

## ENCODING MANY-VALUED LOGIC IN $\lambda$ -CALCULUS

FER-JAN DE VRIES

Informatics, University of Leicester, UK  
*e-mail address:* fer.jan.de.vries@gmail.com

**ABSTRACT.** We will extend the well-known Church encoding of Boolean logic into  $\lambda$ -calculus to an encoding of McCarthy’s 3-valued logic into a suitable infinitary extension of  $\lambda$ -calculus that identifies all unsolvables by  $\perp$ , where  $\perp$  is a fresh constant. This encoding refines to  $n$ -valued logic, for  $n \in \{4, 5\}$ . Such encodings also exist for Church’s original  $\lambda\mathbf{I}$ -calculus.

By way of motivation we consider Russell’s paradox, exploiting the fact that the same encoding allows us also to calculate truth values of infinite closed propositions in this infinitary setting.

### IN MEMORY OF CORRADO BÖHM

Böhm’s theorem [Böhm68] was instrumental in proving the equivalence between an operational semantics and a denotational semantics of the  $\lambda$ -calculus and inspired Barendregt [Bar77a, Bar84] to define the concept of Böhm tree, a first version of which had been introduced by Böhm and Dezani [BDC74]. Böhm trees have later been redefined as the normal forms in a suitable infinitary extension of  $\lambda$ -calculus by Kennaway et al. [KKSdV97]. Böhm trees and their generalisations are now another established way to capture the semantic content of a  $\lambda$ -term [KdV03, SdV11a]. In this paper Böhm trees play a crucial role: we use Böhm trees to encode (even infinite) propositions in  $\lambda$ -calculus and to calculate their values.

### 1. MOTIVATION AND OVERVIEW

In this paper we will extend the well-known Church encoding of Boolean logic into  $\lambda$ -calculus to an encoding of  $n$ -valued logic (for  $3 \leq n \leq 5$ ) into an appropriate infinitary extension of  $\lambda$ -calculus. The extension we use in case of  $n = 3$  is the extension that identifies all unsolvables by  $\perp$  such that the normal forms of the lambda terms are their Böhm trees. By way of motivation we will now consider Russell’s paradox. Any notation that is used in this section will be explained in Section 2 and 3.

---

*Key words and phrases:* lambda calculus, many-valued logic, meaningless term, generalised Böhm tree, Russell’s paradox.

**1.1. Russell's Paradox.** This paradox arises if we, somewhat naïvely, consider the set  $R$  of all sets that are not a member of themselves and then wonder whether  $R \in R$ . We get as paradoxical consequence that  $R \in R$  if and only if  $R \notin R$ . As noted by Church [Chu41] at the heart of this paradox lies the  $\lambda$ -term (the application  $pp$  is interpreted as  $p \ni p$ )

$$P \equiv (\lambda p. \neg(pp))(\lambda p. \neg(pp))$$

which has no finite normal form and therefore is neither true or false.<sup>1</sup> Like the well-known  $\lambda$ -term  $\Omega \equiv (\lambda x.xx)(\lambda x.xx)$ , the term  $P$  has the property that any of its reducts can be further reduced to a redex. Therefore it has no head normal form. Hence  $P$  is an unsolvable term. It allows the following infinite reduction

$$P \rightarrow \neg P \rightarrow \neg(\neg P) \rightarrow \neg(\neg(\neg P)) \rightarrow \dots$$

The limit  $\neg(\neg(\neg(\neg(\dots))))$  of this reduction is an infinite proposition. Unexpected, perhaps, but not necessarily paradoxical.

**1.2. Infinite  $\lambda$ -calculus and Böhm trees.** In the past [KdV03, SdV11a] we have developed a family of infinitary  $\lambda$ -calculi, each depending on a set of meaningless terms  $\mathcal{U}$ . The set of terms underlying these extensions is the set  $\Lambda_{\perp}^{\infty}$  of lambda terms obtained by interpreting the usual  $\lambda$ -calculus syntax extended with one fresh symbol  $\perp$  coinductively. We use this set  $\mathcal{U} \subseteq \Lambda_{\perp}^{\infty}$  of meaningless terms to add a new rewrite rule to  $\lambda_{\beta}$  that allows us to rewrite meaningless terms to  $\perp$ . Cf. Section 2 for a precise description of this rule.

The set  $\mathcal{US}$  of unsolvable  $\lambda$ -terms is the best known example of such a set of meaningless terms. The corresponding infinitary extension  $\lambda_{\beta \perp \mathcal{US}}^{\infty}$  of the finite  $\lambda$ -calculus  $\lambda_{\beta}$  is confluent and normalising for a suitable notion of possibly infinite reduction. The Böhm tree of a finite  $\lambda$ -term is precisely its normal form in  $\lambda_{\beta \perp \mathcal{US}}^{\infty}$ . In particular the Böhm tree of an unsolvable is  $\perp$ .

So with this encoding in the  $\lambda$ -calculus in mind we no longer need be afraid of infinite propositions. By inspecting the Böhm trees of the encoding of infinite closed propositions we will find that they are either lambda terms representing a Boolean or they are unsolvable.

**1.3. Encoding three-valued logic in infinitary  $\lambda$ -calculus.** Thus we are led to extend the Church encoding to an encoding of three-valued logic in infinitary  $\lambda$ -calculus  $\lambda_{\beta \perp \mathcal{US}}^{\infty}$ , by mapping the third value to  $\perp$ . Inspection of the truth tables then reveals that the Church encoding of Boolean logic now has naturally been extended to a Church encoding of what is called McCarthy's three-valued logic [McC63]. In particular we find that the infinite term  $\neg(\neg(\neg(\dots)))$  that we encountered in our analysis of Russell's paradox is neither true nor false but  $\perp$ .

**1.4. Encoding four- and five-valued logic.** We will further note that the set of unsolvable  $\lambda$ -terms that get identified by  $\perp$  can be split into three subsets closed under infinite reduction and substitution. Repeating the above construction now with three new truth values instead of  $\perp$  we find that the Church encoding also encodes a five-valued McCarthyan logic. That five-valued logic and its four-valued sub-logic have been studied earlier by Bergstra and Van de Pol [BvdP96, BvdP11].

<sup>1</sup>The notation  $\neg$  for the  $\lambda$ -term  $\lambda b.b\mathbf{TF}$  is recalled at page 6.

**1.5. Church's  $\lambda\mathbf{I}$ -calculus.** When Church started his work on  $\lambda$ -calculus around or before 1928, his motivation was to use the  $\lambda$ -calculus as the basis for a symbolic logic that could serve as the foundation of mathematics [Chu32]. Church's hope was that by using non-classical logic (in which he had shown an early interest [Chu28]) he could side step the Paradoxes without having to introduce Zermelo's set axioms or Russell's type theory, that he both judged as somewhat artificial.

This is not what happened. He discovered with his students Kleene and Rosser that the lambda definable functions corresponded exactly to the recursive functions [Kle36b, Kle36a, Chu36b]. In the build-up to that result Kleene and Rosser managed to prove the inconsistency of his logical system [KR35] while Church himself was still publicly hopeful that not only his system could be paradox-free but also escape Gödel's incompleteness theorem [Chu34]. A disaster. Fortunately, the  $\lambda$ -calculus itself was consistent by the Church-Rosser theorem [CR36]. Various papers under preparation had to be rewritten. Church rebounded almost immediately with his formulation of the Church-Turing thesis [Chu36b] and his negative solution of Hilbert's *Entscheidungsproblem* [Chu36a] (there is no algorithm that can decide whether a given formula of the first order arithmetic is provable or not).

Church's goal, a paradox free system of symbolic logic, led him to the choice of the  $\lambda\mathbf{I}$ -calculus in which an abstraction  $\lambda x.M$  is only accepted as well-formed term if it contains  $x$  as a free variable. For him only terms with a finite normal form were significant and for this he rejected, what we now call, the classical lambda calculus which has terms that have a normal form although they also have subterms which do not [Chu41].

