mathematics

MDPI

Article

Evaluation Trees and Normalisation for Proposition Algebra [†]

Jan A. Bergstra [1], Alban Ponse [1,*] and Daan J. C. Staudt [2]

[1] Section Theory of Computer Science, Informatics Institute, Faculty of Science, University of Amsterdam, 1098 XH Amsterdam, The Netherlands; j.a.bergstra@uva.nl
[2] Independent Researcher, 1011 XH Amsterdam, The Netherlands; daan@daanstaudt.nl
[*] Correspondence: a.ponse@uva.nl
[†] This paper is an extended version of the conference paper from Correct System Design: Symposium in Honor of Ernst-Rüdiger Olderog on the Occasion of His 60th Birthday, Oldenburg, Germany, 8–9 September 2015.

Abstract

Proposition algebra is based on Hoare's conditional, a ternary connective comparable to if–then–else and used in the context of propositional logic. Conditional statements are composed from atomic propositions (propositional variables), constants for the Boolean truth values, and the conditional. In previous work, various equational axiomatisations have been defined, each of which leads to a so-called valuation congruence. The weakest of these is "free valuation congruence" and the strongest is "static valuation congruence", which is equivalent to propositional logic. Free valuation congruence is axiomatised by four simple equational axioms, and we use evaluation trees to give a simple semantics: two conditional statements are free valuation congruent if, and only if, they have equal evaluation trees. Increasingly stronger valuation congruences arise by adding axioms to the four that define free valuation congruence: repetition-proof, contractive, memorising, and static valuation congruence. We prove that each such valuation congruence C can be characterised using a transformation on evaluation trees: two conditional statements are C-valuation congruent if, and only if, their C-transformed evaluation trees are equal. In order to prove that these transformations preserve the congruence property, we use normalisation functions: two conditional statements are C-valuation congruent if, and only if, the C-normalisation function returns equal images. Our framework provides the first comprehensive tree-based semantics that unifies all major valuation congruences in proposition algebra, offering both conceptual clarity and practical decision procedures.

Keywords: conditional composition; evaluation tree; proposition algebra; short-circuit evaluation; short-circuit logic; side effect

MSC: 03B05; 03B70

Check for updates

Academic Editor: Ignacio Javier Perez Galvez

Received: 1 December 2025
Revised: 24 December 2025
Accepted: 8 January 2026
Published: 12 January 2026

1. Introduction

This paper is an extended journal version of the conference paper [1] entitled *Evaluation trees for proposition algebra: The case for free and repetition-proof valuation congruence*, with new results for three other valuation congruences discussed in Sections 4–6. Much of the text in Sections 1–3 is taken from [1]. We have included this material because it is necessary for the new results. The conclusions in Section 7 are based on the concluding section of [1] and include further references to related work.

In 1985, Hoare's paper *A couple of novelties in the propositional calculus* [2] was published. In this paper, the ternary connective $\_ \triangleleft \_ \triangleright \_$ is introduced as the conditional. A more common expression for a conditional statement $P \triangleleft Q \triangleright R$ is the following:

$$\texttt{if}\ Q\ \texttt{then}\ P\ \texttt{else}\ R.$$

However, 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, either P or R is evaluated (and the other is not) and determines the evaluation value. This evaluation strategy is a form of short-circuit evaluation (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). In [2], Hoare proves that propositional logic is characterised by eleven equational axioms, some of which employ constants T and F for the truth values *true* and *false*. However, the same result was proven in 1948 by Church in ref. [3], where he introduced conditioned disjunction, notation $[p, q, r]$, that also models "`if` q `then` p `else` r", or in Church's words, "p or r, according as q or not q". See also Church's 1956 book [4] (§24), which reiterates all the main findings. Furthermore, Church emphasized the duality property of $[p, q, r]$, and even referred to older related work, notably that of Post from 1942 on functionally complete systems of independent primitive connectives for two-valued logic.

In 2011, Bergstra and Ponse introduced "Proposition Algebra" in ref. [5] as a general approach to the study of the conditional and defined several valuation congruences and equational axiomatisations of these congruences. The most basic and least identifying valuation congruence is free valuation congruence, which is axiomatised by the axioms in Table 1.

These axioms stem from [2] and define the conditional as a primitive connective. The name CP (for Conditional Propositions) was chosen 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 [5], 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 fewer.

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

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

In Section 2 we characterise free valuation congruence with help of evaluation trees, which provide a simple semantics for the conditional; we return to this point in Section 7. Given a conditional statement, its evaluation tree directly represents all its evaluations (in the way a truth table does in the case of propositional logic). Two conditional statements are equivalent with respect to free valuation congruence if their evaluation trees are equal.

Evaluation trees are simple binary trees, proposed by Staudt in ref. [6] (that appeared in 2012). Free valuation congruence identifies less than the equivalence defined by Hoare's axioms in ref. [2]. 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 characterises propositional logic.

In Section 3 we consider repetition-proof valuation congruence, a valuation congruence that identifies more than free and less than static valuation congruence. Repetition-proof valuation congruence is axiomatised by CP extended with two (schematic) axioms, one of which reads

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

for any atomic proposition a, and thus expresses that if 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 repetition-proof valuation congruent, but not free valuation congruent. A more concrete example of this valuation congruence in the field of programming languages is given in Example 3 (in Section 7). We characterise repetition-proof valuation congruence by defining a transformation on evaluation trees that yields repetition-proof evaluation trees: two conditional statements are repetition-proof valuation congruent 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 characterisation, which is phrased as a completeness result, is non-trivial and we could not find a proof that is essentially simpler.

Valuation congruences that identify more conditional statements than repetition-proof valuation congruence are contractive, memorising, and static valuation congruence. These are defined and axiomatised in ref. [5]. In Sections 4–6, each of these valuation congruences is characterised using a transformation on evaluation trees: two conditional statements are C-valuation congruent if, and only if, their C-transformed evaluation trees are equal. These transformations are simple and natural, and only for static valuation congruence do we use a slightly more complex transformation.

In Section 7, we discuss the general structure of the proofs of the axiomatisation results in Sections 3–6, each of which is based on a normalisation function for conditional statements. Then we end the paper with a brief digression on short-circuit logic and give the above-mentioned example of the use of repetition-proof valuation congruence, and Example 4 of contractive valuation congruence. We conclude with some remarks about related and future work. Finally, in Appendix A, we provide some independence results.

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 containing at least two atoms. 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:

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

Observe that CP is a self-dual axiomatisation: 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-circuit evaluation, which is similar to how we consider the evaluation of an "`if` y `then` x `else` z" expression. The following three definitions are taken from [6].

Definition 1. *The set $\mathcal{T}_A$ of evaluation trees over A with leaves in $\{\mathsf{T}, \mathsf{F}\}$ is defined inductively by*

$$\mathsf{T} \in \mathcal{T}_A, \quad \mathsf{F} \in \mathcal{T}_A, \quad (X \trianglelefteq a \trianglerighteq Y) \in \mathcal{T}_A \text{ for any } X, Y \in \mathcal{T}_A \text{ and } a \in A.$$

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 thread algebra; the thread $P \trianglelefteq a \trianglerighteq Q$ first performs a and then proceeds with P if *true* was returned or with Q otherwise (see [7] for more information). Evaluation trees play a crucial role in the main results of [6]. In order to define our "evaluation tree semantics", we first define an auxiliary function on trees.

Definition 2 (Leaf replacement)**.** *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 postfix 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 are listed is irrelevant and 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 replacements satisfy the following equation, which follows easily by structural induction on X:

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

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 3. *The short-circuit evaluation function*

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

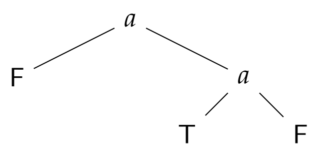
is defined as follows, where $a \in A$*:*

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

Example 1. *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 4. *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 1 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-circuit conjunction* (we return to this connective in Section 7).

In turn, each evaluation tree gives rise to a *unique* conditional statement. For Example 1, this is $\mathsf{F} \triangleleft a \triangleright (\mathsf{T} \triangleleft a \triangleright \mathsf{F})$ (note the syntactical correspondence).

Definition 5. *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* $d(Q \triangleleft a \triangleright R) = 1 + \max\{d(Q), d(R)\}$*.*

Following [5], we speak of "basic forms" instead of normal forms in order to avoid intuitions from term rewriting; for example, the basic form associated with atom a is

$\mathsf{T} \triangleleft a \triangleright \mathsf{F}$, whereas one would expect that the normal form of the latter is a. The following lemmas exploit the structure of basic forms and are stepping stones to our first completeness result (Theorem 1).

Lemma 1. *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, and 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. *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 6. *The binary relation se-congruence, notation $=_{se}$, is defined on C_A as follows:*

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

Lemma 3. *Se-congruence is a congruence relation.*

Proof. Assume $P =_{se} P'$, $Q =_{se} Q'$, and $R =_{se} R'$. Then

$$\begin{aligned} se(P \triangleleft Q \triangleright R) &= se(Q)[\mathsf{T} \mapsto se(P), \mathsf{F} \mapsto se(R)] \\ &= se(Q')[\mathsf{T} \mapsto se(P'), \mathsf{F} \mapsto se(R')] \\ &= se(P' \triangleleft Q' \triangleright R'), \end{aligned}$$

so $P \triangleleft Q \triangleright R =_{se} P' \triangleleft Q' \triangleright R'$. □

Theorem 1 (Completeness of CP). *For all $P, Q \in C_A$,*

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

Proof. ($\Rightarrow$) Without loss of generality it can be assumed that substitutions happen first in equational proofs (see, e.g., [8]). By Lemma 3, $=_{se}$ is a congruence relation and it easily follows that all CP-axioms are sound. For example, the soundness of axiom (CP4) follows from

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

($\Leftarrow$) Let $P =_{se} Q$. According to Lemma 1, there exist basic forms P' and Q', such that $\mathrm{CP} \vdash P = P'$ and $\mathrm{CP} \vdash Q = Q'$, and so $\mathrm{CP} \vdash P' = Q'$. By soundness ($\Rightarrow$) we find $P' =_{se} Q'$, so by Lemma 2, $P' = Q'$. 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 $\text{CP} \vdash P = P'$, and 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 7. *The basic form function $bf : C_A \to BF_A$ is defined as follows, where $a \in A$:*

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

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:

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

(The notational overloading with the leaf replacement functions 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 *normalisation function*.

Lemma 4. *For all $P \in C_A$, $bf(P)$ is a basic form.*

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

Lemma 5. *For each basic form P, $bf(P) = P$.*

Proof. By structural induction on P. □

Definition 8. *The binary relation $=_{bf}$ on C_A is defined as follows:*

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

Lemma 6. *The relation $=_{bf}$ is a congruence relation.*

Proof. Replace $=_{se}$ by $=_{bf}$ in the proof of Lemma 3. □

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

Lemma 7. *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

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

By Lemma 4, $\mathit{bf}(P)$, $\mathit{bf}(P_i)$, and $\mathit{bf}(Q_i)$ are basic forms. We prove (1) by structural induction on the form that $\mathit{bf}(P)$ can have. If $\mathit{bf}(P)=\mathsf{T}$, then

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

and

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

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

The inductive case $\mathit{bf}(P) = R_1 \triangleleft a \triangleright R_2$ is trivial (by definition of the last defining clause of the auxiliary function $[\mathsf{T}\mapsto Q,\mathsf{F}\mapsto R]$ in Definition 7). □

Theorem 2. *For all $P,Q \in C_A$, $\mathrm{CP} \vdash P = Q \iff P =_{\mathit{bf}} Q$.*

Proof. ($\Rightarrow$) By Lemma 6, $=_{\mathit{bf}}$ is a congruence relation and it easily follows that closed instances of the CP-axioms (CP1)–(CP3) satisfy $=_{\mathit{bf}}$. By Lemma 7 it follows that closed instances of axiom (CP4) also satisfy $=_{\mathit{bf}}$.

($\Leftarrow$) Assume $P =_{\mathit{bf}} Q$. According to Lemma 1, 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 $P' =_{\mathit{bf}} Q'$, which implies by Lemma 5 that $P' = Q'$. Hence, $\mathrm{CP} \vdash P = P' = Q' = Q$. □

Corollary 1. *For all $P \in C_A$, $P =_{\mathit{bf}} \mathit{bf}(P)$ and $P =_{se} \mathit{bf}(P)$.*

Proof. By Lemmas 4 and 5, $\mathit{bf}(P) = \mathit{bf}(\mathit{bf}(P))$; thus, $P =_{\mathit{bf}} \mathit{bf}(P)$. By Theorem 2, $\mathrm{CP} \vdash P = \mathit{bf}(P)$, and by Theorem 1, $P =_{se} \mathit{bf}(P)$. □

3. Evaluation Trees for Repetition-Proof Valuation Congruence

In ref. [5], Bergstra and Ponse defined *repetition-proof* CP as the extension of the axiom set CP with the following two axiom schemes, where a ranges over A:

$$\begin{aligned} &\text{(CPrp1)} && (x \triangleleft a \triangleright y) \triangleleft a \triangleright z = (x \triangleleft a \triangleright x) \triangleleft a \triangleright z, \\ &\text{(CPrp2)} && x \triangleleft a \triangleright (y \triangleleft a \triangleright z) = x \triangleleft a \triangleright (z \triangleleft a \triangleright z). \end{aligned}$$

We write $\mathrm{CP}_{rp}(A)$ for this extension. These axiom schemes characterise 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 other's dual.

Following [5], we define a proper subset of basic forms with the property that each propositional statement can be proved equal to such a basic form.

