Delimited control operators prove Double-negation Shift

Danko Ilik[1]

Ecole Polytechnique, INRIA, CNRS & Université Paris Diderot
Address: INRIA PI-R2, 23 avenue d'Italie, CS 81321, 75214 Paris Cedex 13, France
E-mail: danko.ilik@polytechnique.edu

Abstract

We propose an extension of minimal intuitionistic predicate logic, based on delimited control operators, that can derive the predicate-logic version of the Double-negation Shift schema, while preserving the disjunction and existence properties.

Keywords: delimited control operators, Double-negation Shift, disjunction property, existence property, intermediate logic
2000 MSC: 03B20, 03B40, 68N18, 03F55, 03F50, 03B55

1. Introduction

In [22], Hugo Herbelin showed that, by extending the proof-term calculus of intuitionistic predicate logic with a restricted form of *delimited control operators*, one can obtain a logical system able to derive a predicate-logic version of Markov's Principle, $\neg\neg\exists x A(x) \Rightarrow \exists x A(x)$ (for $A(x)$ a $\{\Rightarrow, \forall\}$-free formula), while remaining essentially intuitionistic – satisfying the disjunction and existence properties.

Separately, [21] he also observed that using the full power of delimited control operators one can derive the predicate-logic version of the Double-negation Shift schema, $\forall x \neg\neg A(x) \Rightarrow \neg\neg\forall x A(x)$ (where $A(x)$ is arbitrary), and posed the question whether there is a corresponding logical system which also posseses the disjunction and existence properties. With this article, we answer Herbelin's question in the affirmative.

Delimited control operators have appeared in Theoretical Computer Science, in Semantics of Programming Languages, as a powerful abstraction to account for so-called *computational effects*. While being pervasive in the practise of writing computer programs (for, they include facilities as basic as reading from and writing into memory, stopping the execution of the program, or parallel computation), giving a good mathematical explanation of effects is still one of the major research topics in Semantics.

An important step in that direction was a result of Filinski [13, 14], who showed that every monadic computational effect can be operationally simulated by the delimited control operators *shift/reset*, introduced previously by himself and Danvy [9, 10].

[1]Present address: Faculty of Informatics, University "Goce Delčev", PO Box 201, 2000 Štip, Macedonia; E-mail: danko.ilik@ugd.edu.mk

However, the logical status of shift/reset themselves, and other such operators in general, remains to be fully established, something we hope to being contributing to with this article.

Another interest in delimited control operators, and actually our original interest in them, comes from the role they promise to be playing in a future constructive proof of completeness of full intuitionistic logic (with $\vee$ and $\exists$) with respect to Kripke semantics, something described in [25, 24].

We illustrate the utility of delimited control operators, by considering some examples in a λ-calculus extended with them, containing also natural numbers with the plus operation. The extension consists of two constructs, a *delimiter* (# – "reset") and a *control operator* ($\mathcal{S}$ – "shift"). The delimiter is used as a special kind of brackets in a λ-term, so that the control operator, which can only appear inside such "brackets", be able to gain control of its surrounding context, up to the delimiter. For example, in the following λ-term reduction,

$$1 + \#2 + \mathcal{S}k.4 \quad\to\quad 1 + \#4\,\{(\lambda a.\#2 + a)/\,k\} \quad=\quad 1 + \#4 \quad\to\quad 1 + 4 \to 5,$$

reset is used to delimit the sub-term $2 + \mathcal{S}k.4$. Shift then behaves as a binder, alike λ-abstraction, that names the abstracted surroundings of shift, $2 + \Box$, by k, and replaces in its sub-expression, 4, all occurrences of k by the abstracted surroundings. In this case, k is not used inside shift – this corresponds to the so-called "exceptions" effect that Herbelin found out to be the computational contents behind Markov's Principle. In the next example, k is used; the sub-term inside shift uses its surrounding context twice:

$$\begin{aligned}
&1 + \#2 + \mathcal{S}k.k4 + k8\\
\to\,&1 + \#(\lambda a.\#2 + a)4 + (\lambda a.\#2 + a)8\\
\to^{+}&1 + \#(\#6) + (\#10)\\
\to^{+}&1 + \#6 + 10\\
\to^{+}&17
\end{aligned}$$

From the logical perspective, considering natural deduction formalisms which can be isomorphically presented by proof-λ-terms, we see delimited control, when added to the syntax of such proof terms, as a means of being able to access *a certain part* of the surroundings of a proof term from inside the proof term itself.[2] The part of the surrounding that we want to be able to access will be defined as a "pure evaluation context" in Section 2; logically, it is the surroundings of a proof term for a $\{\Rightarrow, \forall\}$-free formula,[3] which is the predicate logic equivalent of arithmetic Σ^0_1-formulae, for

[2]This is to be contrasted to what happens with the (undelimited) control operator *call/cc*, which is better known in Logic for its role in the development of classical realisability [20, 31, 32, 33] – call/cc amounts computationally to aborting the entire computation and, since its effect is not delimited, one has no hope of getting a natural computational interpretation from classical realisability: a realiser of an existential statement needs not be a program which computes a witness for the existential quantifier.

[3]Following Berger [8], we call the $\{\Rightarrow, \forall\}$-free formulae, Σ-formulae, and denote them by S, T, U, while general formulae are denoted by A, B, C.

which we know that classical and intuitionistic provability coincide. In other words, we propose a proof-term calculus for a logic which is essentially intuitionistic, except that at the fragment "Σ^0_1" we are allowed to use classical reasoning to obtain more (succinct) proofs.

The paper is organised as follows. In the next Section 2, we introduce our system MQC^+. The acronym comes from Troelstra: IQC is intuitionistic predicate logic, MQC is minimal predicate logic (IQC without the $\perp_E$ rule), and CQC is classical predicate logic. In Section 3, we characterise the relationship between MQC^+, MQC, and CQC; in particular, we show that an extension to *predicate* logic of Glivenko's Theorem holds for our system, unlike for MQC. In Section 4, we prove properties of the reduction relation on proof terms, from which we obtain the Disjunction and Existence Property for closed derivations of MQC^+. In the final Section 5, we discuss related and future work.

2. The system MQC^+

The natural deduction system of MQC^+ is shown in Table 1. It consists of the proof rules of minimal intuitionistic predicate logic MQC, plus two new ones, "shift" ($\mathcal{S}$) and "reset" (#).

The turnstile symbol "$\vdash$" can carry an annotation – a Σ-formula T – which is neither used nor changed by the intuitionistic rules. We use the wild-card symbol $\diamond$ for this purpose, to mean that there either is an annotating formula T, or that there is none. In the proof rules where the wild-card appears both above and below the line, it means that either there is the same annotation both above and below, or that there is no annotation above and no annotation below.

The rule (#) can only be applied when the conclusion is a Σ-formula T. It acts as a delimiter in the proof tree, (re-)initialising the annotation with the formula T; from that point upwards in the tree, classical reasoning is allowed – but, only so because we are ultimately proving a Σ-formula. The rule ($\mathcal{S}$) can then be used, inside a sub-tree with (#) at its root, as a kind of ($\neg\neg_E$) rule. Its role is to "escape" to the nearest enclosing delimiter once a witness for the Σ-formula from the annotation has been found.

However, note that, although there can be arbitrarily many uses of the (#) and ($\mathcal{S}$) rules in a derivation tree, *only one* formula T is allowed to appear in annotations, globally, of a derivation tree. This means that the global T is set once and for all, hence it is not possible to mix derivations using different T and T'. Were we in IQC, a natural choice for the global T would have been $\perp$.

