Open Higher-Order Logic

Ugo Dal Lago, Francesco Gavazzo, Alexis Ghyselen

Open Higher-Order Logic

Ugo Dal Lago
Department of Computer Science and Engineering, University of Bologna, Italy

Francesco Gavazzo
Department of Computer Science, University of Pisa, Italy

Alexis Ghyselen
Department of Computer Science and Engineering, University of Bologna, Italy

Abstract

We introduce a variation on Barthe et al.'s higher-order logic in which formulas are interpreted as predicates over *open* rather than *closed* objects. This way, concepts which have an intrinsically functional nature, like continuity, differentiability, or monotonicity, can be expressed and reasoned about in a very natural way, following the structure of the underlying program. We give open higher-order logic in distinct flavors, and in particular in its *relational* and *local* versions, the latter being tailored for situations in which properties hold only in *part* of the underlying function's domain of definition.

2012 ACM Subject Classification Theory of computation → Higher order logic; Theory of computation → Logic and verification

Keywords and phrases Formal Verification, Relational Logic, First-Order Properties

Digital Object Identifier 10.4230/LIPIcs.CSL.2023.17

Related Version *Full Version*: `https://arxiv.org/abs/2211.06671`

Funding This work was funded by the ERC CoG "DIAPASoN" GA 818616.

1 Introduction

Reasoning about functional programs poses a whole series of challenges due, in particular, to the presence of higher-order constructions. A class of methodologies particularly suitable for compositional reasoning on such programs is that of *type systems*, which can be seen as lightweight formal methods for the verification of relatively simple properties, mainly safety ones. In the last thirty years, it has become apparent that properties beyond mere safety are actually amenable to be verified through types, e.g, termination [6, 9], complexity bounds [13, 7], and noninterference [18]. In all these cases, types serve as expressions meant to abstractly describe the input and output interfaces to functions. Various levels of abstractions in turn give rise to distinct levels of expressiveness, and to type inference problems of varying degrees of difficulty.

A somehow different, although related, approach is the one of program logics, in which types are replaced or complemented by formulas written in a logical language. This approach, pioneered by Floyd and Hoare in the context of first-order imperative languages [10, 12], is nowadays common also in the realm of higher-order programming languages [4], where it stands out for its expressive power. Indeed, relative completeness results [5], which hold in many contexts within the realm program logics, are rarer in type systems.

A simple, yet powerful, form of program logic among those capable of dealing with higher-order programs is *higher-order logic*, as formulated by Aguirre et al. in a series of recent works [2, 11, 1, 19], most of which focusing on relational reasoning about higher-order programs. One common trait between the many introduced dialects of higher-order logic is the fact that object programs are taken to be terms of a simply-typed λ-calculus, while

properties are expressed in predicate logic. In the words of Aguirre et al. [2], higher-order logic *can be understood as an attempt to internalize the versatility of relational logical relations in a syntactic framework*. Indeed, rules of higher-order logic are built in such a way that syntax-directed reasoning about programs can be done following the rules provided by logical relations [17, 21]: any basic property at ground types is generalized at higher types by stipulating, conceptually, that related functions should map related inputs to related outputs.

As observed in several works [3, 15, 22], however, ordinary logical relations cannot easily deal with properties which are higher-order extensions of *first-order*, rather than ground, properties. This includes continuity, differentiability, or monotonicity properties. Indeed, what would be the base case in the (recursive) definition of the corresponding logical relation? Properties like continuity hold for functions and are meaningless if formulated for, say, real numbers. As we show in Section 2 below, expectedly, similar difficulties show up when dealing with the same kind of properties in higher-order logic. Reasoning is indeed possible, but becomes cumbersome and difficult to be carried out compositionally, i.e. in a syntax-directed way. Going back to logical relations, however, there is a way out, which consists in switching to an *open* version of logical relations in which the base case is indeed the one of first-order types. This way, one is allowed to start from, say, continuity and generalize it to higher-order types naturally.

In this paper, we show that open logical relations can themselves be given a formal counterpart in the realm of higher-order logic. We do so by introducing a variation on Aguirre et al.'s higher-order logic, called *open higher-order logic*. The salient feature of this new system, compared to those from the literature, is the fact that the mathematical objects that proofs implicitly deal with, namely higher-order functions, are assumed to in turn depend on a sequence of ground global parameter $\Theta = \boldsymbol{x}_1 : \mathtt{B}_1, \ldots, \boldsymbol{x}_n : \mathtt{B}_n$, hence the attribute "open". This way, a predicate P about objects of type τ is taken as a subset of $[\![\tau]\!]^{[\![\Theta]\!]}$, rather than just $[\![\tau]\!]$.

Technically, the contributions of this paper are threefold:

- We define four concrete logical systems, all built around the aforementioned ideas, and capable of dealing with formulas, programs, relations between programs (see Section 4 for those three systems), and local reasoning (Section 5), respectively.
- For each of the presented systems, we give an interpretation of formulas and sequents into an underlying semantic universe.
- We provide a series of examples, dealing with properties like continuity and differentiability, showing how they can be handled in the logical systems. These are spread out in Section 4 and Section 5.

2 Reasoning about First-Order Properties, Compositionally

Before moving to the technical development of open higher-order logic, we informally describe the kind of problems such a logic is meant to solve. We do so by means of an example that higher-order logic is *in principle* capable of dealing with, although doing that compositionally is highly nontrivial: continuity in the presence of higher-order functions.

Let us consider a simply typed λ-calculus extended with a base type `Real` for real numbers, as well as with constants and symbols for functions of first-order type, not necessarily interpreted as continuous functions. It is clear that a term t of type $\mathtt{Real} \to \mathtt{Real}$, in a situation like the one just mentioned, computes a function whose continuity properties depend on how it is constructed and on which constants occur within it. In UHOL [11], the unary variant of higher order logic (HOL), the fact that such a term t actually computes

a continuous function can be expressed as the formula $\mathcal{C}(t)$, where $\mathcal{C}$ is a unary predicate on terms of type $\texttt{Real} \to \texttt{Real}$. Clearly, this predicate's properties, at least some of them, should be captured by way of logical formulas, which become axioms in the underlying formal system. We would thus have, e.g., an axiom about stability of continuity by composition, namely the following formula:

$$\forall p, q : \texttt{Real} \to \texttt{Real}, (\mathcal{C}(p) \wedge \mathcal{C}(q)) \Rightarrow \mathcal{C}(\lambda x.q\ (p\ x))$$

Now, let t be the term $\lambda x.h\ (g\ (f\ x))$, where f, g, h are all constants of type $\texttt{Real} \to \texttt{Real}$ interpreted as continuous functions. We would like to prove within UHOL, possibly in a compositional way, that t – itself a term of type $\texttt{Real} \to \texttt{Real}$ – is continuous too. By design, UHOL's rules – and thus UHOL's proofs – follow the term structure target formulas refer to; and in the case of λ-abstractions, the corresponding rule can be read as follows: "if, whenever the argument x satisfies a precondition ϕ, the body u satisfies the postcondition ψ, then $\lambda x.u$ satisfies the formula $\forall x.\phi \Rightarrow \psi$." When it comes to $\mathcal{C}(t)$, this means we need a postcondition satisfied by the term $h\ (g\ (f\ x))$ *for every* x. But continuity on x cannot be defined looking only at (the semantics of) $h\ (g\ (f\ x))$, namely at a single real number, and thus it is not clear how one should proceed.

Open higher-order logic (OHOL), instead, considers open objects as first-class citizens, without altering the rest of the framework in any other way, so still decomposing terms in the style of logical relations. In other words, in open higher-order logic we can reason on the open term $h\ (g\ (f\ x))$ seeing it *as a function* of type $\texttt{Real} \to \texttt{Real}$. Formally, in this case, we consider open terms for the context $\Theta = x : \texttt{Real}$. The proof can then proceed as follows:

$$\frac{\Gamma \mid \Psi \vdash^{\Theta} f : \texttt{Real} \to \texttt{Real}\ \{\forall y : \texttt{Real}, \mathcal{C}(y) \Rightarrow \mathcal{C}(\boldsymbol{r}\ y)\} \qquad \Gamma \mid \Psi \vdash^{\Theta} (g\ (h\ x)) : \texttt{Real}\ \{\mathcal{C}(\boldsymbol{r})\}}{\Gamma \mid \Psi \vdash^{\Theta} f\ (g\ (h\ x)) : \texttt{Real}\ \{\mathcal{C}(\boldsymbol{r})\}}$$

with the following tree for the second premise.

$$\frac{\Gamma \mid \Psi \vdash^{\Theta} g : \texttt{Real} \to \texttt{Real}\ \{\forall y : \texttt{Real}, \mathcal{C}(y) \Rightarrow \mathcal{C}(\boldsymbol{r}\ y)\} \qquad \Gamma \mid \Psi \vdash^{\Theta} (h\ x) : \texttt{Real}\ \{\mathcal{C}(\boldsymbol{r})\}}{\Gamma \mid \Psi \vdash^{\Theta} (g\ (h\ x)) : \texttt{Real}\ \{\mathcal{C}(\boldsymbol{r})\}}$$

The application rule that we use for this proof is basically an elimination of the implication constructor for the target formula. The derivation for the functions f and g would then be obtained using an axiom assessing that continuity composes, whereas the continuity of $h\ x$ would then follow by hypothesis, as we assumed that all three functions were continuous and thus, in particular, we have $\mathcal{C}(h\ x)$.