Definition 9. *Rp-basic forms are inductively defined:*

- T *and* F *are rp-basic forms;*
- $P_1 \triangleleft a \triangleright P_2$ *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* $Q_i \triangleleft a \triangleright Q_i$.

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

Definition 10. *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:*

$$\begin{aligned} rpf(B) &= B \text{ for } B \in \{\mathsf{T}, \mathsf{F}\}, \\ rpf(P \triangleleft a \triangleright Q) &= rpf(f_a(P)) \triangleleft a \triangleright rpf(g_a(Q)). \end{aligned}$$

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*

$$\begin{aligned} f_a(B) = g_a(B) &= B \text{ for } B \in \{\mathsf{T}, \mathsf{F}\}, \\ f_a(P \triangleleft b \triangleright Q) &= \begin{cases} f_a(P) \triangleleft a \triangleright f_a(P) & \text{if } b = a, \\ P \triangleleft b \triangleright Q & \text{otherwise,} \end{cases} \\ g_a(P \triangleleft b \triangleright Q) &= \begin{cases} g_a(Q) \triangleleft a \triangleright g_a(Q) & \text{if } b = a, \\ P \triangleleft b \triangleright Q & \text{otherwise.} \end{cases} \end{aligned}$$

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 8. *For all* $a \in A$ *and* $P \in BF_A$,

$$g_a(f_a(P)) = f_a(f_a(P)) = f_a(P) \text{ and } f_a(g_a(P)) = g_a(g_a(P)) = g_a(P).$$

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 9. *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 10. *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 of the following:

- By Lemma 9, $f_a(Q)$ 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 in the form of Q:
  1. If $Q \in \{\mathsf{T}, \mathsf{F}\}$, then $rpf(f_a(Q)) = Q$, and 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 8, $rpf(f_a(Q)) = rpf(f_a(Q_1)) \triangleleft a \triangleright rpf(f_a(Q_1))$.
  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 so $rpf(f_a(Q)) = rpf(f_b(Q_1)) \triangleleft b \triangleright rpf(g_b(Q_2))$ and 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 (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 4, and by auxiliary result (2), $rpf(S)$ is an rp-basic form. □

The following result is used in Proposition 1 and Lemma 12.

Lemma 11. *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:

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

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 $Q = Q_1 \lhd a \rhd Q_1$ and $R = R_1 \lhd a \rhd R_1$, then Q and R are rp-basic forms with central condition a, and so

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

and the equality for R follows in a similar way. If $Q = Q_1 \lhd a \rhd Q_1$ and $R \neq R_1 \lhd a \rhd R_1$, 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 $Q = Q_1 \lhd a \rhd Q_1$ and $R = R_1 \lhd a \rhd R_1$, then by auxiliary result (3),

$$rpf(Q) = rpf(f_a(Q_1)) \lhd a \rhd rpf(f_a(Q_1)),$$

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

$$\begin{aligned} rpf(f_a(Q)) &= rpf(f_a(f_a(Q_1))) \lhd a \rhd rpf(g_a(f_a(Q_1))) \\ &= rpf(f_a(Q_1)) \lhd a \rhd rpf(f_a(Q_1)) && \text{by Lemma 8} \\ &= Q_1 \lhd a \rhd Q_1, \end{aligned}$$

and the equalities for R follow in a similar way.

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

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

Proposition 1 (*rpbf* is a normalisation 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 10. For the second statement, it suffices by Lemma 5 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 $P = Q \lhd a \rhd R$, and thus $rpf(P) = rpf(f_a(Q)) \lhd a \rhd rpf(g_a(R))$. By Lemma 11, $rpf(f_a(Q)) = Q$ and $rpf(g_a(R)) = R$, hence $rpf(P) = P$. □

Lemma 12. *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 $P = P_1 \lhd a \rhd P_2$. By induction, $\mathrm{CP}_{rp}(A) \vdash P_i = rpf(P_i)$. We proceed by the following case distinction on the form that P_1 and P_2 can have:

1. If $P_i \in \{\mathsf{T}, \mathsf{F}, Q_i \lhd b_i \rhd Q_i'\}$ with $b_i \neq a$, then $f_a(P_1) = P_1$ and $g_a(P_2) = P_2$, and hence $rpf(P) = rpf(P_1) \lhd a \rhd rpf(P_2)$. Thus, $\mathrm{CP}_{rp}(A) \vdash P = rpf(P)$.

2. If $P_1 = R_1 \lhd a \rhd R_2$ and $P_2 \in \{\mathsf{T}, \mathsf{F}, Q' \lhd b \rhd Q''\}$ with $b \neq a$, then $g_a(P_2) = P_2$ and by auxiliary result (2) in the proof of Lemma 10, $rpf(R_1)$ and $rpf(P_2)$ are rp-basic forms. We derive

$$\begin{aligned}
\mathrm{CP}_{rp}(A) \vdash P &= (R_1 \lhd a \rhd R_2) \lhd a \rhd P_2 \\
&= (R_1 \lhd a \rhd R_1) \lhd a \rhd P_2 && \text{by (CPrp1)} \\
&= (rpf(R_1) \lhd a \rhd rpf(R_1)) \lhd a \rhd rpf(P_2) && \text{by IH} \\
&= (rpf(f_a(R_1)) \lhd a \rhd rpf(f_a(R_1))) \lhd a \rhd rpf(g_a(P_2)) && \text{by Lemma 11} \\
&= rpf(f_a(R_1 \lhd a \rhd R_2)) \lhd a \rhd rpf(g_a(P_2)) \\
&= rpf((R_1 \lhd a \rhd R_2) \lhd a \rhd P_2) \\
&= rpf(P).
\end{aligned}$$

3. If $P_1 \in \{\mathsf{T}, \mathsf{F}, Q' \lhd b \rhd Q''\}$ with $b \neq a$ and $P_2 = S_1 \lhd a \rhd S_2$, 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 $P_1 = R_1 \lhd a \rhd R_2$ and $P_2 = S_1 \lhd a \rhd S_2$, 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. *For all $P \in C_A$, $\mathrm{CP}_{rp}(A) \vdash P = rpbf(P)$.*

Proof. By Theorem 2 and Corollary 1, we find $\mathrm{CP}_{rp}(A) \vdash P = bf(P)$. By Lemma 12, $\mathrm{CP}_{rp}(A) \vdash bf(P) = rpf(bf(P))$, and $rpf(bf(P)) = rpbf(P)$. □

Definition 11. *The binary relation $=_{rpbf}$ on C_A is defined as follows:*

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

Theorem 4. *For all $P, Q \in C_A$, $\mathrm{CP}_{rp}(A) \vdash P = Q \iff P =_{rpbf} Q$.*

Proof. ($\Rightarrow$) Assume $\mathrm{CP}_{rp}(A) \vdash P = Q$. By Theorem 3, $\mathrm{CP}_{rp}(A) \vdash rpbf(P) = rpbf(Q)$. In ref. [5], the following two statements are proven (Theorem 6.3 and an auxiliary result in its proof), where $=_{rp}$ is the binary relation on C_A that specifies repetition-proof valuation congruence:

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

By Lemma 10, these statements imply $rpbf(P) = rpbf(Q)$, that is, $P =_{rpbf} Q$.
($\Leftarrow$) Assume $P =_{rpbf} Q$. By Theorem 3, $\mathrm{CP}_{rp}(A) \vdash P = Q$. □

So, the relation $=_{rpbf}$ is axiomatised by $\mathrm{CP}_{rp}(A)$, and is thus a congruence that coincides with repetition-proof valuation congruence as defined in ref [5]. With this observation in mind, we define a transformation on evaluation trees that mimics the function *rpbf* and proves that the equality of two such transformed trees characterises repetition-proof valuation congruence.

Definition 12. *The unary repetition-proof evaluation function*

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

yields repetition-proof evaluation trees and is defined by $rpse(P) = rp(se(P))$.

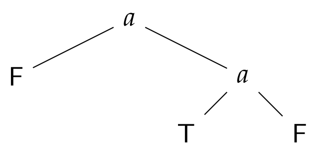
The auxiliary function $rp : \mathcal{T}_A \to \mathcal{T}_A$ is defined as follows ($a \in A$):

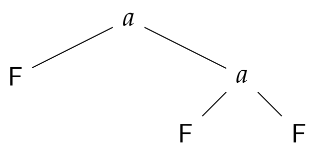
$$\begin{aligned} rp(B) &= B \ \textit{for}\ B \in \{\mathsf{T}, \mathsf{F}\}, \\ rp(X \trianglelefteq a \trianglerighteq Y) &= rp(F_a(X)) \trianglelefteq a \trianglerighteq rp(G_a(Y)). \end{aligned}$$

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

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

Example 2. *Let $P = a \triangleleft (\mathsf{F} \triangleleft a \triangleright \mathsf{T}) \triangleright \mathsf{F}$. We depict $se(P)$ (as in Example 1) and the repetition-proof evaluation tree $rpse(P) = \mathsf{F} \trianglelefteq a \trianglerighteq (\mathsf{F} \trianglelefteq a \trianglerighteq \mathsf{F})$ as follows:*





The similarities between *rpse* and the function *rpbf* can be exploited: We use the following lemma in the proof of this section's last completeness result.

Lemma 13. *For all $P \in BF_A$, $rp(se(P)) = se(rpf(P))$.*

Proof. We first prove an auxiliary result:

$$\text{For all } a \in A \text{ and } P \in BF_A,\ F_a(se(P)) = se(f_a(P)) \text{ and } G_a(se(P)) = se(g_a(P)). \tag{4}$$

Fix some $a \in A$. We prove (4) 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

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

and if $b \neq a$, then

$$\begin{aligned} F_a(se(Q \triangleleft b \triangleright R)) &= F_a(se(Q) \trianglelefteq b \trianglerighteq se(R)) \\ &= se(Q) \trianglelefteq b \trianglerighteq se(R) \\ &= se(Q \triangleleft b \triangleright R) \\ &= se(f_a(Q \triangleleft b \triangleright R)). \end{aligned}$$

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

We prove the lemma's statement by induction on $d(P)$. The base cases $P \in \{\mathsf{T}, \mathsf{F}\}$ follow immediately. Assume $P = Q \triangleleft a \triangleright R$; then,

$$\begin{aligned}
rp(se(Q \triangleleft a \triangleright R)) &= rp(se(Q) \trianglelefteq a \trianglerighteq se(R)) \\
&= rp(F_a(se(Q))) \trianglelefteq a \trianglerighteq rp(G_a(se(R))) \\
&= rp(se(f_a(Q))) \trianglelefteq a \trianglerighteq rp(se(g_a(R))) && \text{by (4)} \\
&= se(rpf(f_a(Q))) \trianglelefteq a \trianglerighteq se(rpf(g_a(R))) && \text{by IH (and Lemma 9)} \\
&= se(rpf(f_a(Q)) \triangleleft a \triangleright rpf(g_a(R))) \\
&= se(rpf(Q \triangleleft a \triangleright R)).
\end{aligned}$$

□

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

Definition 13. *Repetition-proof se-congruence, notation* $=_{rpse}$*, is defined on* C_A *as follows:*

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

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

Proposition 2. *For all* $P, Q \in C_A$, $P =_{rpse} Q \iff P =_{rpbf} Q$.

Proof. ($\Rightarrow$) Assume $rpse(P) = rpse(Q)$; thus, $rp(se(P)) = rp(se(Q))$. By Corollary 1, $rp(se(bf(P))) = rp(se(bf(Q)))$, and so by Lemma 13, $se(rpf(bf(P))) = se(rpf(bf(Q)))$. By Lemma 2 and auxiliary result (2) (see the proof of Lemma 10), it follows that $rpf(bf(P)) = rpf(bf(Q))$; that is, $P =_{rpbf} Q$.

($\Leftarrow$) Assume $P =_{rpbf} Q$; thus, $rpf(bf(P)) = rpf(bf(Q))$ and $se(rpf(bf(P))) = se(rpf(bf(Q)))$. By Lemma 13, $rp(se(bf(P))) = rp(se(bf(Q)))$. By Corollary 1, $se(bf(P)) = se(P)$ and $se(bf(Q)) = se(Q)$, and so $rp(se(P)) = rp(se(Q))$; that is, $P =_{rpse} Q$. □

We end this section with the completeness result we were seeking.

Theorem 5 (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 4 and Proposition 2. □

4. Evaluation Trees for Contractive Valuation Congruence

In ref. [5], Bergstra and Ponse introduced $\mathrm{CP}_{cr}(A)$, contractive CP, as the extension of CP with the following two axiom schemes, where a ranges over A:

$$\begin{aligned}
&(\text{CPcr1}) && (x \triangleleft a \triangleright y) \triangleleft a \triangleright z = x \triangleleft a \triangleright z, \\
&(\text{CPcr2}) && x \triangleleft a \triangleright (y \triangleleft a \triangleright z) = x \triangleleft a \triangleright z.
\end{aligned}$$

These schemes prescribe the contraction for each atom a for respectively the *true* case and the *false* case, and are each others dual. It easily follows that the axiom schemes (CPrp1) and (CPrp2) are derivable from $\mathrm{CP}_{cr}(A)$, and so $\mathrm{CP}_{cr}(A)$ is also an axiomatic extension of $\mathrm{CP}_{rp}(A)$.

Again, we define a proper subset of basic forms with the property that each propositional statement can be proved equal to such a basic form.

Definition 14. *Cr-basic forms are inductively defined:*

- T *and* F *are cr-basic forms;*
- $P_1 \triangleleft a \triangleright P_2$ *is a cr-basic form if* P_1 *and* P_2 *are cr-basic forms, and if* P_i *is not equal to* T *or* F*, the central condition in* P_i *is different from* a*.*

It will become useful to define a function that transforms conditional statements into cr-basic forms, and that is comparable to the function *bf* (see Definition 7).

