A hyperdoctrinal reconstruction of conditional calculus

Anonymous Author(s)

ABSTRACT

KEYWORDS

conditional calculus, categorical logic

ACM Reference Format:
Anonymous Author(s). 2023. A hyperdoctrinal reconstruction of conditional calculus. In *Proceedings of 40th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS) (LICS '25).* ACM, New York, NY, USA, 3 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

In this paper, we show that Bruggink et al.'s *conditional calculus* [1] is an inductively generated, subhyperdoctrine of a very simple hyperdoctrine, obtained using basic, well-known results.

Defining this hyperdoctrine is a mere one-liner; the paper is only a bit longer to explain how to interpret the various constructions of conditional calculus in this setting.

Plan. In §2, we briefly recall the definition of hyperdoctrines and construct the relevant one for us, which we call the *subhom* hyperdoctrine. In §3, we review the constructions of conditional calculus and interpret them in the subhom hyperdoctrine. We then explain the sense in which some of the peculiarities of conditional calculus may be viewed as arising from good properties of the subhom hyperdoctrine, namely the Beck-Chevalley condition (§4) and Frobenius reciprocity (§5). Finally, we conclude in §6. Some proofs of merely technical interest are deferred to Appendix A.

2 THE SUBHOM HYPERDOCTRINE

In this section, we briefly recall the definition of hyperdoctrines and construct the subhom hyperdoctrine.

Hyperdoctrines were introduced by Lawvere as a categorical approach to first-order logic. There are a few variants, but for us:

Definition 2.1. A *hyperdoctrine* is a functor from some base category to the category of Heyting algebras, such that the image of any morphism is both a left and right adjoint.

Our semantical hyperdoctrine is so simple that we can introduce it here, based on a well-known example hyperdoctrine and two easy results.

Example 2.2. The well-known example hyperdoctrine that we need is the contravariant powerset functor:

$$\mathcal{P}\colon \mathbf{Set}^{op} \to \mathbf{Set}$$
$$A \mapsto \mathcal{P}(A)$$
$$(f\colon A \to B) \mapsto (f^{-1}\colon \mathcal{P}(B) \to \mathcal{P}(A)).$$

The hyperdoctrine structure will be detailed below.

The basic results we need are as follows.

LEMMA 2.3. *Hyperdoctrines are closed under precomposition by arbitrary functors.*

LEMMA 2.4. *Hyperdoctrines are closed under products.*

PROOF. □

The former lemma is nearly trivial; the latter is only slightly more involved.

Our definition now goes in two steps. We first define a hyperdoctrine $\mathcal{P}_X$ for each object of a given category.

Definition 2.5. For any objet X in a locally small category $\mathbf{C}$, let $\mathcal{P}_X$ denote the composite

$$\mathbf{C} \xrightarrow{\mathbf{y}_X{}^{op}} \mathbf{Set}^{op} \xrightarrow{\mathcal{P}} \mathbf{Set},$$

where $\mathbf{y}_X(C) = \mathbf{C}(C, X)$ denotes the hom-functor at X.

Remark 2.6. We here view $(-)^{op}$ as a covariant endofunctor on **CAT**, the very large category of locally small categories, which in particular maps any functor $F\colon \mathcal{C} \to \mathcal{D}$ to the functor $F^{op}\colon \mathcal{C}^{op} \to \mathcal{D}^{op}$ defined exactly as F, though with different types.

Each $\mathcal{P}_X$ is a hyperdoctrine by example 2.2 and lemma 2.3.

We may now define our semantical hyperdoctrine using lemma 2.4.

Definition 2.7. For any locally small category $\mathbf{C}$, the *subhom* hyperdoctrine is

$$\mathcal{P}_{\mathbf{C}} = \prod_{X \in \mathbf{C}} \mathcal{P}_X.$$

Remark 2.8. Strictly speaking, this is a hyperdoctrine on $\mathbf{C}^{op}$.

Let us conclude this section by briefly unfolding some of the hyperdoctrine structure, without proof as this is all easy computation.

PROPOSITION 2.9. *Conjunction, disjunction, and negation in each $\mathcal{P}_{\mathbf{C}}(A)$ are given by pointwise intersection, union, and complementation, respectively.*

Furthermore, each $\mathcal{P}_{\mathbf{C}}(A)$ in fact has infinite conjunctions and disjunctions, given pointwise.

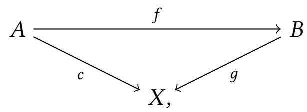
PROPOSITION 2.10. *For any morphism $f\colon A \to B$, $\varphi \in \mathcal{P}_X(A)$, and $\psi \in \mathcal{P}_X(B)$,*

- *the reindexing $f^*(\varphi)$ is $\mathbf{C}(f, X)^{-1}(\varphi)$, i.e., the set of morphisms $d\colon B \to X$ such that $d \circ f$ is in φ;*

LICS '25, when, where

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

- $\forall_f \psi$ *is the set of morphisms* $c : A \to X$ *whose extensions* g *along* f*, as in*



(1)

are all in $\mathcal{P}_X(B)$*; and*
- $\exists_f \psi$ *is the set of morphisms* $c : A \to X$ *which admit some extension* g *along* f*, as in (1), such that* $g \in \mathcal{P}_X(B)$.

