'}{\sigma :: \tau\ \overset{*}{\rightsquigarrow}\ \tau'} \quad \frac{\sigma\ x \neq hd\ \tau\ x\ \ \ \tau\ \approx\ \tau'}{\sigma :: \tau\ \overset{*}{\rightsquigarrow}\ \tau'}$$

$$\frac{\sigma\ x = n\ \ \ \sigma :: \tau\ \overset{*}{\rightsquigarrow}\ \tau' \quad \tau' \models up\ (n + 1)}{\sigma :: \tau \models up\ n}$$

These definitions are tailored to our example. But a more general weak trace equivalence can be defined similarly [16]. We note that our formulation is not the only one possible nor the most elegant. In particular, with a logic supporting nesting of induction into coinduction as primitive [6], there is no need to separate the definition into an inductive part,  $\tau \overset{*}{\rightsquigarrow} \tau'$ , and a coinductive part,  $up\ n$ . Yet our formulation is amenable for formalization in Coq.

We also use an auxiliary trace predicate  $\langle x \rangle^*$  that is true of a finite trace in which the value of  $x$  does not change and is non-negative at the end and hence everywhere. It is defined inductively as follows:

$$\frac{x \geq 0}{\langle \sigma \rangle \models \langle x \rangle^*} \quad \frac{\sigma\ x = hd\ \tau\ x\ \ \ \tau \models \langle x \rangle^*}{\sigma :: \tau \models \langle x \rangle^*}$$

**Proposition 5.6.**  $\{x = 0\}\ s\ \{up\ 0\}$  where  $s \equiv \text{while true do } (y := x; (\text{while } y \neq 0 \text{ do } y := y - 1); x := x + 1)$ .

*Proof.* A sketch of the derivation is given in Figure 7.

$$\begin{array}{c}
\frac{\{y \neq 0 \wedge y \geq 0\} y := y - 1 \{y > 0[y \mapsto y - 1] ** \langle y \geq 0 \rangle\}}{\{x \geq 0 \wedge y \geq 0\} \text{ while } y \neq 0 \text{ do } y := y - 1} \quad (4) \\
\frac{\{x \geq 0 \wedge y \geq 0\}^2 ** ((y > 0)[y \mapsto y - 1] ** \langle y \geq 0 \rangle^2)^\dagger ** \langle y = 0 \rangle\}}{\{x \geq 0 \wedge y \geq 0\} \text{ while } y \neq 0 \text{ do } y := y - 1 \{ \langle x \rangle^* \}} \quad (3) \\
\frac{\{x \geq 0\} y := x \{ \langle x \geq 0 \rangle [y \mapsto x] \} \quad \frac{\{x \geq 0\} x := x + 1 \{ \langle x \geq 0 \rangle [x \mapsto x + 1] \}}{\{x \geq 0 \wedge y \geq 0\} (\text{while } y \neq 0 \text{ do } y := y - 1); x := x + 1} \\
\{ \langle x \rangle^* ** (x \geq 0)[x \mapsto x + 1] \}}{\{x \geq 0\} y := x; (\text{while } y \neq 0 \text{ do } y := y - 1); x := x + 1 \{ \langle x \rangle^* ** (x \geq 0)[x \mapsto x + 1] ** \langle x \geq 0 \rangle \}} \quad (2) \\
\frac{\{x = 0\} \text{ while true do } (y := x; (\text{while } y \neq 0 \text{ do } y := y - 1); x := x + 1) \\
\{ \langle x = 0 \rangle^2 ** (\langle x \rangle^* ** (x \geq 0)[x \mapsto x + 1] ** \langle x \geq 0 \rangle^2)^\dagger ** \langle \text{false} \rangle \}}{\{x = 0\} \text{ while true do } (y := x; (\text{while } y \neq 0 \text{ do } y := y - 1); x := x + 1) \{up\ 0\}} \quad (1)
\end{array}$$

