Evaluation Trees for Proposition Algebra

The Case for Free and Repetition-Proof Valuation Congruence

Jan A. Bergstra and Alban Ponse(✉)

Section Theory of Computer Science, Informatics Institute,
Faculty of Science, University of Amsterdam, Amsterdam, The Netherlands
{j.a.bergstra,a.ponse}@uva.nl
https://staff.fnwi.uva.nl/

Abstract. Proposition algebra is based on Hoare's conditional connective, which is a ternary connective comparable to if-then-else and used in the setting of propositional logic. Conditional statements are provided with a simple semantics that is based on evaluation trees and that characterizes so-called free valuation congruence: two conditional statements are free valuation congruent if, and only if, they have equal evaluation trees. Free valuation congruence is axiomatized by the four basic equational axioms of proposition algebra that define the conditional connective. A valuation congruence that is axiomatized in proposition algebra and that identifies more conditional statements than free valuation congruence is repetition-proof valuation congruence, which we characterize by a simple transformation on evaluation trees.

Keywords: Conditional composition · Evaluation tree · Proposition algebra · Short-circuit evaluation · Short-circuit logic

1 Introduction

In 1985, Hoare's paper *A couple of novelties in the propositional calculus* [12] was published. In this paper the ternary connective $\_ \triangleleft \_ \triangleright \_$ is introduced as the *conditional*.[1] A more common expression for a conditional statement

$$P \triangleleft Q \triangleright R$$

Dedicated to Ernst-Rüdiger Olderog on the occasion of his sixtieth birthday. Jan Bergstra recalls many discussions during various meetings as well as joint work with Ernst-Rüdiger and Jan Willem Klop on readies, failures, and chaos back in 1987. Alban Ponse has pleasant memories of the process of publishing [8], the *Selected Papers from the Workshop on Assertional Methods*, of which Ernst-Rüdiger, who was one of the invited speakers at this workshop (held at CWI in November 1992), is one of the guest editors. An extended version of this paper appeared as report [6].

[1] To be distinguished from Hoare's *conditional* introduced in his 1985 book on CSP [11] and in his well-known 1987 paper *Laws of Programming* [10] for expressions $P \triangleleft b \triangleright Q$ with P and Q denoting programs and b a Boolean expression.

R. Meyer et al. (Eds.): Olderog-Festschrift, LNCS 9360, pp. 44–61, 2015.
DOI: 10.1007/978-3-319-23506-6_6

Table 1. The set CP of equational axioms for free valuation congruence

$$x \triangleleft \mathsf{T} \triangleright y = x \tag{CP1}$$
$$x \triangleleft \mathsf{F} \triangleright y = y \tag{CP2}$$
$$\mathsf{T} \triangleleft x \triangleright \mathsf{F} = x \tag{CP3}$$
$$x \triangleleft (y \triangleleft z \triangleright u) \triangleright v = (x \triangleleft y \triangleright v) \triangleleft z \triangleright (x \triangleleft u \triangleright v) \tag{CP4}$$

is "`if` Q `then` P `else` R", but in order to reason systematically with conditional statements, a notation such as $P \triangleleft Q \triangleright R$ is preferable. In a conditional statement $P \triangleleft Q \triangleright R$, first Q is evaluated, and depending on that evaluation result, then either P or R is evaluated (and the other is not) and determines the final evaluation result. This evaluation strategy is reminiscent of *short-circuit* evaluation.[2] In [12], Hoare proves that propositional logic can be characterized by extending equational logic with eleven axioms on the conditional, some of which employ constants for the truth values *true* and *false*.

In 2011, we introduced *Proposition Algebra* in [4] as a general approach to the study of the conditional: we defined several *valuation congruences* and provided equational axiomatizations of these congruences. The most basic and least identifying valuation congruence is *free* valuation congruence, which is axiomatized by the axioms in Table 1, where we use constants T and F for the truth values *true* and *false*. These axioms stem from [12] and define the conditional as a primitive connective. We use the name CP (for Conditional Propositions) for this set of axioms. Interpreting a conditional statement as an if-then-else expression, axioms (CP1)-(CP3) are natural, and axiom (CP4) (distributivity) can be clarified by case analysis: if z evaluates to *true* and y as well, then x determines the result of evaluation; if z evaluates to *true* and y evaluates to *false*, then v determines the result of evaluation, and so on and so forth. A simple example, taken from [4], is the conditional statement that a pedestrian evaluates before crossing a road with two-way traffic driving on the right:

$$(\textit{look-left-and-check} \triangleleft \textit{look-right-and-check} \triangleright \mathsf{F}) \triangleleft \textit{look-left-and-check} \triangleright \mathsf{F}.$$

This statement requires one, or two, or three atomic evaluations and cannot be simplified to one that requires less.[3]

In Section 2 we characterize free valuation congruence with help of *evaluation trees*, which are simple binary trees proposed by Daan Staudt in [13] (that appeared in 2012). Given a conditional statement, its evaluation tree represents all possible consecutive atomic evaluations followed by the final evaluation result (comparable to a truth table in the case of propositional logic).

[2] Short-circuit evaluation denotes the semantics of binary propositional connectives in which the second argument is evaluated only if the first argument does not suffice to determine the value of the expression.

[3] Note that $\textit{look-left-and-check} \triangleleft (\textit{look-right-and-check} \triangleleft \textit{look-left-and-check} \triangleright \mathsf{F}) \triangleright \mathsf{F}$ prescribes by axioms (CP4) and (CP2) the same evaluation.

Two conditional statements are equivalent with respect to free valuation congruence if their evaluation trees are equal. Free valuation congruence identifies less than the equivalence defined by Hoare's axioms in [12]. For example, the atomic proposition a and the conditional statement $\mathsf{T} \triangleleft a \triangleright a$ are not equivalent with respect to free valuation congruence, although they are equivalent with respect to *static* valuation congruence, which is the valuation congruence that characterizes propositional logic.

A valuation congruence that identifies more than free and less than static valuation congruence is *repetition-proof* valuation congruence, which is axiomatized by CP extended with two (schematic) axioms, one of which reads

$$x \triangleleft a \triangleright (y \triangleleft a \triangleright z) = x \triangleleft a \triangleright (z \triangleleft a \triangleright z),$$

and thus expresses that if atomic proposition a evaluates to *false*, a consecutive evaluation of a also evaluates to *false*, so the conditional statement at the y-position will not be evaluated and can be replaced by any other. As an example, $\mathsf{T} \triangleleft a \triangleright a = \mathsf{T} \triangleleft a \triangleright (\mathsf{T} \triangleleft a \triangleright \mathsf{F}) = \mathsf{T} \triangleleft a \triangleright (\mathsf{F} \triangleleft a \triangleright \mathsf{F})$, and the left-hand and right-hand conditional statements are equivalent with respect to repetition-proof valuation congruence, but not with respect to free valuation congruence.

In Section 3 we characterize repetition-proof valuation congruence by defining a transformation on evaluation trees that yields *repetition-proof evaluation trees*: two conditional statements are equivalent with respect to repetition-proof valuation congruence if, and only if, they have equal repetition-proof evaluation trees. Although this transformation on evaluation trees is simple and natural, our proof of the mentioned characterization—which is phrased as a completeness result—is non-trivial and we could not find a proof that is essentially simpler.

In section 4 we discuss the general structure of the proof of this last result, which is based on normalization of conditional statements, and we conclude with a brief digression on *short-circuit logic* and an example on the use of repetition-proof valuation congruence.

The approach followed in this paper also works for most other valuation congruences defined in [4] and the case for repetition-proof valuation congruence is prototypical, as we show in [6].

2 Evaluation Trees for Free Valuation Congruence

Consider the signature $\Sigma_{\mathrm{CP}}(A) = \{\_ \triangleleft \_ \triangleright \_, \mathsf{T}, \mathsf{F}, a \mid a \in A\}$ with constants T and F for the truth values *true* and *false*, respectively, and constants a for atomic propositions, further called *atoms*, from some countable set A. We write

$$C_A$$

for the set of closed terms, or *conditional statements*, over the signature $\Sigma_{\mathrm{CP}}(A)$. Given a conditional statement $P \triangleleft Q \triangleright R$, we refer to Q as its *central condition*.