As examples, we give the derivations for (generalisations of) the minimal-predicate-logic versions[4] of Markov's Principle,

$$(T \Rightarrow S) \Rightarrow ((S \Rightarrow T) \Rightarrow T) \Rightarrow S, \qquad (\mathrm{MP}_T)$$

and Double-negation Shift,

$$\forall x\, ((A(x) \Rightarrow T) \Rightarrow T) \Rightarrow (\forall x A(x) \Rightarrow T) \Rightarrow T, \qquad (\mathrm{DNS}_T)$$

[4] The distinguished formula T plays the role of $\perp$ and the hypothesis $T \Rightarrow S$ plays the role of the $\perp_E$ rule.

$$\frac{A \in \Gamma}{\Gamma \vdash_\diamond A}\ \mathrm{Ax}$$

$$\frac{\Gamma \vdash_\diamond A_1 \qquad \Gamma \vdash_\diamond A_2}{\Gamma \vdash_\diamond A_1 \wedge A_2}\ \wedge_I \qquad\qquad \frac{\Gamma \vdash_\diamond A_1 \wedge A_2}{\Gamma \vdash_\diamond A_i}\ \wedge_E^i$$

$$\frac{\Gamma \vdash_\diamond A_i}{\Gamma \vdash_\diamond A_1 \vee A_2}\ \vee_I^i \qquad\qquad \frac{\Gamma \vdash_\diamond A_1 \vee A_2 \qquad \Gamma, A_1 \vdash_\diamond C \qquad \Gamma, A_2 \vdash_\diamond C}{\Gamma \vdash_\diamond C}\ \vee_E$$

$$\frac{\Gamma, A_1 \vdash_\diamond A_2}{\Gamma \vdash_\diamond A_1 \Rightarrow A_2}\ \Rightarrow_I \qquad\qquad \frac{\Gamma \vdash_\diamond A_1 \Rightarrow A_2 \qquad \Gamma \vdash_\diamond A_1}{\Gamma \vdash_\diamond A_2}\ \Rightarrow_E$$

$$\frac{\Gamma \vdash_\diamond A(x) \qquad x\text{-fresh}}{\Gamma \vdash_\diamond \forall x A(x)}\ \forall_I \qquad\qquad \frac{\Gamma \vdash_\diamond \forall x A(x)}{\Gamma \vdash_\diamond A(t)}\ \forall_E$$

$$\frac{\Gamma \vdash_\diamond A(t)}{\Gamma \vdash_\diamond \exists x.A(x)}\ \exists_I \qquad\qquad \frac{\Gamma \vdash_\diamond \exists x.A(x) \qquad \Gamma, A(x) \vdash_\diamond C \qquad x\text{-fresh}}{\Gamma \vdash_\diamond C}\ \exists_E$$

$$\frac{\Gamma \vdash_T T}{\Gamma \vdash_\diamond T}\ \#\ (\text{“reset”}) \qquad\qquad \frac{\Gamma, A \Rightarrow T \vdash_T T}{\Gamma \vdash_T A}\ \mathcal{S}\ (\text{“shift”})$$

Table 1: Natural deduction system of MQC$^+$

where, according to the already set convention, T and S are Σ-formulae, while $A(x)$ is a general one.

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\cdots \vdash_S T \Rightarrow S}\,\text{Ax} \qquad \dfrac{\dfrac{}{\cdots \vdash_S (S \Rightarrow T) \Rightarrow T}\,\text{Ax} \qquad \dfrac{\dfrac{\dfrac{}{\cdots, S \vdash_S S}\,\text{Ax}}{\cdots, S \vdash_S T}\,\mathcal{S}}{\cdots \vdash_S S \Rightarrow T}\,\Rightarrow_I}{\cdots \vdash_S T}\,\Rightarrow_E}{T \Rightarrow S, (S \Rightarrow T) \Rightarrow T \vdash_S S}\,\Rightarrow_E}{T \Rightarrow S, (S \Rightarrow T) \Rightarrow T \vdash S}\,\#}{T \Rightarrow S \vdash ((S \Rightarrow T) \Rightarrow T) \Rightarrow S}\,\Rightarrow_I}{\vdash (T \Rightarrow S) \Rightarrow ((S \Rightarrow T) \Rightarrow T) \Rightarrow S}\,\Rightarrow_I
$$

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\cdots \vdash \forall x A(x) \Rightarrow T}\,\text{Ax} \qquad \dfrac{\dfrac{\dfrac{\vdots}{\cdots, \forall x((A(x) \Rightarrow T) \Rightarrow T), A(x) \Rightarrow T \vdash_T T}\,\forall_E, \Rightarrow_E, \text{Ax}}{\cdots, \forall x((A(x) \Rightarrow T) \Rightarrow T) \vdash_T A(x)}\,\mathcal{S}}{\cdots, \forall x((A(x) \Rightarrow T) \Rightarrow T) \vdash_T \forall x A(x)}\,\forall_I, x\text{-fresh}}{\forall x A(x) \Rightarrow T, \forall x((A(x) \Rightarrow T) \Rightarrow T) \vdash_T T}\,\Rightarrow_E}{\forall x A(x) \Rightarrow T, \forall x((A(x) \Rightarrow T) \Rightarrow T) \vdash T}\,\#}{\forall x((A(x) \Rightarrow T) \Rightarrow T) \vdash (\forall x A(x) \Rightarrow T) \Rightarrow T}\,\Rightarrow_I}{\vdash \forall x((A(x) \Rightarrow T) \Rightarrow T) \Rightarrow (\forall x A(x) \Rightarrow T) \Rightarrow T}\,\Rightarrow_I
$$

We now define a calculus of proof-term annotations for the natural deduction system of MQC$^+$, a version of simply typed λ-calculus with constants for handling all logical connectives and the delimited control operators, and then a reduction system for proof terms; the idea is that reducing a proof term describes the process of normalising a natural deduction derivation.

The definitions are based on standard treatments of Logic as λ-calculus (see, for example, [41]), and standard treatment of λ-calculus with shift/reset from Semantics of Programming Languages (for example, [2]).

2.1 Definition. The set of *proof terms* is defined by the following inductive definition,

$$
\begin{aligned}
p, q ::= a \mid \iota_1 p \mid \iota_2 p \mid \mathsf{case}\ p\ \mathsf{of}\ (a_1.q_1 \| a_2.q_2) \mid (p, q) \mid \pi_1 p \mid \pi_2 p \mid \lambda a.p \mid pq \mid \\
\lambda x.p \mid pt \mid (t, p) \mid \mathsf{dest}\ p\ \mathsf{as}\ (x.a)\ \mathsf{in}\ q \mid \#p \mid \mathcal{S}k.p
\end{aligned}
$$

where a, b, k, l denote hypothesis variables, x, y, z denote quantifier variables, and t, u, v denote quantifier terms (individuals); hence, $\lambda a.p$ is a constructor for implication, while $\lambda x.p$ is a constructor for universal quantification; (p, q) is a constructor for conjunction while (t, p) is a constructor for existential quantification, and pq is a destructor for implication while pt is a destructor for universal quantification.