The above example relies on continuity to show a limitation of (U)HOL. Such a limitation, however, is not specific to continuity, and it virtually affects any first order property. In this paper, we shall study another interesting example involving a first-order property, this time at a relational level: correctness of a state-of-the-art algorithm for automatic differentiation. In fact, an algorithm for forward mode differentiation of simply-typed terms has been recently proved correct by way of open logical relations [3]. Formalizing the aforementioned correctness proof in higher-order logic, however, would pose problems of exactly the same kind as those we described above, since derivability and (automatic) differentiation only make sense when spelled out on functions. In Section 4.2 , we show how to solve this problem by a relational version of OHOL capable of dealing both with (open) terms t *and* with their derivatives $D(t)$.

$$t, u ::= x \mid \underline{c} \mid \underline{f} \mid \lambda y.t \mid t\ u \mid \langle t, u\rangle \mid \pi_1(t) \mid \pi_2(t) \qquad \tau, \sigma ::= \mathtt{B} \mid \tau \times \sigma \mid \tau \to \sigma$$

$$\frac{}{\Gamma, x : \tau \vdash x : \tau} \qquad \frac{c : \mathtt{B} \in \mathcal{C}}{\Gamma \vdash \underline{c} : \mathtt{B}} \qquad \frac{f : \tilde{\mathtt{B}} \to \mathtt{B} \in \mathcal{F}}{\Gamma \vdash \underline{f} : \tilde{\mathtt{B}} \to \mathtt{B}} \qquad \frac{\Gamma, x : \tau \vdash t : \sigma}{\Gamma \vdash \lambda x.t : \tau \to \sigma}$$

$$\frac{\Gamma \vdash t : \tau \to \sigma \qquad \Gamma \vdash u : \tau}{\Gamma \vdash t\ u : \sigma} \qquad \frac{\Gamma \vdash t : \tau_1 \qquad \Gamma \vdash u : \tau_2}{\Gamma \vdash \langle t, u\rangle : \tau_1 \times \tau_2} \qquad \frac{\Gamma \vdash t : \tau_1 \times \tau_2}{\Gamma \vdash \pi_i(t) : \tau_i}$$

Figure 1 Syntax and Static Semantics for $\Lambda_{\mathcal{C},\mathcal{F}}$.

Suppose given for each base type $\mathtt{B}$ an interpretation $[\![\mathtt{B}]\!]$ (for example, the type for reals could be given as an interpretation the actual set of real numbers). Suppose also that for each constant $c : \mathtt{B} \in \mathcal{C}$, we have an interpretation $[\![c]\!] \in [\![\mathtt{B}]\!]$ and for each function $f : \tilde{\mathtt{B}} \to \mathtt{B} \in \mathcal{F}$, we have an interpretation $[\![f]\!] : [\![\mathtt{B}]\!]^{[\![\tilde{\mathtt{B}}]\!]}$.

Then, the interpretation of types is defined by an object in the category of **Set**:

$$\tau_1 \times \tau_2 = [\![\tau_1]\!] \times [\![\tau_2]\!] \qquad \tau_1 \to \tau_2 = [\![\tau_2]\!]^{[\![\tau_1]\!]}$$

And, a typing derivation $\Gamma \vdash t : \tau$ is interpreted as a map $[\![\Gamma \vdash t : \tau]\!] : [\![\Gamma]\!] \to [\![\tau]\!]$, this is defined by induction on the type derivation with:

$$[\![\Gamma, x : \tau \vdash x : \tau]\!](\tilde{y}, y) = y \qquad [\![\Gamma \vdash \underline{c} : \mathtt{B}]\!](\tilde{y}) = [\![c]\!] \qquad [\![\Gamma \vdash \underline{f} : \tilde{\mathtt{B}} \to \mathtt{B}]\!](\tilde{y}) = [\![f]\!]$$

$$[\![\Gamma \vdash \lambda x.t : \tau_1 \to \tau_2]\!](\tilde{y}) = \mathtt{fun}\ (y : [\![\tau_1]\!]) \mapsto [\![\Gamma, x : \tau_1 \vdash t : \tau_2]\!](\tilde{y}, y)$$

$$[\![\Gamma \vdash t_1\ t_2 : \tau_2]\!](\tilde{y}) = ([\![\Gamma \vdash t_1 : \tau_1 \to \tau_2]\!](\tilde{y}))([\![\Gamma \vdash t_2 : \tau_1]\!](\tilde{y}))$$

$$[\![\Gamma \vdash \langle t_1, t_2\rangle : \tau_1 \times \tau_2]\!](\tilde{y}) = ([\![\Gamma \vdash t_1 : \tau_1]\!](\tilde{y}), [\![\Gamma \vdash t_2 : \tau_2]\!](\tilde{y}))$$

$$[\![\Gamma \vdash \pi_i(t) : \tau_i]\!](\tilde{y}) = \pi_i([\![\Gamma \vdash t : \tau_1 \times \tau_2]\!](\tilde{y}))$$

Figure 2 Denotational Semantics for $\Lambda_{\mathcal{C},\mathcal{F}}$.

3 Preliminaries

The (higher-order) logic we deal with in this work is, in its bare essence, a formal framework to prove properties about higher-order programs. Consequently, a precise definition of such a logic requires a formal definition of what a program is. Here, we consider a λ-calculus with base types, constants, and functions [3]. We write $\mathcal{C}$ and $\mathcal{F}$ for the collections of constants and symbols for first-order functions upon which terms are defined. The syntax and statics of the resulting calculus, that we denote by $\Lambda_{\mathcal{C},\mathcal{F}}$, are given in Figure 1. The metavariable $\mathtt{B}$ ranges over *base types* (we assume a fixed collection of base types as given), such as real numbers or booleans. Moreover, we assume each constant $c \in \mathcal{C}$ to inhabit a base type $\mathtt{B}$ (notation $c : \mathtt{B} \in \mathcal{C}$) and each function $f \in \mathcal{F}$ to inhabit a function type $\mathtt{B}_1 \times \cdots \times \mathtt{B}_m \to \mathtt{B}$ (notation $f : \mathtt{B}_1 \times \cdots \times \mathtt{B}_m \to \mathtt{B} \in \mathcal{F}$). We oftentimes employ the notation $\tilde{\mathtt{B}}$ to denote a tuple $\mathtt{B}_1 \times \cdots \times \mathtt{B}_m$ when m is clear from the context. This notation generalizes to other objects, e.g. terms, types, and typing contexts.

Finally, we endow $\Lambda_{\mathcal{C},\mathcal{F}}$ with a standard set-theoretic denotational semantics in the usual way. Accordingly, each type τ is interpreted as a set $[\![\tau]\!]$, and any derivable typing judgment $\Gamma \vdash t : \tau$ is interpreted as a function $[\![\Gamma \vdash t : \tau]\!] : [\![\Gamma]\!] \to [\![\tau]\!]$, where $[\![\Gamma]\!] = \prod_{y:\sigma \in \Gamma} [\![\sigma]\!]$.

$$\frac{}{\Gamma \mid \Psi, \phi \vdash \phi} \qquad \frac{\Gamma \vdash t_i : \tau \qquad t_1 =_{(\to)} t_2}{\Gamma \mid \Psi \vdash (t_1 = t_2)} \qquad \frac{\Gamma \mid \Psi \vdash \phi[t_1/y] \qquad \Gamma \mid \Psi \vdash (t_1 = t_2)}{\Gamma \mid \Psi \vdash \phi[t_2/y]}$$

$$\frac{\Gamma, y : \tau \mid \Psi \vdash \phi}{\Gamma \mid \Psi \vdash \forall y : \tau.\phi} \qquad \frac{\Gamma \mid \Psi \vdash \forall y : \tau.\phi \qquad \Gamma \vdash t : \tau}{\Gamma \mid \Psi \vdash \phi[t/y]}$$

Figure 3 Selected Rules of HOL.

3.1 Higher-Order Logic

Having defined $\Lambda_{\mathcal{C},\mathcal{F}}$, we now recall basic definitions of *Higher-Order Logic* [11], the logic we will build upon.

▶ **Definition 1.** *The syntax of HOL formulas is given by the following grammar:*

$$\phi ::= P(t_1, \dots, t_m) \mid \top \mid \bot \mid \phi \wedge \phi \mid \phi \vee \phi \mid \phi \Rightarrow \phi \mid \forall y : \tau.\phi \mid \exists y : \tau.\phi.$$

HOL formulas are defined starting from a given collection of atomic predicates on $\Lambda_{\mathcal{C},\mathcal{F}}$ terms, from which more complex formulas are then constructed relying on connectives. Each predicate P comes with an *arity* $(\tau_1, \dots, \tau_m)$ stating that in an atomic formula $P(t_1, \dots, t_m)$ each term t_i must have type τ_i. Moreover, notice that variables introduced by quantifiers are (typed) term variables, meaning that they can occur in terms themselves occurring in atomic formulas. This intuitive description is formalized by means of well-typing judgments of the form $\Gamma \vdash \phi$ whose defining rules are as expected. Due to space constraints, we omit the formal definition of such rules and address the reader to one of the many papers on the subject [2, 11]. As it is customary, we also assume an equality predicate to be available for *all* types.

HOL inherits the set-theoretical semantics of $\Lambda_{\mathcal{C},\mathcal{F}}$: given an interpretation $[\![P]\!] \subseteq [\![\tau_1]\!] \times \cdots \times [\![\tau_m]\!]$ for each predicate P of arity $(\tau_1, \dots, \tau_m)$, well-typed assertions $\Gamma \vdash \phi$ are interpreted as subsets $[\![\Gamma \vdash \phi]\!] \subseteq [\![\Gamma]\!]$. Intuitively, this semantics is defined for non-atomic formulas by the standard interpretation of boolean constructors and quantification, whereas we define the interpretation $[\![P(t_1, \dots, t_m)]\!]$ of an atomic formula $P(t_1, \dots, t_m)$ as the set of elements $\gamma \in [\![\Gamma]\!]$ such that $([\![\Gamma \vdash t_i : \tau_i]\!](\gamma))_{1 \leq i \leq m} \in [\![P]\!]$.