Figure 7: Derivation of  $\{x = 0\} s \{up\ 0\}$ 

- (1)  $x$  is incremented by one in every iteration, starting from zero, and the loop is iterated forever. Each iteration is finite, although globally unbounded. So  $x$  must increase forever.
- (2) We take  $x \geq 0$  as the invariant.
- (3)  $y$  is decremented by one in every iteration, starting from a non-negative number. It must become zero after a finite number of iterations.  $x$  does not change.
- (4) We take  $y \geq 0$  as the invariant. □

## 6. RELATED WORK

Coinductive big-step semantics for nontermination have been considered by Cousot and Cousot [5] and by Leroy and Grall [11, 12] (in the context of the CompCert project, which is a major demonstration of feasibility of certified compilation). Cousot and Cousot [5] study fixpoints on bi-semantic domains, partitioned into domains of terminating and diverging behaviors; they prove a specific fixed-point theorem for such domains (bi-induction), and then produce a bi-inductive big-step semantics for lambda-calculus (avoiding some duplication of rules between what would otherwise be distinct inductive and coinductive definitions of terminating resp. diverging evaluation). Leroy and Grall [12] investigate two big-step semantics approaches for lambda-calculus. The first, based on Cousot and Cousot [4], has different evaluation relations for terminating and diverging runs, one inductive (with finite traces), the other coinductive (with infinite traces). To conclude that any program either terminates or diverges, they need the law of excluded middle (amounting to decidability of the halting problem), and, as a result, cannot prove the standard small-step semantics sound wrt. the big-step semantics constructively. The second approach, applied in [2], uses a coinductively defined evaluation relation with possibly infinite traces. While-loops are not ensured to be progressive in terms of growing traces (an infinite number of consecutive silent small steps may be collapsed) and this leads to problems.

Some other works on coinductive big-step semantics include Glesner [7] and Nestra [18, 19]. In these it is accepted that a program evaluation can somehow continue after an infinite

number of small steps. With Glesner, this seems to have been a curious unintended side-effect of the design, which she was experimenting with just for the interest of it. Nestra developed a nonstandard semantics with transfinite traces on purpose in order to obtain a soundness result for a widely used slicing transformation that is unsound standardly (can turn nonterminating runs into terminating runs).

Our trace-based coinductive big-step semantics [17] was heavily inspired by Capretta’s [3] modelling of nontermination in a constructive setting similar to ours. Rather than using coinductive possibly infinite traces, he works with a coinductive notion of a possibly infinitely delayed value (for statements, this corresponds to delaying the final state). The categorical basis appears in Rutten’s work [21]. But Rutten only studied the classical setting (any program terminates or not), where a delayed state collapses to a choice of between a state or a designated token signifying nontermination.

A general categorical account of small-step trace-based semantics has been given by Hasuo et al. [9].

While Hoare logics for big-step semantics based on inductive, finite traces have been considered earlier (to reason about traces of terminating runs), Hoare or VDM-style logics for reasoning about properties of nonterminating runs do not seem to have been studied before, with one exception, see below. Neither do we in fact know about dynamic logic or KAT (Kleene algebra with tests) approaches that would have assertions about possibly infinite traces. Rather, nonterminating runs have been typically reasoned about in temporal logics like LTL and CTL\* or in interval temporal logic [14, 8]. These are however essentially different in spirit by their “endogeneity”: assertions are made about traces in a fixed transition system rather than traces of runs of different programs. Notably, however, interval temporal logic has connectives similar to ours—in fact they were a source of inspiration for our design.

Hofmann and Pavlova [10] consider a VDM-style logic with finite trace assertions that are applied to all finite prefixes of the trace of a possibly nonterminating run of a program. This logic allows reasoning about safety, but not liveness. We expect that we should be able to embed a logic like this in ours.

## 7. CONCLUSIONS

We have presented a sound and complete Hoare logic for the coinductive trace-based big-step semantics of While. The logic naturally extends both the standard state-based partial and total correctness Hoare logics. Its design may be exploratory at this stage—in the sense that one might wish to consider alternative choices of primitive connectives. We see our logic as a viable unifying foundational framework facilitating translations from more applied logics.

*Acknowledgements.* We are grateful to Martin Hofmann, Thierry Coquand and Adam Chlipala for discussions. Our anonymous reviewers provided very detailed and useful feedback.