2.2 Remark. The $\mathcal{S}$ in $\mathcal{S}k.p$ is a binder, it binds k in p just as λ binds a in q in a lambda abstraction $\lambda a.q$. Following standard terminology, we sometimes call k a *continuation* variable.

$$\dfrac{(a : A) \in \Gamma}{\Gamma \vdash_\diamond a : A}\ \text{Ax}$$

$$\dfrac{\Gamma \vdash_\diamond p : A_1 \qquad \Gamma \vdash_\diamond q : A_2}{\Gamma \vdash_\diamond (p, q) : A_1 \wedge A_2}\ \wedge_I \qquad\qquad \dfrac{\Gamma \vdash_\diamond p : A_1 \wedge A_2}{\Gamma \vdash_\diamond \pi_i p : A_i}\ \wedge_E^i$$

$$\dfrac{\Gamma \vdash_\diamond p : A_i}{\Gamma \vdash_\diamond \iota_i p : A_1 \vee A_2}\ \vee_I^i$$

$$\dfrac{\Gamma \vdash_\diamond p : A_1 \vee A_2 \qquad \Gamma, a_1 : A_1 \vdash_\diamond q_1 : C \qquad \Gamma, a_2 : A_2 \vdash_\diamond q_2 : C}{\Gamma \vdash_\diamond \mathsf{case}\ p\ \mathsf{of}\ (a_1.q_1 \| a_2.q_2) : C}\ \vee_E$$

$$\dfrac{\Gamma, a : A_1 \vdash_\diamond p : A_2}{\Gamma \vdash_\diamond \lambda a.p : A_1 \Rightarrow A_2}\ \Rightarrow_I \qquad\qquad \dfrac{\Gamma \vdash_\diamond p : A_1 \Rightarrow A_2 \qquad \Gamma \vdash_\diamond q : A_1}{\Gamma \vdash_\diamond pq : A_2}\ \Rightarrow_E$$

$$\dfrac{\Gamma \vdash_\diamond p : A(x) \qquad x\text{-fresh}}{\Gamma \vdash_\diamond \lambda x.p : \forall x A(x)}\ \forall_I \qquad\qquad \dfrac{\Gamma \vdash_\diamond p : \forall x A(x)}{\Gamma \vdash_\diamond pt : A(t)}\ \forall_E$$

$$\dfrac{\Gamma \vdash_\diamond p : A(t)}{\Gamma \vdash_\diamond (t, p) : \exists x.A(x)}\ \exists_I$$

$$\dfrac{\Gamma \vdash_\diamond p : \exists x.A(x) \qquad \Gamma, a : A(x) \vdash_\diamond q : C \qquad x\text{-fresh}}{\Gamma \vdash_\diamond \mathsf{dest}\ p\ \mathsf{as}\ (x.a)\ \mathsf{in}\ q : C}\ \exists_E$$

$$\dfrac{\Gamma \vdash_T p : T}{\Gamma \vdash_\diamond \# p : T}\ \#\ (\text{“reset”}) \qquad\qquad \dfrac{\Gamma, k : A \Rightarrow T \vdash_T p : T}{\Gamma \vdash_T \mathcal{S}k.p : A}\ \mathcal{S}\ (\text{“shift”})$$

Table 2: Proof term annotation for the natural deduction system of MQC$^+$

2.3 Definition. The subset of proof terms known as *values* is defined by:

$$V ::= a \mid \iota_1 V \mid \iota_2 V \mid (V, V) \mid (t, V) \mid \lambda a.p \mid \lambda x.p$$

2.4 Definition. The set of *pure evaluation contexts*, a subset of all proof terms with one placeholder or "hole", is defined by:

$$P ::= [\,] \mid \mathsf{case}\ P\ \mathsf{of}\ (a_1.p_1 \| a_2.p_2) \mid \pi_1 P \mid \pi_2 P \mid \mathsf{dest}\ P\ \mathsf{as}\ (x.a)\ \mathsf{in}\ p \mid$$
$$Pq \mid (\lambda a.q)P \mid Pt \mid \iota_1 P \mid \iota_2 P \mid (P, p) \mid (V, P) \mid (t, P)$$

The association of proof terms to natural deduction derivations is given in Table 2. $P[p]$ denotes the proof term obtained from P by replacing its placeholder $[\,]$ with the proof term p.

In order to define a reduction relation on proof terms we also need the notion of (non-pure) evaluation context.

2.5 Definition. The set of *evaluation contexts* is given by the following inductive definition:

$$E ::= [\,] \mid \mathsf{case}\ E\ \mathsf{of}\ (a_1.p_1 \| a_2.p_2) \mid \pi_1 E \mid \pi_2 E \mid \mathsf{dest}\ E\ \mathsf{as}\ (x.a)\ \mathsf{in}\ p \mid$$
$$Eq \mid (\lambda a.q)E \mid Et \mid \iota_1 E \mid \iota_2 E \mid (E, p) \mid (V, E) \mid (t, E) \mid \#E$$

The set of evaluation contexts is larger than the set of pure evaluation contexts, because it includes #. As before, $E[p]$ denotes the proof term obtained from E by replacing its placeholder $[\,]$ with the proof term p.

2.6 Definition. The reduction relation on proof terms "$\to$" is defined by the following rewrite rules:

$$\begin{array}{rcl}
(\lambda a.p)V \to p\{V/a\} & \quad & \mathsf{case}\ \iota_i V\ \mathsf{of}\ (a_1.p_1 \| a_2.p_2) \to p_i\{V/a_i\} \\
(\lambda x.p)t \to p\{t/x\} & & \mathsf{dest}\ (t, V)\ \mathsf{as}\ (x.a)\ \mathsf{in}\ p \to p\{t/x\}\{V/a\} \\
\pi_i(V_1, V_2) \to V_i & & \#P[\mathcal{S}k.p] \to \#p\,\{(\lambda a.\#P[a])\,/k\} \\
\#V \to V & & E[p] \to E[p'] \text{ when } p \to p'
\end{array}$$

The last rule is known as the "congruent closure" of the preceding rules. The rule for $\mathcal{S}$ applies only when the evaluation context P is pure. The reduction strategy determined by the rules is standard call-by-value reduction. [40]

2.7 Example. The following are the proof terms corresponding to the derivation trees for MP_T and DNS_T from page 3.

$$\lambda e.\lambda a.\#e(a(\lambda b.\mathcal{S}k.b))$$

$$\lambda a.\lambda b.\#b(\lambda x.\mathcal{S}k.axk)$$

Remark that the proof term for MP_T does not make use of the continuation variable k, but only uses the $\mathcal{S}$ operator to pass the value b, once it has been found in the course of the computation, back to the control delimiter #.

3. Relationship to MQC and CQC

To connect provability in MQC$^+$with provability in MQC and CQC, we use the following double-negation translation.

3.1 Definition. The *superscript* translation A^T of a formula A with respect to a Σ-formula T is defined via the *subscript* translation A_T, which is in turn defined by recursion on the structure of A:

$$A^T := (A_T \Rightarrow T) \Rightarrow T$$

$$\begin{aligned} A_T &:= A && \text{if } A \text{ is atomic} \\ (A \Box B)_T &:= A_T \Box B_T && \text{for } \Box = \vee, \wedge \\ (A \Rightarrow B)_T &:= A_T \Rightarrow B^T \\ (\exists A)_T &:= \exists A_T \\ (\forall A)_T &:= \forall A^T \end{aligned}$$

We write Γ_T for the translation $(-)_T$ applied to each formula of the context Γ individually.

This translation is the standard call-by-value CPS translation of types [40], and is similar to the Kuroda translation [44], the difference being that we add a double negation, not only after $\forall$, but also after $\Rightarrow$. Interestingly, when interpreting, using DNS, the negative translation of the Axiom of Countable Choice AC_0, a transformation from the Kuroda translation of AC_0 into our form, with $\neg\neg$ after $\Rightarrow$, appears to be needed [27, p. 200]. Also, Avigad has remarked in [4] that the Kuroda translation makes essential use of the $\perp_E$ rule.