It is a natural question to ask whether an encoding of 3-valued logic is possible in the  $\lambda\mathbf{I}$ -calculus. We recall that there is a Church encoding for Boolean logic in the  $\lambda\mathbf{I}$ -calculus. Barendregt has shown that the unsolvable terms in  $\lambda\mathbf{I}$ -calculus are exactly the  $\lambda\mathbf{I}$ -terms without a finite normal form. This means that the Böhm tree of a  $\lambda\mathbf{I}$ -term is either its finite normal form or  $\perp$ . No infinite terms or reductions are needed in case of  $\lambda\mathbf{I}$ -calculus to define Böhm trees. Thus the above encoding of McCarthy's three-valued logic can be quite simply repeated in Church's  $\lambda\mathbf{I}$ -calculus. For the details see Section 4.2.

Yet, while this encoding is undoubtedly well within Church's technical means, the Böhm tree concept seems in conflict with his intuition of meaning. The Böhm tree construction gives meaning to any term: the terms without a finite normal form which Church considers meaningless/insignificant are given the "meaning"  $\perp$  in this extension of  $\lambda\mathbf{I}$ -calculus with the  $\perp$ -rule.as

**1.6. Overview of this paper.** In Section 2 we assume familiarity with the finite  $\lambda$ -calculus and briefly introduce relevant notation and facts from the infinitary  $\lambda$ -calculus. In Section 3 we recall the encoding of Boolean valued logic and explain how to extend this to an encoding of three-valued logic. Then we show how this encoding can be refined to four- and five valued logic. In Section 4 we discuss Böhm trees for Church's  $\lambda\mathbf{I}$ -calculus and show that three valued logic can also be encoded in  $\lambda\mathbf{I}$ -calculus. Finally Section 5 is a brief conclusion.

## 2. INFINITE $\lambda$ -CALCULUS

We will recall notation, concepts and facts from infinitary  $\lambda$ -calculus, while assuming familiarity with  $\lambda_\beta$ , by which we denote the finite  $\lambda$ -calculus with  $\beta$ -reduction and no  $\eta$ -reduction [Chu41, Bar84]. We will use  $\rightarrow$  and  $\rightarrow^*$  for respectively one step  $\beta$ -reduction

and finite  $\beta$ -reduction. We will use  $\equiv$  to indicate syntactical identity modulo  $\alpha$ . We will use the following special terms.

$$\begin{array}{ll} \mathbf{K} \equiv \lambda xy.y & \mathbf{\Omega} \equiv (\lambda x.xx)\lambda x.xx \\ \mathbf{I} \equiv \lambda x.x & \mathbf{\Theta} \equiv (\lambda xy.y(xxy))\lambda xy.y(xxy) \end{array}$$

We will now explain how to construct infinite extensions of the finite  $\lambda$ -calculus that are confluent and normalising. We begin with the observation that finite reduction is not finitely normalising: for instance, the finite term  $\mathbf{\Theta}x$  has an infinite reduction

$$\mathbf{\Theta}x \rightarrow x(\mathbf{\Theta}x) \rightarrow x(x(\mathbf{\Theta}x)) \rightarrow \dots$$

This is a converging reduction (think of terms as trees and take the standard metric on trees) with an infinite term as limit:

$$x(x(x(\dots)))$$

We can add infinite  $\lambda$ -terms to the finite  $\lambda$ -terms by reading the usual syntax definition (where  $x$  ranges over some countable set of variables) of finite  $\lambda$ -terms coinductively:

$$M ::= x \mid \lambda x.M \mid (MM)$$

We will write  $\Lambda^\infty$  for this set of finite and infinite  $\lambda$ -terms. Using  $\dashrightarrow$  for a possibly infinite converging reduction, we can now write

$$\mathbf{\Theta}x \dashrightarrow x(x(x(\dots)))$$

Later in the paper we will encounter the infinite term  $\lambda y\lambda y\lambda y\dots$  as the limit of the converging reduction

$$\mathbf{\Theta}\mathbf{K} \rightarrow \mathbf{K}(\mathbf{\Theta}\mathbf{K}) \rightarrow \lambda y.\mathbf{\Theta}\mathbf{K} \dashrightarrow \lambda y\lambda y.\mathbf{\Theta}\mathbf{K} \dashrightarrow \lambda y\lambda y\lambda y.\mathbf{\Theta}\mathbf{K} \dashrightarrow \lambda y\lambda y\lambda y\dots$$

These two examples show that by adding infinite terms and infinite reductions to the finite lambda calculus, we obtain that some finite terms without a finite normal form now have converging reductions to an infinite normal form. But we have lost confluence of the finite  $\lambda$ -calculus. E.g. the finite term  $(\lambda x.\mathbf{I}(xx))(\lambda x.\mathbf{I}(xx))$  has a finite reduction to  $\mathbf{\Omega}$  and an infinite converging reduction to  $\mathbf{I}(\mathbf{I}(\mathbf{I}(\dots)))$ . Both reducts have the property that they can only reduce to themselves. Hence they cannot be joined by either finite or converging reductions. This example also shows that this extension of the finite lambda calculus is not normalising.

Yet, it is possible to build (in fact many different) infinitary extensions of  $\lambda_\beta$  which are confluent and normalising for finite and convergent reductions, and finite and infinite terms [KKSdV97, KvOdV99, KdV03, SdV11a]. We need to do three things. First, we add a new symbol  $\perp$  to the syntax of  $\lambda$ -terms and consider the set  $\Lambda_\perp^\infty$  of finite and infinite terms over the extended coinductive syntax. Second, we choose a set  $\mathcal{U}$  of  $\lambda$ -terms in  $\Lambda^\infty$ . Third, we add a new reduction  $\perp_{\mathcal{U}}$ -rule on  $\Lambda_\perp^\infty$  that will allow us to identify the terms of  $\mathcal{U}$  by the new symbol  $\perp$ :

$$\frac{M[\perp := \mathbf{\Omega}] \in \mathcal{U} \quad M \neq \perp}{M \rightarrow \perp} (\perp_{\mathcal{U}})$$

For a given  $\mathcal{U}$  we denote this infinite extension by  $\lambda_{\beta\perp_{\mathcal{U}}}^\infty$ . In a series of papers [KKSdV97, KvOdV99, KdV03, SdV11a] we have determined a collection of necessary and sufficient axioms that the set  $\mathcal{U}$  must satisfy in order for  $\lambda_{\beta\perp_{\mathcal{U}}}^\infty$  to be a converging and normalising infinite  $\lambda$ -calculus.

We call such sets *sets of meaningless terms*. The choice of a set  $\mathcal{U}$  of meaningless terms is akin to the choice of a semantics for lambda calculus: together the normal forms in  $\lambda_{\beta \perp \mathcal{U}}^\infty$  form a model of the  $\lambda$ -calculus. The intuition is that the elements of a meaningless set are undefined, that is, have no meaning or are insignificant. In order for such a model to be consistent the set  $\mathcal{U}$  has to be a proper subset of  $\Lambda^\infty$ .

**Definition 2.1** ([SdV11a]).  $\mathcal{U} \subseteq \Lambda^\infty$  is called a *set of (finite or infinite) meaningless terms*, if it satisfies the axioms of meaninglessness:

- (1) Axiom of Root-activeness:  $\mathcal{R} \subseteq \mathcal{U}$ . ( $\mathcal{R}$  defined below)
- (2) Axiom of Closure under  $\beta$ -reduction: If  $M \twoheadrightarrow_\beta N$  then  $N \in \mathcal{U}$  for all  $M \in \mathcal{U}$ .
- (3) Axiom of Closure under Substitution: If  $M \in \mathcal{U}$  then any substitution instance of  $M$  is an element of  $\mathcal{U}$ .
- (4) Axiom of (Weak) Overlap: Either for each  $\lambda x.P \in \mathcal{U}$ , there is some  $W \in \mathcal{U}$  such that  $P \twoheadrightarrow_\beta Wx$ , or alternatively  $(\lambda x.P)Q \in \mathcal{U}$ , for any  $Q \in \Lambda_\perp^\infty$ .
- (5) Axiom of Indiscernibility: Define  $M \overset{\mathcal{U}}{\rightsquigarrow} N$  if  $M$  can be transformed into  $N$  by replacing pairwise disjoint subterms of  $M$  in  $\mathcal{U}$  by terms in  $\mathcal{U}$ . If  $M \overset{\mathcal{U}}{\rightsquigarrow} N$  then  $M \in \mathcal{U} \Leftrightarrow N \in \mathcal{U}$ .
- (6) Axiom of Consistency:  $\mathcal{U} \neq \Lambda$ .