Definition 15. *The cr-basic form function* $cbf : C_A \to C_A$ *is defined by*

$$cbf(P) = cf(bf(P)).$$

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

$$\begin{aligned} cf(B) &= B \ \text{ for } B \in \{\mathsf{T}, \mathsf{F}\}, \\ cf(P \triangleleft a \triangleright Q) &= cf(h_a(P)) \triangleleft a \triangleright cf(j_a(Q)). \end{aligned}$$

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

$$\begin{aligned} h_a(B) = j_a(B) &= B \ \text{ for } B \in \{\mathsf{T}, \mathsf{F}\}, \\ h_a(P \triangleleft b \triangleright Q) &= \begin{cases} h_a(P) & \text{if } b = a, \\ P \triangleleft b \triangleright Q & \text{otherwise,} \end{cases} \\ j_a(P \triangleleft b \triangleright Q) &= \begin{cases} j_a(Q) & \text{if } b = a, \\ P \triangleleft b \triangleright Q & \text{otherwise.} \end{cases} \end{aligned}$$

Thus, *cbf* maps a conditional statement P to $bf(P)$ and then transforms $bf(P)$ according to the auxiliary functions cf, h_a, and j_a.

Lemma 14. *For all* $a \in A$ *and* $P \in BF_A$*,* $d(P) \geq d(h_a(P))$ *and* $d(P) \geq d(j_a(P))$*.*

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

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

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

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

Lemma 15. *For all* $P \in C_A$*,* $cbf(P)$ *is a cr-basic form.*

Proof. We first prove an auxiliary result:

$$\text{For all } P \in BF_A,\ cf(P) \text{ is a cr-basic form.} \tag{5}$$

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

$$cf(Q \triangleleft a \triangleright R) = cf(h_a(Q)) \triangleleft a \triangleright cf(j_a(R)),$$

which is a cr-basic form because of the following:

- By Lemma 14, $h_a(Q)$ and $j_a(R)$ are basic forms with a depth smaller than or equal to n, and so by the induction hypothesis, $cf(h_a(Q))$ and $cf(j_a(R))$ are cr-basic forms,
- By the definition of the auxiliary functions h_a and j_a, the central condition of $h_a(Q)$ and $j_a(R)$ is not equal to a; hence, $cf(h_a(Q)) \triangleleft a \triangleright cf(j_a(R))$ is a cr-basic form.

This completes the proof of (5).

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,

$$cbf(P \triangleleft Q \triangleright R) = cf(bf(P \triangleleft Q \triangleright R)) = cf(S)$$

for some basic form S by Lemma 4, and by (5), $cf(S)$ is a cr-basic form. $\square$

The following lemma is used in Proposition 3 and Lemma 17.

Lemma 16. *If $Q \triangleleft a \triangleright R$ is a cr-basic form, then $Q = cf(Q) = cf(h_a(Q))$ and $R = cf(R) = cf(j_a(R))$.*

Proof. 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 $Q = Q_1 \triangleleft b \triangleright Q_2$ and $R = R_1 \triangleleft c \triangleright R_2$, then $b \neq a \neq c$ and thus $h_a(Q) = Q$ and $j_a(R) = R$. Moreover, Q_1 and Q_2 have no central condition b; hence, $h_b(Q_1) = Q_1$ and $j_b(Q_2) = Q_2$, and thus

$$\begin{aligned} cf(Q) &= cf(h_b(Q_1)) \triangleleft b \triangleright cf(j_b(Q_2)) \\ &= cf(Q_1) \triangleleft b \triangleright cf(Q_2) \\ &= Q_1 \triangleleft b \triangleright Q_2. && \text{by IH} \end{aligned}$$

The equalities for R follow in a similar way.

If $Q = Q_1 \triangleleft b \triangleright Q_1$ and $R \in \{\mathsf{T}, \mathsf{F}\}$, the lemma's equalities follow in a similar way, and this is also the case if $Q \in \{\mathsf{T}, \mathsf{F}\}$ and $R = Q_1 \triangleleft b \triangleright Q_1$. $\square$

With Lemma 16 we can easily prove the following result.

Proposition 3 (*cbf* is a normalisation function)**.** *For each $P \in C_A$, $cbf(P)$ is a cr-basic form, and for each cr-basic form P, $cbf(P) = P$.*

Proof. The first statement is Lemma 15. For the second statement, it suffices by Lemma 5 to prove that $cf(P) = P$. We prove this by case distinction on P. The cases $P \in \{\mathsf{T}, \mathsf{F}\}$ follow immediately, and otherwise $P = Q \triangleleft a \triangleright R$, and thus $cf(P) = cf(h_a(Q)) \triangleleft a \triangleright cf(j_a(R))$. By Lemma 16, $cf(h_a(Q)) = Q$ and $cf(j_a(R)) = R$; hence, $cf(P) = P$. $\square$

Lemma 17. *For all $P \in BF_A$, $\mathrm{CP}_{cr}(A) \vdash P = cf(P)$.*

Proof. We first prove two auxiliary results:

$$\text{For all } a \in A \text{ and } P, Q \in BF_A, \mathrm{CP}_{cr}(A) \vdash P \triangleleft a \triangleright Q = P \triangleleft a \triangleright j_a(Q), \tag{6}$$

$$\mathrm{CP}_{cr}(A) \vdash P \triangleleft a \triangleright Q = h_a(P) \triangleleft a \triangleright Q. \tag{7}$$

Fix some $a \in A$. We prove (6) by structural induction on Q. The base cases $Q \in \{\mathsf{T}, \mathsf{F}\}$ are trivial. For the inductive case $Q = Q_1 \triangleleft b \triangleright Q_2$, we have to distinguish the cases $b = a$ and $b \neq a$. If $b = a$, then $j_a(Q) = j_a(Q_2)$ and

$$\begin{aligned} \mathrm{CP}_{cr}(A) \vdash P \triangleleft a \triangleright (Q_1 \triangleleft a \triangleright Q_2) &= P \triangleleft a \triangleright Q_2 && \text{by (CPcr2)} \\ &= P \triangleleft a \triangleright j_a(Q_2) && \text{by IH} \\ &= P \triangleleft a \triangleright j_a(Q). \end{aligned}$$

If $b \neq a$ then $j_a(Q) = Q$, and hence $\mathrm{CP}_{cr}(A) \vdash P \triangleleft a \triangleright Q = P \triangleleft a \triangleright j_a(Q)$.

Auxiliary result (7) follows in a similar way by structural induction on P and with the help of axiom scheme (CPcr1).

The lemma's statement follows by induction on $d(P)$. The base cases $P \in \{\mathsf{T}, \mathsf{F}\}$ are trivial. For the inductive case, assume $P = Q \triangleleft a \triangleright R$. We derive

$$\begin{aligned} \mathrm{CP}_{cr}(A) \vdash Q \triangleleft a \triangleright R &= h_a(Q) \triangleleft a \triangleright j_a(R) && \text{by (6), (7)} \\ &= cf(h_a(Q)) \triangleleft a \triangleright cf(j_a(R)) && \text{by IH (and Lemma 14)} \\ &= cf(Q \triangleleft a \triangleright R). \end{aligned}$$

□

Theorem 6. *For all $P \in C_A$, $\mathrm{CP}_{cr}(A) \vdash P = cbf(P)$.*

Proof. By Theorem 2 and Corollary 1, $\mathrm{CP}_{cr}(A) \vdash P = bf(P)$, and by Lemma 17, $\mathrm{CP}_{cr}(A) \vdash bf(P) = cf(bf(P))$, and $cf(bf(P)) = cbf(P)$. □

Definition 16. *The binary relation $=_{cbf}$ on C_A is defined as follows:*

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

Theorem 7. *For all $P, Q \in C_A$, $\mathrm{CP}_{cr}(A) \vdash P = Q \iff P =_{cbf} Q$.*

Proof. ($\Rightarrow$) Assume $\mathrm{CP}_{cr}(A) \vdash P = Q$. Then, by Theorem 6, $\mathrm{CP}_{cr}(A) \vdash cbf(P) = cbf(Q)$. In ref. [5], the following two statements are proven (Theorem 6.4 and an auxiliary result in its proof), where $=_{cr}$ is the binary relation on C_A that specifies contractive valuation congruence:

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

By Lemma 15, these statements imply $cbf(P) = cbf(Q)$; that is, $P =_{cbf} Q$.

($\Leftarrow$) Assume $P =_{cbf} Q$. By Theorem 6, $\mathrm{CP}_{cr}(A) \vdash P = Q$. □

Hence, the relation $=_{cbf}$ is axiomatised by $\mathrm{CP}_{cr}(A)$, and is thus a congruence that coincides with contractive valuation congruence defined in [9]. We now define a transformation on evaluation trees that mimics the function cbf and prove that the equality of two such transformed trees characterises this congruence.

Definition 17. *The unary contractive evaluation function*

$$cse : \mathsf{C}_A \to \mathcal{T}_A$$

yields contractive evaluation trees and is defined by $cse(P) = cr(se(P))$.

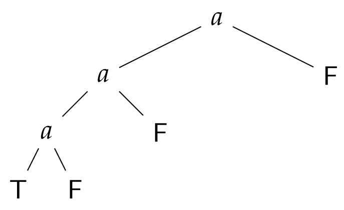
The auxiliary function $cr : \mathcal{T}_A \to \mathcal{T}_A$ *is defined as follows* ($a \in A$):

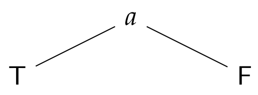
$$cr(B) = B \text{ for } B \in \{\mathsf{T}, \mathsf{F}\},$$
$$cr(X \trianglelefteq a \trianglerighteq Y) = cr(H_a(X)) \trianglelefteq a \trianglerighteq cr(J_a(Y)).$$

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

$$H_a(B) = J_a(B) = B \text{ for } B \in \{\mathsf{T}, \mathsf{F}\},$$
$$H_a(X \trianglelefteq b \trianglerighteq Y) = \begin{cases} H_a(X) & \text{if } b = a, \\ X \trianglelefteq b \trianglerighteq Y & \text{otherwise,} \end{cases}$$
$$J_a(X \trianglelefteq b \trianglerighteq Y) = \begin{cases} J_a(Y) & \text{if } b = a, \\ X \trianglelefteq b \trianglerighteq Y & \text{otherwise.} \end{cases}$$

As a simple example, we depict $se((a \triangleleft a \triangleright \mathsf{F}) \triangleleft a \triangleright \mathsf{F})$ and the contractive evaluation tree $cse((a \triangleleft a \triangleright \mathsf{F}) \triangleleft a \triangleright \mathsf{F})$:





The similarities between the evaluation function *cse* and the function *cbf* can be exploited, and we use the following lemma in the proof of the next completeness result.

Lemma 18. *For all* $P \in BF_A$, $cr(se(P)) = se(cf(P))$.

Proof. We first prove the following auxiliary result:

$$\text{For all } a \in A \text{ and } P \in BF_A,\ H_a(se(P)) = se(h_a(P)) \text{ and } J_a(se(P)) = se(j_a(P)). \tag{8}$$

Fix some $a \in A$. We prove (8) 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

$$\begin{aligned} H_a(se(Q \triangleleft a \triangleright R)) &= H_a(se(Q) \trianglelefteq a \trianglerighteq se(R)) \\ &= H_a(se(Q)) \\ &= se(h_a(Q)) && \text{by IH} \\ &= se(h_a(Q \triangleleft a \triangleright R)), \end{aligned}$$

and if $b \neq a$, then

$$\begin{aligned} H_a(se(Q \lhd b \rhd R)) &= H_a(se(Q) \unlhd b \unrhd se(R)) \\ &= se(Q) \unlhd b \unrhd se(R) \\ &= se(Q \lhd b \rhd R) \\ &= se(h_a(Q \lhd b \rhd R)). \end{aligned}$$

The second equality can be derived in a similar way, and this finishes the proof of (8).

We prove the lemma's statement by induction on $d(P)$. The base cases $P \in \{\mathsf{T}, \mathsf{F}\}$ follow immediately. Assume $P = Q \lhd a \rhd R$; then,

$$\begin{aligned} cr(se(Q \lhd a \rhd R)) &= cr(se(Q) \unlhd a \unrhd se(R)) \\ &= cr(H_a(se(Q))) \unlhd a \unrhd cr(J_a(se(R))) \\ &= cr(se(h_a(Q))) \unlhd a \unrhd cr(se(j_a(R))) && \text{by (8)} \\ &= se(cf(h_a(Q))) \unlhd a \unrhd se(cf(j_a(R))) && \text{by IH (and Lemma 14)} \\ &= se(cf(h_a(Q)) \lhd a \rhd cf(j_a(R))) \\ &= se(cf(Q \lhd a \rhd R)). \end{aligned}$$

□

Finally, we relate conditional statements by means of their contractive evaluation trees.

Definition 18. *Contractive se-congruence, notation $=_{cse}$, is defined on C_A as follows:*

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

The following characterisation result implies that $=_{cse}$ is a congruence relation on C_A.

Proposition 4. *For all $P, Q \in C_A$, $P =_{cse} Q \iff P =_{cbf} Q$.*

Proof. ($\Rightarrow$) Assume $cse(P) = cse(Q)$, and thus $cr(se(P)) = cr(se(Q))$. By Corollary 1, $cr(se(bf(P))) = cr(se(bf(Q)))$, and so by Lemma 18, $se(cf(bf(P))) = se(cf(bf(Q)))$. By Lemma 2 and auxiliary result (2) (see the proof of Lemma 10), it follows that $cf(bf(P)) = cf(bf(Q))$; that is, $P =_{cbf} Q$.