The real power of HOL is not its set-theoretic semantics, but its proof theory. In fact, HOL comes with a proof system that we recall here in a sequent calculus-style. Such a system employs judgments of the form $\Gamma \mid \Psi \vdash \phi$, where Ψ is a set of well-typed assertions, and ϕ is a well-typed assertion. A valid judgment $\Gamma \mid \Psi \vdash \phi$ attests that in the typing context Γ and under the assumptions in Ψ, ϕ is true. Accordingly, predicates P must come with axioms defining their (logical) meaning, as there is no rule for arbitrary predicates. We recall some of the most important rules in Figure 3, other rules being the usual inference rules for logical constructors (we write $=_{(\to)}$ for the smallest equivalence relation including $\to$, the reduction relation on $\Lambda_{\mathcal{C},\mathcal{F}}$, which can be defined as expected). Note that in addition to the logical rules, there are rules specific to the equality predicate allowing, in particular, term substitutions.

3.2 Unary HOL

In practical examples, especially in presence of unary predicates, (the proof system of) HOL may be difficult to use, its rules being ultimately formula-directed. In those cases, it is desirable to have a system (whose rules are) directed by the structure of the terms involved.

$$\frac{\Gamma, y : \tau \mid \Psi \vdash \phi[y/\boldsymbol{r}]}{\Gamma, y : \tau \mid \Psi \vdash y : \tau \ \{\phi\}} \qquad \frac{\Gamma \mid \Psi \vdash \phi[\underline{c}/\boldsymbol{r}]}{\Gamma \mid \Psi \vdash \underline{c} : \mathtt{B} \ \{\phi\}}$$

$$\frac{\Gamma \mid \Psi \vdash \phi[\underline{f}/\boldsymbol{r}]}{\Gamma \mid \Psi \vdash \underline{f} : \tilde{\mathtt{B}} \to \mathtt{B} \ \{\phi\}} \qquad \frac{\Gamma, y : \tau \mid \Psi, \psi \vdash t : \sigma \ \{\phi\}}{\Gamma \mid \Psi \vdash \lambda y.t : \tau \to \sigma \ \{\forall y.\psi \Rightarrow \phi[\boldsymbol{r}\ y/\boldsymbol{r}]\}}$$

$$\frac{\Gamma \mid \Psi \vdash t : \tau \to \sigma \ \{\forall y.\psi[y/\boldsymbol{r}] \Rightarrow \phi[\boldsymbol{r}\ y/\boldsymbol{r}]\} \qquad \Gamma \mid \Psi \vdash u : \tau \ \{\psi\}}{\Gamma \mid \Psi \vdash t\ u : \sigma \ \{\phi[u/y]\}}$$

$$\frac{\Gamma \mid \Psi \vdash t_i : \tau_i \ \{\phi_i\} \qquad \Gamma \mid \Psi \vdash \forall y, z.\phi_1[y/\boldsymbol{r}] \wedge \phi_2[z/\boldsymbol{r}] \Rightarrow \phi[\langle y, z\rangle/\boldsymbol{r}]}{\Gamma \mid \Psi \vdash \langle t_1, t_2\rangle : \tau_1 \times \tau_2 \ \{\phi\}}$$

$$\frac{\Gamma \mid \Psi \vdash t : \tau_1 \times \tau_2 \ \{\phi[\pi_i(\boldsymbol{r})/\boldsymbol{r}]\}}{\Gamma \mid \Psi \vdash \pi_i(t) : \tau_i \ \{\phi\}} \qquad \frac{\Gamma \mid \Psi \vdash t : \tau \ \{\psi\} \qquad \Gamma \mid \Psi \vdash \psi[t/\boldsymbol{r}] \Rightarrow \phi[t/\boldsymbol{r}]}{\Gamma \mid \Psi \vdash t : \tau \ \{\phi\}}$$

Figure 4 Rules of UHOL.

Moving from this observation, Aguirre et al. [11, 2] have developed term-directed proof systems for relational higher-order logics. We recall how such systems work, focusing on the unary case only.

▶ **Definition 2.** Unary *higher-order logic (UHOL) is an inference system based on judgments of the form* $\Gamma \mid \Psi \vdash t : \tau \ \{\phi\}$*, where* Γ *and* t *are as usual,* Ψ *is a set of HOL formulas, and* ϕ *is a HOL formula with a distinguished free variable* $\boldsymbol{r}$ *not appearing in* Γ *(i.e.* $\boldsymbol{r}$ *acts as a placeholder for* t *in the* target formula ϕ*). The defining rules of UHOL are given in Figure 4.*

A UHOL judgment $\Gamma \mid \Psi \vdash t : \tau \ \{\phi\}$ attests that t has type τ in the typing context Γ, and that the formula $\phi[t/\boldsymbol{r}]$ is true under the assumptions in Ψ. To prove such judgments, we rely on term-directed rules. In particular, the first three rules in Figure 4 state that for base terms, satisfiability of the target formula must be verified in the HOL judgment system. Other rules reshape the target formula according to the structure of t. For example, in the λ-abstraction rule the target formula expresses that if an argument satisfies a precondition ψ, then the application satisfies a postcondition ϕ. This can be verified assuming ψ and trying to prove the target formula ϕ for the body of the λ-abstraction. Finally, the last rule, that does not depend on the structure of t, allows us to use HOL judgments to weaken target formulas.

Aguirre et al.'s results [11, 2] show that HOL and UHOL are equivalent – i.e. $\Gamma \mid \Psi \vdash t : \sigma \ \{\phi\}$ if and only if $\Gamma \mid \Psi \vdash \phi[t/\boldsymbol{r}]$ – and that they are both sound in regard to the previously introduced set-theoretic semantics: if $\Gamma \mid \Psi \vdash \phi$ is valid, then $[\![\Gamma \vdash (\bigwedge_{\psi \in \Psi} \psi)]\!] \subseteq [\![\Gamma \vdash \phi]\!]$. But even if equivalent from a semantic and provability perspective, they are not so if one looks at proof effectiveness and automation.

In the HOL system, it is difficult to know when to use a term substitution to simplify a term, as well as when to apply the cut rule. Consequently, the system turns out to work well on judgments speaking about simple enough terms, but it may be difficult to use when more complex terms are involved. In UHOL, instead, the last rule of Figure 4 – which is crucial to guarantee expressiveness of the logic – turns out to be problematic for automation, and it is thus desirable to avoid its usage as much as possible. This makes the proof system of UHOL effective when applied to formulas with possibly complex terms, but *only* if there is little or no need to adjust the target formula, i.e. to use the last rule in Figure 4. Additionally, UHOL relies on HOL to prove properties of base terms, and thus it inherits the same weaknesses of the latter. Finally, the system is designed to work on terms and formulas whose shape

matches the semantic behavior of the terms they refer to. In absence of such a correspondence, proofs become remarkably difficult. This is indeed problematic, as there are many natural examples lacking such a correspondence and for which, consequently, UHOL is ineffective. A typical pattern of such a behavior is, for instance, the one we described in Section 2: when a formula for a λ-abstraction focuses on the whole function – and not on the actual application of this function to an argument – the rule for λ-abstraction cannot be used.

The just described scenario, namely the one in which one aims to prove *first-order* properties of programs, highlights an important weakness of both HOL and UHOL: none of the them can easily – not to say compositionally – cope with first-order properties of programs. The main contribution of the present work, namely the definition of Open HOL, provides a way to go beyond the aforementioned weakness, this way achieving effective and compositional reasoning about first-order program properties.

4 Open Higher-Order Logic

In this section, we introduce *Open* HOL (OHOL, for short), a higher-order logic designed to natively deal with *first-order properties* of programs. To do so, we follow the idea behind open logical relations [3] where, in the base case of the (recursive) definition of a logical relation, *closed* terms of a ground type $\mathtt{B}$ – which, semantically, correspond to elements in $[\![\mathtt{B}]\!]$ – are replaced by *open* terms having free variables in a fixed typing context Θ – so that the semantics of such terms is given by *functions* from $[\![\Theta]\!]$ to $[\![\mathtt{B}]\!]$, which are then required to satisfy the first-order property of interest.