This construction is inspired by the definition of Böhm tree [Bar84]. If one takes for  $\mathcal{U}$  the set  $\mathcal{US}$  of unsolvables [Bar77b], then the resulting infinite  $\lambda$ -calculus  $\lambda_{\beta \perp \mathcal{US}}^\infty$  is confluent and normalising for  $\beta \perp_{\mathcal{US}}$  reduction. The Böhm tree of a finite  $\lambda$ -term  $M$  can equivalently be described as its unique normal form in  $\lambda_{\beta \perp \mathcal{US}}^\infty$  [KKSdV97]. Here a (possibly infinite) closed term  $M$  in  $\Lambda^\infty$  is called *solvable* if  $MN_1 \dots N_k \twoheadrightarrow_\beta \mathbf{I}$  for some sequence  $N_1, \dots, N_k$  with  $k \geq 0$ . An open lambda term is called *solvable* if its closure is solvable. A  $\lambda$ -term is called *unsolvable* if it is not solvable. The set of unsolvables is the largest set for which this construction works. A  $\lambda$ -term is unsolvable if and only if it has no finite  $\beta$ -reduction to a head normal form [Bar84].

The smallest set of meaningless terms [KKSdV97, Ber96] is the set  $\mathcal{R}$  of terms that are root-active (or mute). A  $\lambda$ -term  $M$  is root-active if any reduct of  $M$  can further reduce to a redex. The classical root-active term is  $\mathbf{\Omega}$ . The unsolvable  $\mathbf{\Omega I}$  is not root-active. Note that the definition of a root-active term allows for free variables. The normal forms in  $\lambda_{\beta \perp \mathcal{R}}^\infty$  are exactly the Berarducci trees.

The Lévy-Longo trees can be obtained if one performs this construction over the set of terms without a weak head normal form. In general there are uncountably many sets of meaningless terms [SdV11a]. The collection of normal forms of each such  $\lambda_{\beta \perp \mathcal{U}}^\infty$  is a model of the  $\lambda$ -calculus  $\lambda_\beta$ . The axioms are chosen such that different sets of meaningless terms give rise to different consistent models.

Church considered the terms without finite normal form as insignificant [Chu41, Bar84]. We recognise that the set of terms in  $\Lambda^\infty$  without a finite normal form is not a set of meaningless terms [KvOdV99, KdV03] in the sense of Definition 2.1, because it is not closed under reduction. The term  $\mathbf{KI\Omega}$  has an infinite reduction, because its subterm  $\mathbf{\Omega}$  has. Yet  $\mathbf{KI\Omega}$  reduces to the finite normal form  $\mathbf{I}$ . We will come back to this in Section 4.2.

### 3. ENCODING MANY-VALUED LOGIC IN $\lambda$ -CALCULUS

In this section we will extend the familiar Church encoding of Boolean logic to many-valued logic using ideas from Böhm trees and infinitary  $\lambda$ -calculus. We don't know precise reference to the original Church encoding. As Landin remarks in [Lan64]:

In particular Church and Curry, and McCarthy and the ALGOL 60 authors, are so large a part of the history of their respective disciplines as to make detailed attributions inevitably incomplete and probably impertinent.

Berarducci and Böhm have vastly generalised the Church encoding [BB85].

**3.1. Encoding Boolean logic in  $\lambda$ -calculus.** In “the History of Lisp” [McC78] John McCarthy mentions his “invention of the true conditional expression

$$\mathbf{if } M \mathbf{ then } N_1 \mathbf{ else } N_2$$

which evaluates only one of  $N_1$  and  $N_2$  according to whether  $M$  is true or false” and also his “desire for a programming language that would allow its use” in the period 1957-8. He also recalls “the conditional expression interpretation of Boolean connectives” as one of the characterising ideas of LISP. By this he means concretely the if-then-else construct (when applied to Boolean expressions only) which in combination with the truth values **T** and **F** can be used as a basis for propositional logic [McC60] with the following natural definitions:

$$\begin{aligned} \neg &\equiv \lambda m.\mathbf{if } m \mathbf{ then } \mathbf{F} \mathbf{ else } \mathbf{T} \\ \wedge &\equiv \lambda mn.\mathbf{if } m \mathbf{ then } n \mathbf{ else } \mathbf{F} \\ \vee &\equiv \lambda mn.\mathbf{if } m \mathbf{ then } \mathbf{T} \mathbf{ else } n \\ \rightarrow &\equiv \lambda mn.\mathbf{if } m \mathbf{ then } n \mathbf{ else } \mathbf{T} \end{aligned} \tag{3.1}$$

Barendregt's book [Bar84] records two elegant encodings of the Booleans and the if-then-else construct. One encodes into the classical  $\lambda$ -calculus and the other into the more restricted  $\lambda\mathbf{I}$ -calculus preferred by Church [Chu32, Chu41]. The latter we will discuss in Section 4.2. The former is the simplest:

$$\begin{aligned} \mathbf{T} &\equiv \lambda xy.x \\ \mathbf{F} &\equiv \lambda xy.y \\ \mathbf{if } B \mathbf{ then } M \mathbf{ else } N &\equiv BMN \end{aligned}$$

It is easy to see that if-then-else behaves as intended in this encoding. When  $B$  reduces to **T** and **F**, we have respectively:

$$\begin{aligned} \mathbf{if } \mathbf{T} \mathbf{ then } M \mathbf{ else } N &\rightarrow M \\ \mathbf{if } \mathbf{F} \mathbf{ then } M \mathbf{ else } N &\rightarrow N \end{aligned} \tag{3.2}$$

With help of (3.2) it is straightforward to verify that the standard truth tables of Figure 1 for Boolean valued propositional logic hold in  $\lambda$ -calculus. Boolean logic commonly deals with

$\neg$	$\mathbf{T}$	$\mathbf{F}$	$\wedge$	$\mathbf{T}$	$\mathbf{F}$	$\vee$	$\mathbf{T}$	$\mathbf{F}$	$\rightarrow$	$\mathbf{T}$	$\mathbf{F}$
$\mathbf{T}$	$\mathbf{F}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{F}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{F}$
$\mathbf{F}$	$\mathbf{T}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{T}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{T}$	$\mathbf{T}$

Figure 1: Boolean-valued propositional logic

finite propositions. The set of finite propositions can be defined formally with an inductive syntax, where  $p$  ranges over some possibly infinite set of propositional variables:

$$\phi ::= p \mid \mathbf{T} \mid \mathbf{F} \mid (\phi \wedge \phi) \mid (\phi \vee \phi) \mid (\phi \rightarrow \phi) \mid \neg\phi \quad (3.3)$$

It is not hard to prove by induction that all closed finite propositions have a unique finite normal form:

**Lemma 3.1.** *Let  $\phi$  be a finite closed proposition. Then  $\phi$  has a unique finite normal form, which is either  $\mathbf{T}$  or  $\mathbf{F}$ .*

**3.2. Encoding infinitary propositions in infinitary  $\lambda$ -Calculus.** Infinite propositions can be used to model certain while statements. For instance Bergstra and Ponse [BP11] model *while*  $\neg a$  *test*  $b$  as the potentially infinite solution of the recursive equation

$$W = \mathbf{if} \ a \ \mathbf{then} \ \mathbf{T} \ \mathbf{else} \ (\mathbf{if} \ b \ \mathbf{then} \ W \ \mathbf{else} \ W)$$

By reading the syntax definition (3.3) coinductively we obtain the set of finite and infinite propositions.

In the introduction we showed how Russell's paradox leads to the infinite proposition  $\neg(\neg(\neg(\dots)))$ . The encoding of this infinite proposition in  $\lambda$ -calculus is the infinite term

$$(((\dots)\mathbf{FT})\mathbf{FT})\mathbf{FT}$$

which is an infinite normal form with an infinite left spine and no head normal form. Hence this encoding of the Russell's paradox is unsolvable.