## REFERENCES

- [1] Y. Bertot, P. Castéran. *Interactive Theorem Proving and Program Development: Coq'Art: The Calculus of Inductive Constructions, Texts in Theoretical Computer Science: An EATCS Series*. Springer, 2004.
- [2] S. Blazy, X. Leroy. Mechanized semantics for the Clight subset of the C language. *J. of Autom. Reasoning*, 43(3):263–288, 2009.
- [3] V. Capretta. General recursion via coinductive types. *Log. Methods in Comput. Sci.*, 1(2:1), 2005.
- [4] P. Cousot, R. Cousot. Inductive definitions, semantics and abstract interpretation. In *Conf. Record of 19th ACM SIGPLAN-SIGACT Symp. on Principles of Programming Languages, POPL '92 (Albuquerque, NM, Jan. 1992)*, pp. 83–94. ACM Press, 1992.
- [5] P. Cousot, R. Cousot. Bi-inductive structural semantics. *Inform. and Comput.*, 207(2):258–283, 2009.
- [6] N. A. Danielsson, T. Altenkirch. Subtyping, declaratively. In C. Bolduc, J. Desharnais, B. Ktari, eds., *Proc. of 10th Int. Conf. on Mathematics of Program Construction, MPC 2000 (Québec City, June 2010)*, vol. 6120 of *Lect. Notes in Comput. Sci.*, pp. 100–118. Springer, 2010.
- [7] S. Glesner. A proof calculus for natural semantics based on greatest fixed point semantics. In J. Knoop, G. C. Necula, W. Zimmermann, eds., *Proc. of 3rd Int. Wksh. on Compiler Optimization Meets Compiler Verification, COCV 2004 (Barcelona, Apr. 2004)*, vol. 132(1) of *Electron. Notes in Theor. Comput. Sci.*, pp. 73–93. Elsevier, 2004.
- [8] J. Halpern, Z. Manna, B. Moszkowski. A hardware semantics based on temporal intervals. In J. Díaz, ed., *Proc. of 10th Int. Coll. on Automata, Languages and Programming, ICALP '83 (Barcelona, July 1983)*, vol. 154 of *Lect. Notes in Comput. Sci.*, pp. 278–191. Springer, 1983.
- [9] I. Hasuo, B. Jacobs, A. Sokolova. Generic trace semantics via coinduction. *Log. Methods in Comput. Sci.*, 3(4:11), 2007.
- [10] M. Hofmann, M. Pavlova. Elimination of ghost variables in program logics. In G. Barthe, C. Fournet, eds., *Revised Selected Papers from 3rd Symp. on Trustworthy Global Computing, TGC 2007 (Sophia Antipolis, Nov. 2007)*, vol. 4912 of *Lect. Notes in Comput. Sci.*, pp. 1–20. Springer, 2008.
- [11] X. Leroy. Coinductive big-step operational semantics. In P. Sestoft, ed., *Proc. of 15th Europ. Symp. on Programming, ESOP 2006 (Vienna, March 2006)*, vol. 3924 of *Lect. Notes in Comput. Sci.*, pp. 54–68. Springer, 2006.
- [12] X. Leroy, H. Grall. Coinductive big-step operational semantics. *Inform. and Comput.*, 207(2):285–305, 2009.
- [13] T. Kleymann. Hoare logic and auxiliary variables. *Formal Asp. in Comput.* 11(5):541–566, 1999.
- [14] B. Moszkowski. A temporal logic for reasoning about hardware. *Computer*, 18(2):10–19, 1985.
- [15] K. Nakata, T. Uustalu. A Hoare logic for the coinductive trace-based big-step semantics of While. In A. D. Gordon, ed., *Proc. of 19th European Symp. on Programming, ESOP 2010 (Paphos, March 2010)*, vol. 6012 of *Lect. Notes in Comput. Sci.*, pp. 488–506. Springer, 2010.
- [16] K. Nakata, T. Uustalu. Resumptions, weak bisimilarity and big-step semantics for While with interactive I/O: an exercise in mixed induction-coinduction. In L. Aceto, P. Sobocinski, eds., *Proc. of 7th Wksh. on Structural Operational Semantics, SOS 2010 (Paris, Aug. 2010)*, vol. 32 of *Electron. Proc. in Theor. Comput. Sci.*, pp. 57–75. Open Publishing Assoc., 2010.
- [17] K. Nakata, T. Uustalu. Trace-based coinductive operational semantics for While: big-step and small-step, relational and functional styles. In S. Berghofer, T. Nipkow, C. Urban, M. Wenzel, eds., *Proc. of 22nd Int. Conf. on Theorem Proving in Higher Order Logics, TPHOLs 2009 (Munich, Aug. 2009)*, vol. 5674 of *Lect. Notes in Comput. Sci.*, pp. 375–390. Springer, 2009.
- [18] H. Nestra. Fractional semantics. In M. Johnson, V. Vene, eds., *Proc. of 11th Int. Conf. on Algebraic Methodology and Software Technology, AMAST 2006 (Kuressaare, July 2006)*, vol. 4019 of *Lect. Notes in Comput. Sci.*, pp. 278–292. Springer, 2006.
- [19] H. Nestra. Transfinite semantics in the form of greatest fixpoint. *J. of Log. and Algebr. Program.*, 78(7):574–593, 2009.
- [20] H. Riis Nielson, F. Nielson. *Semantics with Applications: A Formal Introduction*. Wiley, 1992.
- [21] J. Rutten. A note on coinduction and weak bisimilarity for While programs. *Theor. Inform. and Appl.*, 33(4–5):393–400, 1999.