Let us fix a typing context $\Theta = \boldsymbol{x}_1 : \mathtt{B}_1, \dots, \boldsymbol{x}_n : \mathtt{B}_n$. For the sake of clarity, we consider variables $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ as disjoint from the usual term variables, which we denote by $y, y_1, \dots$. OHOL's formulas are defined parametrically with respect to Θ (meaning that each typing context Θ gives a OHOL's formulas' grammar) by the following grammar:

$$\phi ::= P^{\Theta}(t_1, \dots, t_m) \mid \top \mid \bot \mid \phi \wedge \phi \mid \phi \vee \phi \mid \phi \Rightarrow \phi \mid \forall y : \tau.\phi \mid \exists y : \tau.\phi.$$

Notice that there is only one difference between HOL's and OHOL's formulas, namely atomic predicates. In OHOL, in fact, atomic predicates P^{Θ} are parametrized by the typing context Θ, with the intended meaning that arguments of P^{Θ} may have free variables in Θ. That is, if a predicate P^{Θ} has arity $(\tau_1, \dots, \tau_n)$, then in an atomic formula $P^{\Theta}(t_1, \dots, t_m)$ we require each term t_i to have type τ_i in the typing context Θ. Formally, this implies that to derive well-typed OHOL's judgments $\Gamma \vdash^{\Theta} \phi$, we shall rely on the following rule for atomic predicates:

$$\frac{P^{\Theta} \text{ has arity } (\tau_1, \dots, \tau_m) \qquad \forall 1 \leq i \leq m,\ \Gamma, \Theta \vdash t_i : \tau_i}{\Gamma \vdash^{\Theta} P^{\Theta}(t_1, \dots, t_m)}$$

Let us now see how the admittedly minor differences between the defining grammars of HOL and OHOL formulas (and, consequently, in their judgment rules) impact on the logics themselves. We begin with set-theoretic semantics, where we see that OHOL indeed interprets a term t of type τ with free variables in Θ as a function from $[\![\Theta]\!]$ to $[\![\tau]\!]$. More generally, we interpret well-typed assertions $\Gamma \vdash^{\Theta} \phi$ as subsets $[\![\Gamma \vdash^{\Theta} \phi]\!] \subseteq [\![\Gamma^{\Theta}]\!]$ with $[\![\Gamma^{\Theta}]\!] \cong \prod_{y:\tau \in \Gamma} [\![\tau]\!]^{[\![\Theta]\!]}$. For readability, it is convenient to introduce the following notation: given $f \in B^{A \times C}$ and $g \in A^C$, we define $f \star g \in B^C$ by $(f \star g)(x) = f(g(x), x)$.

▶ **Definition 3.** *Given an interpretation* $[\![P^{\Theta}]\!] \subseteq \prod_{1 \leq i \leq m} [\![\tau_i]\!]^{[\![\Theta]\!]}$ *of predicates* P^{Θ} *of arity* $(\tau_1, \dots, \tau_m)$*, the semantics* $[\![\Gamma \vdash^{\Theta} P^{\Theta}(t_1, \dots, t_m)]\!] \subseteq [\![\Gamma]\!]^{[\![\Theta]\!]}$ *of the judgment* $\Gamma \vdash^{\Theta} P^{\Theta}(t_1, \dots, t_m)$ *is defined as follows:*

$$[\![\Gamma \vdash^{\Theta} P^{\Theta}(t_1, \ldots, t_m)]\!] = \{\tilde{y} \in [\![\Gamma^{\Theta}]\!] \mid \prod_{1 \leq i \leq m} ([\![\Gamma, \Theta \vdash t_i : \tau_i]\!] \star \tilde{y}) \in [\![P^{\Theta}]\!]\}.$$

We define the semantics $[\![\Gamma \vdash^{\Theta} \phi]\!] \subseteq [\![\Gamma]\!]^{[\![\Theta]\!]}$ *of a well-typed judgment* $\Gamma \vdash^{\Theta} \phi$ *by inductively extending* $[\![\Gamma \vdash^{\Theta} P^{\Theta}(t_1, \ldots, t_m)]\!]$ *in the usual way. For instance:*

$$[\![\Gamma \vdash^{\Theta} \phi \vee \psi]\!] = [\![\Gamma \vdash^{\Theta} \phi]\!] \cup [\![\Gamma \vdash^{\Theta} \psi]\!];$$
$$[\![\Gamma \vdash^{\Theta} \forall y : \tau.\phi]\!] = \{\tilde{y} \in [\![\Gamma^{\Theta}]\!] \mid \forall y \in [\![\tau]\!]^{[\![\Theta]\!]}, (\tilde{y}, y) \in [\![\Gamma, y : \tau \vdash^{\Theta} \phi]\!]\}.$$

Notice that even if OHOL's formulas are essentially those of HOL, Definition 3 interprets terms and variables of type τ as function from $[\![\Theta]\!]$ to $[\![\tau]\!]$. In particular, notice that in the formula $\forall y : \tau.\phi$ the variable y has type τ in the underlying context, but it is interpreted as a function in $[\![\tau]\!]^{[\![\Theta]\!]}$.

Having defined the set-theoretic semantics of OHOL, we move to its proof system. We consider judgments of the form $\Gamma \mid \Psi \vdash^{\Theta} \phi$ and adjust the judgment system of HOL (Figure 3) by adding the typing context Θ to all premises of a type derivation of $\Lambda_{\mathcal{C},\mathcal{F}}$ terms. For example the rule

$$\frac{\Gamma \mid \Psi \vdash \forall y : \tau.\phi \quad \Gamma \vdash t : \tau}{\Gamma \mid \Psi \vdash \phi[t/y]} \quad \text{is replaced by} \quad \frac{\Gamma \mid \Psi \vdash^{\Theta} \forall y : \tau.\phi \quad \Gamma, \Theta \vdash t : \tau}{\Gamma \mid \Psi \vdash^{\Theta} \phi[t/y]}$$

▶ **Proposition 4.** *OHOL is sound with respect to the set-theoretic semantics. That is, if* $\Gamma \mid \Psi \vdash^{\Theta} \phi$ *is derivable, then* $[\![\Gamma \vdash^{\Theta} \bigwedge_{\psi \in \Psi} \psi]\!] \subseteq [\![\Gamma \vdash^{\Theta} \phi]\!]$.

Proof. By induction on $\Gamma \mid \Psi \vdash^{\Theta} \phi$ proceeding as in the corresponding proof for HOL. ◀

4.1 Unary OHOL

The OHOL proof system, being ultimately defined upon the one for HOL, shares (some) strengths and weaknesses with the latter. In particular, as for HOL, the formula-directed nature of OHOL rules makes OHOL difficult to use in presence of complex terms. To overcome this problem, we proceed as in the design of UHOL and introduce a new judgment system, which we dub *Unary Open HOL* (UOHOL, for short), whose proof rules are term-directed. Formally, UOHOL has judgments of the form $\Gamma \mid \Psi \vdash^{\Theta} t : \tau \ \{\phi\}$, where Ψ is a set of OHOL formulas, ϕ is a OHOL formula with a distinguished variable $\boldsymbol{r}$, and t is a $\Lambda_{\mathcal{C},\mathcal{F}}$ term. A judgment $\Gamma \mid \Psi \vdash^{\Theta} t : \tau \ \{\phi\}$ states that $\Gamma, \Theta \vdash t : \tau$ and that, under the assumptions in Ψ, $\phi[t/\boldsymbol{r}]$ is satisfied. UOHOL rules consist of the previously mentioned rules of Figure 4, with one additional rule:

$$\frac{(\boldsymbol{x}_i : \mathtt{B}_i) \in \Theta \qquad \Gamma \mid \Psi \vdash^{\Theta} \phi[\boldsymbol{x}_i/\boldsymbol{r}]}{\Gamma \mid \Psi \vdash^{\Theta} \boldsymbol{x}_i : \mathtt{B}_i \ \{\phi\}}$$

Such a rule corresponds to the axiom rule in Figure 4, but for variables in Θ. Finally, following the case of HOL and UHOL, we prove equivalence of OHOL and UOHOL.

▶ **Proposition 5.** *UOHOL and OHOL are equivalent, i.e.* $\Gamma \mid \Psi \vdash^{\Theta} t : \tau \ \{\phi\}$ *if and only if* $\Gamma \mid \Psi \vdash^{\Theta} \phi[t/\boldsymbol{r}]$.

Compared to UHOL, we see that UOHOL allows for more satisfactory proofs of higher-order properties, as mentioned in Section 2. To see that, let ϕ be the following OHOL formula: $\forall y : \tau \to \sigma, \forall z : \tau, P^{\Theta}(y\ z)$. It is easy to see that ϕ could be written as a HOL formula too: for, let us consider the formula ψ below, with T being product of types in Θ:

$$\forall y : T \to (\tau \to \sigma), \forall z : (T \to \tau), P(\lambda x.(y\ x)\ (z\ x)).$$

Compared to ψ, the formula ϕ is considerably simpler, and thus it facilitates derivation in OHOL or UOHOL. Moreover, it is precisely the simpler (logical) structure of ϕ that allows us to deal with first-order properties smoothly, in contrast to the previously mentioned UHOL's weaknesses. This is the main point of OHOL and UOHOL. We do not claim any increase of expressiveness over the standard HOL. What we claim, instead, is that OHOL and UOHOL constitute a way to simplify the process of proving some formulas of HOL that cannot be readily proved valid in HOL or UHOL.

4.2 OHOL with Multiple Domains and ROHOL

The discussion at the end of Section 3 on the drawbacks and benefits of HOL and UHOL remains relevant for OHOL and UOHOL. Moreover, even if OHOL solves some problems suffered by HOL when applied to first-order properties, it does *not* solve *all* of them. In particular, OHOL focuses on a single domain – the typing context Θ – that cannot be changed in a formula. Consequently, all terms in a given formula are relative to the *same* domain Θ. As we are going to see, however, it is sometimes useful to allow different terms to be associated to different typing contexts within the same formula. Here, we present a simple generalization of OHOL designed to tackle this problem, showing also its usefulness for the relational version of UOHOL, that we briefly sketch below.

4.2.1 Handling Multiples Domains

Let us begin with OHOL with multiple domains of definition. Due to space constraints, the formal description of the logic has to be omitted (but see [8] for details). Nonetheless, we can still outline its main features here, although at an informal level only. In OHOL with multiple domains of definition, we associate to each variable and term a unique typing context, but contrary to OHOL, we allow different terms to be associated with *distinct* typing contexts. In particular, we assign to variables not only types but also typing contexts, which now play the role of kinds. Consequently, we have typed variables $y : \tau :: \Theta$ stating that y has type τ, and τ has kind Θ. Semantically, we let kinds specify the domains of the semantic interpretation of variables: given typing contexts Θ_1 and Θ_2, the variables $y_1 : \tau_1 :: \Theta_1$ and $y_2 : \tau_2 :: \Theta_2$ are interpreted as functions in $[\![\tau_1]\!]^{[\![\Theta_1]\!]}$ and $[\![\tau_2]\!]^{[\![\Theta_2]\!]}$, respectively (recall that in OHOL each variable $y : \tau$ is interpreted as a function $[\![\tau]\!]^{[\![\Theta]\!]}$, with Θ *fixed*).