We define the *dual* P^d of $P \in C_A$ as follows:

$$\mathsf{T}^d = \mathsf{F}, \qquad a^d = a \quad (\text{for } a \in A),$$
$$\mathsf{F}^d = \mathsf{T}, \qquad (P \triangleleft Q \triangleright R)^d = R^d \triangleleft Q^d \triangleright P^d.$$

Observe that CP is a self-dual axiomatization: when defining $x^d = x$ for each variable x, the dual of each axiom is also in CP, and hence

$$\mathrm{CP} \vdash P = Q \iff \mathrm{CP} \vdash P^d = Q^d.$$

A natural view on conditional statements in C_A involves short-circuited evaluation, similar to how we consider the evaluation of an "`if` y `then` x `else` z" expression. The following definition is taken from [13].

Definition 2.1. *The set $\mathcal{T}_A$ of* ***evaluation trees over*** *A* ***with leaves in*** *$\{\mathsf{T},\mathsf{F}\}$ is defined inductively by*

$$\begin{aligned} \mathsf{T} &\in \mathcal{T}_A, \\ \mathsf{F} &\in \mathcal{T}_A, \\ (X \trianglelefteq a \trianglerighteq Y) &\in \mathcal{T}_A \textit{ for any } X, Y \in \mathcal{T}_A \textit{ and } a \in A. \end{aligned}$$

The function $\_ \trianglelefteq a \trianglerighteq \_$ is called ***post-conditional composition over*** *a. In the evaluation tree $X \trianglelefteq a \trianglerighteq Y$, the root is represented by a, the left branch by X and the right branch by Y.*

We refer to trees in $\mathcal{T}_A$ as evaluation trees, or trees for short. Post-conditional composition and its notation stem from [2]. Evaluation trees play a crucial role in the main results of [13]. In order to define our "evaluation tree semantics", we first define an auxiliary function on trees.

Definition 2.2. *Given evaluation trees $Y, Z \in \mathcal{T}_A$, the* ***leaf replacement*** *function $[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z] : \mathcal{T}_A \to \mathcal{T}_A$, for which post-fix notation*

$$X[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z]$$

is adopted, is defined as follows, where $a \in A$:

$$\begin{aligned} \mathsf{T}[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z] &= Y, \\ \mathsf{F}[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z] &= Z, \\ (X_1 \trianglelefteq a \trianglerighteq X_2)[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z] &= X_1[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z] \trianglelefteq a \trianglerighteq X_2[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z]. \end{aligned}$$

We note that the order in which the replacements of leaves of X is listed is irrelevant and we adopt the convention of not listing identities inside the brackets, e.g., $X[\mathsf{F} \mapsto Z] = X[\mathsf{T} \mapsto \mathsf{T}, \mathsf{F} \mapsto Z]$. Furthermore, repeated leaf replacements satisfy the following equation:

$$\begin{aligned} &\big(X[\mathsf{T} \mapsto Y_1, \mathsf{F} \mapsto Z_1]\big)\ [\mathsf{T} \mapsto Y_2, \mathsf{F} \mapsto Z_2] \\ &\qquad = X[\mathsf{T} \mapsto Y_1[\mathsf{T} \mapsto Y_2, \mathsf{F} \mapsto Z_2],\ \mathsf{F} \mapsto Z_1[\mathsf{T} \mapsto Y_2, \mathsf{F} \mapsto Z_2]]. \end{aligned}$$

We now have the terminology and notation to define the interpretation of conditional statements in C_A as evaluation trees by a function *se* (abbreviating short-circuit evaluation).

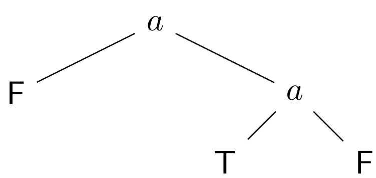
Definition 2.3. *The **short-circuit evaluation function** $se : C_A \to \mathcal{T}_A$ is defined as follows, where $a \in A$:*

$$\begin{aligned} se(\mathsf{T}) &= \mathsf{T}, \\ se(\mathsf{F}) &= \mathsf{F}, \\ se(a) &= \mathsf{T} \trianglelefteq a \trianglerighteq \mathsf{F}, \\ se(P \triangleleft Q \triangleright R) &= se(Q)[\mathsf{T} \mapsto se(P), \mathsf{F} \mapsto se(R)]. \end{aligned}$$

Example 2.4. *The conditional statement $a \triangleleft (\mathsf{F} \triangleleft a \triangleright \mathsf{T}) \triangleright \mathsf{F}$ yields the following evaluation tree:*

$$\begin{aligned} se(a \triangleleft (\mathsf{F} \triangleleft a \triangleright \mathsf{T}) \triangleright \mathsf{F}) &= se(\mathsf{F} \triangleleft a \triangleright \mathsf{T})[\mathsf{T} \mapsto se(a), \mathsf{F} \mapsto se(\mathsf{F})] \\ &= (\mathsf{F} \trianglelefteq a \trianglerighteq \mathsf{T})[\mathsf{T} \mapsto se(a)] \\ &= \mathsf{F} \trianglelefteq a \trianglerighteq (\mathsf{T} \trianglelefteq a \trianglerighteq \mathsf{F}). \end{aligned}$$

A more pictorial representation of this evaluation tree is the following, where $\trianglelefteq$ yields a left branch and $\trianglerighteq$ a right branch:



As we can see from the definition on atoms, evaluation continues in the left branch if an atom evaluates to *true* and in the right branch if it evaluates to *false*. We shall often use the constants T and F to denote the result of an evaluation (instead of *true* and *false*).

Definition 2.5. *Let $P \in C_A$. An **evaluation** of P is a pair (σ, B) where $\sigma \in (A\{\mathsf{T}, \mathsf{F}\})^*$ and $B \in \{\mathsf{T}, \mathsf{F}\}$, such that if $se(P) \in \{\mathsf{T}, \mathsf{F}\}$, then $\sigma = \epsilon$ (the empty string) and $B = se(P)$, and otherwise,*

$$\sigma = a_1 B_1 a_2 B_2 \cdots a_n B_n,$$

where $a_1 a_2 \cdots a_n B$ is a complete path in $se(P)$ and

- *for $i < n$, if a_{i+1} is a left child of a_i then $B_i = \mathsf{T}$, and otherwise $B_i = \mathsf{F}$,*
- *if B is a left child of a_n then $B_n = \mathsf{T}$, and otherwise $B_n = \mathsf{F}$.*

*We refer to σ as the **evaluation path** and to B as the **evaluation result**.*

So, an evaluation of a conditional statement P is a complete path in $se(P)$ (from root to leaf) and contains evaluation values for all occurring atoms. For instance, the evaluation tree $\mathsf{F} \trianglelefteq a \trianglerighteq (\mathsf{T} \trianglelefteq a \trianglerighteq \mathsf{F})$ from Example 2.4 encodes the evaluations $(a\mathsf{T}, \mathsf{F})$, $(a\mathsf{F}a\mathsf{T}, \mathsf{T})$, and $(a\mathsf{F}a\mathsf{F}, \mathsf{F})$. As an aside, we note that this particular evaluation tree encodes all possible evaluations of $\neg a$ `&&` a, where `&&` is the connective that prescribes *short-circuited conjunction* (we return to this connective in Section 4).

In turn, each evaluation tree gives rise to a *unique* conditional statement. For Example 2.4, this is $\mathsf{F} \triangleleft a \triangleright (\mathsf{T} \triangleleft a \triangleright \mathsf{F})$ (note the syntactical correspondence).

Definition 2.6. ***Basic forms over*** A *are defined by the following grammar*

$$t ::= \mathsf{T} \mid \mathsf{F} \mid t \triangleleft a \triangleright t \quad \textit{for } a \in A.$$

We write BF_A *for the set of basic forms over* A*. The* ***depth*** $d(P)$ *of* $P \in BF_A$ *is defined by* $d(\mathsf{T}) = d(\mathsf{F}) = 0$ *and* $d(Q \triangleleft a \triangleright R) = 1 + \max\{d(Q), d(R)\}$*.*

The following two lemmas exploit the structure of basic forms and are stepping stones to our first completeness result (Theorem 2.11).