Not all infinite propositions reduce to infinite left spines: for instance, the infinite proposition

$$P_1 \equiv \mathbf{T} \wedge (\mathbf{T} \wedge \dots) \equiv \mathbf{T} \wedge P_1 \equiv \mathbf{TP}_1\mathbf{T} \equiv (\lambda xy.x)P_1\mathbf{T} \rightarrow P_1$$

is root-active. Also, some infinite propositions reduce just to  $\mathbf{T}$  or  $\mathbf{F}$ : for instance, the term

$$P_2 \equiv \mathbf{T} \vee (\mathbf{T} \vee (\mathbf{T} \vee (\dots))) \equiv \mathbf{T} \vee P_2 \equiv \mathbf{TP}_2\mathbf{T} \equiv (\lambda xy.x)\mathbf{TP}_2 \rightarrow \mathbf{T}$$

These examples show that some infinite propositions reduce to a Boolean, but not all. The latter have in common that their Böhm tree is  $\perp$ .

**Theorem 3.2.** *Let  $\phi$  be a finite or infinite closed proposition. Then the Böhm tree of  $\phi$  is either  $\mathbf{T}$ ,  $\mathbf{F}$  or  $\perp$ .*

*Proof.* By coinduction! □

The missing detail in the above “proof” follows from the corollary of the next lemma:

**Lemma 3.3.** *Let  $U$  be an unsolvable  $\lambda$ -term in  $\Lambda$ . Then  $\neg U$ ,  $U \wedge N$ ,  $U \vee N$  and  $U \rightarrow N$  are all unsolvable terms.*

*Proof.* Immediate from the definitions. For instance, suppose  $\neg U$  is solvable; then

$$(\neg U)N_1 \dots N_n \twoheadrightarrow \mathbf{I}$$

for some  $N_1, \dots, N_n$ . But  $\neg U \equiv U\mathbf{FT}$ . Hence  $U$  is solvable. Therefore, unsolvability of  $U$  implies the unsolvability of  $\neg U$ . □

**Corollary 3.4.** *The Böhm trees of  $\neg\perp$ ,  $\perp \wedge N$ ,  $\perp \vee N$  and  $\perp \rightarrow N$  are all equal to  $\perp$ .*

*Proof.* Eg.,  $B(\neg\perp) = B(\neg U) = B(U\mathbf{FT}) = \perp$  □

$\neg$	$\mathbf{T}$	$\mathbf{F}$	$\overset{\circ}{\wedge}$	$\mathbf{T}$	$\mathbf{F}$	$\perp$	$\overset{\circ}{\vee}$	$\mathbf{T}$	$\mathbf{F}$	$\perp$	$\rightarrow$	$\mathbf{T}$	$\mathbf{F}$	$\perp$
$\mathbf{T}$	$\mathbf{F}$	$\perp$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{F}$	$\perp$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{F}$	$\perp$
$\mathbf{F}$	$\mathbf{T}$	$\perp$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{T}$	$\mathbf{F}$	$\perp$	$\mathbf{F}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$
$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$

Figure 2: McCarthy's left-sequential three-valued propositional logic

**3.3. Encoding three-valued McCarthy logic with help of Böhm trees.** Theorem 3.2 suggests an experiment: what logic do we obtain if we repeat the encoding of Section 3.1 with three truth values  $\{\mathbf{T}, \mathbf{F}, \perp\}$  instead of two? Using Lemma 3.4 we can extend the truth tables of Boolean-valued logic to the truth tables of Figure 2. These are exactly the truth tables of McCarthy's left-sequential three-valued propositional logic. We will use the notation of [BBR95] and write  $\overset{\circ}{\wedge}$  and  $\overset{\circ}{\vee}$  for the conjunction and disjunction in left-sequential logic.

McCarthy discovered left-sequential three-valued propositional logic in his search for a suitable formalism for a mathematical theory of computation [McC63]. In the context of a language for computational (partial) functions he introduced conditional expressions of the form

$$(p_1 \rightarrow e_1, \dots, p_n \rightarrow e_n)$$

where the  $p_i$  are propositional expressions that evaluate to true or false. The idea is that the value of the whole conditional expression is the value of the expression  $e_i$  for the first  $p_i$  with value true. If all  $p_i$  have value false then the conditional expression is undefined. To allow that the evaluation of an expression can be inconclusive, McCarthy stated the rule to evaluate conditional expressions more precisely:

If an undefined  $p$  occurs before a true  $p$  or if all  $p$ 's are false or if the  $e$  corresponding to the first true  $p$  is undefined, then the form is undefined. Otherwise, the value of the form is the value of the  $e$  corresponding to the first true  $p$ .

Now the propositional connectives can be defined with help of conditional expressions.

$$\begin{aligned}
 \neg p &\equiv (p \rightarrow \mathbf{F}, \mathbf{T} \rightarrow \mathbf{T}) \\
 p \overset{\circ}{\wedge} q &\equiv (p \rightarrow q, \mathbf{T} \rightarrow \mathbf{F}) \\
 p \overset{\circ}{\vee} q &\equiv (p \rightarrow \mathbf{T}, \mathbf{T} \rightarrow q) \\
 p \rightarrow q &\equiv (p \rightarrow q, \mathbf{T} \rightarrow \mathbf{T})
 \end{aligned} \tag{3.4}$$

for which McCarthy then derives the very same truth tables of Figure 2. In the presence of the third truth value, undefined, the left sequential conjunction and disjunction are no longer commutative.

Guzman and Squier [GS90] gave a complete axiomatisation of McCarthy's logic, cf. Figure 3. They also gave the following definition of conditional using the left-sequential connectives.

**Lemma 3.5.** *if  $B_0$  then  $B_1$  else  $B_2 = (B_0 \overset{\circ}{\wedge} B_1) \overset{\circ}{\vee} (\neg B_0 \overset{\circ}{\wedge} B_2)$  for all  $B_0, B_1, B_2 \in \{\mathbf{T}, \perp, \mathbf{F}\}$ .*

*Proof.* After applying the definitions of the logical operators it remains to show that

$$B_0 B_1 B_2 = (B_0 B_1 \mathbf{F}) \mathbf{T} ((B_0 \mathbf{F} \mathbf{T}) B_2 \mathbf{F}) \tag{3.5}$$

The argument now is by inspection.



(1)	$\neg \mathbf{T} = \mathbf{F}$
(2)	$\neg \perp = \perp$
(3)	$\neg \neg x = x$
(4)	$\neg(x \wp y) = \neg x \wp \neg y$
(5)	$x \rightarrow y = \neg x \wp y$
(6)	$x \wp (y \wp z) = (x \wp y) \wp z$
(7)	$\mathbf{T} \wp x = x$
(8)	$x \wp (x \wp y) = x$
(9)	$x \wp (y \wp z) = (x \wp y) \wp (x \wp z)$
(10)	$(x \wp y) \wp z = (x \wp z) \wp (\neg x \wp y \wp z)$
(11)	$(x \wp y) \wp (y \wp x) = (y \wp x) \wp (x \wp y)$

Figure 3: Complete axiomatisation of McCarthy’s left-sequential three-valued propositional logic by Guzman and Squire

- $B_0 = \mathbf{T}$ . Since  $\mathbf{T}xy = x$ , it is sufficient to show that  $B_1 = B_1\mathbf{T}\mathbf{F}$ . This follows by inspection of the three options for  $B_1 \in \{\mathbf{T}, \perp, \mathbf{F}\}$ .
- $B_0 = \perp$ . Then  $\perp B_1 \perp = \perp = (\perp B_1 \mathbf{F})\mathbf{T}((\perp \mathbf{F}\mathbf{T})B_2\mathbf{F})$ .
- $B_0 = \mathbf{F}$ . Since  $\mathbf{F}xy = y$ , it is enough to show that  $B_2 = \mathbf{T}B_2\mathbf{F}$ , which follows by  $\mathbf{T}xy = x$ . □

In the remainder of the paper we will ignore implication as it can be defined from  $\neg$  and  $\wp$ .

**3.4. Refining the encoding from three-valued to four- and five-valued logic.** In the previous section we identified unsolvable  $\lambda$ -terms with  $\perp$ , their (possibly infinite) normal form in the infinitary  $\lambda$ -calculus  $\lambda_{\beta\perp_{US}}^\infty$ . We used  $\perp$  as third truth value besides  $\mathbf{T}$  and  $\mathbf{F}$ . We can refine this idea using the observation of [SdV11b] that the set of unsolvables is the union of three pairwise disjoint sets, each closed under substitution and infinite reduction.