($\Leftarrow$) Assume $P =_{cbf} Q$; thus, $cf(bf(P)) = cf(bf(Q))$ and $se(cf(bf(P))) = se(cf(bf(Q)))$. By Lemma 18, $cr(se(bf(P))) = cr(se(bf(Q)))$. By Corollary 1, $se(bf(P)) = se(P)$ and $se(bf(Q)) = se(Q)$, so $cr(se(P)) = cr(se(Q))$: that is, $P =_{cse} Q$. □

Our final result in this section is a completeness result for contractive se-congruence.

Theorem 8 (Completeness of $\mathrm{CP}_{cr}(A)$)**.** *For all $P, Q \in C_A$,*

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

Proof. Combine Theorem 7 and Proposition 4. □

5. Evaluation Trees for Memorising Valuation Congruence

In ref. [5], Bergstra and Ponse introduced CP_{mem}, memorising CP, as the extension of CP with the following axiom:

$$\text{(CPmem)} \qquad x \lhd y \rhd (z \lhd u \rhd (v \lhd y \rhd w)) = x \lhd y \rhd (z \lhd u \rhd w).$$

Axiom (CPmem) expresses that the first evaluation value of y is memorised. More precisely, a "memorising evaluation" is one with the property that upon the evaluation of a compound propositional statement, the first evaluation value of each atom is memorised throughout the evaluation and can have no side effect that affects the evaluation of any other atom. We write CP_{mem} for the set $\mathrm{CP} \cup \{(\mathrm{CPmem})\}$ of axioms.

Replacing the variable y in axiom (CPmem) by $\mathsf{F} \triangleleft y \triangleright \mathsf{T}$ and/or the variable u by $\mathsf{F} \triangleleft u \triangleright \mathsf{T}$ yields all other memorising patterns:

$$\begin{aligned}
&\text{(CPm1)} \qquad && (z \triangleleft u \triangleright (w \triangleleft y \triangleright v)) \triangleleft y \triangleright x = (z \triangleleft u \triangleright w) \triangleleft y \triangleright x,\\
&\text{(CPm2)} \qquad && x \triangleleft y \triangleright ((v \triangleleft y \triangleright w) \triangleleft u \triangleright z) = x \triangleleft y \triangleright (w \triangleleft u \triangleright z),\\
&\text{(CPm3)} \qquad && ((w \triangleleft y \triangleright v) \triangleleft u \triangleright z) \triangleleft y \triangleright x = (w \triangleleft u \triangleright z) \triangleleft y \triangleright x.
\end{aligned}$$

Hence, the duality principle also holds in CP_{mem}. Furthermore, if we replace in axiom (CP-mem) u by F, we find the *contraction law*

$$x \triangleleft y \triangleright (v \triangleleft y \triangleright w) = x \triangleleft y \triangleright w, \tag{9}$$

and replacing y by $\mathsf{F} \triangleleft y \triangleright \mathsf{T}$ then yields the dual contraction law

$$(w \triangleleft y \triangleright v) \triangleleft y \triangleright x = w \triangleleft y \triangleright x. \tag{10}$$

Hence, CP_{mem} is an axiomatic extension of $\mathrm{CP}_{cr}(A)$.

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

Definition 19. *Let A' be a subset of A. Mem-basic forms over A' are inductively defined as follows:*

- T *and* F *are mem-basic forms over A';*
- *$P \triangleleft a \triangleright Q$ is a mem-basic form over A' if $a \in A'$ and P and Q are mem-basic forms over $A' \setminus \{a\}$.*

P is a mem-basic form if for some $A' \subset A$, P is a mem-basic form over A'.

Note that if A is finite, the number of mem-basic forms is also finite. It will turn out useful to define a function that transforms conditional statements into mem-basic forms.

Definition 20. *The mem-basic form function $mbf : C_A \to C_A$ is defined by*

$$mbf(P) = mf(bf(P)).$$

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

$$\begin{aligned}
mf(B) &= B \;\; \text{for } B \in \{\mathsf{T}, \mathsf{F}\},\\
mf(P \triangleleft a \triangleright Q) &= mf(\ell_a(P)) \triangleleft a \triangleright mf(r_a(Q)).
\end{aligned}$$

For $a \in A$, the auxiliary functions $\ell_a : BF_A \to BF_A$ (left a-reduction) and $r_a : BF_A \to BF_A$ (right a-reduction) are defined by

$$\ell_a(B) = r_a(B) = B \text{ for } B \in \{\mathsf{T}, \mathsf{F}\},$$
$$\ell_a(P \triangleleft b \triangleright Q) = \begin{cases} \ell_a(P) & \text{if } b = a, \\ \ell_a(P) \triangleleft b \triangleright \ell_a(Q) & \text{otherwise,} \end{cases}$$
$$r_a(P \triangleleft b \triangleright Q) = \begin{cases} r_a(Q) & \text{if } b = a, \\ r_a(P) \triangleleft b \triangleright r_a(Q) & \text{otherwise.} \end{cases}$$

Thus, mbf maps a conditional statement P to $bf(P)$ and then transforms $bf(P)$ according to the auxiliary functions mf, ℓ_a, and r_a. We will use the following inequalities.

Lemma 19. *For all $a \in A$ and $P \in BF_A$, $d(P) \geq d(\ell_a(P))$ and $d(P) \geq d(r_a(P))$.*

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

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

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

If $b \neq a$, then

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

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

Lemma 20. *For all $P \in C_A$, $mbf(P)$ is a mem-basic form.*

Proof. We first prove an auxiliary result:

$$\text{For all } P \in BF_A,\ mf(P) \text{ is a mem-basic form.} \tag{11}$$

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

$$mf(Q \triangleleft a \triangleright R) = mf(\ell_a(Q)) \triangleleft a \triangleright mf(r_a(R)),$$

which is a mem-basic form because by Lemma 19, $\ell_a(Q)$ and $r_a(R)$ are basic forms with depth smaller than or equal to n, so by the induction hypothesis, $mf(\ell_a(Q))$ is a mem-basic form over A_Q and $mf(r_a(R))$ is a mem-basic form over A_R for suitable subsets A_Q and A_R of A. Notice that by the definition of ℓ_a and r_a we can assume that the atom a does not occur in $A_Q \cup A_R$. Hence, $mf(\ell_a(Q)) \triangleleft a \triangleright mf(r_a(R))$ is a mem-basic form over $A_Q \cup A_R \cup \{a\}$, which completes the proof of (11).

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,

$$mbf(P \triangleleft Q \triangleright R) = mf(bf(P \triangleleft Q \triangleright R)) = mf(S)$$

For some basic form S by Lemma 4, and by (11), $mf(S)$ is a mem-basic form. □

With Lemma 20 we can easily prove the following result.

Proposition 5 (mbf is a normalisation function)**.** *For each $P \in C_A$, $mbf(P)$ is a mem-basic form, and for each mem-basic form P, $mbf(P) = P$.*

Proof. The first statement is Lemma 20. For the second statement, it suffices by Lemma 5 to prove that $mf(P) = P$. We prove this by induction on $d(P)$. The base cases $P \in \{\mathsf{T}, \mathsf{F}\}$ are trivial, and for the inductive case, assume $P = Q \triangleleft a \triangleright R$; thus, $mf(P) = mf(\ell_a(Q)) \triangleleft a \triangleright mf(r_a(R))$. Because P is a mem-basic form, Q and R are mem-basic forms in which a does not occur, and thus $\ell_a(Q) = Q$ and $r_a(R) = R$. By induction, $mf(Q) = Q$ and $mf(R) = R$, and thus $mf(P) = P$. □

Lemma 21. *For all $P \in BF_A$, $\mathrm{CP}_{mem} \vdash P = mf(P)$.*

Proof. We first prove an auxiliary result:

$$\text{For all } a \in A \text{ and } P, Q \in BF_A,\ \mathrm{CP}_{mem} \vdash\ P \triangleleft a \triangleright Q = P \triangleleft a \triangleright r_a(Q), \tag{12}$$

$$P \triangleleft a \triangleright Q = \ell_a(P) \triangleleft a \triangleright Q. \tag{13}$$

Fix some $a \in A$. We prove (12) by structural induction on Q. The base cases $Q \in \{\mathsf{T}, \mathsf{F}\}$ are trivial. For the inductive case $Q = Q_1 \triangleleft b \triangleright Q_2$ we have to distinguish the cases $b = a$ and $b \neq a$. If $b = a$, then $r_a(Q) = r_a(Q_2)$ and

$$\begin{aligned}
\mathrm{CP}_{mem} \vdash P \triangleleft a \triangleright (Q_1 \triangleleft a \triangleright Q_2) &= P \triangleleft a \triangleright Q_2 && \text{by (9)}\\
&= P \triangleleft a \triangleright r_a(Q_2) && \text{by IH}\\
&= P \triangleleft a \triangleright r_a(Q).
\end{aligned}$$

If $b \neq a$, then $r_a(Q) = r_a(Q_1) \triangleleft b \triangleright r_a(Q_2)$ and

$$\begin{aligned}
&\mathrm{CP}_{mem} \vdash P \triangleleft a \triangleright (Q_1 \triangleleft b \triangleright Q_2)\\
&\quad = P \triangleleft a \triangleright ((\mathsf{T} \triangleleft a \triangleright Q_1) \triangleleft b \triangleright (\mathsf{T} \triangleleft a \triangleright Q_2)) && \text{by (CPmem), (CPm2)}\\
&\quad = P \triangleleft a \triangleright ((\mathsf{T} \triangleleft a \triangleright r_a(Q_1)) \triangleleft b \triangleright (\mathsf{T} \triangleleft a \triangleright r_a(Q_2))) && \text{by IH (twice)}\\
&\quad = P \triangleleft a \triangleright (r_a(Q_1) \triangleleft b \triangleright r_a(Q_2)) && \text{by (CPmem), (CPm2)}\\
&\quad = P \triangleleft a \triangleright r_a(Q).
\end{aligned}$$

Auxiliary result (13) follows in a similar way with help of axioms (CPm1) and (CPm3).

The lemma's statement follows by induction on $d(P)$. The base cases $P \in \{\mathsf{T}, \mathsf{F}\}$ are trivial. For the inductive case, assume $P = Q \triangleleft a \triangleright R$. We derive

$$\begin{aligned}
\mathrm{CP}_{mem} \vdash Q \triangleleft a \triangleright R &= \ell_a(Q) \triangleleft a \triangleright r_a(R) && \text{by (12), (13)}\\
&= mf(\ell_a(Q)) \triangleleft a \triangleright mf(r_a(R)) && \text{by IH (and Lemma 19)}\\
&= mf(Q \triangleleft a \triangleright R).
\end{aligned}$$

□

Theorem 9. *For all $P \in C_A$, $\mathrm{CP}_{mem} \vdash P = mbf(P)$.*

Proof. By Theorem 2 and Corollary 1, $\mathrm{CP}_{mem} \vdash P = bf(P)$, and by Lemma 21, $\mathrm{CP}_{mem} \vdash bf(P) = mf(bf(P))$ and $mf(bf(P)) = mbf(P)$. □

Definition 21. *The binary relation $=_{mbf}$ on C_A is defined as follows:*

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

Theorem 10. *For all $P, Q \in C_A$, $\mathrm{CP}_{mem} \vdash P = Q \iff P =_{mbf} Q$.*

Proof. ($\Rightarrow$) Assume $\mathrm{CP}_{mem} \vdash P = Q$. Then, by Theorem 9, $\mathrm{CP}_{mem} \vdash mbf(P) = mbf(Q)$. In ref. [5], the following two statements are proven (Theorem 8.1 and Lemma 8.4), where $=_{mem}$ is the binary relation on C_A that specifies memorising valuation congruence:

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

By Lemma 20, these statements imply $mbf(P) = mbf(Q)$; that is, $P =_{mbf} Q$.
($\Leftarrow$) Assume $P =_{mbf} Q$. By Theorem 9, $\mathrm{CP}_{mem} \vdash P = Q$. □

Hence, the relation $=_{mbf}$ is axiomatised by CP_{mem} and is thus a congruence that coincides with memorising valuation congruence as defined in ref. [5]. We define a transformation on evaluation trees that mimics the function mbf, and prove that equality of two such transformed trees characterises the congruence that is axiomatised by CP_{mem}.

Definition 22. *The unary memorising evaluation function*

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

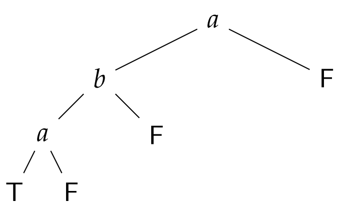
yields memorising evaluation trees and is defined by $mse(P) = m(se(P))$.
The auxiliary function $m : \mathcal{T}_A \to \mathcal{T}_A$ is defined as follows ($a \in A$):

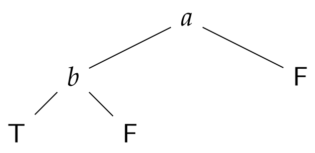
$$m(B) = B \text{ for } B \in \{\mathsf{T}, \mathsf{F}\},$$
$$m(X \trianglelefteq a \trianglerighteq Y) = m(L_a(X)) \trianglelefteq a \trianglerighteq m(R_a(Y)).$$

For $a \in A$, the auxiliary functions $L_a : \mathcal{T}_A \to \mathcal{T}_A$ (left a-reduction) and $R_a : \mathcal{T}_A \to \mathcal{T}_A$ (right a-reduction) are defined by

$$L_a(B) = R_a(B) = B \text{ for } B \in \{\mathsf{T}, \mathsf{F}\},$$
$$L_a(X \trianglelefteq b \trianglerighteq Y) = \begin{cases} L_a(X) & \text{if } b = a, \\ L_a(X) \trianglelefteq b \trianglerighteq L_a(Y) & \text{otherwise,} \end{cases}$$
$$R_a(X \trianglelefteq b \trianglerighteq Y) = \begin{cases} R_a(Y) & \text{if } b = a, \\ R_a(X) \trianglelefteq b \trianglerighteq R_a(Y) & \text{otherwise.} \end{cases}$$

As a simple example, we depict $se((a \triangleleft b \triangleright \mathsf{F}) \triangleleft a \triangleright \mathsf{F})$ and the memorising evaluation tree $mse((a \triangleleft b \triangleright \mathsf{F}) \triangleleft a \triangleright \mathsf{F})$:





The similarities between *mse* and the function *mbf* will of course be exploited.