Lemma 2.7. *For each* $P \in C_A$ *there exists* $Q \in BF_A$ *such that* $\mathrm{CP} \vdash P = Q$*.*

Proof. First we establish an auxiliary result: if P, Q, R are basic forms, then there is a basic form S such that $\mathrm{CP} \vdash P \triangleleft Q \triangleright R = S$. This follows by structural induction on Q.

The lemma's statement follows by structural induction on P. The base cases $P \in \{\mathsf{T}, \mathsf{F}, a \mid a \in A\}$ are trivial, and if $P = P_1 \triangleleft P_2 \triangleright P_3$ there exist by induction basic forms Q_i such that $\mathrm{CP} \vdash P_i = Q_i$, hence $\mathrm{CP} \vdash P_1 \triangleleft P_2 \triangleright P_3 = Q_1 \triangleleft Q_2 \triangleright Q_3$. Now apply the auxiliary result. □

Lemma 2.8. *For all basic forms* P *and* Q*,* $se(P) = se(Q)$ *implies* $P = Q$*.*

Proof. By structural induction on P. The base cases $P \in \{\mathsf{T}, \mathsf{F}\}$ are trivial. If $P = P_1 \triangleleft a \triangleright P_2$, then $Q \notin \{\mathsf{T}, \mathsf{F}\}$ and $Q \neq Q_1 \triangleleft b \triangleright Q_2$ with $b \neq a$, so $Q = Q_1 \triangleleft a \triangleright Q_2$ and $se(P_i) = se(Q_i)$. By induction we find $P_i = Q_i$, and hence $P = Q$. □

Definition 2.9. ***Free valuation congruence****, notation* $=_{se}$*, is defined on* C_A *as follows:*

$$P =_{se} Q \iff se(P) = se(Q).$$

Lemma 2.10. *Free valuation congruence is a congruence relation.*

Proof. Let $P, Q, R \in C_A$ and assume $P =_{se} P'$, thus $se(P) = se(P')$. Then $se(P \triangleleft Q \triangleright R) = se(Q)[\mathsf{T} \mapsto se(P), \mathsf{F} \mapsto se(R)] = se(Q)[\mathsf{T} \mapsto se(P'), \mathsf{F} \mapsto se(R)] = se(P' \triangleleft Q \triangleright R)$, and thus $P \triangleleft Q \triangleright R =_{se} P' \triangleleft Q \triangleright R$. The two remaining cases can be proved in a similar way. □

Theorem 2.11 (Completeness of CP). *For all* $P, Q \in C_A$*,*

$$\mathrm{CP} \vdash P = Q \iff P =_{se} Q.$$

Proof. We first prove ⇒. By Lemma 2.10, $=_{se}$ is a congruence relation and it easily follows that all CP-axioms are sound. For example, soundness of axiom (CP4) follows from

$$\begin{aligned}
&se(P \triangleleft (Q \triangleleft R \triangleright S) \triangleright U)\\
&\qquad = se(Q \triangleleft R \triangleright S)[\mathsf{T} \mapsto se(P), \mathsf{F} \mapsto se(U)]\\
&\qquad = \big(se(R)[\mathsf{T} \mapsto se(Q), \mathsf{F} \mapsto se(S)]\big)\,[\mathsf{T} \mapsto se(P), \mathsf{F} \mapsto se(U)]\\
&\qquad = se(R)[\mathsf{T} \mapsto se(Q)[\mathsf{T} \mapsto se(P), \mathsf{F} \mapsto se(U)],\\
&\qquad\qquad\qquad \mathsf{F} \mapsto se(S)[\mathsf{T} \mapsto se(P), \mathsf{F} \mapsto se(U)]]\\
&\qquad = se(R)[\mathsf{T} \mapsto se(P \triangleleft Q \triangleright U), \mathsf{F} \mapsto se(P \triangleleft S \triangleright U)]\\
&\qquad = se((P \triangleleft Q \triangleright U) \triangleleft R \triangleright (P \triangleleft S \triangleright U)).
\end{aligned}$$

In order to prove $\Leftarrow$, let $P =_{se} Q$. According to Lemma 2.7 there exist basic forms P' and Q' such that $\mathrm{CP} \vdash P = P'$ and $\mathrm{CP} \vdash Q = Q'$, so $\mathrm{CP} \vdash P' = Q'$. By soundness ($\Rightarrow$) we find $P' =_{se} Q'$, so by Lemma 2.8, $P' = Q'$. Hence, $\mathrm{CP} \vdash P = P' = Q' = Q$. □

A consequence of the above results is that for each $P \in C_A$ there is a *unique* basic form P' with $\mathrm{CP} \vdash P = P'$, and that for each basic form, its *se*-image has exactly the same syntactic structure (replacing $\triangleleft$ by $\trianglelefteq$, and $\triangleright$ by $\trianglerighteq$). In the remainder of this section, we make this precise.

Definition 2.12. *The* ***basic form function*** $\mathit{bf} : C_A \to BF_A$ *is defined as follows, where* $a \in A$*:*

$$\begin{aligned}
\mathit{bf}(\mathsf{T}) &= \mathsf{T},\\
\mathit{bf}(\mathsf{F}) &= \mathsf{F},\\
\mathit{bf}(a) &= \mathsf{T} \triangleleft a \triangleright \mathsf{F},\\
\mathit{bf}(P \triangleleft Q \triangleright R) &= \mathit{bf}(Q)[\mathsf{T} \mapsto \mathit{bf}(P), \mathsf{F} \mapsto \mathit{bf}(R)].
\end{aligned}$$

Given $Q, R \in BF_A$*, the auxiliary function* $[\mathsf{T} \mapsto Q, \mathsf{F} \mapsto R] : BF_A \to BF_A$ *for which post-fix notation* $P[\mathsf{T} \mapsto Q, \mathsf{F} \mapsto R]$ *is adopted, is defined as follows:*

$$\begin{aligned}
\mathsf{T}[\mathsf{T} \mapsto Q, \mathsf{F} \mapsto R] &= Q,\\
\mathsf{F}[\mathsf{T} \mapsto Q, \mathsf{F} \mapsto R] &= R,\\
(P_1 \triangleleft a \triangleright P_2)[\mathsf{T} \mapsto Q, \mathsf{F} \mapsto R] &= P_1[\mathsf{T} \mapsto Q, \mathsf{F} \mapsto R] \triangleleft a \triangleright P_2[\mathsf{T} \mapsto Q, \mathsf{F} \mapsto R].
\end{aligned}$$

(The notational overloading with the leaf replacement function on evaluation trees is harmless).

So, for given $Q, R \in BF_A$, the auxiliary function $[\mathsf{T} \mapsto Q, \mathsf{F} \mapsto R]$ applied to $P \in BF_A$ (thus, $P[\mathsf{T} \mapsto Q, \mathsf{F} \mapsto R]$) replaces all T-occurrences in P by Q, and all F-occurrences in P by R. The following two lemmas imply that bf is a normalization function.

Lemma 2.13. *For all* $P \in C_A$*,* $\mathit{bf}(P)$ *is a basic form.*

Proof. By structural induction. The base cases are trivial. For the inductive case we find $\mathit{bf}(P \triangleleft Q \triangleright R) = \mathit{bf}(Q)[\mathsf{T} \mapsto \mathit{bf}(P), \mathsf{F} \mapsto \mathit{bf}(R)]$, so by induction, $\mathit{bf}(P)$, $\mathit{bf}(Q)$, and $\mathit{bf}(R)$ are basic forms. Furthermore, replacing all T-occurrences and F-occurrences in $\mathit{bf}(Q)$ by basic forms $\mathit{bf}(P)$ and $\mathit{bf}(R)$, respectively, yields a basic form. □

Lemma 2.14. *For each basic form* P*,* $\mathit{bf}(P) = P$*.*

Proof. By structural induction on P. □

Definition 2.15. *The binary relation* $=_{\mathit{bf}}$ *on* C_A *is defined as follows:*

$$P =_{\mathit{bf}} Q \iff \mathit{bf}(P) = \mathit{bf}(Q).$$

Lemma 2.16. *The relation* $=_{bf}$ *is a congruence relation.*

Proof. Let $P, Q, R \in C_A$ and assume $P =_{bf} P'$, thus $bf(P) = bf(P')$. Then $bf(P \triangleleft Q \triangleright R) = bf(Q)[\mathsf{T} \mapsto bf(P), \mathsf{F} \mapsto bf(R)] = bf(Q)[\mathsf{T} \mapsto bf(P'), \mathsf{F} \mapsto bf(R)] = bf(P' \triangleleft Q \triangleright R)$, and thus $P \triangleleft Q \triangleright R =_{bf} P' \triangleleft Q \triangleright R$. The two remaining cases can be proved in a similar way. □

Before proving that CP is an axiomatization of the relation $=_{bf}$, we show that each instance of the axiom (CP4) satisfies $=_{bf}$.