3.2 Remark. When A is a Σ-formula, we have that $A_T = A$.

We will denote derivability in MQC$^+$ by "$\vdash^+$", derivability in MQC by "$\vdash^m$", and the one in CQC by "$\vdash^c$". When we say CQC, we have in mind a standard natural deduction calculus, but where $\perp$ is replaced by a distinguished formula T – which one, will be clear from context – and correspondingly, the $\perp_E$ rule says that $T \Rightarrow A$, and the $\neg\neg_E$ rule is $(A \Rightarrow T) \Rightarrow T \vdash^c A$. The following theorem is not surprising, since, after all, our system is a subsystem of classical logic, but we give it for the sake of completeness, since this version of Kuroda's translation does not use the $\perp_E$ rule in the target system.

3.3 Theorem (Equiconsistency with MQC)**.** *Given a derivation of* $\Gamma \vdash^+ A$*, which uses* $\mathcal{S}$ *and* # *for the* Σ*-formula* T*, we can build a derivation of* $\Gamma_T \vdash^m A^T$.

Proof. By induction on the derivation, using the proof terms listed below. A line above a sub-term marks the place where the induction hypothesis is applied.

$$
\begin{aligned}
\overline{a} &= \lambda k.ka \\
\overline{\lambda a.p} &= \lambda k.k\,(\lambda a.\lambda k'.\overline{p}\,(\lambda b.k'b)) \\
\overline{pq} &= \lambda k.\overline{p}\,(\lambda f.\overline{q}\,(\lambda a.fa\,(\lambda b.kb))) \\
\overline{(p,q)} &= \lambda k.\overline{p}\,(\lambda a.\overline{q}\,(\lambda b.k\,(a,b))) \\
\overline{\pi_1 p} &= \lambda k.\overline{p}\,(\lambda c.k\,(\pi_1 c)) \\
\overline{\iota_1 p} &= \lambda k.\overline{p}\,(\lambda a.k\,(\iota_1 a)) \\
\overline{\mathsf{case}\ p\ \mathsf{of}\ (a_1.q_1 \| a_2.q_2)} &= \lambda k.\overline{p}\,(\lambda c.\ \mathsf{case}\ c\ \mathsf{of}\ (a_1.\overline{q_1}k \| a_2.\overline{q_2}k)) \\
\overline{\lambda x.p} &= \lambda k.k\,(\lambda x.\lambda k'.\overline{p}\,(\lambda b.k'b)) \\
\overline{pt} &= \lambda k.\overline{p}\,(\lambda f.ftk) \\
\overline{(t,p)} &= \lambda k.\overline{p}\,(\lambda a.k(t,a)) \\
\overline{\mathsf{dest}\ p\ \mathsf{as}\ (x.a)\ \mathsf{in}\ q} &= \lambda k.\overline{p}\,(\lambda c.\ \mathsf{dest}\ c\ \mathsf{as}\ (x.a)\ \mathsf{in}\ \overline{q}k) \\
\overline{\#ap} &= \lambda k.k\,(\overline{p}(\lambda a.a)) \\
\overline{\mathcal{S}l.p} &= \lambda k.\,(\overline{p}(\lambda a.a))\,\{\lambda a.\lambda k'.k'\,(ka)/\,l\}
\end{aligned}
$$

□

In order to relate MQC$^+$-provability of certain forms of formulae to their provability in MQC and CQC, we need the following version of the DNS schema, which is extended with a clause handling implication, something that is not needed when one has the $\bot_E$ rule. We denote by $\neg_T A$ the formula $A \Rightarrow T$; when it is clear from the context, we omit the subscript T from $\neg_T$.

3.4 Definition. The *Double Negation Shift for T* (DNS$_T$) is the following generalisation of the minimal-predicate-logic version of the usual DNS schema, extended with a clause handling implication:

$$\forall x.\neg_T\neg_T A(x) \Rightarrow \neg_T\neg_T\,(\forall x.A(x)) \qquad (\mathrm{DNS}_T^{\forall})$$
$$(A \Rightarrow \neg_T\neg_T B) \Rightarrow \neg_T\neg_T\,(A \Rightarrow B) \qquad (\mathrm{DNS}_T^{\Rightarrow})$$

The following proposition is given for IQC as Exercise 2.3.3 of [44], we give the proof here to emphasise the role of DNS$_T^{\Rightarrow}$ when $\bot_E$ is not present.

3.5 Proposition. *$DNS_T \vdash^m \neg_T\neg_T A \Leftrightarrow A^T$.*

Proof. Induction on the complexity of A. When A is atomic, $A^T = \neg\neg A$.

($\wedge$) Both directions are via the proof term

$$
\begin{aligned}
&\lambda c.\lambda k.\mathrm{IH}_A\,(\lambda k'.c\,(\lambda d.k'\,(\pi_1 d))) \\
&\qquad (\lambda a.\mathrm{IH}_B\,(\lambda k'.c\,(\lambda d.k'\,(\pi_2 d)))\,(\lambda b.k\,(a,b)))\,.
\end{aligned}
$$

($\vee$) Both directions are via the proof term

$$\lambda a.\lambda k.a\,(\lambda c.\\ \mathsf{case}\ c\ \mathsf{of}\ (a_1.\mathrm{IH}_A\,(\lambda l.la_1)\,(\lambda b.k\,(\iota_1 b))\,\|a_2.\mathrm{IH}_B\,(\lambda l.la_2)\,(\lambda b.k\,(\iota_2 b))))$$

($\exists$) Analogous to case ($\vee$).

($\Rightarrow$) In this case it is crucial to use $\mathrm{DNS}_T^{\Rightarrow}$, since in minimal logic we do not have $\bot_E$. The direction left-to-right is via the proof term

$$\lambda c.\lambda k.\mathrm{IH}_A^{\leftarrow}\,(\lambda k'.\mathrm{DNS}_T^{\Rightarrow}\,(\lambda a.\lambda k''.k'a)\,k)\\ (\lambda a.\mathrm{IH}_B^{\rightarrow}\,(\lambda k'.c\,(\lambda f.k'\,(fa)))\,(\lambda b.k\,(\lambda a'.\lambda k'.k'b)))\,.$$

The direction right-to-left is via the proof term

$$\lambda c.\lambda k.\mathrm{IH}_A^{\rightarrow}\,(\lambda k'.\mathrm{DNS}_T^{\Rightarrow}\,(\lambda a.\lambda k''.k'a)\,k)\\ (\lambda a.\mathrm{IH}_B^{\leftarrow}\,(\lambda k'.c\,(\lambda f.fak'))\,(\lambda b.k\,(\lambda a'.b)))\,.$$

The arrows in the superscript of “IH” determine the direction in which the induction hypotheses are used.

($\forall$) We have:

$$(\forall x A(x))^T = \neg\neg(\forall x A^T(x)) \overset{\mathrm{IH}}{\leftrightarrow} \neg\neg(\forall x \neg\neg A(x))\\ \overset{\mathrm{DNS}_T^{\forall}}{\leftrightarrow} \neg\neg\neg\neg\forall x A(x) \leftrightarrow \neg\neg\forall x A(x)$$

□

3.6 Lemma. $\Gamma \vdash^c A$ *if and only if* $\Gamma_T \vdash^m A^T$.

Proof. The direction right-to-left follows from the previous proposition, because DNS is a classical theorem. The other direction is by induction on the derivation of $\Gamma \vdash^c A$. Actually, we can use the translation table of the proof of Theorem 3.3 to treat all cases, except for the $\neg\neg_E$ rule which was not covered by the translation. We remark that there is no need to translate the $\bot_E$ rule, since it comes for free in classical logic – it is derivable from the $\neg\neg_E$ rule.