The semantic interpretation for formulas using kinds is a rather straightforward generalization of the semantics in Definition 3: we only need to modify the notion of arity for predicates, which are now of the form $(\tau_1 :: \Theta_1, \tau_2 :: \Theta_2, \dots, \tau_m :: \Theta_m)$, with the intended meaning that type τ_i has kind Θ_i. Therefore, a predicate of arity $(\tau_1 :: \Theta_1, \tau_2 :: \Theta_2, \dots, \tau_m :: \Theta_m)$ is now interpreted as a subset of $\prod_{1 \leq i \leq m} [\![\tau_i]\!]^{[\![\Theta_i]\!]}$.

As for the logical rules, the only relevant point is to specify a kind for each term and for each variable, and to impose that all variables in a term have the same kind. For example, we have the following rule for the introduction and elimination of the $\forall$ constructor:

$$\frac{\Gamma, y : \tau :: \Theta \mid \Psi \vdash \phi}{\Gamma \mid \Psi \vdash \forall y : \tau :: \Theta.\phi} \qquad \frac{\Gamma \mid \Psi \vdash \forall y : \tau :: \Theta.\phi \qquad \Delta \vdash t : \tau \qquad \forall (z : \sigma) \in \Delta, (z : \sigma :: \Theta) \in \Gamma}{\Gamma \mid \Psi \vdash \phi[t/y]}$$

4.2.2 Going Relationally

Even if theoretically fine, having access to different kinds is not that useful for UOHOL. Indeed, in a UOHOL proof, the focus is on a *single* term (as well as its subterms) only, so that if a term has kind Θ, then we do not gain much from using OHOL with multiple

domains of definition. The potential of such a logic, instead, becomes evident when we move to *relational* reasoning. Aguirre et al. [2, 11] have introduced RHOL, a relational version of UHOL talking about *pairs* of terms, rather than a single term in isolation. Because of the similarities between UHOL and UOHOL, the design of ROHOL from UOHOL can be done by mimicking the original construction by Aguirre et al. A RHOL judgment has the shape $\Gamma \mid \Psi \vdash t_1 : \tau_1 \sim t_2 : \tau_2 \ \{\phi\}$, where ϕ contains two special variables $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ acting as placeholders for t_1 and t_2, respectively. The intuitive meaning is that t_1 and t_2 have types τ_1 and t_2, respectively, in the typing context Γ, and that $\phi[t_1/\boldsymbol{r}_1][t_2/\boldsymbol{r}_2]$ holds under the assumptions in Ψ.

As an example, a RHOL rule for λ-abstraction is

$$\frac{\Gamma, y_1 : \sigma_1, y_2 : \sigma_2 \mid \Psi, \psi \vdash t_1 : \tau_1 \sim t_2 : \tau_2 \ \{\phi\}}{\Gamma \mid \Psi \vdash \lambda y_1.t_1 : \sigma_1 \to \tau_1 \sim \lambda y_2.t_2 : \sigma_2 \to \tau_2 \ \{\forall y_1, y_2.\psi \Rightarrow \phi[\boldsymbol{r}_1 \ y_1/\boldsymbol{r}_1][\boldsymbol{r}_2 \ y_2/\boldsymbol{r}_2]\}}$$

Such a rule can be seen as two instances of the rule for λ-abstractions from UHOL merged together. In a similar fashion, we can extend such a rule to our open setting with a fixed context Θ. More interesting is the case of multiple domains of definition, where we can consider the first term having kind Θ_1 and the second one having kind Θ_2. This way, we obtain the following rule in what we may call ROHOL with multiple domains of definition:

$$\frac{\Gamma, y_1 : \sigma_1 :: \Theta_1, y_2 : \sigma_2 :: \Theta_2 \mid \Psi, \psi \vdash t_1 : \tau_1 \sim t_2 : \tau_2 \ \{\phi\}}{\Gamma \mid \Psi \vdash \lambda y_1.t_1 : \sigma_1 \to \tau_1 \sim \lambda y_2.t_2 : \sigma_2 \to \tau_2 \ \{\forall y_1 : \sigma_1 :: \Theta_1, y_2 : \sigma_2 :: \Theta_2.\psi \Rightarrow \phi[\boldsymbol{r}_1 \ y_1/\boldsymbol{r}_1][\boldsymbol{r}_2 \ y_2/\boldsymbol{r}_2]\}}$$

Such a rule also allows us to see the benefits of having different kinds, as we are now allowed to relate functions with different initial domains. As before, space constraints force us to omit formal details (for which we refer to the long version of this paper [8]). Instead, we witness how our logic work by means of a nontrivial example.

▶ **Example 6** (Automatic Differentiation)**.** Let us consider the problem of formally proving correctness of forward mode automatic differentiation (AD) [20, 14] as treated by means of open logical relations [3]. In a nutshell, AD can be presented as a mapping D on terms and types such that if $\Gamma \vdash t : \tau$ then $D(\Gamma) \vdash D(t) : D(\tau)$, with $D(t)$ computing the derivative of t. We will not detail the transformation here (a complete description can be found in [3, 14], as well as in the long version of this paper [8]). Proving the correctness of AD amounts to showing that for any first order term $\Theta \vdash t : \mathtt{Real}$, where $\Theta = \boldsymbol{x}_1 : \mathtt{Real}, \ldots, \boldsymbol{x}_n : \mathtt{Real}$, the (semantics of the) term $D(\Theta) \vdash D(t) : \mathtt{Real} \times \mathtt{Real}$ – with $D(\Theta) = d\boldsymbol{x}_1 : \mathtt{Real} \times \mathtt{Real}, \ldots, d\boldsymbol{x}_n : \mathtt{Real} \times \mathtt{Real}$ – is the derivative of (the semantics of) t. Correctness proofs have beed recently given, both semantically [14] and syntactically [3]. Here, we sketch how the syntactic proof by Barthe et al. [3] can be embedded in RHOL. First, given a proof of $\Gamma, \Theta \vdash t : \tau$, we obtain the judgment

$$\Gamma :: \Theta, D(\Gamma) :: D(\Theta) \mid \Psi \vdash t : \tau \sim D(t) : D(\tau) \ \{\boldsymbol{r}_1 \ \mathcal{R}_\tau^\Theta \ \boldsymbol{r}_2\}$$

where Ψ is a set of assertions relating elements of Γ and $D(\Gamma)$ and $\boldsymbol{r}_1 \ \mathcal{R}_\tau^\Theta \ \boldsymbol{r}_2$ is a formula corresponding to the logical relation by Barthe et al.[3]. The key point to stress is that in our (ROHOL) proof the flow of the derivation follows naturally from the definition of an open logical relation, as elegantly showed by the case of λ-abstractions. For a λ-abstraction, in fact, we have $D(\lambda y : \sigma.t) = \lambda dy : D(\sigma).D(t)$ and $D(\sigma \to \tau) = D(\sigma) \to D(\tau)$. Consequently, we have to derive the following judgment (assuming $\Gamma \vdash \lambda y.t : \sigma \to \tau$)

$$\Gamma :: \Theta, D(\Gamma) :: D(\Theta) \mid \Psi \vdash \lambda y.t : \sigma \to \tau \sim \lambda dy.D(t) : D(\sigma) \to D(\tau) \ \{\boldsymbol{r}_1 \ \mathcal{R}_{\sigma\to\tau}^\Theta \ \boldsymbol{r}_2\}.$$

The logical relation $t_1 \ \mathcal{R}^{\Theta}_{\sigma\to\tau}\ t_2$ for arrow types in this case is defined by

$$\forall y_1 : \sigma :: \Theta. \forall y_2 : D(\sigma) :: D(\Theta). (y_1 \ \mathcal{R}^{\Theta}_{\sigma}\ y_2) \Rightarrow (t_1\ y_1)\ \mathcal{R}^{\Theta}_{\tau}\ (t_2\ y_2)$$

and this is exactly the shape of the λ-abstraction rule in ROHOL. Thus, we can apply the rule:

$$\frac{\Gamma :: \Theta, y : \sigma :: \Theta, D(\Gamma) :: D(\Theta), dy : D(\sigma) :: D(\Theta) \mid \Psi, (y_1 \ \mathcal{R}^{\Theta}_{\sigma}\ y_2) \vdash t : \tau \sim D(t) : D(\tau)\ \{\boldsymbol{r}_1\ \mathcal{R}^{\Theta}_{\tau}\ \boldsymbol{r}_2\}}{\Gamma :: \Theta, D(\Gamma) :: D(\Theta) \mid \Psi \vdash \lambda y.t : \sigma \to \tau \sim \lambda dy. D(t) : D(\sigma) \to D(\tau)\ \{\boldsymbol{r}_1\ \mathcal{R}^{\Theta}_{\sigma\to\tau}\ \boldsymbol{r}_2\}}$$

and conclude the proof by induction.