Lemma 2.17. *For all* $P, P_1, P_2, Q_1, Q_2 \in C_A$,

$$bf(Q_1 \triangleleft (P_1 \triangleleft P \triangleright P_2) \triangleright Q_2) = bf((Q_1 \triangleleft P_1 \triangleright Q_2) \triangleleft P \triangleright (Q_1 \triangleleft P_2 \triangleright Q_2)).$$

Proof. By definition, the lemma's statement is equivalent with

$$\begin{aligned}\big(bf(P)[\mathsf{T} \mapsto bf(P_1), \mathsf{F} \mapsto bf(P_2)]\big)\ &[\mathsf{T} \mapsto bf(Q_1), \mathsf{F} \mapsto bf(Q_2)] \\ &= bf(P)[\mathsf{T} \mapsto bf(Q_1 \triangleleft P_1 \triangleright Q_2), \mathsf{F} \mapsto bf(Q_1 \triangleleft P_2 \triangleright Q_2)].\end{aligned} \tag{1}$$

By Lemma 2.13, $bf(P)$, $bf(P_i)$, and $bf(Q_i)$ are basic forms. We prove (1) by structural induction on the form that $bf(P)$ can have. If $bf(P) = \mathsf{T}$, then

$$\begin{aligned}\big(\mathsf{T}[\mathsf{T} \mapsto bf(P_1), \mathsf{F} \mapsto bf(P_2)]\big)\ &[\mathsf{T} \mapsto bf(Q_1), \mathsf{F} \mapsto bf(Q_2)] \\ &= bf(P_1)[\mathsf{T} \mapsto bf(Q_1), \mathsf{F} \mapsto bf(Q_2)]\end{aligned}$$

and

$$\begin{aligned}\mathsf{T}[\mathsf{T} \mapsto bf(Q_1 \triangleleft P_1 \triangleright Q_2), \mathsf{F} \mapsto &bf(Q_1 \triangleleft P_2 \triangleright Q_2)] \\ &= bf(Q_1 \triangleleft P_1 \triangleright Q_2) \\ &= bf(P_1)[\mathsf{T} \mapsto bf(Q_1), \mathsf{F} \mapsto bf(Q_2)].\end{aligned}$$

If $bf(P) = \mathsf{F}$, then equation (1) follows in a similar way.

The inductive case $bf(P) = R_1 \triangleleft a \triangleright R_2$ is trivial (by definition of the last defining clause of the auxiliary functions $[\mathsf{T} \mapsto Q, \mathsf{F} \mapsto R]$ in Definition 2.12). □

Theorem 2.18. *For all* $P, Q \in C_A$, $\mathrm{CP} \vdash P = Q \iff P =_{bf} Q$.

Proof. We first prove $\Rightarrow$. By Lemma 2.16, $=_{bf}$ is a congruence relation and it easily follows that arbitrary instances of the CP-axioms (CP1)-(CP3) satisfy $=_{bf}$. By Lemma 2.17 it follows that arbitrary instances of axiom (CP4) also satisfy $=_{bf}$.

In order to prove $\Leftarrow$, assume $P =_{bf} Q$. According to Lemma 2.7, there exist basic forms P' and Q' such that $\mathrm{CP} \vdash P = P'$ and $\mathrm{CP} \vdash Q = Q'$, so $\mathrm{CP} \vdash P' = Q'$. By $\Rightarrow$ it follows that $P' =_{bf} Q'$, which implies by Lemma 2.14 that $P' = Q'$. Hence, $\mathrm{CP} \vdash P = P' = Q' = Q$. □

Corollary 2.19. *For all* $P \in C_A$, $P =_{bf} bf(P)$ *and* $P =_{se} bf(P)$.

Proof. By Lemma 2.13 and Lemma 2.14, $bf(P) = bf(bf(P))$, thus $P =_{bf} bf(P)$. By Theorem 2.18, $\mathrm{CP} \vdash P = bf(P)$, and by Theorem 2.11, $P =_{se} bf(P)$. □

3 Evaluation Trees for Repetition-proof Valuation Congruence

In [4] we defined *repetition-proof* CP as the extension of the axiom set CP with the following two axiom schemes, where a ranges over A:

$$(x \lhd a \rhd y) \lhd a \rhd z = (x \lhd a \rhd x) \lhd a \rhd z, \tag{CPrp1}$$

$$x \lhd a \rhd (y \lhd a \rhd z) = x \lhd a \rhd (z \lhd a \rhd z). \tag{CPrp2}$$

We write $\mathrm{CP}_{rp}(A)$ for this extension. These axiom schemes characterize that for each atom a, a consecutive evaluation of a yields the same result, so in both cases the conditional statement at the y-position will not be evaluated and can be replaced by any other. Note that (CPrp1) and (CPrp2) are each others dual.

We define a proper subset of basic forms with the property that each conditional statement can be proved equal to such a basic form.

Definition 3.1. ***Rp-basic forms*** *are inductively defined:*

- T *and* F *are rp-basic forms, and*
- $P_1 \lhd a \rhd P_2$ *is an rp-basic form if* P_1 *and* P_2 *are rp-basic forms, and if* P_i *is not equal to* T *or* F*, then either the central condition in* P_i *is different from* a*, or* P_i *is of the form* $Q_i \lhd a \rhd Q_i$*.*

It will turn out useful to define a function that transforms conditional statements into rp-basic forms and that is comparable to the function bf.

Definition 3.2. *The* ***rp-basic form function*** $rpbf : C_A \to C_A$ *is defined by*

$$rpbf(P) = rpf(bf(P)).$$

The auxiliary function $rpf : BF_A \to BF_A$ *is defined as follows:*

$$\begin{aligned} rpf(\mathsf{T}) &= \mathsf{T}, \\ rpf(\mathsf{F}) &= \mathsf{F}, \\ rpf(P \lhd a \rhd Q) &= rpf(f_a(P)) \lhd a \rhd rpf(g_a(Q)). \end{aligned}$$

For $a \in A$*, the auxiliary functions* $f_a : BF_A \to BF_A$ *and* $g_a : BF_A \to BF_A$ *are defined by*

$$\begin{aligned} f_a(\mathsf{T}) &= \mathsf{T}, \\ f_a(\mathsf{F}) &= \mathsf{F}, \\ f_a(P \lhd b \rhd Q) &= \begin{cases} f_a(P) \lhd a \rhd f_a(P) & \text{if } b = a, \\ P \lhd b \rhd Q & \text{otherwise,} \end{cases} \end{aligned}$$

and

$$\begin{aligned} g_a(\mathsf{T}) &= \mathsf{T}, \\ g_a(\mathsf{F}) &= \mathsf{F}, \\ g_a(P \lhd b \rhd Q) &= \begin{cases} g_a(Q) \lhd a \rhd g_a(Q) & \text{if } b = a, \\ P \lhd b \rhd Q & \text{otherwise.} \end{cases} \end{aligned}$$

Thus, *rpbf* maps a conditional statement P to $bf(P)$ and then transforms $bf(P)$ according to the auxiliary functions rpf, f_a, and g_a.