By construction, we have:

Proposition 2.11. *The hyperdoctrine* $\mathcal{P}_{\mathbf{C}}$ *is* classical, *i.e., each fibre* $\mathcal{P}_{\mathbf{C}}(A)$ *is in fact a boolean algebra. Furthermore, for any* $f : A \to B$ *and* $\varphi \in \mathcal{P}_{\mathbf{C}}(B)$*, we have*

$$\neg\forall_f \varphi = \exists_f \neg\varphi$$
$$\neg\exists_f \varphi = \forall_f \neg\varphi.$$

3 CONDITIONAL CALCULUS IN THE SUBHOM HYPERDOCTRINE

In this section, we prove:

Theorem 3.1. *Conditional calculus on any category* $\mathbf{C}$ *is the smallest subhyperdoctrine of* $\mathcal{P}_{\mathbf{C}}$ *containing the true and false predicates.*

This is not a very precise statement, so let us briefly recall the basics of conditional calculus, and then give a more concrete statement (theorem 3.10).

Definition 3.2. Conditions are defined inductively by the following inference rule,

$$\frac{\ldots \quad f_i : A \to B_i \quad B_i \vdash \varphi_i \quad \ldots \quad (i \in I)}{A \vdash \varepsilon_{i \in I}(f_i, \varphi_i)}$$

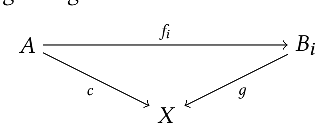
where
- A and B_i range over objects of $\mathbf{C}$,
- f_i ranges over morphisms,
- I denotes any set, and
- ε ranges over *quantifiers*, i.e., elements of $\{\forall, \exists\}$.

We let $\mathbf{Cond}_{\mathbf{C}}(A)$ denote the set of conditions φ over A, i.e., such that $A \vdash \varphi$.

Notation 3.3. We often omit the base objects of conditions, writing φ instead of $A \vdash \varphi$, when it is clear from context.

From this definition, it is not at all clear that a condition $A \vdash \varphi$ should denote a subset of $\prod_X \mathbf{C}(A, X)$, as claimed by the theorem. Indeed, this is only visible in the satisfaction predicate $c \vDash \varphi$, which relates a condition $A \vdash \varphi$ to morphisms from A:

Definition 3.4. Satisfaction $c \vDash \varphi$ is defined by induction on φ, as follows:
- if $\varphi = \forall_{i \in I}(f_i, \varphi_i)$, $c \vDash \varphi$ means that, for all $i \in I$ and g making the following triangle commute



(2)

we have $g \vDash \varphi_i$;
- if $\varphi = \exists_{i \in I}(f_i, \varphi_i)$, $c \vDash \varphi$ means that there exists $i \in I$ and g making (2) commute, such that $g \vDash \varphi_i$.

Satisfaction may thus be viewed as associating a "semantics" to each condition $A \vdash \varphi$ in $\mathcal{P}_{\mathbf{C}}$:

Definition 3.5. We define

$$[\![-]\!]_A : \mathbf{Cond}_{\mathbf{C}}(A) \to \mathcal{P}_{\mathbf{C}}(A) = \prod_X \mathcal{P}(\mathbf{C}(C, X))$$

by $$[\![\varphi]\!]_A(X) = \{c : A \to X \mid c \vDash \varphi\}.$$

Our main result will state that these assignments assemble into a "universal" hyperdoctrine morphism $\mathbf{Cond}_{\mathbf{C}} \to \mathcal{P}_{\mathbf{C}}$. In order to state it, we must first present the hyperdoctrine structure on $\mathbf{Cond}_{\mathbf{C}}$, which is given by the operations given on conditions by Bruggink et al.

The base case for the induction that defines conditions is of course when $I = \emptyset$. This gives the interpretation of true and false:

Notation 3.6. We let $\top_A := (A \vdash \forall_{i \in \emptyset} \emptyset)$ and $\bot_A := (A \vdash \exists_{i \in \emptyset} \emptyset)$, for any object A.