At the basis of this observation lies the simple and well known fact that any *finite*  $\lambda$ -term has one of two forms, where  $m, n$  range over natural numbers:

$$\begin{aligned} &\lambda x_1 \dots x_n. x M_m \dots M_1 \\ &\lambda x_1 \dots x_n. (\lambda x P) Q M_m \dots M_1 \end{aligned}$$

The former expression is called a *head normal form* and the redex  $(\lambda x P)Q$  in the latter is called the head redex. Wadsworth has shown that repeated head reduction of a term  $M$  terminates in a head normal form if and only if  $M$  has one, and also that having a head normal form is equivalent to being solvable. A reduction in which each step reduces a head redex is called a head reduction [Bar84, see Section 8.3]

Hence any unsolvable term  $M$  has an infinite head reduction. One of the following scenarios must hold for  $M$ .

- $M$  has an infinite head reduction to a term of the form  $\lambda x \lambda x \lambda x \dots$  (modulo renaming).  
Example  $\Theta \mathbf{K} \twoheadrightarrow \lambda x \lambda x \lambda x \dots$
- $M$  has an infinite head reduction to a term of the form  $\lambda x_1 \dots \lambda x_n. (((\dots) M_3) M_2) M_1$ .  
An example  $\Theta \lambda x. xy \twoheadrightarrow (((\dots y) y) y)$ .

- $M$  has a finite head reduction to a term of the form  $\lambda x_1 \dots x_n. (\lambda x P) Q M_m \dots M_1$  in which the head redex  $(\lambda x. P) Q$  is root active. The term  $\Omega$  is an example.

These three mutually exclusive fates of unsolvable terms lead to the following definition.

- Definition 3.6** ([SdV11b]). (1)  $\mathcal{HA} = \{M \in \Lambda^\infty \mid M \twoheadrightarrow_\beta N \text{ and } N \text{ is a head active form}\}$  where  $N$  is a *head active form* if  $M = \lambda x_1 \dots x_n. R P_1 \dots P_k$  and  $R$  is root-active.
- (2)  $\mathcal{IL} = \{M \in \Lambda^\infty \mid M \twoheadrightarrow_\beta N \text{ and } N \text{ is an infinite left spine form}\}$  where  $N$  is an *infinite left spine form* if  $N = \lambda x_1 \dots x_n. ((\dots P_2) P_1)$ .
- (3)  $\mathcal{O} = \{M \in \Lambda^\infty \mid M \twoheadrightarrow_\beta \mathbf{O}\}$ .

The three sets can be characterised alternatively using the notion of Berarducci tree which can reveal more detail of a term than Böhm trees do.

**Lemma 3.7.** (1)  $M \in \mathcal{HA}$  if and only if the Berarducci tree of  $M$  is of the form

$$\lambda x_1 \dots x_n. \perp N_m \dots N_1$$

for some natural numbers  $n, m$ .

(2)  $M \in \mathcal{IL}$  if and only if the Berarducci tree of  $M$  is of the form

$$\lambda x_1 \dots x_n. (((\dots) N_3) N_2) N_1$$

for some natural numbers  $n$ .

(3)  $M \in \mathcal{O}$  if and only if the Berarducci tree of  $M$  is  $\lambda x_1 x_2 x_3 \dots$ , ie.  $\mathbf{O}$ .

The union of the  $\mathcal{HA}$ ,  $\mathcal{IL}$  and  $\mathcal{O}$  is the set of unsolvables. With help of these three sets we can refine the notion of Böhm reduction. We will represent each set by its own truth value. Instead of replacing unsolvable all  $\lambda$ -terms by  $\perp$  we will now replace the elements in  $\mathcal{HA}$ ,  $\mathcal{IL}$  and  $\mathcal{O}$  by, respectively, the constants  $\perp_{\mathcal{HA}}$ ,  $\perp_{\mathcal{IL}}$  and  $\perp_{\mathcal{O}}$ , so that instead of one  $\perp$ -reduction  $\rightarrow_\perp$  we have now three reduction rules, that we denote by  $\rightarrow_{\perp_{\mathcal{HA}}}$ ,  $\rightarrow_{\perp_{\mathcal{IL}}}$  and  $\rightarrow_{\perp_{\mathcal{O}}}$ . We will use  $\perp_{\mathcal{HA}}$ ,  $\perp_{\mathcal{IL}}$  and  $\perp_{\mathcal{O}}$  as truth values next to  $\mathbf{T}$  and  $\mathbf{F}$  to interpret five-valued propositional logic.

In the same fashion, if we split the unsolvables in only two sets  $\mathcal{HA}$  and  $\mathcal{IL} \cup \mathcal{O}$  and introduce besides  $\perp_{\mathcal{HA}}$  a single constant  $\perp_{\mathcal{IL} \cup \mathcal{O}}$  to replace the elements in  $\mathcal{IL} \cup \mathcal{O}$ , we have the ingredients to interpret four-valued propositional logic.

These constructions work because of the following theorem.

- Theorem 3.8.** (1) Let  $\Lambda_{\perp_{\mathcal{HA}} \perp_{\mathcal{IL}} \perp_{\mathcal{O}}}^\infty$  be the set of finite and infinite  $\lambda$ -terms constructed with the symbols  $\perp_{\mathcal{HA}}$ ,  $\perp_{\mathcal{IL}}$  and  $\perp_{\mathcal{O}}$ . Then the infinitary  $\lambda$ -calculus  $\lambda_{\beta \perp_{\mathcal{HA}} \perp_{\mathcal{IL}} \perp_{\mathcal{O}}}^\infty$  is confluent and normalising for (strongly) convergent reduction.
- (2) Let  $\Lambda_{\perp_{\mathcal{HA}} \perp_{\mathcal{IL} \cup \mathcal{O}}}^\infty$  be the set of finite and infinite  $\lambda$ -terms constructed with the symbols  $\perp_{\mathcal{HA}}$  and  $\perp_{\mathcal{IL} \cup \mathcal{O}}$ . Then the infinitary  $\lambda$ -calculus  $\lambda_{\beta \perp_{\mathcal{HA}} \perp_{\mathcal{IL} \cup \mathcal{O}}}^\infty$  is confluent and normalising for (strongly) convergent reduction.

*Proof.* Both follow from Lemma 3.7 and two facts from [KKSdV97], namely that  $\lambda_{\beta \perp_{\mathcal{R}}}^\infty$  is confluent and normalising, and that  $\perp_{\mathcal{R}}$ -reduction can be postponed over  $\beta$ -reduction.  $\square$

We will now encode five-valued logic in  $\lambda$ -calculus using the same logical operators as before together with the five truth values from  $\{\mathbf{T}, \mathbf{F}, \perp_{\mathcal{HA}}, \perp_{\mathcal{IL}}, \perp_{\mathcal{O}}\}$ . Similarly using the four truth values from  $\{\mathbf{T}, \mathbf{F}, \perp_{\mathcal{HA}}, \perp_{\mathcal{IL} \cup \mathcal{O}}\}$  we will encode four-valued logic.

We need an analogue of Corollary 3.4.

**Lemma 3.9.** Let  $U$  be a  $\lambda$ -term in  $\mathcal{HA}$  ( $\mathcal{IL}$ ,  $\mathcal{O}$  and  $\mathcal{IL} \cup \mathcal{O}$ ). Then  $\neg U$ ,  $U \wedge N$ ,  $U \vee N$  and  $U \rightarrow N$  are all terms in  $\mathcal{HA}$  ( $\mathcal{IL}$ ,  $\mathcal{O}$  and  $\mathcal{IL} \cup \mathcal{O}$ ).