Lemma 3.3. *For all $a \in A$ and $P \in BF_A$, $g_a(f_a(P)) = f_a(f_a(P)) = f_a(P)$ and $f_a(g_a(P)) = g_a(g_a(P)) = g_a(P)$.*

Proof. By structural induction on P. The base cases $P \in \{\mathsf{T}, \mathsf{F}\}$ are trivial. For the inductive case $P = Q \triangleleft b \triangleright R$ we have to distinguish the cases $b = a$ and $b \neq a$. If $b = a$, then

$$\begin{aligned} g_a(f_a(Q \triangleleft a \triangleright R)) &= g_a(f_a(Q)) \triangleleft a \triangleright g_a(f_a(Q)) \\ &= f_a(Q) \triangleleft a \triangleright f_a(Q) && \text{by IH} \\ &= f_a(Q \triangleleft a \triangleright R), \end{aligned}$$

and $f_a(f_a(Q \triangleleft a \triangleright R)) = f_a(Q \triangleleft a \triangleright R)$ follows in a similar way. If $b \neq a$, then $f_a(P) = g_a(P) = P$, and hence $g_a(f_a(P)) = f_a(f_a(P)) = f_a(P)$.

The second pair of equalities can be derived in a similar way. □

In order to prove that for all $P \in C_A$, $rpbf(P)$ is an rp-basic form, we use the following auxiliary lemma.

Lemma 3.4. *For all $a \in A$ and $P \in BF_A$, $d(P) \geq d(f_a(P))$ and $d(P) \geq d(g_a(P))$.*

Proof. Fix some $a \in A$. We prove these inequalities by structural induction on P. The base cases $P \in \{\mathsf{T}, \mathsf{F}\}$ are trivial. For the inductive case $P = Q \triangleleft b \triangleright R$ we have to distinguish the cases $b = a$ and $b \neq a$. If $b = a$, then

$$\begin{aligned} d(Q \triangleleft a \triangleright R) &= 1 + \max\{d(Q), d(R)\} \\ &\geq 1 + d(Q) \\ &\geq 1 + d(f_a(Q)) && \text{by IH} \\ &= d(f_a(Q) \triangleleft a \triangleright f_a(Q)) \\ &= d(f_a(Q \triangleleft a \triangleright R)), \end{aligned}$$

and $d(Q \triangleleft a \triangleright R) \geq d(g_a(Q \triangleleft a \triangleright R))$ follows in a similar way.

If $b \neq a$, then $f_a(P) = g_a(P) = P$, and hence $d(P) \geq d(f_a(P))$ and $d(P) \geq d(g_a(P))$. □

Lemma 3.5. *For all $P \in C_A$, $rpbf(P)$ is an rp-basic form.*

Proof. We first prove an auxiliary result:

$$\text{For all } P \in BF_A,\ rpf(P) \text{ is an rp-basic form.} \tag{2}$$

This follows by induction on the depth $d(P)$ of P. If $d(P) = 0$, then $P \in \{\mathsf{T}, \mathsf{F}\}$, and hence $rpf(P) = P$ is an rp-basic form. For the inductive case $d(P) = n + 1$ it must be the case that $P = Q \triangleleft a \triangleright R$. We find

$$rpf(Q \triangleleft a \triangleright R) = rpf(f_a(Q)) \triangleleft a \triangleright rpf(g_a(R)),$$

which is an rp-basic form because

- by Lemma 3.4, $f_a(Q)$ and $g_a(R)$ are basic forms with depth smaller than or equal to n, so by the induction hypothesis, $rpf(f_a(Q))$ and $rpf(g_a(R))$ are rp-basic forms,
- $rpf(f_a(Q))$ and $rpf(g_a(R))$ both satisfy the following property: if the central condition (if present) is a, then the outer arguments are equal. We show this first for $rpf(f_a(Q))$ by a case distinction on the form of Q:
  1. If $Q \in \{\mathsf{T}, \mathsf{F}\}$, then $rpf(f_a(Q)) = Q$, so there is nothing to prove.
  2. If $Q = Q_1 \triangleleft a \triangleright Q_2$, then $f_a(Q) = f_a(Q_1) \triangleleft a \triangleright f_a(Q_1)$ and thus by Lemma 3.3, $rpf(f_a(Q)) = rpf(f_a(Q_1)) \triangleleft a \triangleright rpf(f_a(Q_1))$.
  3. If $Q = Q_1 \triangleleft b \triangleright Q_2$ with $b \neq a$, then $f_a(Q) = Q_1 \triangleleft b \triangleright Q_2$ and thus $rpf(f_a(Q)) = rpf(f_b(Q_1)) \triangleleft b \triangleright rpf(g_b(Q_2))$, so there is nothing to prove.

  The fact that $rpf(g_a(R))$ satisfies this property follows in a similar way.

This finishes the proof of auxiliary result (2).

The lemma's statement now follows by structural induction: the base cases (comprising a single atom a) are again trivial, and for the inductive case,

$$rpbf(P \triangleleft Q \triangleright R) = rpf(bf(P \triangleleft Q \triangleright R)) = rpf(S)$$

for some basic form S by Lemma 2.13, and by auxiliary result (2), $rpf(S)$ is an rp-basic form. □

The following, rather technical result is used in Proposition 3.7 and Lemma 3.8.

Lemma 3.6. *If $Q \triangleleft a \triangleright R$ is an rp-basic form, then $Q = rpf(Q) = rpf(f_a(Q))$ and $R = rpf(R) = rpf(g_a(R))$.*

Proof. We first prove an auxiliary result:

$$\text{If } Q \triangleleft a \triangleright R \text{ is an rp-basic form, then } f_a(Q) = g_a(Q) \text{ and } f_a(R) = g_a(R). \tag{3}$$

We prove both equalities by simultaneous induction on the structure of Q and R. The base case, thus $Q, R \in \{\mathsf{T}, \mathsf{F}\}$, is trivial. If $Q = Q_1 \triangleleft a \triangleright Q_1$ and $R = R_1 \triangleleft a \triangleright R_1$, then Q and R are rp-basic forms with central condition a, so

$$\begin{aligned} f_a(Q) &= f_a(Q_1) \triangleleft a \triangleright f_a(Q_1) \\ &= g_a(Q_1) \triangleleft a \triangleright g_a(Q_1) && \text{by IH} \\ &= g_a(Q), \end{aligned}$$

and the equality for R follows in a similar way. If $Q = Q_1 \triangleleft a \triangleright Q_1$ and $R \neq R_1 \triangleleft a \triangleright R_1$, then $f_a(R) = g_a(R) = R$, and the result follows as above. All remaining cases follow in a similar way, which finishes the proof of (3).

We now prove the lemma's statement by simultaneous induction on the structure of Q and R. The base case, thus $Q, R \in \{\mathsf{T}, \mathsf{F}\}$, is again trivial. If $Q = Q_1 \triangleleft a \triangleright Q_1$ and $R = R_1 \triangleleft a \triangleright R_1$, then by auxiliary result (3),

$$rpf(Q) = rpf(f_a(Q_1)) \triangleleft a \triangleright rpf(f_a(Q_1)),$$

and by induction, $Q_1 = rpf(Q_1) = rpf(f_a(Q_1))$. Hence, $rpf(Q) = Q_1 \triangleleft a \triangleright Q_1$, and

$$\begin{aligned} rpf(f_a(Q)) &= rpf(f_a(f_a(Q_1))) \triangleleft a \triangleright rpf(g_a(f_a(Q_1))) \\ &= rpf(f_a(Q_1)) \triangleleft a \triangleright rpf(f_a(Q_1)) && \text{by Lemma 3.3} \\ &= Q_1 \triangleleft a \triangleright Q_1, \end{aligned}$$

and the equalities for R follow in a similar way.

If $Q = Q_1 \triangleleft a \triangleright Q_1$ and $R \neq R_1 \triangleleft a \triangleright R_1$, the lemma's equalities follow in a similar way, although a bit simpler because $g_a(R) = f_a(R) = R$.