Conjunction and disjunction are straightforward:

Definition 3.7. For any conditions $A \vdash \varphi_1, \varphi_2$, let

$$\varphi_1 \wedge \varphi_2 := \forall_{i \in \{1,2\}}(id_A, \varphi_i)$$
$$\varphi_1 \vee \varphi_2 := \exists_{i \in \{1,2\}}(id_A, \varphi_i)$$

Negation is defined by induction:

Definition 3.8. We define $\neg : \prod_A \mathbf{Cond}_{\mathbf{C}}(A) \to \mathbf{Cond}_{\mathbf{C}}(A)$ by induction, as follows:

$$\neg(\forall_i(f_i, \varphi_i)) := \exists_i(f_i, \neg\varphi_i)$$
$$\neg(\exists_i(f_i, \varphi_i)) := \forall_i(f_i, \neg\varphi_i)$$

It remains to extend the assignment $A \mapsto \mathbf{Cond}_{\mathbf{C}}(A)$ to morphisms, and prove that the obtained maps have left and right adjoints. This is, in fact, the trickiest part. Assuming for now that conditions, or, rather, conditions, up to satisfaction equivalence, assemble into a hyperdoctrine, uniqueness of adjoints tells us that it suffices to give one of the adjoint functors to get all three. E.g., if we give the right adjoint $\forall_f$, then the whole chain $\exists_f \dashv f^* \dashv \forall_f$ follows. It turns out that both $\exists_f$ and $\forall_f$ are much easier to define than f^*:

Definition 3.9. For all $\varepsilon \in \{\forall, \exists\}$, $f : A \to B$, and $B \vdash \varphi$, let

$$\varepsilon_f.\varphi := \varepsilon_{\star \in 1}(f, \varphi).$$

The definition is so tautological that it might be hard to parse: $\varepsilon_f.\varphi$ is the condition consisting of the same quantifier ε, applied to the singleton family (f, φ).

The catch is that conditions (modulo satisfaction equivalence) may not form a hyperdoctrine, because reindexing, although definable semantically, may not be expressible in the rigid syntax of conditions.

A first, cheap result that we can readily prove is:

Theorem 3.10. *Let* $\mathbf{C}$ *be any locally small category. The maps* $[\![-]\!]_A$ *commute with fibrewise logical operations* $\neg$, $\wedge$, $\vee$*, and with all* $\exists_f$ *and* $\forall_f$.

We first prove:

Lemma 3.11. *For all I and family of (f_i, φ_i), we have*

$$[\![\forall_{i\in I}(f_i,\varphi_i)]\!]_A = \bigwedge_{i\in I} \forall_{f_i} [\![(id,\varphi_i)]\!]_A$$
$$[\![\exists_{i\in I}(f_i,\varphi_i)]\!]_A = \bigvee_{i\in I} \exists_{f_i} [\![(id,\varphi_i)]\!]_A.$$

Proof. By a straightforward induction and propositions 2.9 to 2.10. □

Proof of theorem 3.10. Conjuction and disjunction work similarly, so let us only treat, say, disjunction. The semantics of $\varphi \vee \psi$ is by definition of $[\![-]\!]_A$ the set of morphisms $c\colon A \to X$ to some X which satisfy either φ or ψ, which is precisely $[\![\varphi]\!]_A \vee [\![\psi]\!]_A$, as computed in $\mathcal{P}_C(A)$.

The statement about $\forall_f$ and $\exists_f$ follows directly from lemma 3.11.

For the same reason, the constants $\top_A$ and $\bot_A$ are mapped to the constantly full and empty families, respectively.

For negation, we need to proceed by induction. The base case is given by $\top_A$ and $\bot_A$. For the induction step, consider, e.g., any $\varphi = \forall_i(f_i, \varphi_i)$. We have:

$$\begin{aligned}
[\![\neg\varphi]\!]_A &= [\![\exists_i(f_i,\neg\varphi_i)]\!]_A \\
&= \bigvee_i \exists_{f_i} [\![\neg\varphi_i]\!]_A && \text{(by lemma 3.11)} \\
&= \bigvee_i \exists_{f_i} \neg[\![\varphi_i]\!]_A && \text{(by induction hypothesis)} \\
&= \neg \bigwedge_i \forall_{f_i} [\![\varphi_i]\!]_A && \text{(by proposition 2.11)} \\
&= \neg[\![\forall_i(f_i,\varphi_i)]\!]_A && \text{(by lemma 3.11 again)} \\
&= \neg[\![\varphi]\!]_A.
\end{aligned}$$
□

Corollary 3.12. *If conditions form a hyperdoctrine, with entailment given by inclusion of the satisfaction predicates, then the maps $[\![-]\!]_A$ assemble into a hyperdoctrine morphism, which is furthermore unique.*

At this point, we have proved that the semantics of most operations on conditions is given by the hyperdoctrine structure on $\mathcal{P}_C$. There is one operation that we have not treated, though, which Bruggink et al. call the shift.

4 ON THE BECK-CHEVALLEY CONDITION

5 ON FROBENIUS RECIPROCITY

6 CONCLUSION AND PERSPECTIVES

REFERENCES

[1] H. J. Sander Bruggink, Raphaël Cauderlier, Mathias Hülsbusch, and Barbara König. [n. d.]. Conditional Reactive Systems. In *DROPS-IDN/v2/Document/10.4230/LIPIcs.FSTTCS.2011.191* (2011). Schloss Dagstuhl – Leibniz-Zentrum für Informatik. https://doi.org/10.4230/LIPIcs.FSTTCS.2011.191

A TECHNICAL PROOFS