Lemma 22. *For all $P \in BF_A$, $m(se(P)) = se(mf(P))$.*

Proof. We first prove an auxiliary result:

$$\text{For all } a \in A \text{ and } P \in BF_A,\ L_a(se(P)) = se(\ell_a(P)) \text{ and } R_a(se(P)) = se(r_a(P)). \tag{14}$$

Fix some $a \in A$. We prove (14) 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

$$\begin{aligned}
L_a(se(Q \triangleleft a \triangleright R)) &= L_a(se(Q) \trianglelefteq a \trianglerighteq se(R)) \\
&= L_a(se(Q)) \\
&= se(\ell_a(Q)) && \text{by IH} \\
&= se(\ell_a(Q \triangleleft a \triangleright R)),
\end{aligned}$$

and if $b \neq a$, then

$$\begin{aligned}
L_a(se(Q \triangleleft b \triangleright R)) &= L_a(se(Q) \trianglelefteq b \trianglerighteq se(R)) \\
&= L_a(se(Q)) \trianglelefteq b \trianglerighteq L_a(se(R)) \\
&= se(\ell_a(Q)) \triangleleft b \triangleright se(\ell_a(R)) && \text{by IH} \\
&= se(\ell_a(Q \triangleleft b \triangleright R)).
\end{aligned}$$

The second equality can be derived in a similar way, and this finishes the proof of (14).

We prove the lemma's statement by induction on $d(P)$. The base cases $P \in \{\mathsf{T}, \mathsf{F}\}$ follow immediately. Assume $P = Q \triangleleft a \triangleright R$; then,

$$\begin{aligned}
m(se(Q \triangleleft a \triangleright R)) &= m(se(Q) \trianglelefteq a \trianglerighteq se(R)) \\
&= m(L_a(se(Q))) \trianglelefteq a \trianglerighteq m(R_a(se(R))) \\
&= m(se(\ell_a(Q))) \trianglelefteq a \trianglerighteq m(se(r_a(R))) && \text{by (14)} \\
&= se(mf(\ell_a(Q))) \trianglelefteq a \trianglerighteq se(mf(r_a(R))) && \text{by IH (and Lemma 19)} \\
&= se(mf(\ell_a(Q)) \triangleleft a \triangleright mf(r_a(R))) \\
&= se(mf(Q \triangleleft a \triangleright R)).
\end{aligned}$$

□

Definition 23. *Memorising se-congruence, notation $=_{mse}$, is defined on C_A as follows:*

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

The following characterisation result immediately implies that $=_{mse}$ is indeed a congruence relation on C_A.

Proposition 6. *For all $P, Q \in C_A$, $P =_{mse} Q \iff P =_{mbf} Q$.*

Proof. ($\Rightarrow$) Assume $mse(P) = mse(Q)$; thus, $m(se(P)) = m(se(Q))$. By Corollary 1, $m(se(bf(P))) = m(se(bf(Q)))$, and so by Lemma 22,

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

By Lemma 2, it follows that $mf(bf(P)) = mf(bf(Q))$; that is, $P =_{mbf} Q$.

($\Leftarrow$) If $P =_{mbf} Q$, then $se(mf(bf(P))) = se(mf(bf(Q)))$, and by Lemma 22,

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

By Corollary 1, $m(se(P)) = m(se(Q))$; that is, $P =_{mse} Q$. □

We end this section with a completeness result for memorising se-congruence.

Theorem 11 (Completeness of CP_{mem})**.** *For all $P, Q \in C_A$,*

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

Proof. Combine Theorem 10 and Proposition 6. □

6. Evaluation Trees for Static Valuation Congruence

The most identifying axiomatic extension of CP we consider can be defined by adding the following axiom to CP_{mem}:

$$\text{(CPs)} \qquad \mathsf{F} \lhd x \rhd \mathsf{F} = \mathsf{F}.$$

So, the evaluation value of each atom in a conditional statement is memorised, and by axiom (CPs), no atom a can have a side effect because $\mathsf{T} \lhd (\mathsf{F} \lhd a \rhd \mathsf{F}) \rhd P = \mathsf{T} \lhd \mathsf{F} \rhd P = P$ for all $P \in C_A$. We write CP_s for the set of these axioms, and thus

$$\mathrm{CP}_s = \mathrm{CP}_{mem} \cup \{\text{(CPs)}\} = \mathrm{CP} \cup \{\text{(CPmem)}, \text{(CPs)}\}.$$

Observe that $\mathrm{CP}_s \vdash \mathsf{T} = \mathsf{T} \lhd (\mathsf{F} \lhd x \rhd \mathsf{F}) \rhd \mathsf{T} = (\mathsf{T} \lhd \mathsf{F} \rhd \mathsf{T}) \lhd x \rhd (\mathsf{T} \lhd \mathsf{F} \rhd \mathsf{T}) = \mathsf{T} \lhd x \rhd \mathsf{T}$; the duality principle holds in CP_s. The following lemma is a direct consequence of axiom (CPs).

Lemma 23. $\mathrm{CP}_s \vdash x \lhd y \rhd x = x$.

Proof.

$$\begin{aligned} \mathrm{CP}_s \vdash x \lhd y \rhd x &= (\mathsf{T} \lhd \mathsf{F} \rhd x) \lhd y \rhd (\mathsf{T} \lhd \mathsf{F} \rhd x) && \text{by (CP2)} \\ &= \mathsf{T} \lhd (\mathsf{F} \lhd y \rhd \mathsf{F}) \rhd x && \text{by (CP4)} \\ &= x. && \text{by (CPs), (CP2)} \end{aligned}$$

□

A simple example on CP_s illustrates how the order of evaluation of x and y in $x \triangleleft y \triangleright \mathsf{F}$ can be swapped:

$$x \triangleleft y \triangleright \mathsf{F} = y \triangleleft x \triangleright \mathsf{F}. \tag{15}$$

Equation (15) can be derived as follows:

$$\begin{aligned}
\mathrm{CP}_s \vdash x \triangleleft y \triangleright \mathsf{F} &= ((\mathsf{T} \triangleleft x \triangleright \mathsf{F}) \triangleleft y \triangleright \mathsf{F}) \triangleleft x \triangleright ((\mathsf{T} \triangleleft x \triangleright \mathsf{F}) \triangleleft y \triangleright \mathsf{F}) && \text{by (CP3), Lemma 23}\\
&= (\mathsf{T} \triangleleft y \triangleright \mathsf{F}) \triangleleft x \triangleright (\mathsf{F} \triangleleft y \triangleright \mathsf{F}) && \text{by (CPm3), (CPm2)}\\
&= y \triangleleft x \triangleright \mathsf{F}. && \text{by (CP3), (CPs)}
\end{aligned}$$

In ref, [5], Bergstra and Ponse defined CP_{st} as the extension of CP with the following two axioms:

$$\begin{aligned}
&\text{(CPstat)} && (x \triangleleft y \triangleright z) \triangleleft u \triangleright v = (x \triangleleft u \triangleright v) \triangleleft y \triangleright (z \triangleleft u \triangleright v),\\
&\text{(the contraction law (10))} && (x \triangleleft y \triangleright z) \triangleleft y \triangleright u = x \triangleleft y \triangleright u.
\end{aligned}$$

Axiom (CPstat) expresses how the order of evaluation of u and y can be swapped. Because we will rely on the results for CP_{st} recorded in ref. [5], we first prove the following result.

Proposition 7. *The axiom sets* CP_{st} *and* CP_s *are equally strong.*

Proof. We show that all axioms in the one set are derivable from the other set. We first prove that the axiom (CPmem) is derivable from CP_{st}:

$$\begin{aligned}
&\mathrm{CP}_{st} \vdash x \triangleleft y \triangleright (z \triangleleft u \triangleright (v \triangleleft y \triangleright w))\\
&\quad = x \triangleleft y \triangleright ((v \triangleleft y \triangleright w) \triangleleft (\mathsf{F} \triangleleft u \triangleright \mathsf{T}) \triangleright z) && \text{by (CP4), (CP2), (CP1)}\\
&\quad = x \triangleleft y \triangleright ((v \triangleleft (\mathsf{F} \triangleleft u \triangleright \mathsf{T}) \triangleright z) \triangleleft y \triangleright (w \triangleleft (\mathsf{F} \triangleleft u \triangleright \mathsf{T}) \triangleright z)) && \text{by (CPstat)}\\
&\quad = x \triangleleft y \triangleright (w \triangleleft (\mathsf{F} \triangleleft u \triangleright \mathsf{T}) \triangleright z) && \text{by (9)}\\
&\quad = x \triangleleft y \triangleright (z \triangleleft u \triangleright w), && \text{by (CP4), (CP2), (CP1)}
\end{aligned}$$

where the contraction law (9) is derivable from CP_{st}; replace y by $\mathsf{F} \triangleleft y \triangleright \mathsf{T}$ in (10). Hence, $\mathrm{CP}_s \vdash$ (CPmem). If $u = v = \mathsf{F}$ in axiom (CPstat), we find $\mathsf{F} = \mathsf{F} \triangleleft y \triangleright \mathsf{F}$, and hence $\mathrm{CP}_{st} \vdash \mathrm{CP}_s$.

In order to show that $\mathrm{CP}_s \vdash \mathrm{CP}_{st}$ we have to derive $\mathrm{CP}_s \vdash$ (CPstat):

$$\begin{aligned}
\mathrm{CP}_s \vdash\ (x \triangleleft y \triangleright z) \triangleleft u \triangleright v &= (x \triangleleft y \triangleright (z \triangleleft u \triangleright v)) \triangleleft u \triangleright v && \text{by (CPm1)}\\
&= (x \triangleleft y \triangleright (z \triangleleft u \triangleright v)) \triangleleft u \triangleright (z \triangleleft u \triangleright v) && \text{by (9)}\\
&= x \triangleleft (y \triangleleft u \triangleright \mathsf{F}) \triangleright (z \triangleleft u \triangleright v) && \text{by (CP4), (CP2)}\\
&= x \triangleleft (u \triangleleft y \triangleright \mathsf{F}) \triangleright (z \triangleleft u \triangleright v) && \text{by (15)}\\
&= (x \triangleleft u \triangleright (z \triangleleft u \triangleright v)) \triangleleft y \triangleright (z \triangleleft u \triangleright v) && \text{by (CP4), (CP2)}\\
&= (x \triangleleft u \triangleright v) \triangleleft y \triangleright (z \triangleleft u \triangleright v). && \text{by (9)}
\end{aligned}$$

□

Given a finite, ordered subset of atoms we define a proper subset of basic forms with the property that each propositional statement over these atoms can be proven equal to such a basic form.

Definition 24. *St-basic forms over $\sigma \in A^s$ are defined as follows:*

- T *and* F *are st-basic forms over* ϵ*;*
- $P \triangleleft a \triangleright Q$ *is an st-basic form over* $a\rho \in A^s$ *if* P *and* Q *are st-basic forms over* ρ*.*

P is an st-basic form if for some $\sigma \in A^s$, P is an st-basic form over σ.

For example, an st-basic form over $ab \in A^s$ has the following form:

$$(B_1 \triangleleft b \triangleright B_2) \triangleleft a \triangleright (B_3 \triangleleft b \triangleright B_4)$$

with $B_i \in \{\mathsf{T}, \mathsf{F}\}$. For $\sigma = a_1 a_2 \cdots a_n \in A^s$, there exist 2^{2^n} different st-basic forms over σ.

It will be useful to define a function that transforms conditional statements to st-basic forms.

Definition 25. *Let $\sigma \in A^s$. Then, C_σ is the set of closed terms in C_A with atoms in σ, and BF_σ is the subset of basic forms in C_σ. The conditional statement $E_\sigma \in BF_\sigma$ is defined as*

$$E_\epsilon = \mathsf{F} \quad \textit{and, if } \sigma = a\rho, \quad E_\sigma = E_\rho \triangleleft a \triangleright E_\rho.$$

So, for each $\sigma \in A^s$, E_σ is an st-basic form over σ in which the constant T does not occur, e.g.,

$$E_{ab} = (\mathsf{F} \triangleleft b \triangleright \mathsf{F}) \triangleleft a \triangleright (\mathsf{F} \triangleleft b \triangleright \mathsf{F}).$$

Definition 26. *The st-basic form function $sbf_\sigma : C_\sigma \to C_\sigma$ is defined by*

$$sbf_\sigma(P) = mbf(\mathsf{T} \triangleleft E_\sigma \triangleright P),$$

where mbf is defined in Definition 20.

For example, $sbf_{ab}(a) = (\mathsf{T} \triangleleft b \triangleright \mathsf{T}) \triangleleft a \triangleright (\mathsf{F} \triangleleft b \triangleright \mathsf{F})$,
$sbf_{ba}(a) = (\mathsf{T} \triangleleft a \triangleright \mathsf{F}) \triangleleft b \triangleright (\mathsf{T} \triangleleft a \triangleright \mathsf{F})$.