For all remaining cases, the lemma's equalities follow in a similar way. □

Proposition 3.7 ($rpbf$ is a normalization function). *For all $P \in C_A$, $rpbf(P)$ is an rp-basic form, and for each rp-basic form P, $rpbf(P) = P$.*

Proof. The first statement is Lemma 3.5. For the second statement, it suffices by Lemma 2.14 to prove that for each rp-basic form P, $rpf(P) = P$. This follows by case distinction on P. The cases $P \in \{\mathsf{T}, \mathsf{F}\}$ follow immediately, and otherwise $P = P_1 \triangleleft a \triangleright P_2$, and thus $rpf(P) = rpf(f_a(P_1)) \triangleleft a \triangleright rpf(g_a(P_2))$. By Lemma 3.6, $rpf(f_a(P_1)) = P_1$ and $rpf(g_a(P_2)) = P_2$, hence $rpf(P) = P$. □

Lemma 3.8. *For all $P \in BF_A$, $\mathrm{CP}_{rp}(A) \vdash P = rpf(P)$.*

Proof. We apply structural induction on P. The base cases $P \in \{\mathsf{T}, \mathsf{F}\}$ are trivial. Assume $P = P_1 \triangleleft a \triangleright P_2$. By induction $\mathrm{CP}_{rp}(A) \vdash P_i = rpf(P_i)$. We proceed by a case distinction on the form that P_1 and P_2 can have:

1. If $P_i \in \{\mathsf{T}, \mathsf{F}, Q_i \triangleleft b_i \triangleright Q_i'\}$ with $b_i \neq a$, then $f_a(P_1) = P_1$ and $g_a(P_2) = P_2$, and hence $rpf(P) = rpf(P_1) \triangleleft a \triangleright rpf(P_2)$, and thus $\mathrm{CP}_{rp}(A) \vdash P = rpf(P)$.
2. If $P_1 = R_1 \triangleleft a \triangleright R_2$ and $P_2 \in \{\mathsf{T}, \mathsf{F}, Q' \triangleleft b \triangleright Q''\}$ with $b \neq a$, then $g_a(P_2) = P_2$ and by auxiliary result (2) in the proof of Lemma 3.5, $rpf(R_1)$ and $rpf(P_2)$ are rp-basic forms. We derive

$$\begin{aligned} \mathrm{CP}_{rp}(A) \vdash P &= (R_1 \triangleleft a \triangleright R_2) \triangleleft a \triangleright P_2 \\ &= (R_1 \triangleleft a \triangleright R_1) \triangleleft a \triangleright P_2 && \text{by (CPrp1)} \\ &= (rpf(R_1) \triangleleft a \triangleright rpf(R_1)) \triangleleft a \triangleright rpf(P_2) && \text{by IH} \\ &= (rpf(f_a(R_1)) \triangleleft a \triangleright rpf(f_a(R_1))) \triangleleft a \triangleright rpf(g_a(P_2)) && \text{by Lemma 3.6} \\ &= rpf(f_a(R_1 \triangleleft a \triangleright R_2)) \triangleleft a \triangleright rpf(g_a(P_2)) \\ &= rpf((R_1 \triangleleft a \triangleright R_2) \triangleleft a \triangleright P_2) \\ &= rpf(P). \end{aligned}$$

3. If $P_1 \in \{\mathsf{T}, \mathsf{F}, Q' \triangleleft b \triangleright Q''\}$ with $b \neq a$ and $P_2 = S_1 \triangleleft a \triangleright S_2$, we can proceed as in the previous case, but now using axiom scheme (CPrp2) and the identity $f_a(P_1) = P_1$, and the fact that $rpf(P_1)$ and $rpf(S_2)$ are rp-basic forms.
4. If $P_1 = R_1 \triangleleft a \triangleright R_2$ and $P_2 = S_1 \triangleleft a \triangleright S_2$, we can proceed as in two previous cases, now using both (CPrp1) and (CPrp2), and the fact that $rpf(R_1)$ and $rpf(S_2)$ are rp-basic forms.

□

Theorem 3.9. *For all $P \in C_A$, $\mathrm{CP}_{rp}(A) \vdash P = rpbf(P)$.*

Proof. By Theorem 2.18 and Corollary 2.19 we find $\mathrm{CP}_{rp}(A) \vdash P = bf(P)$. By Lemma 3.8, $\mathrm{CP}_{rp}(A) \vdash bf(P) = rpf(bf(P))$, and $rpf(bf(P)) = rpbf(P)$. □

Definition 3.10. *The binary relation $=_{rpbf}$ on C_A is defined as follows:*

$$P =_{rpbf} Q \iff rpbf(P) = rpbf(Q).$$

Theorem 3.11. *For all $P, Q \in C_A$, $\mathrm{CP}_{rp}(A) \vdash P = Q \iff P =_{rpbf} Q$.*

Proof. Assume $\mathrm{CP}_{rp}(A) \vdash P = Q$. By Theorem 3.9, $\mathrm{CP}_{rp}(A) \vdash rpbf(P) = rpbf(Q)$. In [4] the following two statements are proved (Theorem 6.3 and an auxiliary result in its proof), where $=_{rpf}$ is a binary relation on C_A:

1. For all $P, Q \in C_A$, $\mathrm{CP}_{rp}(A) \vdash P = Q \iff P =_{rpf} Q$.
2. For all rp-basic forms P and Q, $P =_{rpf} Q \Rightarrow P = Q$.

By Lemma 3.5 these statements imply $rpbf(P) = rpbf(Q)$, that is, $P =_{rpbf} Q$.

Assume $P =_{rpbf} Q$. By Lemma 2.14, $bf(rpbf(P)) = bf(rpbf(Q))$. By Theorem 2.18, $\mathrm{CP} \vdash rpbf(P) = rpbf(Q)$. By Theorem 3.9, $\mathrm{CP}_{rp}(A) \vdash P = Q$. □

So, the relation $=_{rpbf}$ is axiomatized by $\mathrm{CP}_{rp}(A)$ and is thus a congruence. With this observation in mind, we define a transformation on evaluation trees that mimics the function $rpbf$ and prove that equality of two such transformed trees characterizes the congruence that is axiomatized by $\mathrm{CP}_{rp}(A)$.

Definition 3.12. *The unary* ***repetition-proof evaluation function***

$$rpse : C_A \to \mathcal{T}_A$$

yields ***repetition-proof evaluation trees*** *and is defined by*

$$rpse(P) = rp(se(P)).$$

The auxiliary function $rp : \mathcal{T}_A \to \mathcal{T}_A$ is defined as follows ($a \in A$):

$$\begin{aligned} rp(\mathsf{T}) &= \mathsf{T},\\ rp(\mathsf{F}) &= \mathsf{F},\\ rp(X \trianglelefteq a \trianglerighteq Y) &= rp(F_a(X)) \trianglelefteq a \trianglerighteq rp(G_a(Y)). \end{aligned}$$

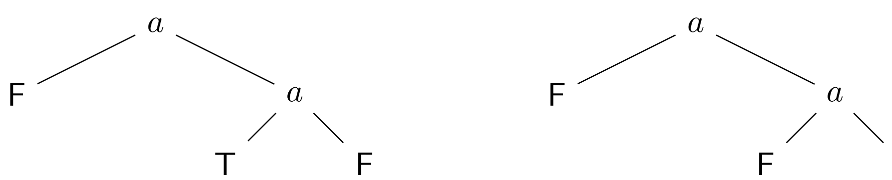
For $a \in A$, the auxiliary functions $F_a : \mathcal{T}_A \to \mathcal{T}_A$ and $G_a : \mathcal{T}_A \to \mathcal{T}_A$ are defined by

$$\begin{aligned} F_a(\mathsf{T}) &= \mathsf{T},\\ F_a(\mathsf{F}) &= \mathsf{F},\\ F_a(X \trianglelefteq b \trianglerighteq Y) &= \begin{cases} F_a(X) \trianglelefteq a \trianglerighteq F_a(X) & \text{if } b = a,\\ X \trianglelefteq b \trianglerighteq Y & \text{otherwise,} \end{cases} \end{aligned}$$

and

$$\begin{aligned}
G_a(\mathsf{T}) &= \mathsf{T},\\
G_a(\mathsf{F}) &= \mathsf{F},\\
G_a(X \trianglelefteq b \trianglerighteq Y) &= \begin{cases} G_a(Y) \trianglelefteq a \trianglerighteq G_a(Y) & \text{if } b = a,\\ X \trianglelefteq b \trianglerighteq Y & \text{otherwise.}\end{cases}
\end{aligned}$$

Example 3.13. *Let* $P = a \triangleleft (\mathsf{F} \triangleleft a \triangleright \mathsf{T}) \triangleright \mathsf{F}$. *We depict* $se(P)$ *(as in Example 2.4) and the repetition-proof evaluation tree* $rpse(P) = \mathsf{F} \trianglelefteq a \trianglerighteq (\mathsf{F} \trianglelefteq a \trianglerighteq \mathsf{F})$*:*



The similarities between *rpse* and the function *rpbf* can be exploited:

Lemma 3.14. *For all* $a \in A$ *and* $X \in \mathcal{T}_A$, $G_a(F_a(X)) = F_a(F_a(X)) = F_a(X)$ *and* $F_a(G_a(X)) = G_a(G_a(X)) = G_a(X)$.