5 Local Open Higher-Order Logic

In this section, we go beyond OHOL and introduce a *local* version of OHOL allowing formulas to account for the domain of definition of functions. The motivation behind the introduction of such a logic is ultimately found in the interaction between the openness of OHOL – whereby terms of a ground type are actually regarded as first-order functions – and constructs of the language, whose correct semantic behavior relies on having actual terms of ground type – and not first-order functions – as arguments. To clarify, let us consider *the* main construct badly interacting with the current formulation of OHOL: the conditional. Let us consider the term construct $\mathsf{if}\ t\ \mathsf{then}\ u_1\ \mathsf{else}\ u_0$. The standard UHOL rule for this conditional would be:

$$\frac{\Gamma \vdash t : \mathtt{Bool} \qquad \Gamma \mid \Psi, t = \mathtt{tt} \vdash u_1 : \tau\ \{\phi\} \qquad \Gamma \mid \Psi, t = \mathtt{ff} \vdash u_0 : \tau\ \{\phi\}}{\Gamma \mid \Psi \vdash \mathsf{if}\ t\ \mathsf{then}\ u_1\ \mathsf{else}\ u_0 : \tau\ \{\phi\}}$$

expressing that this conditional satisfies a formula ϕ when both branches satisfy ϕ with the obvious assumption that in the first branch t is true and in the second branch t is false.

If we naively generalize this rule to UOHOL, we see that $t =^{\Theta} \mathtt{tt}$ does not describe equality between the boolean *values* t and $\mathtt{tt}$: instead, it gives equality between the boolean *function* t from $[\![\Theta]\!]$ to $[\![\mathtt{Bool}]\!]$ and the boolean *constant function* $\mathtt{tt}$ from $[\![\Theta]\!]$ to $[\![\mathtt{Bool}]\!]$. Consequently, the resulting rule would be unsound, as in the first branch we cannot assume t to be constantly equal to true (there are subsets of Θ in which t is true and others in which t is false).

To overcome this problem, we extend OHOL to take into account restricted (sub)domains of $[\![\Theta]\!]$. We do so by defining a new logic, *Local OHOL* (LOHOL, for short). Compared to OHOL, the main novelty of LOHOL is the introduction of formulas of the form $[S]\phi$, where S represents a subset of $[\![\Theta]\!]$, expressing that ϕ is a formula on functions with domain of definition S, instead of the whole $[\![\Theta]\!]$.

▶ **Definition 7.** *LOHOL's formulas are defined by the following grammar (we define disjunction, implication, and existential quantification by way of De Morgan's laws).*

$$\phi ::= P^{\Theta}(S_1, \dots, S_k; t_1, \dots, t_m) \mid \bot \mid \neg\phi \mid \phi \wedge \phi \mid \forall y : \tau.\phi \mid [S]\phi \mid \forall X.\phi$$
$$S ::= X \mid S \cup S \mid S \cap S \mid \emptyset \mid S^{\complement} \mid \{\Theta \mid P^{S}(t_1, \dots, t_m)\}.$$

Here, X is a set variable, $P^S(t_1, \dots, t_m)$ a predicate on terms that can use variables in Θ, and $P^{\Theta}(S_1, \dots, S_k; t_1, \dots, t_m)$ a predicate on arbitrary numbers of sets and terms.

We denote by $(k; \tau_1, \dots, \tau_m)$ the arity of a predicate P^{Θ}, with k the number of sets and τ_i type of t_i. This generic definition allows us to have predicates on terms only – as in OHOL – as well as predicates on sets only – such as the subset inclusion predicate. Finally,

$$\frac{}{X, \tilde{X} \mid \Gamma \vdash^{\Theta} X} \qquad \frac{(\Gamma, \Theta \vdash t_i : \tau_i)_{1 \le i \le m} \qquad P^S \text{ has arity } (\tau_1, \ldots, \tau_m)}{\tilde{X} \mid \Gamma \vdash^{\Theta} \{\Theta \mid P^S(t_1, \ldots, t_m)\}}$$

$$\frac{(\Gamma, \Theta \vdash t_i : \tau_i)_{1 \le i \le m} \qquad (\tilde{X} \mid \Gamma \vdash^{\Theta} S_j)_{1 \le j \le k} \qquad P^{\Theta} \text{ has arity } (k; \tau_1, \ldots, \tau_m)}{\tilde{X} \mid \Gamma \vdash^{\Theta} P^{\Theta}(S_1, \ldots, S_k; t_1, \ldots, t_m)}$$

$$\frac{\tilde{X} \mid \Gamma, y : \tau \vdash^{\Theta} \phi}{\tilde{X} \mid \Gamma \vdash^{\Theta} \forall y : \tau.\phi} \qquad \frac{X, \tilde{X} \mid \Gamma \vdash^{\Theta} \phi}{\tilde{X} \mid \Gamma \vdash^{\Theta} \forall X.\phi} \qquad \frac{\tilde{X} \mid \Gamma \vdash^{\Theta} S_2 \qquad \tilde{X} \mid \Gamma \vdash^{\Theta} \phi}{\tilde{X} \mid \Gamma \vdash^{\Theta} [S_2]\phi}$$

Figure 5 Well-Typing Judgments.

$\{\Theta \mid P^S(t_1, \ldots, t_m)\}$ is called the comprehension formula, as it gives a simple comprehension schema. For instance, $[\ \{\boldsymbol{x} : \texttt{Real} \mid \boldsymbol{x} \le 2\}]$ restricts the domain of definition from $\mathbb{R}$ to $]-\infty, 2]$.

Let us now move to LOHOL's semantics, beginning with the semantic of well-typed formulas. First, we define the well-typed judgments $\tilde{X} \mid \Gamma \vdash^{\Theta} S$ and $\tilde{X} \mid \Gamma \vdash^{\Theta} \phi$, where $\tilde{X}$ denotes a sequence of set variables occurring free in S and and ϕ. The most important rules are given in Figure 5. Such rules should be clear. In particular, the first rule, similar to an axiom rule, states that in a judgment of the form $\tilde{X} \mid \Gamma \vdash^{\Theta} S$ we cannot use set variables outside those in $\tilde{X}$. Notice also that in the last rule, the second premise shows that restricting an already restricted domain amounts to intersecting the domains themselves.

Next, we define the denotational semantics $[\![\tilde{X} \mid \Gamma \vdash^{\Theta} S]\!] : \widetilde{2^{[\![\Theta]\!]}} \times [\![\Gamma^{\Theta}]\!] \to 2^{[\![\Theta]\!]}$ of a well-typed set $\tilde{X} \mid \Gamma \vdash^{\Theta} S$, as mapping subsets of $[\![\Theta]\!]$ (one for each set variable in $\tilde{X}$) and functions in $[\![\tau]\!]^{[\![\Theta]\!]}$ to subsets of $[\![\Theta]\!]$. And finally, we define the semantics $[\![\tilde{X} \mid \Gamma \vdash^{\Theta} \phi]\!]_S \subseteq \widetilde{2^{[\![\Theta]\!]}} \times [\![\Gamma^{\Theta}]\!]$ of a well-typed assertion $\tilde{X} \mid \Gamma \vdash^{\Theta} \phi$ parameterized by a well-typed set $\tilde{X} \mid \Gamma \vdash^{\Theta} S$ as giving the possible values for set variables in $\tilde{X}$ and variables in Γ for which the formula ϕ, on the restricted domain represented by S, is true. The use of this parameter S is useful for the inductive definition, but in practice for a well-typed assertion $\tilde{X} \mid \Gamma \vdash^{\Theta} \phi$, its semantics is defined as $[\![\tilde{X} \mid \Gamma \vdash^{\Theta} \phi]\!]_{\emptyset^{\complement}}$, taking as parameter the set representing $[\![\Theta]\!]$.

▶ **Definition 8.** *Let* $[\![P^S]\!] \subseteq [\![\tau_1]\!] \times \cdots \times [\![\tau_m]\!]$ *and* $[\![P^{\Theta}]\!] \subseteq 2^{[\![\Theta]\!]^k} \times \coprod_{S \subseteq [\![\Theta]\!]} \prod_{1 \le i \le m} [\![\tau_i]\!]^S$ *be interpretations of predicates* P^S *of arity* $(\tau_1, \ldots, \tau_m)$ *and* P^{Θ} *of arity* $(k; \tau_1, \ldots, \tau_m)$*, respectively. Then, the semantics of well-typed judgments and of well-typed assertions is inductively defined thus:*

$$[\![X, \tilde{X} \mid \Gamma \vdash^{\Theta} X]\!]((X, \tilde{X}), \tilde{y}) = X$$

$$[\![\tilde{X} \mid \Gamma \vdash^{\Theta} \{\Theta \mid P^S(t_1, \ldots, t_m)\}]\!](\tilde{X}, \tilde{y}) = \{\tilde{x} \in [\![\Theta]\!] \mid [\![P^S]\!] \ni ([\![\Gamma, \Theta \vdash t_i : \tau_i]\!](\tilde{y}(\tilde{x}), \tilde{x}))_{1 \le i \le m}\}$$

$$[\![\tilde{X} \mid \Gamma \vdash^{\Theta} S_1 \cup S_2]\!](\tilde{X}, \tilde{y}) = [\![\tilde{X} \mid \Gamma \vdash^{\Theta} S_1]\!](\tilde{X}, \tilde{y}) \cup [\![\tilde{X} \mid \Gamma \vdash^{\Theta} S_2]\!](\tilde{X}, \tilde{y})$$

$$[\![\tilde{X} \mid \Gamma \vdash^{\Theta} P^{\Theta}(S_1, \ldots, S_k; t_1, \ldots, t_m)]\!]_S =$$
$$\{(\tilde{X}, \tilde{y}) \mid [\![P^{\Theta}]\!] \ni (([\![\tilde{X} \mid \Gamma \vdash^{\Theta} S_i]\!](\tilde{X}, \tilde{y}))_{1 \le i \le k}, ((\![\Gamma, \Theta \vdash t_i : \tau_i]\!] \star \tilde{y})_{|[\![\tilde{X} \mid \Gamma \vdash^{\Theta} S]\!](\tilde{X}, \tilde{y})})_{1 \le i \le m})\}$$

$$[\![\tilde{X} \mid \Gamma \vdash^{\Theta} \forall y : \tau.\phi]\!]_S = \{(\tilde{X}, \tilde{y}) \mid \forall y \in [\![\Theta \Rightarrow \tau]\!], (\tilde{X}, (\tilde{y}, y)) \in [\![\tilde{X} \mid \Gamma, y : \tau \vdash^{\Theta} \phi]\!]_S\}$$

$$[\![\tilde{X} \mid \Gamma \vdash^{\Theta} \forall X.\phi]\!]_S = \{(\tilde{X}, \tilde{y}) \mid \forall X \in 2^{[\![\Theta]\!]}, ((X, \tilde{X}), \tilde{y}) \in [\![X, \tilde{X} \mid \Gamma \vdash^{\Theta} \phi]\!]_S\}$$

$$[\![\tilde{X} \mid \Gamma \vdash^{\Theta} [S']\phi]\!]_S = [\![\tilde{X} \mid \Gamma \vdash^{\Theta} \phi]\!]_{S \cap S'}.$$