The reason that $sbf_\sigma(P)$ is defined relative to some $\sigma \in A^s$ that covers the alphabet of P is that in order to prove completeness of CP_s (and CP_{st}), we need to be able to relate conditional statements that contain different sets of atoms, but have equal st-basic forms for all appropriate σ, such as

$$sbf_{ab}(\mathsf{F}) = sbf_{ab}(\mathsf{F} \triangleleft a \triangleright \mathsf{F}) = sbf_{ab}(\mathsf{F} \triangleleft b \triangleright \mathsf{F}) = (\mathsf{F} \triangleleft b \triangleright \mathsf{F}) \triangleleft a \triangleright (\mathsf{F} \triangleleft b \triangleright \mathsf{F}).$$

Lemma 24. *Let $\sigma \in A^s$. For all $P \in C_A$, $\mathrm{CP}_s \vdash P = \mathsf{T} \triangleleft E_\sigma \triangleright P$.*

Proof. By induction on the structure of σ. If $\sigma = \epsilon$, then $E_\sigma = \mathsf{F}$ and by axiom (CP2), $\mathrm{CP}_s \vdash P = \mathsf{T} \triangleleft E_\epsilon \triangleright P$. If $\sigma = a\rho$ for some $a \in A$ and $\rho \in A^s$, then $E_\sigma = E_\rho \triangleleft a \triangleright E_\rho$, and hence

$$\begin{aligned}
\mathrm{CP}_s \vdash P &= P \triangleleft a \triangleright P && \text{by Lemma 23}\\
&= (\mathsf{T} \triangleleft E_\rho \triangleright P) \triangleleft a \triangleright (\mathsf{T} \triangleleft E_\rho \triangleright P) && \text{by IH}\\
&= \mathsf{T} \triangleleft (E_\rho \triangleleft a \triangleright E_\rho) \triangleright P. && \text{by (CP4)}
\end{aligned}$$

□

Lemma 25. *Let $\sigma \in A^s$. For all $P \in C_\sigma$, $sbf_\sigma(P)$ is an st-basic form.*

Proof. We first prove an auxiliary result:

$$\begin{aligned}&\text{For all } a \in A \text{ and } P, Q \in BF_A,\ \ell_a(P[\mathsf{F} \mapsto Q]) = (\ell_a(P))[\mathsf{F} \mapsto \ell_a(Q)]\\ &\text{and } \ r_a(P[\mathsf{F} \mapsto Q]) = (r_a(P))[\mathsf{F} \mapsto r_a(Q)].\end{aligned} \tag{16}$$

Both equalities follow easily by induction on the structure of P and we only show the inductive case for the first one. Choose $a \in A$. If $P = P_1 \lhd a \rhd P_2$, then $\ell_a(P) = \ell_a(P_1)$ and

$$\begin{aligned}&\ell_a(P[\mathsf{F} \mapsto Q]) = \ell_a(P_1[\mathsf{F} \mapsto Q] \lhd a \rhd P_2[\mathsf{F} \mapsto Q]) = \ell_a(P_1[\mathsf{F} \mapsto Q])\\ &\overset{\text{IH}}{=} (\ell_a(P_1))[\mathsf{F} \mapsto \ell_a(Q)] = (\ell_a(P))[\mathsf{F} \mapsto \ell_a(Q)],\end{aligned}$$

and if $P = P_1 \lhd b \rhd P_2$ with $b \neq a$, then $\ell_a(P) = \ell_a(P_1) \lhd b \rhd \ell_a(P_2)$ and

$$\begin{aligned}&\ell_a(P[\mathsf{F} \mapsto Q]) = \ell_a(P_1[\mathsf{F} \mapsto Q] \lhd b \rhd P_2[\mathsf{F} \mapsto Q]) = \ell_a(P_1[\mathsf{F} \mapsto Q]) \lhd b \rhd \ell_a(P_1[\mathsf{F} \mapsto Q])\\ &\overset{\text{IH}}{=} (\ell_a(P_1))[\mathsf{F} \mapsto \ell_a(Q)] \lhd b \rhd (\ell_a(P_2))[\mathsf{F} \mapsto \ell_a(Q)] = (\ell_a(P))[\mathsf{F} \mapsto \ell_a(Q)].\end{aligned}$$

We prove the lemma's statement by induction on the structure of σ. If $\sigma = \epsilon$, then each $P \in C_\sigma$ contains no atoms. Hence, $bf(P) \in \{\mathsf{T}, \mathsf{F}\}$. If $bf(P) = \mathsf{T}$, then

$$sbf_\epsilon(P) = mbf(\mathsf{T} \lhd \mathsf{F} \rhd P) = mf(bf(\mathsf{T} \lhd \mathsf{F} \rhd P)) = mf(bf(P)) = \mathsf{T},$$

which is an st-basic form over ϵ. The case for $bf(P) = \mathsf{F}$ is similar. If $\sigma = a\rho$ for some $a \in A$ and $\rho \in A^s$, then for each $P \in C_\sigma$,

$$\begin{aligned}sbf_\sigma(P) &= mbf(\mathsf{T} \lhd E_\sigma \rhd P)\\ &= mf(bf(\mathsf{T} \lhd E_\sigma \rhd P))\\ &= mf(E_\sigma[\mathsf{F} \mapsto bf(P)]) && \text{by Lemma 5}\\ &= mf(E_\rho[\mathsf{F} \mapsto bf(P)] \lhd a \rhd E_\rho[\mathsf{F} \mapsto bf(P)])\\ &= mf(\ell_a(E_\rho[\mathsf{F} \mapsto bf(P)])) \lhd a \rhd mf(r_a(E_\rho[\mathsf{F} \mapsto bf(P)]))\\ &= mf(\ell_a(E_\rho)[\mathsf{F} \mapsto \ell_a(bf(P))]) \lhd a \rhd mf(r_a(E_\rho)[\mathsf{F} \mapsto r_a(bf(P))]) && \text{by (16)}\\ &= mf(E_\rho[\mathsf{F} \mapsto \ell_a(bf(P))]) \lhd a \rhd mf(E_\rho[\mathsf{F} \mapsto r_a(bf(P))]) && \text{by } a \notin C_\rho\\ &= mf(E_\rho[\mathsf{F} \mapsto bf(\ell_a(bf(P)))]) \lhd a \rhd mf(E_\rho[\mathsf{F} \mapsto bf(r_a(bf(P)))]) && \text{by Lemma 5}\\ &= mf(bf(\mathsf{T} \lhd E_\rho \rhd \ell_a(bf(P)))) \lhd a \rhd mf(bf(\mathsf{T} \lhd E_\rho \rhd r_a(bf(P))))\\ &= sbf_\rho(\ell_a(bf(P))) \lhd a \rhd sbf_\rho(r_a(bf(P))), && \text{by IH} \quad (17)\end{aligned}$$

where (17) follows because $\ell_a(bf(P))$ and $r_a(bf(P))$ are conditional statements in C_ρ (thus, not containing a), and so by induction, $sbf_\rho(\ell_a(bf(P)))$ and $sbf_\rho(r_a(bf(P)))$ are st-basic forms over ρ. Hence, $sbf_\sigma(P)$ is an st-basic form over σ. □

With Lemma 25 we can easily prove the following result.

Proposition 8 (sbf_σ **is a normalisation function).** *Let $\sigma \in A^s$. For each $P \in C_\sigma$, $sbf_\sigma(P)$ is an st-basic form over σ, and for each st-basic form P over σ, $sbf_\sigma(P) = P$.*

Proof. The first statement is Lemma 25. We prove the second statement by induction on the structure of σ. If $\sigma = \epsilon$, $sbf_\sigma(P) = P$ by definition.

If $\sigma = a\rho$, then $P = P_1 \triangleleft a \triangleright P_2$ with P_i st-basic forms over ρ; thus, $\ell_a(P_1) = P_1$ and $r_a(P_2) = P_2$. For brevity, we identify below $bf(Q)$ and Q for all $Q \in BF_A$:

$$\begin{aligned}
sbf_\sigma(P) &= mbf(\mathsf{T} \triangleleft E_{a\rho} \triangleright (P_1 \triangleleft a \triangleright P_2)) \\
&= mf(E_{\rho a}[\mathsf{F} \mapsto P_1 \triangleleft a \triangleright P_2]) && \text{as above} \\
&= mf(E_\rho[\mathsf{F} \mapsto P_1 \triangleleft a \triangleright P_2] \triangleleft a \triangleright E_\rho[\mathsf{F} \mapsto P_1 \triangleleft a \triangleright P_2]) \\
&= mf(\ell_a(E_\rho[\mathsf{F} \mapsto P_1 \triangleleft a \triangleright P_2]) \triangleleft a \triangleright mf(r_a(E_\rho[\mathsf{F} \mapsto P_1 \triangleleft a \triangleright P_2])) \\
&= mf(\ell_a(E_\rho)[\mathsf{F} \mapsto \ell_a(P_1 \triangleleft a \triangleright P_2])) \triangleleft a \triangleright \\
&\qquad mf(r_a(E_\rho)[\mathsf{F} \mapsto r_a(P_1 \triangleleft a \triangleright P_2])) && \text{by (16)} \\
&= mf(E_\rho[\mathsf{F} \mapsto P_1]) \triangleleft a \triangleright mf(E_\rho[\mathsf{F} \mapsto P_2]) && \text{by } a \notin C_\rho \\
&= sbf_\rho(P_1) \triangleleft a \triangleright sbf_\rho(P_2) \\
&= P_1 \triangleleft a \triangleright P_2. && \text{by IH}
\end{aligned}$$

□

Lemma 26. *Let $\sigma \in A^s$. For all $P \in C_\sigma$, $\mathrm{CP}_s \vdash P = sbf_\sigma(P)$.*

Proof. By Lemma 24, $\mathrm{CP}_s \vdash P = \mathsf{T} \triangleleft E_\sigma \triangleright P$. By Theorem 9, $\mathrm{CP}_s \vdash \mathsf{T} \triangleleft E_\sigma \triangleright P = mbf(\mathsf{T} \triangleleft E_\sigma \triangleright P)$, and hence $\mathrm{CP}_s \vdash P = sbf_\sigma(P)$. □

Definition 27. *Let $\sigma \in A^s$. The binary relation $=_{sbf,\sigma}$ on C_σ is defined as follows:*

$$P =_{sbf,\sigma} Q \iff sbf_\sigma(P) = sbf_\sigma(Q).$$

Theorem 12. *Let $\sigma \in A^s$. For all $P, Q \in C_\sigma$, $\mathrm{CP}_s \vdash P = Q \iff P =_{sbf,\sigma} Q$.*

Proof. ($\Rightarrow$) Assume $\mathrm{CP}_s \vdash P = Q$. Then, by Lemma 26, $\mathrm{CP}_s \vdash sbf_\sigma(P) = sbf_\sigma(Q)$, and by Proposition 7, $\mathrm{CP}_{st} \vdash sbf_\sigma(P) = sbf_\sigma(Q)$. In ref. [5], the following two statements are proven (Theorem 9.1 and an auxiliary result in its proof), where $=_{st}$ is the binary relation on C_A that specifies static valuation congruence:

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

By Lemma 25, these statements imply $sbf_\sigma(P) = sbf_\sigma(Q)$, and thus $P =_{sbf,\sigma} Q$.

($\Leftarrow$) Assume $P =_{sbf,\sigma} Q$, thus $\mathsf{T} \triangleleft E_\sigma \triangleright P =_{mbf} \mathsf{T} \triangleleft E_\sigma \triangleright Q$. By Theorem 10, $\mathrm{CP}_{mem} \vdash \mathsf{T} \triangleleft E_\sigma \triangleright P = \mathsf{T} \triangleleft E_\sigma \triangleright Q$, and by Lemma 24, this implies $\mathrm{CP}_s \vdash P = Q$. □

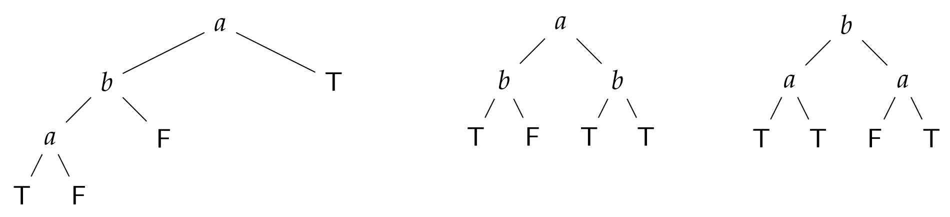
Hence, the relation $=_{sbf,\sigma}$ is a congruence on C_σ that is axiomatised by CP_s and coincides with static valuation congruence as defined in ref. [5]. Below we define static evaluation trees with help of the function sbf_σ and prove that the equality of two such trees characterises this congruence.

Definition 28. *Let $\sigma \in A^s$. The unary static evaluation function $sse_\sigma : C_\sigma \to \mathcal{T}_A$ yields static evaluation trees and is defined as follows:*

$$sse_\sigma(P) = se(sbf_\sigma(P)),$$

where sbf_σ is defined in Definition 26 and se in Definition 3.

As an example, let $P = (a \lhd b \rhd \mathsf{F}) \lhd a \rhd \mathsf{T}$. We depict $se(P)$ at the left-hand side. The static evaluation tree $sse_{ab}(P)$ is depicted in the middle, and the static evaluation tree $sse_{ba}(P)$ is depicted at the right-hand side:



The two different static evaluation trees correspond to the different ways in which one can present truth tables for P, that is, the different possible orderings of the valuation values of the atoms occurring in P:

a	b	$(a \lhd b \rhd \mathsf{F}) \lhd a \rhd \mathsf{T}$
T	T	T
T	F	F
F	T	T
F	F	T

b	a	$(a \lhd b \rhd \mathsf{F}) \lhd a \rhd \mathsf{T}$
T	T	T
T	F	T
F	T	F
F	F	T

The similarities between sse_σ and the function sbf_σ can be exploited and lead to a completeness result.