*Proof.* Immediate from the definitions. For instance, suppose  $U \in \mathcal{HA}$ , that is suppose the Berarducci tree of  $U$  is of the form  $\lambda x_1 \dots x_n. \perp N_m \dots N_1$ . Then the Berarducci tree of  $\neg U$  is the Berarducci tree of  $(\lambda x_1 \dots x_n. \perp N_m \dots N_1) \mathbf{FT}$ . One easily sees that  $\neg U$  is an element of  $\mathcal{HA}$ .  $\square$

**Corollary 3.10.** (1) *The normal forms in  $\lambda_{\beta \perp \mathcal{HA} \perp \mathcal{IL} \perp \mathcal{O}}^\infty$  of  $\neg \perp_{\mathcal{X}}$ ,  $\perp_{\mathcal{X}} \delta N$ ,  $\perp_{\mathcal{X}} \vee N$  and  $\perp_{\mathcal{X}} \rightarrow N$  are all equal to  $\perp_{\mathcal{X}}$  for  $\mathcal{X} \in \{\mathcal{HA}, \mathcal{IL}, \mathcal{O}\}$ .*  
(2) *The normal forms in  $\lambda_{\beta \perp \mathcal{HA} \perp \mathcal{IL} \cup \mathcal{O}}^\infty$  of  $\neg \perp_{\mathcal{X}}$ ,  $\perp_{\mathcal{X}} \delta N$ ,  $\perp_{\mathcal{X}} \vee N$  and  $\perp_{\mathcal{X}} \rightarrow N$  are all equal to  $\perp_{\mathcal{X}}$  for  $\mathcal{X} \in \{\mathcal{HA}, \mathcal{IL} \cup \mathcal{O}\}$ .*

**Theorem 3.11.** (1) *Let  $\phi$  be a finite or infinite closed proposition with truth values from  $\{\mathbf{T}, \mathbf{F}, \perp_{\mathcal{HA}}, \perp_{\mathcal{IL}}, \perp_{\mathcal{O}}\}$ . Then the normal form of  $\phi$  in  $\lambda_{\beta \perp \mathcal{HA} \perp \mathcal{IL}}^\infty \mathcal{O}$  is either  $\mathbf{T}$ ,  $\mathbf{F}$ ,  $\perp_{\mathcal{HA}}$ ,  $\perp_{\mathcal{IL}}$  or  $\perp_{\mathcal{O}}$ .*

(2) *Let  $\phi$  be a finite or infinite closed proposition with truth values from  $\{\mathbf{T}, \mathbf{F}, \perp_{\mathcal{HA}}, \perp_{\mathcal{IL} \cup \mathcal{O}}\}$ . Then the normal form of  $\phi$  in  $\lambda_{\beta \perp \mathcal{HA} \perp \mathcal{IL} \cup \mathcal{O}}^\infty$  is either  $\mathbf{T}$ ,  $\mathbf{F}$ ,  $\perp_{\mathcal{HA}}$  or  $\perp_{\mathcal{IL} \cup \mathcal{O}}$ .*

*Proof.* By coinduction!  $\square$

Using Corollary 3.10, it is straightforward to calculate the truth tables for a four-valued logic encoded in  $\lambda$ -calculus:

$\neg$		$\delta$	$\mathbf{T}$	$\mathbf{F}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$
$\mathbf{T}$	$\mathbf{F}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{F}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{IL}}$
$\mathbf{F}$	$\mathbf{T}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{F}$
$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{HA}}$
$\perp_{\mathcal{IL} \cup \mathcal{O}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$
$\vee$		$\mathbf{T}$	$\mathbf{F}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$	
$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{T}$	
$\mathbf{F}$	$\mathbf{T}$	$\mathbf{T}$	$\mathbf{F}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$	
$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{HA}}$	$\perp_{\mathcal{HA}}$	
$\perp_{\mathcal{IL} \cup \mathcal{O}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$	$\perp_{\mathcal{IL} \cup \mathcal{O}}$	

Figure 4: Left-sequential four-valued propositional logic

As it happens, this four-valued propositional logic has been studied by Bergstra and Van de Pol [BvdP96, BvdP11]. In the context of process algebra enriched with conditional statements the need for many-valued logic arises in case a condition evaluates to a truth value (*e.g.*, error/exceptions and divergences) different from true or false. This led Bergstra and his colleagues to a study of a great many of versions of three-, four- and even five-valued logic [BBR95, BP98, BP00, BP99].

For the four-valued logic of Figure 4 Bergstra and Van de Pol gave a complete axiomatisation in [BvdP96, BvdP11]. See Figure 5. They use  $\mathbf{m}$  (meaningless) for  $\perp_{\mathcal{HA}}$  and  $\mathbf{d}$  (divergence) for  $\perp_{\mathcal{IL} \cup \mathcal{O}}$ . These names make some sense here as well. The terms in  $\mathcal{IL} \cup \mathcal{O}$  can be called diverging as they have limits with infinite left spines. On the other hand, terms in  $\mathcal{HA}$  reduce by definition to terms of the form  $M = \lambda x_1 \dots x_n. RP_1 \dots P_k$  with  $R$  is root-active. This term  $R$  is meaningless, in the sense that it will not reveal any further information how long one may reduce it.

(1)	$\neg \mathbf{d} = \mathbf{d}$
(2)	$\neg \mathbf{m} = \mathbf{m}$
(3)	$\neg \mathbf{T} = \mathbf{F}$
(4)	$\neg \neg x = x$
(5)	$\mathbf{T} \delta x = x$
(6)	$\mathbf{F} \delta x = \mathbf{F}$
(7)	$x \overset{\circ}{\vee} y = \neg(\neg x \delta \neg y)$
(8)	$x \delta (y \delta z) = (x \delta y) \delta z$
(9)	$(x \overset{\circ}{\vee} y) \delta z = (\neg x \delta y \delta z) \vee (x \delta z) \overset{\circ}{\vee} (x \delta z)$

Figure 5: Bergstra and Van de Pol’s axiomatisation of left-sequential four-valued propositional logic

The axioms in Figure 5 have been selected carefully: each is independent of the others. They also note that Axiom (11) of Figure 3,

$$(x \delta y) \overset{\circ}{\vee} (y \delta x) = (y \delta x) \overset{\circ}{\vee} (x \delta y)$$

does not hold in four-valued logic. We can recognise that in our context if we substitute  $\perp_{\mathcal{H}\mathcal{A}}$  for  $x$  and  $\perp_{\mathcal{I}\mathcal{L}\cup\mathcal{O}}$  for  $y$ . Then by Lemma 3.9 we see immediately that the left-hand side of the axiom reduces to  $\perp_{\mathcal{H}\mathcal{A}}$ , while the right-hand side reduces to  $\perp_{\mathcal{I}\mathcal{L}\cup\mathcal{O}}$ .

Similarly, the truth tables for the five-valued logic encoded in  $\lambda$ -calculus are as follows:

$\neg$		$\delta$	<b>T</b>	<b>F</b>	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{O}}$
<b>T</b>	<b>F</b>	<b>T</b>	<b>T</b>	<b>F</b>	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{O}}$
<b>F</b>	<b>T</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>	<b>F</b>
$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{H}\mathcal{A}}$
$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{I}\mathcal{L}}$
$\perp_{\mathcal{O}}$	$\perp_{\mathcal{O}}$	$\perp_{\mathcal{O}}$	$\perp_{\mathcal{O}}$	$\perp_{\mathcal{O}}$	$\perp_{\mathcal{O}}$	$\perp_{\mathcal{O}}$	$\perp_{\mathcal{O}}$

  

$\overset{\circ}{\vee}$	<b>T</b>	<b>F</b>	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{O}}$
<b>T</b>	<b>T</b>	<b>T</b>	<b>T</b>	<b>T</b>	<b>T</b>
<b>F</b>	<b>T</b>	<b>F</b>	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{O}}$
$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{H}\mathcal{A}}$	$\perp_{\mathcal{H}\mathcal{A}}$
$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{I}\mathcal{L}}$	$\perp_{\mathcal{I}\mathcal{L}}$
$\perp_{\mathcal{O}}$	$\perp_{\mathcal{O}}$	$\perp_{\mathcal{O}}$	$\perp_{\mathcal{O}}$	$\perp_{\mathcal{O}}$	$\perp_{\mathcal{O}}$

Figure 6: Left-sequential five-valued propositional logic

Finally using Corollary 3.10, it is also straightforward to calculate the truth tables of a five-valued logic encoded in  $\lambda$ -calculus. See Figure 6. This is the five-valued logic that Bergstra and Van de Pol left implicit in their final remark in [BvdP11] that their complete axiomatisation generalises to five- and higher valued logics, as long as one adds axioms of the form  $\neg p = p$  for each new truth value  $p$ .