Proof. By structural induction on X (cf. the proof of Lemma 3.3). □

We use the following lemma in the proof of our final completeness result.

Lemma 3.15. *For all* $P \in BF_A$, $rp(se(P)) = se(rpf(P))$.

Proof. We first prove an auxiliary result:

$$\begin{aligned}&\text{For all } P \in BF_A \text{ and for all } a \in A,\ \ rp(F_a(se(P))) = se(rpf(f_a(P)))\\ &\text{and }\ rp(G_a(se(P))) = se(rpf(g_a(P))).\end{aligned} \tag{4}$$

We prove the first equality of (4) by structural induction on P. The base cases $P \in \{\mathsf{T}, \mathsf{F}\}$ are trivial. For the inductive case $P = Q \triangleleft a \triangleright R$, let $b \in A$. We have to distinguish the cases $b = a$ and $b \neq a$. If $b = a$, then

$$\begin{aligned}
&rp(F_a(se(Q \triangleleft a \triangleright R)))\\
&\quad = rp(F_a(se(Q) \trianglelefteq a \trianglerighteq se(R)))\\
&\quad = rp(F_a(se(Q)) \trianglelefteq a \trianglerighteq F_a(se(Q)))\\
&\quad = rp(F_a(F_a(se(Q)))) \trianglelefteq a \trianglerighteq rp(G_a(F_a(se(Q))))\\
&\quad = rp(F_a(se(Q))) \trianglelefteq a \trianglerighteq rp(F_a(se(Q))) && \text{by Lemma 3.14}\\
&\quad = se(rpf(f_a(Q))) \trianglelefteq a \trianglerighteq se(rpf(f_a(Q))) && \text{by IH}\\
&\quad = se(rpf(f_a(Q)) \triangleleft a \triangleright rpf(f_a(Q)))\\
&\quad = se(rpf(f_a(f_a(Q))) \triangleleft a \triangleright rpf(g_a(f_a(Q)))) && \text{by Lemma 3.3}\\
&\quad = se(rpf(f_a(Q \triangleleft a \triangleright f_a(Q))))\\
&\quad = se(rpf(f_a(Q \triangleleft a \triangleright R))).
\end{aligned}$$

If $b \neq a$, then

$$\begin{aligned} rp(F_b(se(Q \triangleleft a \triangleright R))) &= rp(F_b(se(Q) \trianglelefteq a \trianglerighteq se(R))) \\ &= rp(se(Q) \trianglelefteq a \trianglerighteq se(R)) \\ &= rp(F_a(se(Q))) \trianglelefteq a \trianglerighteq rp(G_a(se(R))) \\ &= se(rpf(f_a(Q))) \trianglelefteq a \trianglerighteq se(rpf(g_a(R))) \qquad \text{by IH} \\ &= se(rpf(f_a(Q)) \triangleleft a \triangleright rpf(g_a(R))) \\ &= se(rpf(Q \triangleleft a \triangleright R)) \\ &= se(rpf(f_b(Q \triangleleft a \triangleright R))). \end{aligned}$$

The second equality can be proved in a similar way, and this finishes the proof of (4).

The lemma's statement now follows by a case distinction on P. The cases $P \in \{\mathsf{T}, \mathsf{F}\}$ follow immediately, and otherwise $P = Q \triangleleft a \triangleright R$, and thus

$$\begin{aligned} rp(se(Q \triangleleft a \triangleright R)) &= rp(se(Q) \trianglelefteq a \trianglerighteq se(R)) \\ &= rp(F_a(se(Q))) \trianglelefteq a \trianglerighteq rp(G_a(se(R))) \\ &= se(rpf(f_a(Q))) \trianglelefteq a \trianglerighteq se(rpf(g_a(R))) \qquad \text{by (4)} \\ &= se(rpf(f_a(Q)) \triangleleft a \triangleright rpf(g_a(R))) \\ &= se(rpf(Q \triangleleft a \triangleright R)). \end{aligned}$$

□

Finally, we relate conditional statements by means of their repetition-proof evaluation trees.

Definition 3.16. ***Repetition-proof valuation congruence***, *notation* $=_{rpse}$, *is defined on* C_A *as follows:*

$$P =_{rpse} Q \iff rpse(P) = rpse(Q).$$

The following characterization result immediately implies that $=_{rpse}$ is a congruence relation on C_A (and hence justifies calling it a congruence).

Proposition 3.17. *For all* $P, Q \in C_A$, $P =_{rpse} Q \iff P =_{rpbf} Q$.

Proof. In order to prove $\Rightarrow$, assume $rpse(P) = rpse(Q)$, thus $rp(se(P)) = rp(se(Q))$. By Corollary 2.19,

$$rp(se(bf(P))) = rp(se(bf(Q))),$$

so by Lemma 3.15, $se(rpf(bf(P))) = se(rpf(bf(Q)))$. By Lemma 2.8 and auxiliary result (2) (see the proof of Lemma 3.5), it follows that $rpf(bf(P)) = rpf(bf(Q))$, that is, $P =_{rpbf} Q$.

In order to prove $\Leftarrow$, assume $P =_{rpbf} Q$, thus $rpf(bf(P)) = rpf(bf(Q))$. Then $se(rpf(bf(P))) = se(rpf(bf(Q)))$ and thus by Lemma 3.15,

$$rp(se(bf(P))) = rp(se(bf(Q))).$$

By Corollary 2.19, $se(bf(P)) = se(P)$ and $se(bf(Q)) = se(Q)$, so $rp(se(P)) = rp(se(Q))$, that is, $P =_{rpse} Q$. □

We end this section with a last completeness result.

Theorem 3.18 (Completeness of $\mathrm{CP}_{rp}(A)$). *For all $P, Q \in C_A$,*

$$\mathrm{CP}_{rp}(A) \vdash P = Q \iff P =_{rpse} Q.$$

Proof. Combine Theorem 3.11 and Proposition 3.17. □

4 Conclusions

In [4] we introduced proposition algebra using Hoare's conditional $x \triangleleft y \triangleright z$ and the constants T and F. We defined a number of varieties of so-called *valuation algebras* in order to capture different semantics for the evaluation of conditional statements, and provided axiomatizations for the resulting valuation congruences. In [3,5] we introduced an alternative valuation semantics for proposition algebra in the form of *Hoare-McCarthy algebras* (HMA's) that is more elegant than the semantical framework provided in [4]: HMA-based semantics has the advantage that one can define a valuation congruence without first defining the valuation *equivalence* it is contained in.

In this paper, we use Staudt's evaluation trees [13] to define free valuation congruence as the relation $=_{se}$ (see Section 2) and this appears to be a relatively simple and stand-alone exercise, resulting in a semantics that is elegant and much simpler than HMA-based semantics [3,5] and the semantics defined in [4]. By Theorem 2.11, $=_{se}$ coincides with "free valuation congruence as defined in [4]" because both relations are axiomatized by CP (see [4, Thm.4.4andThm.6.2]). The advantage of "evaluation tree semantics" is that for a given conditional statement P, the evaluation tree $se(P)$ determines all relevant atomic evaluations, and $P =_{se} Q$ is determined by evaluation trees that contain no more atoms than those that occur in P and Q; this is comparable to how truth tables can be used in the setting of propositional logic.