Definition 29. *Let $\sigma \in A^s$. Static valuation congruence over σ, notation $=_{sse,\sigma}$, is defined on C_σ as follows:*

$$P =_{sse,\sigma} Q \iff sse_\sigma(P) = sse_\sigma(Q).$$

The following characterisation result immediately implies that for all $\sigma \in A^s$, $=_{sse,\sigma}$ is indeed a congruence relation on C_σ.

Proposition 9. *Let $\sigma \in A^s$. For all $P, Q \in C_\sigma$, $P =_{sse,\sigma} Q \iff P =_{sbf,\sigma} Q$.*

Proof. This follows by Proposition 6. □

We end this section with a completeness result for static valuation congruence.

Theorem 13 (Completeness of CP_s)**.** *Let $\sigma \in A^s$. For all $P, Q \in C_\sigma$,*

$$\mathrm{CP}_s \vdash P = Q \iff P =_{sse,\sigma} Q.$$

Proof. Combine Theorem 12 and Proposition 9. □

7. Conclusions

In 2011, Bergstra and Ponse introduced proposition algebra in ref. [5], which is based on Hoare's conditional connective $x \lhd y \rhd z$ and the constants T and F. Four simple equational axioms specify "free valuation congruence" ($=_{fr}$) and the addition of two axioms specifies "static valuation congruence" ($=_{st}$), which represents propositional logic. Some other valuation congruences that lie in between free and static valuation congruence were also defined, and in this paper we paid attention to repetition-proof, contractive, and memorising valuation congruence, in symbols $=_{rp}$, $=_{cr}$, and $=_{mem}$.

In ref. [10], Bergstra and Ponse introduced an alternative valuation semantics for proposition algebra in the form of *Hoare–McCarthy algebras* (HMA's), which is more elegant than the semantical framework provided in ref. [5]; 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 [6] to characterise free valuation congruence as the relation "se-congruence", the relation $=_{se}$ in 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 [10] and the semantics defined in ref. [5]. By Theorem 1, $=_{se}$ coincides with free valuation congruence $=_{fr}$ because both relations are axiomatised by CP (see [5] (Theorems 4.4 and 6.2)). An advantage of "evaluation tree semantics" is that for a given conditional statement P, the evaluation tree $se(P)$ determines all relevant evaluations, so $P =_{se} Q$ 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 se-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 clear 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

$$\mathrm{CP}_{rp}(A) \vdash P = Q \iff P =_{rpse} Q, \tag{18}$$

by which $=_{rpse}$ coincides with the repetition-proof valuation congruence as defined in ref. [5] because both are axiomatised by $\mathrm{CP}_{rp}(A)$ (see [5] (Theorem 6.3)). So, by equivalence (18), $=_{rpse}$ is a *congruence* relation on C_A. However, we could not find a direct proof of this fact and chose to simulate the transformation *rpse* by the transformation *rpbf* on conditional statements, and to prove that the associated equivalence relation $=_{rpbf}$ is axiomatised by $\mathrm{CP}_{rp}(A)$, and is hence a congruence. This is Theorem 4, the proof of which depends on ref. [5] (Theorem 6.3; this theorem requires that $|A| > 1$, and so does the HMA approach in ref. [10]) *and* on Theorem 3; that is,

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

In order to prove (18) (which is Theorem 5), it is thus sufficient to prove that $=_{rpbf}$ and $=_{rpse}$ coincide, and this is Proposition 2. Although it remains a challenge to find a direct and elegant proof of equivalence (18), we can conclude that repetition-proof evaluation trees and the valuation congruence $=_{rpse}$ provide full-fledged, simple, and elegant semantics for $\mathrm{CP}_{rp}(A)$.

The structure of our completeness proofs of the axiomatisations of the other se-congruences is very similar, although the case for static se-congruence requires a slightly more complex proof (we return to this point below). Moreover, these axiomatisations are incremental: the axiom systems $\mathrm{CP}_{rp}(A)$, up to and including CP_s, all share the axioms of CP, and each succeeding system is defined by the addition of either one or two axioms (schemes), in most cases making the previously added axiom(s) redundant (in Appendix A, we provide some independence results). Given some $\sigma \in A^s$, this implies that in C_σ,

$$=_{se} \;\subsetneq\; =_{rpse} \;\subsetneq\; =_{cse} \;\subsetneq\; =_{mse} \;\subsetneq\; =_{sse,\sigma},$$

where all these inclusions are proper. We conclude that for the se-congruences $=_{se}$ up to $=_{mse}$, the associated evaluation trees provide full-fledged, simple, and elegant semantics.

The case for static se-congruence over C_σ for some $\sigma \in A^s$ is somewhat more involved. This semantics coincides with any standard semantics of propositional logic in the following sense:

$$P =_{sse,\sigma} Q \quad \text{if, and only if,} \quad \overline{P} \leftrightarrow \overline{Q} \text{ is a tautology in propositional logic,}$$

where $\overline{P}$ and $\overline{Q}$ refer to Hoare's definition ref. [2]:

$$\overline{x \triangleleft y \triangleright z} = (\overline{x} \wedge \overline{y}) \vee (\neg\overline{y} \wedge \overline{z}), \quad \overline{a} = a \text{ for } a \in C_\sigma, \quad \overline{\mathsf{T}} = \mathsf{T}, \quad \overline{\mathsf{F}} = \mathsf{F}.$$

Let $\sigma \in A^s$ and $a \in C_\sigma$. The fact that $=_{sse,\sigma}$ is stronger than $=_{mse}$ is immediately clear:

$$\mathsf{F} \triangleleft a \triangleright \mathsf{F} =_{sse,\sigma} \mathsf{F},$$

where it is easy to see that $\mathsf{F} \triangleleft a \triangleright \mathsf{F} \neq_{mse} \mathsf{F}$. Our proof that CP_s (and thus CP_{st} as defined in ref. [5]) is an axiomatisation of static se-congruence is slightly more complex than those for the other axiomatisations because in this case the evaluation of a conditional statement P does not enforce a canonical order for the evaluation of its atoms, and therefore such an ordering should be fixed beforehand in order to construct an adequate evaluation tree. To this purpose, we can use any $\sigma \in A^s$ that covers the atoms in P.

We continue with a brief digression on *short-circuit logic*, which we defined in ref. [11] (see Bergstra and Ponse [10] 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 && (called "logical AND" in the programming language C), are often defined in the following way:

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

independent of the precise syntax of P and Q; hence, $P \mathbin{\&\&} Q =_{\text{def}} Q \triangleleft P \triangleright \mathsf{F}$. It easily follows that && is associative.

In a similar way, negation can be defined by $\neg P =_{\text{def}} \mathsf{F} \triangleleft P \triangleright \mathsf{T}$. In ref. [11], we focus on this question:

Question 1. *Which are the logical laws that characterise short-circuit evaluation of binary propositional connectives?*

A first approach to this question is to adopt the conditional as an auxiliary operator, as is the approach in ref. [10,11], and to answer Question 1 using definitions of the binary propositional connectives as above and the axiomatisation for the se-congruence of interest in proposition algebra (or, if "mixed conditional statements" are at stake, axiomatisations for the appropriate se-congruences). An alternative and more direct approach to Question 1 is to establish axiomatisations for short-circuited binary connectives in which the conditional is *not* used. For free se-congruence, a complete equational axiomatisation of short-circuited binary propositional connectives is provided by Staudt in ref. [6], where

$$se(P \mathbin{\&\&} Q) =_{\text{def}} se(P)[\mathsf{T} \mapsto se(Q)] \quad \text{and} \quad se(\neg P) =_{\text{def}} se(P)[\mathsf{T} \mapsto \mathsf{F}, \mathsf{F} \mapsto \mathsf{T}]$$

(and where the function *se* is also defined for short-circuited disjunction). The completeness proof is based on decomposition properties of evaluation trees. For repetition-proof and contractive se-congruence we conjecture that a finite equational axiomatisation of the short-circuited binary propositional connectives does not exist if $|A| > 2$. We continue with two examples on the use of $\mathrm{CP}_{rp}(A)$ and $\mathrm{CP}_{cr}(A)$, both taken from [11].

Example 3 ([11], Example 4)**.** *Let A be a set of atoms of the form* $(\texttt{e == e}')$ *and* $(\texttt{n = e})$ *with* $\texttt{n}$ *some initialized integer program variable and* $\texttt{e}, \texttt{e}'$ *arithmetical expressions over the integers that may contain* $\texttt{n}$*. Assume that* $(\texttt{e == e}')$ *evaluates to true if* $\texttt{e}$ *and* $\texttt{e}'$ *represent the same value, and* $(\texttt{n = e})$ *always evaluates to true with the effect that the value of* $\texttt{e}$ *is assigned to* $\texttt{n}$*. Then, these atoms satisfy the axioms of* $\mathrm{CP}_{rp}(A)$ *(of course, not all equations that are valid in the setting of Example 3 follow from* $\mathrm{CP}_{rp}(A)$*, e.g.,* $\mathrm{CP}_{rp}(A) \nvdash (\texttt{0 == 0}) = \mathsf{T}$*). Note that if* $\texttt{n}$ *has initial value* $\texttt{0}$ *or* $\texttt{1}$*,* $((\texttt{n = n+1})\ \&\&\ (\texttt{n = n+1}))\ \&\&\ (\texttt{n == 2})$*, and* $(\texttt{n = n+1})\ \&\&\ (\texttt{n == 2})$ *evaluate to different results, so the atom* $(\texttt{n = n+1})$ *does not satisfy the law "*$a\ \&\&\ a = a$*", by which this is a typical example of the repetition-proof characteristic of* $\mathrm{CP}_{rp}(A)$*.*

Example 4 ([11], Example 3)**.** *An example in the same style that illustrates the use of* $\mathrm{CP}_{cr}(A)$ *concerns atoms that define manipulation of Boolean registers. Let* $A = \{(\texttt{set}:\texttt{i}:\texttt{j}), (\texttt{eq}:\texttt{i}:\texttt{j}) \mid \texttt{1} \leq \texttt{i} \leq \texttt{n},\ \texttt{j} \in \{\texttt{T},\texttt{F}\}\}$ *with* $\texttt{n}$ *the number of registers and* $\texttt{j}$ *the value of registers.*

- *Upon evaluation, atom* $(\texttt{set}:\texttt{i}:\texttt{j})$ *can have a side effect (it sets register* $\texttt{i}$ *to value* $\texttt{j}$*) and always yields true.*
- *Upon evaluation, atom* $(\texttt{eq}:\texttt{i}:\texttt{j})$ *has no side effect and yields only true if register* $\texttt{i}$ *has value* $\texttt{j}$*.*

It is clear that the axioms of $\mathrm{CP}_{cr}(A)$ *are valid, as is their consequence "*$a\ \&\&\ a = a$*" for all* $a \in A$*. However,* $t\ \&\&\ t = t$ *is not true for all* $t \in \Sigma_{\mathrm{CP}}(A)$*. Let* $t = (\texttt{eq}:\texttt{1}:\texttt{F})\ \&\&\ (\texttt{set}:\texttt{1}:\texttt{T})$*. If register* $\texttt{1}$ *has value* $\texttt{F}$*, then evaluation of* t *yields true, while evaluation of* $t\ \&\&\ t$ *yields false.*

Furthermore, we note that the se-congruences $=_{se}$ up to $=_{cse}$ can be used as a basis for systematic analysis of the kind of *side effects* that may occur upon the evaluation of short-circuited connectives as in Examples 3 and 4, and we quote these words of Parnas [12]:

> "Most mainline methods disparage side effects as a bad programming practice. Yet even in well-structured, reliable software, many components do have side effects; side effects are very useful in practice. It is time to investigate methods that deal with side effects as the normal case."

Although side effects are well understood in programming, see e.g., Black and Windley [13], they are often explained informally and systematic surveys and analyses do not yet appear to be available.

We finally note that the value of CP_{mem} lies primarily in the fact that it is a stepping stone to defining static se-congruence (and more se-congruences, see the section below on related and future work), rather than in direct applications in the field of programming.

Related and Future Work

As mentioned in the Introduction, this paper is a journal version of the conference paper [1] by Bergstra and Ponse entitled, *Evaluation trees for proposition algebra: The case for free and repetition-proof valuation congruence*. In ref. [1], "free valuation congruence" and "repetition-proof valuation congruence" are defined using evaluation trees. However, in the earlier paper [5], Bergstra and Ponse define both "free valuation congruence" and "repetition-proof valuation congruence" as the smallest congruence arising from the corresponding "valuation equivalence", which in turn is defined using a variety of so-called valuation algebras. In this article, we use "se-congruence" and "repetition-proof valuation congruence" for the congruences defined in ref. [1].