4. ENCODING THREE-VALUED LOGIC IN THE FINITE  $\lambda\mathbf{I}$ -CALCULUS

The  $\lambda$ -calculus that Church used in his unfortunate attempt towards a foundation of mathematics was the  $\lambda\mathbf{I}$ -calculus. This calculus differs from the common  $\lambda$ -calculus  $\lambda_\beta$  by a restriction on the set of  $\lambda$ -terms. Terms in the  $\lambda\mathbf{I}$ -calculus only contain abstractions of the form  $\lambda x.M$  if  $x$  occurs free in  $M$ . For example the terms  $\lambda xy.x$  and  $\lambda xy.y$  that we used for the Booleans are now forbidden. So we cannot use the Church encoding of Boolean logic as before.

The consequence of this restriction is that terms in the  $\lambda\mathbf{I}$ -calculus have two properties Church deemed important: (i) if a term has a finite normal form, it cannot have an infinite reduction, and (ii) if a term has a finite normal form then all its subterms must also have a normal form [Chu41]. These properties don't hold in the classical  $\lambda$ -calculus.

**4.1. Another encoding of the Booleans.** Barendregt gave in fact two encodings for the Booleans in his book [Bar84]. Besides the previous well-known encoding of the Booleans he also defined an encoding of the Booleans in the spirit of Church, because the new encodings of the Booleans are terms in the  $\lambda\mathbf{I}$ -calculus.

$$\begin{aligned}\mathbf{T}_I &= \lambda xy.y\mathbf{I}\mathbf{I}x \\ \mathbf{F}_I &= \lambda x.x\mathbf{I}\mathbf{I}\mathbf{I}\end{aligned}$$

In this case we cannot derive (3.2). Instead we get

$$\begin{aligned}\text{if } \mathbf{T}_I \text{ then } M \text{ else } N &\rightarrow N\mathbf{I}\mathbf{I}M \\ \text{if } \mathbf{F}_I \text{ then } M \text{ else } N &\rightarrow M\mathbf{I}\mathbf{I}\mathbf{I}N\end{aligned}\tag{4.1}$$

Yet by inspection of each of the four concrete options for  $M, N \in \{\mathbf{T}_I, \mathbf{F}_I\}$  we find that

$$\begin{aligned}N\mathbf{I}\mathbf{I}M &\rightarrow M \\ M\mathbf{I}\mathbf{I}\mathbf{I}N &\rightarrow N\end{aligned}\tag{4.2}$$

Combining (4.1) with (4.2) gives us (3.2) for all Booleans  $M, N \in \{\mathbf{T}_I, \mathbf{F}_I\}$ . Hence also this less well-known encoding validates the truth tables of Boolean propositional logic.

**4.2. Böhm trees in the finite  $\lambda\mathbf{I}$ -calculus.** Church strongly preferred the  $\lambda\mathbf{I}$ -calculus over the unrestricted  $\lambda$ -calculus. For him the natural notion of the meaning of a  $\lambda$ -term is its finite normal form, provided it exists. Terms without finite normal form he considered to be meaningless or, in his own alternative wording, insignificant [Chu41]. In the  $\lambda\mathbf{I}$ -calculus terms without finite normal form can safely be identified. In the unrestricted calculus this leads to inconsistency [Bar84, Proposition 2.2.4].

In fact, Barendregt [Bar73] has shown that the unsolvable terms in the  $\lambda\mathbf{I}$ -calculus are precisely the terms without finite normal form. Klop [Klo75] gave a simpler proof. They did not consider the Böhm tree construction. But in setting of the  $\lambda\mathbf{I}$ -calculus the Böhm tree construction simplifies enormously. There is no need to consider infinite terms and infinite reductions. We just add the fresh symbol  $\perp$  to the syntax of the  $\lambda\mathbf{I}$ -calculus plus the rule

$$M \rightarrow_{\perp} \perp, \text{ whenever } M[\perp := \Omega] \text{ has no finite normal form.}$$

Let us denote this extension of the  $\lambda\mathbf{I}$ -calculus by  $\lambda\mathbf{I}_{\beta\perp}$ . The extension  $\lambda\mathbf{I}_{\beta\perp}$  is confluent and normalising in the finitary sense, and the Böhm tree of any  $\lambda\mathbf{I}$ -term equals either  $\perp$  or is a finite  $\perp$ -free normal form. In the past we have overlooked this construction, after

observing that the set of  $\lambda\mathbf{I}$ -terms is not closed under infinite  $\beta$ -reduction. In the limit a bound variable may “drop off”. For instance, consider  $\lambda v.\Theta(\lambda xyz.xy)v$ . Since

$$\lambda v.\Theta(\lambda xyz.xy)v \rightarrow \lambda v.(\lambda xyz.xy)(\Theta(\lambda xyz.xy))v \rightarrow \lambda v\lambda z.\Theta(\lambda xyz.xy)v$$

we find that the finite  $\lambda\mathbf{I}$ -term  $\lambda v.\Theta(\lambda xyz.xy)v$  has an infinite reduction to the infinite term  $\lambda v\lambda z\lambda z\lambda z\dots$  which is no longer a  $\lambda\mathbf{I}$ -term.

The above  $\perp$ -rule resolves this problem, because now the term  $\lambda v.\Theta(\lambda xyz.xy)v$  reduces in one step to  $\perp$ . In  $\lambda\mathbf{I}_{\beta\perp}$  there is no need to consider infinite reduction as any finite  $\lambda\mathbf{I}$ -term has a finite reduction to finite Böhm tree in  $\lambda\mathbf{I}_{\beta\perp}$ . Hence we can encode three valued-logic in  $\lambda\mathbf{I}_{\beta\perp}$  if we take as truth values  $\mathbf{T}_\mathbf{I}$ ,  $\mathbf{F}_\mathbf{I}$  and  $\perp$ .

**Lemma 4.1.** *Let  $U$  be an unsolvable  $\lambda\mathbf{I}$ -term. Then  $\neg U$ ,  $U \wedge N$ ,  $U \vee N$  and  $U \rightarrow N$  are all unsolvable  $\lambda\mathbf{I}$ -terms, when  $N$  is a  $\lambda\mathbf{I}$ -term.*

*Proof.* By Lemma 3.3 it remains to show that  $\neg U$ ,  $U \wedge N$ ,  $U \vee N$  and  $U \rightarrow N$  are  $\lambda\mathbf{I}$ -terms. But this follows from the fact that  $\mathbf{T}_\mathbf{I}$  and  $\mathbf{F}_\mathbf{I}$  are  $\lambda\mathbf{I}$ -terms.  $\square$

Thus, despite a different encoding of the Booleans, we find the same truth tables of McCarthy’s left-sequential three-valued propositional logic of Figure 2. Note that the earlier partition in Section 3.4 of the unsolvables based on the form of the left spine of their Berarducci tree applies verbatim to  $\lambda\mathbf{I}$ -terms. Hence, also this second encoding of the Booleans refines to an encoding of the same earlier four- and five-valued logics in the  $\lambda\mathbf{I}$ -calculus.

**4.3. Why Curry’s Paradox does not apply.** We end with noting that Curry’s Paradox does not apply to the finite  $\lambda\mathbf{I}$ -calculus because the above infinitary extension is consistent.

Contemporaneously with Church, Curry had been searching for a symbolic logic that could serve as foundation of mathematics. The technique by which Kleene and Rosser [KR35] found the inconsistency in the symbolic logic of Church also applied to some of the systems of illative combinatoric logic that Curry was exploring. In contrast to Church, Curry had not committed himself to an underlying philosophy. He considered the Kleene-Rosser paradox an helpful instrument in the search for “stronger and stronger systems which are consistent” as well as “weaker and weaker systems which are inconsistent” [CF58].

In 1942 Curry published a short and self-contained argument to show the inconsistency for the type of symbolic logics that he and Church were working on. Curry showed that that any combinatory complete system, like e.g.  $\lambda$ -calculus, with an implication operator satisfying:

$$\begin{aligned} X &\rightarrow X \\ (X &\rightarrow (X \rightarrow Y)) \rightarrow (X \rightarrow Y) \end{aligned}$$

is inconsistent. The elegant short proof of the Curry’s Paradox can be found in [CF58, Bar84].