Let us comment on some important points of Definition 8. First, the interpretation of set predicates $P^S(t_1, \ldots, t_m)$ coincides with the interpretation of HOL predicates. This way, set predicates can work on ground terms, as in $t = \texttt{tt}$. The interpretation of predicates P^{Θ}, instead, is a subset of $2^{[\![\Theta]\!]^k} \times \coprod_{S \subseteq [\![\Theta]\!]} \prod_{1 \le i \le m} [\![\tau_i]\!]^S$. The first projection $2^{[\![\Theta]\!]^k}$ represents

the k subsets of $[\![\Theta]\!]$, whereas the second projection corresponds to the type of predicates in OHOL, but for all the possible subsets of Θ. As an example, consider $\Theta = \boldsymbol{x} : \texttt{Real}$ and let $\mathcal{C}^\Theta$ be a predicate of arity $(0; \texttt{Real})$ for local continuity. Then, we would interpret $\mathcal{C}^\Theta$ as

$$[\![\mathcal{C}^\Theta]\!] = \{f : A \to \mathbb{R} \mid A \subseteq \mathbb{R} \text{ and } f \text{ is locally continuous on } A\}.$$

The most interesting case of Definition 8 is the one for LOHOL predicates. The key point of this definition, and the main difference with OHOL, is the restriction operation $([\![\Gamma, \Theta \vdash t_i : \tau_i]\!] \star \tilde{y})_{|[\![\tilde{X} \mid \Gamma \vdash^\Theta S]\!](\tilde{X}, \tilde{y})}$. The considered function $([\![\Gamma, \Theta \vdash t_i : \tau_i]\!] \star \tilde{y})$ is the same as the one of OHOL but, as expected for LOHOL, we have to restrict its domain of definition to the subset represented by S, which is given by $[\![\tilde{X} \mid \Gamma \vdash^\Theta S]\!](\tilde{X}, \tilde{y})$.[1]

Definition 8 also shows that the constructor $[S]$ is semantically relevant only for atomic predicates. Syntactically, this implies that it is always possible to commute this constructor with other connectives, hence pushing it just before atomic formulas.

▶ **Proposition 9.** *We have the following logical equivalences:*

$$[S]\top \equiv \top \qquad [S]\neg\phi \equiv \neg[S]\phi \qquad [S][T]\phi \equiv [S \cap T]\phi \qquad [\emptyset^{\complement}]\phi \equiv \phi$$
$$[S](\phi \wedge \psi) \equiv [S]\phi \wedge [S]\psi \qquad [S]\forall y : \tau.\phi \equiv \forall y : \tau.[S]\phi \qquad [S]\forall X.\phi \equiv \forall X.[S]\phi$$

From Proposition 9 it follows that a judgment system for LOHOL needs not have a rule covering $[S]$, as it is enough to consider atomic formulas of the form $[S]P^\Theta(\dots)$.

▶ Remark 10. In the following, we will still use the notation $[S]\phi$ to improve readability. Moreover, we will tacitly assume to have defined an equivalence relation on sets that we can apply in formulas, so to identify sets (trivially) having the same semantics, such as $S \cap (T \cap U)$ and $(S \cap T) \cap U$.

We now give an inference judgment system for LOHOL. The rules are close to the ones presented in Figure 3 and are given in Figure 6. A judgment has the shape $\tilde{X} \mid \Gamma \mid \Psi \vdash^\Theta \phi$, and we ask that all formulas and sets are well-typed. The main difference with OHOL is that we have to keep track of set variables. For the last rule, we use the following definition.

▶ **Definition 11** (Functional Extension)**.** *The pointwise extension of a predicate P^S with arity $(\tau_1, \dots, \tau_m)$ is the predicate P^S_p of arity $(0; \tau_1, \dots, \tau_m)$ semantically interpreted by:*

$$[\![P^S_p]\!] = \{(f_1, \dots, f_m) \in \textstyle\prod_{1 \le i \le m} [\![\tau_i]\!]^S \mid S \subseteq [\![\Theta]\!] \wedge \forall \tilde{x} \in S.(f_1(\tilde{x}), \dots, f_m(\tilde{x})) \in [\![P^S]\!]\}$$

▶ **Example 12.** The pointwise extension $\le_p$ of $\le$ on real numbers corresponds to the usual pointwise comparison (on a possibly restricted domain of definition). Notice that the last rule of Figure 6 proves e.g. the formula $[\boldsymbol{x} + 3 \le 2](\boldsymbol{x} + 3 \le_p 2)$ showing that restricting the domain of definition to $]-\infty, -1]$ ensures that the function $x \mapsto x + 3$ is always smaller that the constant function $x \mapsto 2$ on this domain.

It is easy to show that the system defined in Figure 6 is indeed sound.

▶ **Theorem 13.** *We have $[\![\tilde{X} \mid \Gamma \vdash^\Theta \bigwedge_{\psi \in \Psi} \psi]\!]_{\emptyset^{\complement}} \subseteq [\![\tilde{X} \mid \Gamma \vdash^\Theta \phi]\!]_{\emptyset^{\complement}}$ for any derivable judgment $\tilde{X} \mid \Gamma \mid \Psi \vdash^\Theta \Phi$.*

[1] Notice that in a set-theoretical semantics, the notion of restriction indeed corresponds to the usual notion of function restrictions. If one wants to give a categorical meaning to this and consider other categories than **Set**, she could consider presheaves semantics for terms.

$$\frac{}{\tilde{X} \mid \Gamma \mid \Psi, \phi \vdash^{\Theta} \phi} \qquad \frac{\Gamma, \Theta \vdash t_i : \tau \qquad t_1 =_{(\to)} t_2 \qquad \tilde{X} \mid \Gamma \vdash^{\Theta} S}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} [S](t_1 = t_2)}$$

$$\frac{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} [S]\phi[t_1/y] \qquad \tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} [S](t_1 = t_2)}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} [S]\phi[t_2/y]} \qquad \frac{\tilde{X} \mid \Gamma, y : \tau \mid \Psi \vdash^{\Theta} \phi}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \forall y : \tau.\phi}$$

$$\frac{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \forall y : \tau.\phi \qquad \Gamma, \Theta \vdash^{\Theta} t : \tau}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \phi[t/y]} \qquad \frac{\tilde{X}, X \mid \Gamma \mid \Psi \vdash^{\Theta} \phi}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \forall X.\phi}$$

$$\frac{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \forall X.\phi \quad \tilde{X} \mid \Gamma \vdash^{\Theta} S}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \phi[S/X]} \qquad \frac{P_p^S \text{ pointwise extension of } P^S}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} [S \cap (\ \{\Theta \mid P^S(t_1, \ldots, t_m)\})]P_p^S(t_1, \ldots, t_m)}$$

Figure 6 Inference Judgment System for LOHOL.

$$\frac{\tilde{X} \mid \Gamma, y : \tau \mid \Psi \vdash^{\Theta} \phi[y/\boldsymbol{r}]}{\tilde{X} \mid \Gamma, y : \tau \mid \Psi \vdash^{\Theta} y : \tau\ \{\phi\}} \qquad \frac{(\boldsymbol{x}_i : \mathtt{B}_i) \in \Theta \qquad \tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \phi[\boldsymbol{x}_i/\boldsymbol{r}]}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \boldsymbol{x}_i : \mathtt{B}_i\ \{\phi\}} \qquad \frac{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \phi[\underline{c}/\boldsymbol{r}]}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \underline{c} : \mathtt{B}\ \{\phi\}}$$

$$\frac{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \phi[\underline{f}/\boldsymbol{r}]}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \underline{f} : \tilde{\mathtt{B}} \to \mathtt{B}\ \{\phi\}} \qquad \frac{\tilde{X} \mid \Gamma, y : \tau \mid \Psi, \psi \vdash^{\Theta} t : \sigma\ \{\phi\}}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \lambda y.t : \tau \to \sigma\ \{\forall y.\psi \Rightarrow \phi[\boldsymbol{r}\ y/\boldsymbol{r}]\}}$$

$$\frac{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} t : \tau \to \sigma\ \{\forall y.\psi[y/\boldsymbol{r}] \Rightarrow \phi[\boldsymbol{r}\ y/\boldsymbol{r}]\} \qquad \tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} u : \tau\ \{\psi\}}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} t\ u : \sigma\ \{\phi[u/y]\}}$$