To show that $\Gamma_T \vdash^m A^T$ follows from $\Gamma_T \vdash^m (\neg\neg A)^T$, we use the fact that $\vdash^m \neg\neg(T_T) \leftrightarrow T$:

$$(\neg\neg A)^T = ((A \Rightarrow T) \Rightarrow T)^T = \neg\neg((A_T \Rightarrow \neg\neg T) \Rightarrow \neg\neg T)\\ \Leftrightarrow \neg\neg((A_T \Rightarrow T) \Rightarrow T) = \neg\neg\neg\neg A_T \Leftrightarrow \neg\neg A_T = A^T.$$

□

We proved the following relationships for the provability of an arbitrary formula A in MQC$^+$, MQC, and CQC:

$$\begin{array}{ccccc} \vdash^+ A & \xrightarrow{3.3} & \vdash^m A^T & \xleftrightarrow{3.6} & \vdash^c A \\ & & \big\downarrow{\scriptstyle 3.5} & & \\ \vdash^+ \neg\neg A & \longleftarrow & \mathrm{DNS}_T \vdash^m \neg\neg A & & \end{array}$$

3.7 Corollary. *For any formula A, we have the following diagram:*

$$\begin{array}{ccccc} \vdash^+ \neg\neg A & \xrightarrow{3.3} & \vdash^m (\neg\neg A)^T & \xleftrightarrow{3.6} & \vdash^c A \\ \big\uparrow & & \big\downarrow{\scriptstyle 3.5} & & \\ DNS_T \vdash^m \neg\neg A & \longleftarrow & DNS_T \vdash^m \neg\neg\neg\neg A & & \end{array}$$

In particular, the statement $\vdash^+ \neg\neg A \longleftrightarrow \; \vdash^c A$ represents an extension of Glivenko's theorem [16, 45, 46] to predicate logic.

4. Properties

In this section we will prove that MQC$^+$ has the Normalisation, Disjunction, and Existence Properties, by proving properties of the reduction relation on proof terms.

4.1 Lemma (Annotation Weakening)**.** *If $\Gamma \vdash p : A$, then $\Gamma \vdash_T p : A$ for any T.*

Proof. A simple induction on the derivation. □

4.2 Lemma (Substitutions)**.** *The following hold:*

1. *If $\Gamma, a : A \vdash_\diamond p : B$ and $\Gamma \vdash_\diamond q : A$, then $\Gamma \vdash_\diamond p\{q/a\} : B$.*
2. *If $\Gamma \vdash_\diamond p : B(x)$, where x is fresh, and t is a closed term, then $\Gamma \vdash_\diamond p\{t/x\} : B(t)$.*

Proof. The proof is standard, by induction on the derivation (see for example [41]). The new rules $\mathcal{S}$ and # pose no problems, since we can use the identities $(\#p)\{q/a\} = \#(p\{q/a\})$ and $(\mathcal{S}k.p)\{q/a\} = \mathcal{S}k.(p\{q/a\})$ when k is fresh. □

4.3 Lemma (Decomposition)**.** *If $\Gamma \vdash_T P[\mathcal{S}k.p] : B$, then there is a formula A and derivations $\Gamma, k : A \Rightarrow T \vdash_T p : T$ and $\Gamma, a : A \vdash_T P[a] : B$.*

Proof. The proof is by induction on the derivation. We only need to consider the rules that can generate a pure evaluation context of the required form. Of the rules that we consider, for the intuitionistic rules, the proof is simply by using the induction hypothesis, as shown below for the $\wedge_I$ rule; and the only non-intuitionistic rule to consider is $\mathcal{S}$, because # does not generate a pure evaluation context.

- For $\wedge_I$, there are two cases to consider, depending on whether the pure evaluation context is $(P[\mathcal{S}k.p], q)$ or $(V, P[\mathcal{S}k.p])$, but the proofs are analogous. Let the last rule in the derivation be:

$$\frac{\Gamma \vdash_T P[\mathcal{S}k.p] : B_1 \qquad \Gamma \vdash_T q : B_2}{\Gamma \vdash_T (P[\mathcal{S}k.p], q) : B_1 \wedge B_2}$$

The induction hypothesis gives us a formula A_1 and two derivations, $\Gamma, k : A_1 \Rightarrow T \vdash_T p : T$ and $\Gamma, a : A_1 \vdash_T P[a] : B_1$, from which the goal follows by choosing $A := A_1$.

- For $\mathcal{S}$, the pure evaluation context must be the empty one, so the last used rule is:

$$\frac{\Gamma, k : B \Rightarrow T \vdash_T p : T}{\Gamma \vdash_T [\mathcal{S}k.p] : B}$$

If we set $A := B$, the goal follows from the premise of the rule above and, for $\Gamma, a : A \vdash_T [a] : A$, from the Ax rule.

□

4.4 Lemma (Annotation Strengthening)**.** $\Gamma \vdash_S V : T \longrightarrow \Gamma \vdash V : T$

Proof. The proof is by induction on the derivation and very simple. We only need to consider the intuitionistic rules that introduce a value and that prove a Σ-formula, that is, the rules Ax, $\wedge_I$, $\vee_I^1$, $\vee_I^2$, and $\exists_I$. $\mathcal{S}$ and # do not introduce a value. □

4.5 Theorem (Subject Reduction)**.** *If* $\Gamma \vdash_\diamond p : A$ *and* $p \to q$*, then* $\Gamma \vdash_\diamond q : A$.

Proof. The proof is by induction on the derivation and is standard (see for example [41]), by using Substitutions Lemma 4.2 and Decomposition Lemma 4.3. Below, we consider the new rules and, for illustration, one of the intuitionistic rules.

(#) We have $\Gamma \vdash_\diamond \#p$ and $\#p \to q$ for some q. We look at three possible cases, because there are three rules for reducing a term of form $\#p$. If $q \equiv \#q'$ and the reduction was by the congruence rule, we have $p \to q'$; now use IH and the # rule to finish the proof. If p is a value and $q \equiv p$, then $\Gamma \vdash_T q : T$; now use Strengthening Lemma 4.4 to conclude $\Gamma \vdash q : T$. The third case is when $p \equiv P[\mathcal{S}k.p']$ and $q \equiv \#p'\{(\lambda a.\#P[a])/k\}$ – then, the proof is by combining lemmas 4.2 and 4.3.

($\mathcal{S}$) This case is impossible, since there are no rules for reducing a term of form $\mathcal{S}k.p$ on its own, and the set of evaluation contexts does not include a clause for $\mathcal{S}k.[\ ]$.

($\wedge_E^1$) We have $\Gamma \vdash_\diamond p : A \wedge B$, $\Gamma \vdash_\diamond \pi_1 p : A$, and $\pi_1 p \to q$. If the reduction was by the congruence rule, then $q \equiv \pi_1 q'$ for some q', and we can use IH. Otherwise, $p \equiv (V_1, V_2)$ and $q \equiv V_1$, and $\Gamma \vdash_\diamond p : A \wedge B$ must have been proved by the $\wedge_I$ rule, which is enough.

□

While the last theorem shows that reducing a proof term does not change its logical specification, the next one shows that a proof term which is not in normal form does not get "stuck".

4.6 Theorem (Progress)**.** *If $\vdash_\diamond p : A$, p is not a value, and p is not of form $P[\mathcal{S}k.p']$, then p reduces in one step to some proof term r.*

Proof. By induction on the derivation. The cases Ax, $(\Rightarrow_I)$, and $(\forall_I)$ introduce a value, while the case $(\mathcal{S})$ introduces a $\mathcal{S}k.p$ term, so they are impossible.