APPENDIX A. STATE-BASED PARTIAL CORRECTNESS AND TOTAL CORRECTNESS HOARE LOGICS

The figures below give the rules of the standard, state-based partial correctness and total correctness Hoare logics in the form used in Section 4.

$$\begin{array}{c}
\frac{}{\{U[e/x]\} x := e \{U\}} \quad \frac{}{\{U\} \text{skip} \{U\}} \quad \frac{\{U\} s_0 \{V\} \quad \{V\} s_1 \{Z\}}{\{U\} s_0; s_1 \{Z\}} \\
\frac{\{e \wedge U\} s_t \{Z\} \quad \{\neg e \wedge U\} s_f \{Z\}}{\{U\} \text{if } e \text{ then } s_t \text{ else } s_f \{Z\}} \quad \frac{\{e \wedge I\} s_t \{I\}}{\{I\} \text{while } e \text{ do } s_t \{I \wedge \neg e\}} \\
\frac{\forall z. \{U\} s \{V\}}{\{\exists z. U\} s \{\exists z. V\}} \quad \frac{U \models U' \quad \{U'\} s \{Z'\} \quad Z' \models Z}{\{U\} s \{Z\}}
\end{array}$$

Figure 8: Inference rules of partial-correctness Hoare logic

$$\begin{array}{c}
\frac{}{\{U[e/x]\} x := e \{U\}} \quad \frac{}{\{U\} \text{skip} \{U\}} \quad \frac{\{U\} s_0 \{V\} \quad \{V\} s_1 \{Z\}}{\{U\} s_0; s_1 \{Z\}} \\
\frac{\{e \wedge U\} s_t \{Z\} \quad \{\neg e \wedge U\} s_f \{Z\}}{\{U\} \text{if } e \text{ then } s_t \text{ else } s_f \{Z\}} \\
\frac{\forall n : \text{nat} \{e \wedge I \wedge t = n\} s_t \{I \wedge t < n\}}{\{I \wedge t = m\} \text{while } e \text{ do } s_t \{I \wedge t \leq m \wedge \neg e\}} \text{ while-fun} \\
\text{alt.} \\
\frac{\forall n : \text{nat} \{e \wedge J n\} s_t \{\exists k. k < n \wedge J k\}}{\{J m\} \text{while } e \text{ do } s_t \{\exists k. k \leq m \wedge J k \wedge \neg e\}} \text{ while-rel} \\
\frac{\forall z. \{U\} s \{V\}}{\{\exists z. U\} s \{\exists z. V\}} \quad \frac{U \models U' \quad \{U'\} s \{Z'\} \quad Z' \models Z}{\{U\} s \{Z\}}
\end{array}$$

Figure 9: Inference rules of total-correctness Hoare logic