As the infinitary extensions  $\lambda_\beta$  and the  $\lambda\mathbf{I}$ -calculus  $\lambda$ -calculus are consistent (the normal forms of  $\mathbf{T}$  and  $\mathbf{F}$  are not equal in them) the Curry’s Paradox does not apply to them. More direct: the implication  $X \rightarrow Y$  does not satisfy the above two conditions for implication: if  $X$  is  $\perp$  then both expressions reduce to  $\perp$  for any value of  $Y$ .

## 5. CONCLUSION

The idea to solve Russell's paradox with three-valued logic is not at all new. Feferman gave various pointers in [Fef84]. But the conjunctions and disjunctions of the three-valued logics that are considered for that purpose all seem to be commutative in contrast to those in the left-sequential McCarthy logic that we use here.

It is possible to further refine the encoding to an encoding of  $\infty$ -valued logic in  $\lambda$ -calculus. The new truth values then correspond to the different shapes of left spine that unsolvables can have. We see no further use for that.

## ACKNOWLEDGEMENTS

We would like to thank the editor for her encouragement and infinite patience, and the referees for careful reading of our paper and their constructive comments that helped to improve this paper.

## REFERENCES

- [Bar73] Henk P. Barendregt. A characterization of terms of the  $\lambda$ I-calculus having a normal form. *Journal of Symbolic Logic*, 38(3):441–445, 1973.
- [Bar77a] H. P. Barendregt. The type free lambda calculus. In Jon Barwise, editor, *Handbook of Mathematical Logic*, pages 1091–1132. North-Holland Publishing Company, Amsterdam, 1977.
- [Bar77b] H.P. Barendregt. Solvability in lambda calculi. In M. Guillaume, editor, *Colloque international de logique: Clermont-Ferrand, 18-25 juillet 1975*, pages 209–219. Paris: Éditions du C.N.R.S., 1977.
- [Bar84] H. P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. North-Holland, Amsterdam, Revised edition, 1984.
- [BB85] Corrado Böhm and Alessandro Berarducci. Automatic synthesis of typed lambda-programs on term algebras. *Theor. Comput. Sci.*, 39:135–154, 1985.
- [BBR95] Jan A. Bergstra, Inge Bethke, and Piet Rodenburg. A propositional logic with 4 values: true, false, divergent and meaningless. *Journal of Applied Non-Classical Logics*, 5(2):199–217, 1995.
- [BDC74] Corrado Böhm and Mariangiola Dezani-Ciancaglini. Combinatorial problems, combinator equations and normal forms. In Jacques Loeckx, editor, *Automata, Languages and Programming*, pages 185–199. Springer, 1974.
- [Ber96] A. Berarducci. Infinite  $\lambda$ -calculus and non-sensible models. In *Logic and algebra (Pontignano, 1994)*, pages 339–377. Dekker, New York, 1996.
- [Böhm68] Corrado Böhm. Alcune proprietà delle forme  $\beta$ - $\eta$ -normali del nel  $\lambda$ -K-calcolo. Technical report, Pubbl. IAC, n. 696, Roma, 1968.
- [BP98] Jan A. Bergstra and Alban Ponse. Bochvar-McCarthy logic and process algebra. *Notre Dame Journal of Formal Logic*, 39(4):464–484, 1998.
- [BP99] Jan A. Bergstra and Alban Ponse. Process algebra with five-valued conditions. In *Combinatorics, Computation, and Logic, Proceedings of DMTCS'99 and CATS'99, Auckland*, volume 21(3) of *Australian Computer Science Communications*, pages 128–143. Springer-Verlag, 1999.
- [BP00] Jan A. Bergstra and Alban Ponse. Process algebra with four-valued logic. *Journal of Applied Non-Classical Logics*, 10(1):27–53, 2000.
- [BP11] Jan A. Bergstra and Alban Ponse. Proposition algebra. *ACM Trans. Comput. Log.*, 12(3):21:1–21:36, 2011.
- [BvdP96] J. A. Bergstra and J. van de Pol. A calculus for sequential logic with 4 values. Technical Report 160, Utrecht University, Dept. of Philosophy, 1996.
- [BvdP11] J. A. Bergstra and J. van de Pol. A calculus for four-valued sequential logic. *Theor. Comput. Sci.*, 412(28):3122–3128, 2011.
- [CF58] H.B. Curry and R. Feys. *Combinatory Logic*, volume I. North-Holland Publishing Company, Amsterdam, 1958.

- [Chu28] Alonzo Church. On the law of excluded middle. *Bull. Amer. Math. Soc.*, 34(1):75–78, 01 1928.
- [Chu32] Alonzo Church. A set of postulates for the foundation of logic. *Annals of Mathematics*, 33(2):346–366, 1932.
- [Chu34] Alonzo Church. The Richard paradox. *The American Mathematical Monthly*, 41(6):356–361, 1934.
- [Chu36a] Alonzo Church. A note on the Entscheidungsproblem. *J. Symbolic Logic*, 1(1):40–41, 03 1936.
- [Chu36b] Alonzo Church. An unsolvable problem of elementary number theory. *American Journal of Mathematics*, 58(2):345–363, 1936.
- [Chu41] A. Church. *The Calculi of Lambda Conversion*. Princeton University Press, 1941.
- [CR36] A. Church and J. B. Rosser. Some properties of conversion. *Trans.Amer.Math. Soc.*, 39:472–482, 1936.
- [Fef84] Solomon Feferman. Toward useful type-free theories. I. *The Journal of Symbolic Logic*, 49(1):75–111, 1984.
- [GS90] Fernando Guzmán and Craig C. Squier. The algebra of conditional logic. *Algebra Universalis*, 27(1):88–110, 1990.
- [KdV03] J. R. Kennaway and F. J. de Vries. Infinitary rewriting. In Terese, editor, *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*, pages 668–711. Cambridge University Press, 2003.
- [KKSdV97] J. R. Kennaway, J. W. Klop, M. R. Sleep, and F. J. de Vries. Infinitary lambda calculus. *Theoretical Computer Science*, 175(1):93–125, 1997.
- [Kle36a] S. C. Kleene.  $\lambda$ -definability and recursiveness. *Duke Math. J.*, 2(2):340–353, 06 1936.
- [Kle36b] S.C. Kleene. General recursive functions of natural numbers. *Mathematische Annalen*, 112:727–742, 1936.
- [Klo75] J. W. Klop. On solvability by  $\lambda$ I-terms. In C. Böhm, editor,  *$\lambda$ -Calculus and Computer Science Theory*, pages 342–345, Berlin, Heidelberg, 1975. Springer Berlin Heidelberg.
- [KR35] S. C. Kleene and J. B. Rosser. The inconsistency of certain formal logics. *Annals of Mathematics*, 36(3):630–636, 1935.
- [KvOdV99] J. R. Kennaway, V. van Oostrom, and F. J. de Vries. Meaningless terms in rewriting. *Journal of Functional and Logic Programming*, 1999(1), 1999.
- [Lan64] P. J. Landin. The Mechanical Evaluation of Expressions. *The Computer Journal*, 6(4):308–320, January 1964.
- [McC60] John McCarthy. Recursive functions of symbolic expressions and their computation by machine, part I. *Commun. ACM*, 3(4):184–195, 1960.
- [McC63] John McCarthy. A basis for a mathematical theory of computation. In P. Braffort and D. Hirschberg, editors, *Computer Programming and Formal Systems*, volume 35 of *Studies in Logic and the Foundations of Mathematics*, pages 33 – 70. Elsevier, 1963.
- [McC78] John McCarthy. History of LISP. *SIGPLAN Not.*, 13(8):217–223, August 1978.
- [SdV11a] P. Severi and F. J. de Vries. Weakening the axiom of overlap in infinitary lambda calculus. In M. Schmidt-Schauß, editor, *Proceedings of the 22nd International Conference on Rewriting Techniques and Applications, May 30 - June 1, 2011, Novi Sad, Serbia*, volume 10 of *LIPICs*, pages 313–328. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2011.
- [SdV11b] P. G. Severi and F. J. de Vries. Decomposing the lattice of meaningless sets in the infinitary lambda calculus. In L. D. Beklemishev and R. de Queiroz, editors, *Logic, Language, Information and Computation - 18th International Workshop, WoLLIC 2011, Philadelphia, PA, USA, May 18-20, 2011. Proceedings*, pages 210–227, 2011.