In the journal paper [14], we discuss several *three-valued* versions of proposition algebra and their evaluation trees, with help of a constant U that represents the truth value *undefined*. This paved the way for the recent paper [9], in which Bergstra and Ponse define "conditional short-circuit logic", based on the work of Guzmán and Squier in [15], in which three-valued "conditional logic" is studied. Conditional logic is based on the notion of a regular extension of classical logic, defined by Kleene in ref. [16], and on the observation that up to anti-isomorphism, there is a unique regular extension of the classical boolean algebra $B = \{\mathsf{T}, \mathsf{F}\}$ with ordinary negation ($'$), conjunction ($\wedge$), and disjunction ($\vee$) to a three-valued logic with non-commutative $\wedge$ and $\vee$ which satisfies deMorgan's laws $x'' = x$ and $(x \wedge y)' = x' \vee y'$; the name "conditional logic" stems from Gries [17] and this logic was first studied by McCarthy [18]. In ref. [9], two-valued and three-valued conditional short-circuit logic are distinguished, each of which gives rise to a corresponding "conditional valuation congruence", defined in terms of "CL-basic forms", although it is noted that "free valuation congruence" is defined in ref. [5]. In the two-valued case, this is a congruence that is in between memorising and static se-congruence. A typical identity in the underlying proposition algebra with $a, b \in A$ is

$$(P_1 \triangleleft b \triangleright P_2) \triangleleft a \triangleright (Q_1 \triangleleft b \triangleright Q_2) = (P_1 \triangleleft a \triangleright Q_1) \triangleleft b \triangleright (P_2 \triangleleft b \triangleright Q_2)$$

and an example of a non-valid equation is $\mathsf{F} \triangleleft a \triangleright \mathsf{F} = \mathsf{F}$. Like in the case for static se-congruence, CP_{mem} and its associated short-circuit logic are a technical basis for this work.

In ref. [6], Staudt defined "free fully evaluated logic", which characterises free valuation congruence for fully evaluated connectives (and negation), and came up with a completeness result. The binary connective "fulland", which we represent here by the symbol &, can be characterised by the equation

$$x \mathbin{\&} y = y \triangleleft x \triangleright (\mathsf{F} \triangleleft y \triangleright \mathsf{F}).$$

Hence, y is always evaluated, even when the evaluation result is already known to be *false*. In ref. [6], evaluation trees for expressions with & and its dual connective "fullor" are defined and their equality is axiomatised. This led to [19], in which Ponse and Staudt defined memorising, conditional, and static fully evaluated logic. In ref. [20], Cornets de Groot studied the "left-sequential exclusive OR" ℓXOR, defined by

$$x \,\ell\mathrm{XOR}\, y = (\mathsf{F} \triangleleft y \triangleright \mathsf{T}) \triangleleft x \triangleright y,$$

so that y is always evaluated. With regard to this binary (sequential) connective, there is therefore no difference between short-circuited and full evaluation (in [20], the "left-sequential NAND" ℓNAND, defined by $x \,\ell\mathrm{NAND}\, y = (\mathsf{F} \triangleleft y \triangleright \mathsf{T}) \triangleleft x \triangleright \mathsf{T}$ is also studied, and this work contains several completeness and expressiveness results with respect to free valuation congruence).

In the near future, we plan to investigate three-valued free, memorising, and conditional se-congruence, in particular their axiomatisations and normalisation functions.

Author Contributions: J.A.B., A.P. and D.J.C.S. contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding. The APC was funded by MDPI.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Acknowledgments: We would like to thank the anonymous reviewers for their helpful comments, which have certainly improved the article.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Independent Axiomatisations

All results mentioned below were found by or checked with the theorem prover Prover9 (Version 0.5B), and finite (counter) models were generated with the tool Mace4 (Version 0.5B). For both tools, including free downloads, see [21]. We used these tools on a Macbook Pro with a 2.4 GHz dual-core Intel Core i5 processor and 4 GB of RAM.

Firstly, the axioms (CP1)–(CP4) of CP (see Table 1, and also Table A1) are independent; this is easily verified with help of Mace4 and the same is true for the axioms of $\mathrm{CP}_{rp}(A)$ and $\mathrm{CP}_{cr}(A)$ (see Sections 3 and 4). Furthermore, as was noted in Section 4, $\mathrm{CP}_{cr}(A) \vdash \mathrm{CP}_{rp}(A)$.

Also CP_{mem} is independent; it is, for example, immediately clear that the memorising axiom (CPmem) (also included in Table A1) is independent from those of CP: the evaluation trees of a and $\mathsf{T} \triangleleft a \triangleright a$ are different. Also, all other axioms of CP_{mem} are independent and we briefly explain how the tool Mace4 can be used to prove this for axiom (CP1). Taking for example the CP-axioms (CP2)–(CP4) and (CPmem) as assumptions, a model of size two for these axioms that refutes the (CP1)-instance $\mathsf{F} \triangleleft \mathsf{T} \triangleright \mathsf{F} = \mathsf{F}$ is the following, which was generated by Mace4:

x	y	z	$x \triangleleft y \triangleright z$
T	T	T	T
T	T	F	T
T	F	T	T
T	F	F	F

x	y	z	$x \triangleleft y \triangleright z$
F	T	T	T
F	T	F	T
F	F	T	T
F	F	F	F

Table A1. The set CP_s of equational axioms for static valuation congruence, and three more axioms.

$x \triangleleft \mathsf{T} \triangleright y = x$	(CP1)
$x \triangleleft \mathsf{F} \triangleright y = y$	(CP2)
$\mathsf{T} \triangleleft x \triangleright \mathsf{F} = x$	(CP3)
$x \triangleleft (y \triangleleft z \triangleright u) \triangleright v = (x \triangleleft y \triangleright v) \triangleleft z \triangleright (x \triangleleft u \triangleright v)$	(CP4)
$x \triangleleft y \triangleright (z \triangleleft u \triangleright (v \triangleleft y \triangleright w)) = x \triangleleft y \triangleright (z \triangleleft u \triangleright w)$	(CPmem)
$\mathsf{F} \triangleleft x \triangleright \mathsf{F} = \mathsf{F}$	(CPs)
$x \triangleleft y \triangleright x = x$	(CP1,2)
$(x \triangleleft y \triangleright z) \triangleleft y \triangleright \mathsf{F} = y \triangleleft x \triangleright \mathsf{F}$	(CP3s)
$x \triangleleft y \triangleright (v \triangleleft y \triangleright w) = x \triangleleft y \triangleright w$	(Contr1)

The case for CP_s is more complex, and in Table A1 we give its six axioms and three other axioms that we will use to define alternative axiomatisations. With Prover9 it quickly follows that the axiom (CP1) is derivable from $\mathrm{CP}_s \setminus \{(\mathrm{CP1})\}$ and with Mace4 that the axioms in the latter set are independent. Furthermore, in ref. [14] (Table 5), an alternative set of independent axioms for CP_s is given that is concise and simple, and equally strong (this follows with help of Prover9), and independence follows with the help of Mace4:

$$\{(\mathrm{CP1}),\ (\mathrm{CP2}),\ (\mathrm{CP3s}),\ (\mathrm{CP4})\}. \tag{A1}$$

Note that the axiom (CP3s) with $y = \mathsf{T}$ implies (CP3), and with $y = \mathsf{F}$ the axiom (CPs); hence, its name.

Following Lemma 23, we finally note that there is another axiomatisation of static valuation congruence that consists of only four independent axioms and is at least as elegant:

$$\{(\mathrm{CP1,2}),\ (\mathrm{CP3}),\ (\mathrm{CP4}),\ (\mathrm{Contr1})\}, \tag{A2}$$

where (CP1,2) is the equation of Lemma 23, i.e., $x \triangleleft y \triangleright x = x$, and (Contr1) is the contraction law $x \triangleleft y \triangleright (v \triangleleft y \triangleright w) = x \triangleleft y \triangleright w$ (i.e, (9) in Section 5). With Prover9, it quickly follows that the axiomatisations (A1) and (A2) are equally strong, and the independence of (A2) also quickly follows with the help of Mace4.

Although we believe that CP_s is sufficiently simple and expresses the fundamental intuitions in an appropriate manner, our reasons for presenting these two alternative axiomatisations are not only their independence (in contrast to CP_s), but also their striking simplicity compared to Hoare's eleven axioms in ref. [2].

Ternary Boolean algebras are algebras of type $(3, 1)$ with one ternary operation f and one unary operation g defined by $f(x, y, z) = xy + yz + zx$ and $g(x) = x'$, where $+, \cdot$ and $'$ are the Boolean join, meet, and complementation operations. In ref. [22], Padmanabhan and McCune present several single axioms, each of which constitutes a base for ternary Boolean algebra, and thus axiomatises static valuation congruence. As might be expected, these equations are complex and not easy to read. We finally mention the work of Fatelo and Martins-Ferreira [23], which offers a general study of classical algebras via ternary operations. These algebras all satisfy $x \triangleleft y \triangleright x = x$ and are thus related to the domain of static valuation congruence.

References

1. Bergstra, J.A.; Ponse, A. Evaluation trees for proposition algebra: The case for free and repetition-proof valuation congruence. In *Correct System Design: Symposium in Honor of Ernst-Rüdiger Olderog on the Occasion of His 60th Birthday, Oldenburg, Germany, 8–9 September 2015*; Meyer, R., Platzer, A., Wehrheim, H., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2015; Volume 9360, pp. 44–61. [CrossRef]
2. Hoare, C.A.R. A couple of novelties in the propositional calculus. *Math. Log. Q.* **1985**, *31*, 173–178. [CrossRef]
3. Church, A. Conditioned disjunction as a primitive connective for the propositional calculus. *Port. Math.* **1948**, *7*, 87–90. Available online: https://purl.pt/2171/1/j-5293-b-vol7-fasc2-art3_PDF/j-5293-b-vol7-fasc2-art3_PDF_01-B-R0300/j-5293-b-vol7-fasc2-art3_0000_capa1-90_t01-B-R0300.pdf (accessed on 25 November 2025).
4. Church, A. *Introduction to Mathematical Logic*; Princeton University Press: Princeton, NJ, USA, 1956; Also Published as a Paperback Edition; De Gruyter Brill: 1991. [CrossRef]
5. Bergstra, J.A.; Ponse, A. Proposition algebra. *ACM Trans. Comput. Log.* **2011**, *12*, 36. [CrossRef]
6. Staudt, D.J.C. Completeness for Two Left-Sequential Logics. Master's Thesis, University of Amsterdam, Amsterdam, The Netherlands, 2012. *arXiv* **2012**, arXiv:1206.1936.
7. Vu, T.D. Denotational semantics for thread algebra. *J. Log. Algebr. Program.* **2008**, *74*, 94–111. [CrossRef]
8. Aceto, L.; Chen, T.; Fokkink, W.J.; Ingolfsdottir, A. On the axiomatizability of priority. *Math. Struct. Comput. Sci.* **2008**, 18, 5–28. [CrossRef]
9. Bergstra, J.A.; Ponse, A. Conditional logic as a short-circuit logic. *Sci. Ann. Comput. Sci.* **2025**, *35*, 161–196. [CrossRef]
10. Bergstra, J.A.; Ponse, A. Proposition algebra and short-circuit logic. In Proceedings of the 4th International Conference on Fundamentals of Software Engineering (FSEN 2011), Tehran, Iran, 20–22 April 2012; Arbab, F., Sirjani, M., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7141, pp. 15–31. [CrossRef]
11. Bergstra, J.A.; Ponse, A.; Staudt, D.J.C. Short-circuit logic. *arXiv* **2013**. [CrossRef]
12. Parnas, D.L. Really rethinking 'Formal Methods'. *Computer* **2010**, *43*, 28–34. [CrossRef]
13. Black, P.E.; Windley, P.J. Inference rules for programming Languages with side effects in expressions. In *Theorem Proving in Higher Order Logics*; Goos, G., Hartmanis, J., van Leeuwen, J., von Wright, J., Grundy, J., Harrison, J., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1996; Volume 1125, pp. 51–60. [CrossRef]
14. Bergstra, J.A.; Ponse, A.; Staudt, D.J.C. Non-commutative propositional logic with short-circuit evaluation. *J. Appl.-Non-Class. Logics* **2021**, *31*, 234–278. [CrossRef]
15. Guzmán, F.; Squier, C.C. The algebra of conditional logic. *Algebra Universalis* **1990**, *27*, 88–110. [CrossRef]
16. Kleene, S.C. *Introduction to Metamathematics*; Van Nostrand: New York, NY, USA, 1952; ISBN 9780720421033. Also Published as a Hardback Edition: North Holland: 1980
17. Gries, D. *The Science of Programming*; Springer: New York, NY, USA, 1981. [CrossRef]
18. McCarthy, J. A basis for a mathematical theory of computation. In *Computer Programming and Formal Systems*; Braffort, P., Hirschberg, D., Eds.; Studies in Logic and the Foundations of Mathematics; Elsevier: Amsterdam, The Netherlands, 1963; Volume 35, pp. 33–70. [CrossRef]
19. Ponse, A.; Staudt, D.J.C. Fully evaluated left-sequential logics. *J. Appl.-Non-Class. Logics* **2025**, 1–53. [CrossRef]

20. Cornets de Groot, S.H. Logical Systems with Left-Sequential Versions of NAND and XOR. Master's Thesis, University of Amsterdam, Amsterdam, The Netherlands, 2020. Available online: https://eprints.illc.uva.nl/id/eprint/1743/1/MoL-2020-02.text.pdf (accessed on 2 January 2026).
21. McCune, W. *The GUI: Prover9 and Mace4 with a Graphical User Interface: Prover9-Mace4*, Version 0.5B; The University of New Mexico: Albuquerque, NM, USA, 2008. Available online: https://www.cs.unm.edu/~mccune/prover9/gui/v05.html (accessed on 17 November 2025).
22. Padmanabhan, R.; McCune, W. Single Identities for Ternary Boolean Algebras. *Comput. Math. Appl.* **1995**, *29*, 13–16. [CrossRef]
23. Fatelo, J.P.; Martins-Ferreira, N. Reconstructing classical algebras via ternary operations. *Mathematics* **2025**, *13*, 1407. [CrossRef]