$(\wedge_I)$ We have that $\vdash_\diamond (p, q) : A \wedge B$ and (p, q) is neither a value nor of form $P[\mathcal{S}k.p']$. From $(p, q) \neq V$, we have that at least one of p and q is not a value. From $(p, q) \neq P[\mathcal{S}k.p']$, we have that $p \neq P'[Sk.r]$, and $p \neq V'$ or $q \neq P'[\mathcal{S}k.r]$.

If p is not a value, since it is neither of form $P'[\mathcal{S}k.r]$, by IH, $p \to r$ for some r, hence $(p, q) \to (r, q)$.

If p is a value V', then q is not a value and it is not of form $P'[\mathcal{S}k.r]$, and, by IH, $q \to r$, therefore $(V', q) \to (V', r)$.

$(\wedge_E^1)$ We have that $\vdash_\diamond \pi_1 p : A$ and that $\pi_1 p$, hence p itself, is not of form $P[\mathcal{S}k.p']$. If p is a value, then it must be a pair (V_1, V_2), so $\pi_1(V_1, V_2) \to V_1$. If p is not a value, we can use IH to obtain $\pi_1 p \to \pi_1 r$ for some r.

$(\vee_I^1)$ From $\vdash_\diamond \iota_1 p : A \vee B$ and $\iota_1 p$ a non-value and not of form $P[\mathcal{S}k.p']$, we have that p is not a value and not of that form, so we use IH to obtain an r such that $p \to r$, hence $\iota_1 p \to \iota_1 r$.

$(\vee_E)$ We have $\vdash_\diamond \mathsf{case}\ p\ \mathsf{of}\ (a_1.p_1 \| a_2.p_2) : C$. If p is a value, then it is of form $\iota_i V$, therefore $\mathsf{case}\ \iota_i V\ \mathsf{of}\ (a_1.p_1 \| a_2.p_2) \to p_i\{V/a_i\}$. If p is of form $P[\mathcal{S}k.p']$, then so is $\mathsf{case}\ p\ \mathsf{of}\ (a_1.p_1 \| a_2.p_2)$. Otherwise, we use IH to obtain an r such that $\mathsf{case}\ p\ \mathsf{of}\ (a_1.p_1 \| a_2.p_2) \to \mathsf{case}\ r\ \mathsf{of}\ (a_1.p_1 \| a_2.p_2)$.

$(\Rightarrow_E)$ From $pq \neq P[\mathcal{S}k.r]$, we have that $p \neq P'[\mathcal{S}k.r]$, and $p \neq V'$ or $q \neq P'[\mathcal{S}k.r]$.

If p is not a value, since it is also not of form $P'[\mathcal{S}k.r]$, we can use IH.

If p is a value, it is of form $\lambda a.p'$. If q is a value V, then $pq \to p'\{V/a\}$. If q is not a value, it can not be of form $P'[\mathcal{S}k.r]$, because p is a value; then, we can use IH on q.

$(\forall_E)$ We have $\vdash_\diamond pt : A(t)$. If p is of form $P[\mathcal{S}k.p']$, then so is pt. If p is a value, then it is of form $\lambda x.r$, hence $(\lambda x.r)t \to r\{t/x\}$. Otherwise, by IH, $p \to r$ for some r, so $pt \to rt$.

$(\exists_I)$ From $\vdash_\diamond (t, p) : A(t)$ and (t, p) a non-value and not of form $P[\mathcal{S}k.p']$, we have that p is not a value and not of that form, so we use IH to obtain an r such that $(t, p) \to (t, r)$.

$(\exists_E)$ We have $\vdash_\diamond \mathsf{dest}\ p\ \mathsf{as}\ (x.a)\ \mathsf{in}\ q : C$. If p is a value, then it is of form (t, V), therefore $\mathsf{dest}\ (t, V)\ \mathsf{as}\ (x.a)\ \mathsf{in}\ q \to q\{t/x\}\{V/a\}$. If p is of form $P[\mathcal{S}k.p']$, then so is $\mathsf{dest}\ p\ \mathsf{as}\ (x.a)\ \mathsf{in}\ q$. Otherwise, we use IH to obtain an r such that $\mathsf{dest}\ p\ \mathsf{as}\ (x.a)\ \mathsf{in}\ q \to \mathsf{dest}\ r\ \mathsf{as}\ (x.a)\ \mathsf{in}\ q$.

$(\#)$ We have $\vdash_\diamond \#p : T$. If p is a value, then $\#p \to p$. If $p \equiv P[\mathcal{S}k.p']$, then $\#p \to \#p'\{\lambda a.\#P[a]/k\}$. If p is neither a value nor of form $P[\mathcal{S}k.p']$, by IH, $p \to p'$, so $\#p \to \#p'$.

□

4.7 Corollary (Normalisation)**.** *For every closed proof term p_0, such that $\emptyset \vdash^+ p_0 : A$, there is a finite reduction path $p_0 \to p_1 \to \ldots \to p_n$ ending with a value p_n.*

Proof. This is a consequence of Subject Reduction and Progress, because a derivation tree $\emptyset \vdash^+ p_0 : A$, with no annotations under the turnstile, can not reduce to the form $P[\mathcal{S}k.p]$. That the reduction path has finite length follows from Theorem 4 of [2, 3]. □

4.8 Remark. When proving Normalisation of a variant of λ-calculus, it is customary to distinguish *weak* (there exists a terminating reduction sequence) from *strong* normalisation (all reduction sequences terminate). There is no need to make that distinction in the present case, because the reduction system is of the type known as *weak head reduction*, which only permits one possible reduction sequence.

4.9 Corollary (Disjunction and Existence Properties)**.** *If $\emptyset \vdash^+ A \vee B$, then $\emptyset \vdash^+ A$ or $\emptyset \vdash^+ B$. If $\emptyset \vdash^+ \exists x A(x)$, then there exists a closed term t such that $\emptyset \vdash^+ A(t)$.*

Proof. Let $\emptyset \vdash^+ p : A \vee B$. By Normalisation and Subject Reduction, for some V, $p \to \cdots \to V$ and $\emptyset \vdash^+ V : A \vee B$. Since V is a value, V must be of form $\iota_1 V'$ or $\iota_2 V'$, therefore either $\emptyset \vdash^+ V' : A$ or $\emptyset \vdash^+ V' : B$. The case for "$\exists$" is analogous. □

5. Related and future work

5.1. Double-negation Shift

The first use of a schema equivalent to DNS appears to be in modal logic, by Barcan [6, 5, 15], who introduced what is today known as Barcan's formula,

$$\forall x \Box A(x) \to \Box \forall x A(x),$$

or, equivalently,

$$\Diamond \exists x A(x) \to \exists \Diamond A(x).$$

Veldman kindly pointed to us that DNS is also known as Kuroda's Conjecture [34]. In [30], Kripke showed that Kuroda's Conjecture and Markov's Principle are underivable in intuitionistic logic. (however, see also [28] for criticism of Kripke's argument)

In [29, Section 2.11], Kreisel used the principle

$$\neg \forall n A(n) \Rightarrow \exists n \neg A(n), \tag{GMP}$$

for $A(n)$ an arbitrary formula, to deal with implication while giving a translation of formulae of Analysis into functionals of finite type. In [39], Oliva calls this principle the Generalised Markov's Principle (GMP) and remarks that $\mathrm{HA}^\omega \vdash \mathrm{DNS} \leftrightarrow \neg\neg\mathrm{GMP}$. Kreisel does not give a justification of GMP in his paper.