In Section 3 we define repetition-proof valuation congruence $=_{rpse}$ on C_A by $P =_{rpse} Q$ if, and only if, $rpse(P) = rpse(Q)$, where $rpse(P) = rp(se(P))$ and rp is a transformation function on evaluation trees. It is obvious that this transformation is "natural", given the axiom schemes (CPrp1) and (CPrp2) that are characteristic for $\mathrm{CP}_{rp}(A)$. The equivalence on C_A that we want to prove is

$$\mathrm{CP}_{rp}(A) \vdash P = Q \iff P =_{rpse} Q, \tag{5}$$

by which $=_{rpse}$ coincides with "repetition-proof valuation congruence as defined in [4]" because both are axiomatized by $\mathrm{CP}_{rp}(A)$ (see [4, Thm.6.3]). However, equivalence (5) implies that $=_{rpse}$ is a *congruence* relation on C_A and we could not find a direct proof of this fact. We chose to simulate the transformation $rpse$ by the transformation $rpbf$ on conditional statements and to prove that the resulting equivalence relation $=_{rpbf}$ is a congruence axiomatized by $\mathrm{CP}_{rp}(A)$.

This is Theorem 3.11, the proof of which depends on [4, Thm.6.3]) *and* on Theorem 3.9, that is,

$$\text{For all } P \in C_A,\ \mathrm{CP}_{rp}(A) \vdash P = \mathit{rpbf}(P).$$

In order to prove equivalence (5) (which is Theorem 3.18), it is thus sufficient to prove that $=_{rpbf}$ and $=_{rpse}$ coincide, and this is Proposition 3.17.

In [6] we define evaluation trees for most of the other valuation congruences defined in [4] by transformations on *se*-images that are also "natural", and this also results in elegant "evaluation tree semantics" for each of these congruences.

We conclude with a brief digression on *short-circuit logic*, which we defined in [7] (see [5] for a quick introduction), and an example on the use of $\mathrm{CP}_{rp}(A)$. Familiar binary connectives that occur in the context of imperative programming and that prescribe short-circuit evaluation, such as `&&` (in C called "logical AND"), are often defined in the following way:

$$P \mathbin{\texttt{\&\&}} Q \ =_{\mathrm{def}} \ \texttt{if}\ \ P\ \ \texttt{then}\ \ Q\ \ \texttt{else}\ \ \mathit{false},$$

independent of the precise syntax of P and Q, hence, $P \mathbin{\texttt{\&\&}} Q =_{\mathrm{def}} Q \triangleleft P \triangleright \mathsf{F}$. It easily follows that `&&` is associative (cf. Footnote 3). In a similarly way, negation can be defined by $\neg P =_{\mathrm{def}} \mathsf{F} \triangleleft P \triangleright \mathsf{T}$. In [7] we focus on this question:

Question 4.1. *Which are the logical laws that characterize short-circuit evaluation of binary propositional connectives?*

A first approach to this question is to adopt the conditional as an auxiliary operator, as is done in [5,7], and to answer Question 4.1 using definitions of the binary propositional connectives as above and the axiomatization for the valuation congruence of interest in proposition algebra (or, if "mixed conditional statements" are at stake, axiomatizations for the appropriate valuation congruences). An alternative and more direct approach to Question 4.1 is to establish axiomatizations for short-circuited binary connectives in which the conditional is *not* used. For free valuation congruence, an equational axiomatization of short-circuited binary propositional connectives is provided by Staudt in [13], where $se(P \mathbin{\texttt{\&\&}} Q) =_{\mathrm{def}} se(P)[\mathsf{T} \mapsto se(Q)]$ and $se(\neg P) =_{\mathrm{def}} se(P)[\mathsf{T} \mapsto \mathsf{F}, \mathsf{F} \mapsto \mathsf{T}]$ (and where the function *se* is also defined for short-circuited disjunction), and the associated completeness proof is based on decomposition properties of such evaluation trees. For repetition-proof valuation congruence it is an open question whether a finite, equational axiomatization of the short-circuited binary propositional connectives exists, and an investigation of repetition-proof evaluation trees defined by such connectives might be of interest in this respect. We end with an example on the use of $\mathrm{CP}_{rp}(A)$ that is based on [7, Ex.4].

Example 4.2. *Let A be a set of atoms of the form* $(e\texttt{==}e')$ *and* $(\texttt{n=}e)$ *with* `n` *some initialized program variable and e, e' arithmetical expressions over the integers that may contain* `n`. *Assume that* $(e\texttt{==}e')$ *evaluates to true if e and e' represent the same value, and* $(\texttt{n=}e)$ *always evaluates to true with the effect that* *e's value is assigned to* `n`*. Then these atoms satisfy the axioms of* $\mathrm{CP}_{rp}(A)$.[4] *Notice that if* `n` *has initial value* 0 *or* 1*,* `((n=n+1) && (n=n+1)) && (n==2)` *and* `(n=n+1) && (n==2)` *evaluate to different results, so the atom* `(n=n+1)` *does not satisfy the law* a `&&` $a = a$*, by which this example is typical for the repetition-proof characteristic of* $\mathrm{CP}_{rp}(A)$*.*

We acknowledge the helpful comments of two anonymous reviewers.

References

1. Bergstra, J.A., Bethke, I., Rodenburg, P.H.: A propositional logic with 4 values: true, false, divergent and meaningless. Journal of Applied Non-Classical Logics **5**(2), 199–218 (1995)
2. Bergstra, J.A., Loots, M.E.: Program algebra for sequential code. Journal of Logic and Algebraic Programming **51**(2), 125–156 (2002)
3. Bergstra, J.A., Ponse, A.: On Hoare-McCarthy algebras [cs.LO] (2010). http://arxiv.org/abs/1012.5059
4. Bergstra, J.A., Ponse, A.: Proposition algebra. ACM Transactions on Computational Logic **12**(3), Article 21, 36 pages (2011)
5. Bergstra, J.A., Ponse, A.: Proposition algebra and short-circuit logic. In: Arbab, F., Sirjani, M. (eds.) FSEN 2011. LNCS, vol. 7141, pp. 15–31. Springer, Heidelberg (2012)
6. Bergstra, J.A., Ponse A.: Evaluation trees for proposition algebra. arXiv:1504.08321v2 [cs.LO] (2015)
7. Bergstra, J.A., Ponse, A., Staudt, D.J.C.: Short-circuit logic. arXiv:1010.3674v4 [cs.LO, math.LO] (version v1: October 2010) (2013)
8. de Boer, F.S., de Vries, F.-J., Olderog, E.-R., Ponse, A. (guest editors): Selected papers from the Workshop on Assertional Methods. Formal Aspects of Computing, 6(1 Supplement; Special issue) (1994)
9. Harel, D.: Dynamic logic. In: Gabbay, D., Günthner, F. (eds.) Handbook of Philosophical Logic, vol. II, pp. 497–604. Reidel Publishing Company (1984)
10. Hayes, I.J., He Jifeng, Hoare, C.A.R., Morgan, C.C., Roscoe, A.W., Sanders, J.W., Sorensen, I.H., Spivey, J.M., Sufrin B.A.: Laws of programming. Communications of the ACM **3**(8), 672–686 (1987)
11. Hoare, C.A.R.: Communicating Sequential Processes. Prentice Hall International (1985)
12. Hoare, C.A.R.: A couple of novelties in the propositional calculus. Zeitschrift für Mathematische Logik und Grundlagen der Mathematik **31**(2), 173–178 (1985); Republished. In: Hoare, C.A.R., Jones, C.B. (eds.) Essays in Computing Science. Series in Computer Science, pp. 325–331. Prentice Hall International (1989)
13. Staudt, D.J.C.: Completeness for two left-sequential logics. MSc. thesis Logic, University of Amsterdam (May 2012). arXiv:1206.1936v1 [cs.LO] (2012)
14. Wortel, L.: Side effects in steering fragments. MSc. thesis Logic, University of Amsterdam (September 2011). arXiv:1109.2222v1 [cs.LO] (2011)

[4] Of course, not all equations that are valid in the setting of Example 4.2 follow from $\mathrm{CP}_{rp}(A)$, e.g., $\mathrm{CP}_{rp}(A) \nvdash$ `(0==0)` $= \mathsf{T}$. We note that a particular consequence of $\mathrm{CP}_{rp}(A)$ in the setting of short-circuit logic is $(\neg a$ `&&` $a)$ `&&` $x = \neg a$ `&&` a (cf. Example 3.13), and that Example 4.2 is related to the work of Wortel [14], where an instance of *Propositional Dynamic Logic* [9] is investigated in which assignments can be turned into tests; the assumption that such tests always evaluate to *true* is natural because the assumption that assignments always succeed is natural.