The term "double negation shift" appears for the first time in [42] to denote the formula

$$\forall n \neg\neg A(n) \Rightarrow \neg\neg \forall n A(n). \tag{DNS}$$

There, Spector builds upon previous works of Gödel [17, 18, 19], namely he realises DNS by adding the schema of bar recursion to Gödel's system T. The name "bar recursion" comes from the Bar Principle of Brouwer which is used in justifying it. However, Spector attaches no particular interest to the DNS schema itself; he writes:

> The schema [DNS] is chosen not because we believe it is of intuitionistic significance, but to provide a formal system in which classical analysis is easily interpreted, and whose logical basis is intuitionistic. [42]

We treat DNS at the level of predicate logic, not of arithmetic, an important change in status that we plan to investigate in future.

5.2. Negative translation of Countable Choice

The Axiom of Countable Choice,

$$\forall x^0 \exists y^\rho A(x,y) \Rightarrow \exists f^{0\to\rho} \forall x^0 A(x, f(x)), \tag{AC$_0$}$$

is a formula schema of HA$^\omega$, Heyting Arithmetic in all finite types. The type 0 stands for the set of natural numbers $\mathbb{N}$, the type $1 = 0 \to 0$ stands for the functions $\mathbb{N} \to \mathbb{N}$, 2 stands for the functionals $(0 \to 0) \to 0$, and so on; ρ is a type variable.

Spector showed that Kuroda's [27, p.163] negative translation, $\neg\neg(\mathrm{AC_0}^*)$, of AC$_0$,

$$\forall x^0 \neg\neg\exists y^\rho A^*(x,y) \Rightarrow \exists f^{0\to\rho} \forall x^0 \neg\neg A^*(x, f(x)), \tag{AC$_0^\mathrm{N}$}$$

is provable from DNS and the intuitionistic AC$_0$. Since AC$_0$ is realisable in HA$^\omega$, and DNS is realisable by bar recursion, so is AC_0^N. His approach was extended to the Axiom of Dependent Choice (DC) by Luckhardt [36] and Howard [23]. In more recent years, Kohlenbach, Berger, and Oliva have given their own versions of bar recursion (see [7] for a comparison).

Since we treat DNS at the level of pure logic, without considering arithmetic axioms, we are only able to give an *open* proof term deriving the negative translation of AC$_0$,

$$\forall x^0 \neg_T\neg_T \exists y^\rho A_T(x,y) \Rightarrow \neg_T\neg_T \exists f^{0\to\rho} \forall x^0 \neg_T\neg_T A_T(x, f(x)). \tag{AC$_{0T}$}$$

Given a variable c to denote a proof of the intuitionistic AC$_0$, we can use a proof term similar to the one of DNS$_T$ for deriving the above schema:

$$\lambda a.\lambda k.\#k(c(\lambda x.\mathcal{S}k'.ax(\lambda d.k'(\nu d)))),$$

where ν is a proof term for $\exists y A_T(x,y) \Rightarrow \exists y A^T(x,y)$.

The proof term being open means that we can not immediately use it for computation. We would have to either develop a realisability interpretation for MQC$^+$, or add delimited control operators to an intuitionistic system with strong existential quantifiers, like Martin-Löf's type theory, which can derive AC$_0$.

5.3. Herbelin's calculus for Markov Principle

In [22], Herbelin presented $\mathrm{IQC}_{\mathrm{MP}}$, an intuitionistic predicate logic that can derive the pure predicate-logical version of Markov's Principle. Our MQC^+ has been developed starting from his calculus. There are two important differences between the two.

First, derivations of $\mathrm{IQC}_{\mathrm{MP}}$ are annotated by a *context* of Σ-formulae, not just one formula. This permits to have a derivation which uses multiple and different instances of Markov's Principle. Had we had context-annotations as well, it would have been possible to have the following characterisation of provability of Σ-formulae S:

$$\begin{tikzcd}
\vdash^+ S \arrow[r, "3.3"] & \vdash^i S^\perp \arrow[r, leftrightarrow, "3.6"] \arrow[d, equal, "\text{by def. of } (\cdot)^\perp"] & \vdash^c S \\
\mathrm{MP} \vdash^i S \arrow[u] & \vdash^i \neg\neg S \arrow[l] &
\end{tikzcd}$$

Proving that such a context-annotated version of MQC^+ satisfies the analogues of the properties proven in Section 4 remains future work.

Aside from that, the typing and the reduction rules for delimited control operators of $\mathrm{IQC}_{\mathrm{MP}}$ are a restriction of those for MQC^+. Consider the typing rules:

$$\frac{\Gamma \vdash_{\alpha:T,\Delta} p : T}{\Gamma \vdash_\Delta \mathsf{catch}_\alpha p : T}\ \textsc{Catch} \qquad\qquad \frac{\Gamma \vdash_\Delta p : T \quad (\alpha : T) \in \Delta}{\Gamma \vdash_\Delta \mathsf{throw}_\alpha p : A}\ \textsc{Throw}$$

While catch is just #, the proof term $\mathsf{throw}\ p$ is a particular case of $\mathcal{S}k.p$ that does not use the continuation variable k inside p, something already seen with the proof term deriving MP of Example 2.7.

5.4. Other studies of delimited control operators

Delimited control operators have been studied in the Theoretical Computer Science literature quite extensively in the past twenty years, since their appearance [35, 43, 12]. We mention some of the works that pertain to Logic.

The original typing system for shift/reset of Danvy and Filinski from [9] is a so-called "type-and-effect" system: implication is a quaternary not a binary connective, and is as such difficult to understand in traditional logical terms. A proof of Subject Reduction, Progress, and Normalisation of this system appears in [2, 3].

A typing system which is a specialisation of Danvy and Filinski's, but again has a ternary implication connective, appears in [38].

Ariola, Herbelin and Sabry [1] decompose shift/reset in their own calculus and prove, besides other things, the normalisation property for a typing system where reset is applied at atomic types.

There are a number of works connecting delimited control operators to sub-structural classical logic [26, 47, 37]. Our contribution differs in two respects: we identify delimited control operators as giving rise to an *a priori* constructive logic, rather than classical logic which is only constructive *a posteriori* for certain classes of formulae; and we are connecting delimited control operators with known extra-intuitionistic axioms, rather than analysing the sub-structural properties of the derivation system rules themselves.

5.5. The meaning of DNS in the presence of common axioms

In this paper we were dealing with a purely predicate-logical version of DNS. Further examination is necessary on how DNS interacts with common logical axioms, as Wim Veldman kindly warned us. For instance, DNS is false in some uncountable models: for example, it contradicts the continuity principle proposed by Brouwer.

Acknowledgements

I would like to thank my thesis supervisor Hugo Herbelin for commenting on an earlier draft of this paper, as well as for many inspiring discussions.

References

[1] Zena M. Ariola, Hugo Herbelin, and Amr Sabry. A type-theoretic foundation of delimited continuations. *Higher Order and Symbolic Computation*, 22(3):233–273, September 2009. online from 2007.

[2] Kenichi Asai and Yukiyoshi Kameyama. Polymorphic delimited continuations. In *APLAS*, pages 239–254, 2007.

[3] Kenichi Asai and Yukiyoshi Kameyama. Polymorphic delimited continuations. Technical report, Graduate school of SIE, University of Tsukuba, Japan, 2007.

[4] Jeremy Avigad. A variant of the double-negation translation. Technical report, Carnegie Mellon University, 2006. Technical Report CMU-PHIL 179.

[5] Ruth C. Barcan. The deduction theorem in a functional calculus of first order based on strict implication. *The Journal of Symbolic Logic*, 11(4):115–118, 1946.

[6] Ruth C. Barcan. A functional calculus of first order based on strict implication. *The Journal of Symbolic Logic*, 11(1):1–16, 1946.

[7] U. Berger and P. Oliva. Modified bar recursion and classical dependent choice. In M. Baaz, S.D. Friedman, and J. Kraijcek, editors, *Logic Colloquium '01, Proceedings of the Annual European Summer Meeting of the Association for Symbolic Logic, held in Vienna, Austria, August 6 - 11, 2001*, volume 20 of *Lecture Notes in Logic*, pages 89–107. Springer, 2005.

[8] Ulrich Berger. A computational interpretation of open induction. In F. Titsworth, editor, *Proceedings of the Ninetenth Annual IEEE Symposium on Logic in Computer Science*, pages 326–334. IEEE Computer Society, 2004.

[9] Olivier Danvy and Andrzej Filinski. A functional abstraction of typed contexts. Technical report, Computer Science Department, University of Copenhagen, 1989. DIKU Rapport 89/12.

[10] Olivier Danvy and Andrzej Filinski. Abstracting control. In *LISP and Functional Programming*, pages 151–160, 1990.

[11] Solomon Feferman, editor. *Collected works. Publications 1938–1974*, volume II. The Clarendon Press Oxford University Press, New York, 1990.

[12] Matthias Felleisen, Daniel P. Friedman, Eugene E. Kohlbecker, and Bruce F. Duba. Reasoning with continuations. In *LICS*, pages 131–141, 1986.

[13] Andrzej Filinski. Representing monads. In *Proceedings of the 21st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 446–457, 1994.

[14] Andrzej Filinski. *Controlling Effects*. PhD thesis, School of Computer Science, Carnegie Mellon University, 1996. Technical Report CMU-CS-96-119 (144pp.).

[15] Melvin Fitting. Barcan both ways, 1997.

[16] Valery Ivanovich Glivenko. Sur quelques points de la logique de M. Brouwer. In *Bulletins de la classe des sciences*, volume 15 of *5*, pages 183–188. Academie Royale de Belgique, 1929.

[17] Kurt Gödel. *In what sense is intuitionistic logic constructive*, volume III, pages 189–200. The Clarendon Press Oxford University Press, New York, 1941. early lecture on the Dialectica interpretation.

[18] Kurt Gödel. *On a hitherto unutilized extension of the finitary standpoint*, pages 241–251. Volume II of Feferman [11], 1958.

[19] Kurt Gödel. *On an extension of finitary mathematics which has not yet been used*, pages 271–280. Volume II of Feferman [11], 1972.

[20] Timothy Griffin. A formulae-as-types notion of control. In *POPL*, pages 47–58, 1990.

[21] Hugo Herbelin. personal communication, January 2010.

[22] Hugo Herbelin. An intuitionistic logic that proves Markov's principle. In *Proceedings, 25th Annual IEEE Symposium on Logic in Computer Science (LICS '10), Edinburgh, UK, 11-14 July 2010*, page N/A. IEEE Computer Society Press, 2010.

[23] W. A. Howard. Functional interpretation of bar induction by bar recursion. *Compositio Math.*, 20:107–124 (1968), 1968.

[24] Danko Ilik. *Constructive Completeness Proofs and Delimited Control*. PhD thesis, École Polytechnique, October 2010.

[25] Danko Ilik. Continuation-passing style models complete for intuitionistic logic. submitted, draft available from `http://arxiv.org/abs/1102.1061`, last accessed on 10-March-2011, 2011.

[26] Oleg Kiselyov and Chung chieh Shan. A substructural type system for delimited continuations. In Simona Ronchi Della Rocca, editor, *TLCA*, volume 4583 of *Lecture Notes in Computer Science*, pages 223–239. Springer, 2007.

[27] U. Kohlenbach. *Applied proof theory: proof interpretations and their use in mathematics*. Springer Monographs in Mathematics. Springer-Verlag, Berlin, 2008.

[28] G. Kreisel. Review: [Semantical analysis of intuitionistic logic I. by Saul A. Kripke]. *The Journal of Symbolic Logic*, 35(2):330–332, 1970.

[29] Georg Kreisel. Interpretation of analysis by means of constructive functionals of finite types. Studies in Logic and The Foundations of Mathematics, pages 101–127. North-Holland Publishing Company Amsterdam, 1957.

[30] Saul A. Kripke. Semantical analysis of intuitionistic logic i. In *Formal Systems and Recursive Functions*, pages 92–130. North Holland, 1965.

[31] Jean-Louis Krivine. Typed lambda-calculus in classical zermelo-frænkel set theory. *Arch. Math. Log.*, 40(3):189–205, 2001.

[32] Jean-Louis Krivine. Realizability algebras: a program to well order R. *CoRR*, abs/1005.2395, 2010.

[33] Jean-Louis Krivine. Realizability algebras II : new models of ZF + DC. *CoRR*, abs/1007.0825, 2010.

[34] Sigekatu Kuroda. Intuitionistische untersuchungen der formalistischen logik. *Nagoya Mathematical Journal*, (2):35–47, 1951.

[35] Peter J. Landin. A generalization of jumps and labels. Technical report, UNIVAC Systems Programming Research, August 1965.

[36] Horst Luckhardt. *Extensional Gödel functional interpretation. A consistency proof of classical analysis*. Lecture Notes in Mathematics, Vol. 306. Springer-Verlag, Berlin, 1973.

[37] Guillaume Munch-Maccagnoni. From delimited CPS to polarisation. manuscript, 2011.

[38] Chetan R. Murthy. Control operators, hierarchies, and pseudo-classical type systems: A-translation at work. In *Proceedings of the ACM SIGPLAN Workshop on Continuations CW92*, pages 49–72. Stanford University, 1992. Technical Report STAN-CS-92-1426.

[39] Paulo Oliva. Understanding and using Spector's bar recursive interpretation of classical analysis. In Arnold Beckmann, Ulrich Berger, Benedikt Löwe, and John V. Tucker, editors, *CiE*, volume 3988 of *Lecture Notes in Computer Science*, pages 423–434. Springer, 2006.

[40] G. D. Plotkin. Call-by-name, call-by-value and the [lambda]-calculus. *Theoretical Computer Science*, 1(2):125–159, 1975.

[41] Morten Heine Sørensen and Pawel Urzyczyn. *Lectures on the Curry-Howard Isomorphism, Volume 149 (Studies in Logic and the Foundations of Mathematics)*. Elsevier Science Inc., New York, NY, USA, 2006.

[42] Clifford Spector. Provably recursive functionals of analysis: a consistency proof of analysis by an extension of principles formulated in current intuitionistic mathematics. In *Proc. Sympos. Pure Math., Vol. V*, pages 1–27. American Mathematical Society, Providence, R.I., 1962.

[43] Hayo Thielecke. An introduction to landin's "a generalization of jumps and labels". *Higher Order Symbol. Comput.*, 11:117–123, September 1998.

[44] A. S. Troelstra and D. van Dalen. *Constructivism in mathematics. Vol. I*, volume 121 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, 1988. An introduction.

[45] Mark van Atten. The development of intuitionistic logic. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Summer 2009 edition, 2009.

[46] Wikipedia. Glivenko's theorem — Wikipedia, the free encyclopedia, 2009. [Online; accessed 1-July-2010].

[47] Noam Zeilberger. Polarity and the logic of delimited continuations. In *Proceedings, 25th Annual IEEE Symposium on Logic in Computer Science (LICS '10), Edinburgh, UK, 11-14 July 2010*, page N/A. IEEE Computer Society Press, 2010.