$$\frac{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} t_i : \tau_i\ \{\phi_i\} \qquad \tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \forall y, z.\phi_1[y/\boldsymbol{r}] \wedge \phi_2[z/\boldsymbol{r}] \Rightarrow \phi[\langle y, z\rangle/\boldsymbol{r}]}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \langle t_1, t_2\rangle : \tau_1 \times \tau_2\ \{\phi\}}$$

$$\frac{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} t : \tau_1 \times \tau_2\ \{\phi[\pi_i(\boldsymbol{r})/\boldsymbol{r}]\}}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \pi_i(t) : \tau_i\ \{\phi\}} \qquad \frac{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} t : \tau\ \{\psi\} \qquad \tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \psi[t/\boldsymbol{r}] \Rightarrow \phi[t/\boldsymbol{r}]}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} t : \tau\ \{\phi\}}$$

$$\frac{\tilde{X}, X \mid \Gamma \mid \Psi \vdash^{\Theta} t : \tau\ \{\phi\}}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} t : \tau\ \{\forall X.\phi\}} \qquad \frac{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} t : \tau\ \{\forall X.\phi\} \qquad \tilde{X} \mid \Gamma \vdash^{\Theta} S}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} t : \tau\ \{\phi[S/X]\}}$$

$$\frac{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} t : \mathtt{Bool}\ \{\phi_t\} \qquad \begin{array}{c}\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} u_1 : \tau\ \{[\ \{\Theta \mid t = \mathsf{tt}\}]\phi_1\} \\ \tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} u_0 : \tau\ \{[\ \{\Theta \mid t = \mathsf{ff}\}]\phi_0\}\end{array}}{\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \mathsf{if}\ t\ \mathsf{then}\ u_1\ \mathsf{else}\ u_0 : \tau\ \{\phi_t[t/r] \wedge [\ \{\Theta \mid t = \mathsf{tt}\}]\phi_1 \wedge [\ \{\Theta \mid t = \mathsf{ff}\}]\phi_0\}}$$

Figure 7 Some Rules of ULOHOL.

Finally, we design a framework for unary predicates, called ULOHOL, similarly to UOHOL. Remarkably, ULOHOL allows us to give a satisfactory rule for conditionals. Most of the defining rules of ULOHOL are straightforward (but notice that, contrary to UOHOL, we have to keep track of set variables). We give some of the most relevant ones in Figure 7. Finally, we show the equivalence with the judgment system of LOHOL.

▶ **Theorem 14** (Equivalence between ULOHOL and LOHOL). *For every sequence of set variables $\tilde{X}$, context Γ, type τ, term t, set of assertions Ψ and assertion ϕ, $\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} t : \tau\ \{\phi\}$ is derivable if and only if $\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} \phi[t/\boldsymbol{r}]$ is.*

Proof. The important implication is the right-to-left one. The proof proceeds by induction on $\tilde{X} \mid \Gamma \mid \Psi \vdash^{\Theta} t : \tau\ \{\phi\}$ following the usual proof in UHOL. For the last rule, the desired implication follows from $[t =_{\mathtt{Bool}} \mathsf{ff}](t =_{\Theta \to \mathtt{Bool}} \mathsf{ff})$, as pointwise equality corresponds to functional equality, and so we can substitute t by ff in the formula $[t = \mathsf{ff}]\phi_0$, which is semantically equivalent to replacing $\mathsf{if}\ t\ \mathsf{then}\ u_1\ \mathsf{else}\ u_0$ by u_0 (and similarly for ϕ_1). ◀

$$\models \forall X, Y.[X]\mathcal{O}(Y) \Leftrightarrow \mathcal{O}(Y) \qquad (ORestr)$$
$$\models \mathcal{O}(\ \{\Theta \mid (x < \underline{c}) = \mathsf{tt}\}) \qquad (ORight)$$
$$\models \mathcal{O}(\ \{\Theta \mid (x > \underline{c}) = \mathsf{tt}\}) \qquad (OLeft)$$
$$\models \forall X, Y.(\mathcal{O}(X) \wedge \mathcal{O}(Y)) \Rightarrow (\mathcal{O}(X \cup Y) \wedge \mathcal{O}(X \cap Y)) \qquad (OStab)$$
$$\models \forall X, Y, z.([X]\mathcal{C}(z) \wedge (Y \subseteq X)) \Rightarrow [Y]\mathcal{C}(z) \qquad (CSubsets)$$
$$\models \forall X, Y, z.(\mathcal{O}(X) \wedge \mathcal{O}(Y) \wedge [X]\mathcal{C}(z) \wedge [Y]\mathcal{C}(z)) \Rightarrow [X \cup Y]\mathcal{C}(z) \qquad (COpenUnion)$$

Figure 8 Defining axioms of the predicates $\mathcal{O}$ and $\mathcal{C}$.

5.1 LOHOL at Work

Having introduced L(U)OHOL formally, let us now see the benefits of its characteristic features – namely openness, locality, and a novel rule for conditionals – on a nontrivial example: local continuity of real-valued functions. More specifically, we take inspiration from the work by Mazza and Pagani [16] (where it is shown that the points of error of AD on a PCF language form a set of measure zero) and show how our system can be used to prove *local continuity* on *open sets* of $\mathbb{R}$, this way getting rid of those discontinuity points that may occur due to the presence of conditionals.

Let us consider the context $\Theta = x : \mathtt{Real}$ and let t be the term $\mathsf{if}\ (x < 3)\ \mathsf{then}\ u_1\ \mathsf{else}\ u_0$. We want to show that t is continuous on a set that excludes: *(i)* the possible points of discontinuity of u_1; *(ii)* the possible points of discontinuity of u_0; *(iii)* the point $x = 3$, for which continuity is not assured. To do that, we need predicates and axioms for open sets, including stability of continuity by union of open sets. Consequently, we introduce the predicates $\mathcal{O}(S)$, of arity $(1;\cdot)$, and $\mathcal{C}(t)$, of arity $(0;\mathtt{Real})$. Our goal is then to prove a formula of the shape $(\mathcal{O}(S) \wedge [S]\mathcal{C}(t))$, meaning that t is locally continuous on the open set $S \subseteq \mathbb{R}$. The defining axioms of $\mathcal{O}$ and $\mathcal{C}$ are given in Figure 8, where we assume common predicates for sets (such as $\subseteq$) as given.

The first axiom states that restriction has no effect on $\mathcal{O}$, as there are simply no function to restrict, whereas the remaining axioms describe standard properties of open sets and continuity. Notice that we use the formula $(x \leq 3 = \mathsf{tt})$ instead of $x \leq 3$. This is because we shall construct open sets using the boolean $\Lambda_{\mathcal{C},\mathcal{F}}$ function $<$, and thus it is natural to consider the predicate $(x \leq 3 = \mathsf{tt})$. Nonetheless, it is also possible to use the formula $x \leq 3$ as well, although we would need to add an axiom linking the function $< \in \mathcal{F}$ and the set predicate $<$ of LOHOL.

Let us now consider the formulas ϕ_i, with $i \in \{0, 1\}$, defined by

$$\phi_i = [\ \{\Theta \mid (x < 3) = \mathsf{tt}\}](\mathcal{O}(S_i) \wedge [S_i]\mathcal{C}(\boldsymbol{r}))$$

and the judgments $\cdot \mid \cdot \vdash^{\Theta} u_i : \mathtt{Real}\ \{\phi_i\}$ which, altogether, give derivations of the form

$$\frac{\cdot \mid \cdot \vdash^{\Theta} (x < 3) : \mathtt{Bool}\ \{\phi_b\} \qquad \cdot \mid \cdot \vdash^{\Theta} u_0 : \mathtt{Real}\ \{\phi_0\} \qquad \cdot \mid \cdot \vdash^{\Theta} u_1 : \mathtt{Real}\ \{\phi_1\}}{\cdot \mid \cdot \vdash^{\Theta} \mathsf{if}\ (x < 3)\ \mathsf{then}\ u_1\ \mathsf{else}\ u_0 : \mathtt{Real}\ \{\phi_b \wedge \phi_1 \wedge \phi_0\}}$$

The target formula of this proof is not very satisfactory, as it forces us to use the implication rule. This, after all, is not surprising: conditionals constitute a complex case to prove continuity, and a generic rule for conditional cannot be expected to work directly on such a complex case. We overcome this problem by deriving a new rule for the conditional taking advantages of the axiomatic theory of $\mathcal{O}$ and $\mathcal{C}$.

▶ **Proposition 15.** *The following rule is derivable.*

$$\frac{\Gamma \mid \Psi \vdash^{\Theta} b : \mathtt{Bool}\ \{\mathcal{O}(R_0) \wedge \mathcal{O}(R_1) \wedge (R_0 \subseteq \{\Theta \mid b = \mathtt{ff}\}) \wedge (R_1 \subseteq \{\Theta \mid b = \mathtt{tt}\})\} \quad (\forall i \in \{0,1\})\ \Gamma \mid \Psi \vdash^{\Theta} t_i : \mathtt{Real}\ \{(\mathcal{O}(S_i) \wedge [S_i]\mathcal{C}(\boldsymbol{r}))\}}{\Gamma \mid \Psi \vdash^{\Theta} \mathsf{if}\ b\ \mathsf{then}\ t_1\ \mathsf{else}\ t_0 : \mathtt{Real}\ \{\mathcal{O}((S_0 \cap R_0) \cup (S_1 \cap R_1)) \wedge [(S_0 \cap R_0) \cup (S_1 \cap R_1)]\mathcal{C}(\boldsymbol{r})\}}$$

Proof. First, we show $(\mathcal{O}(S_0) \wedge [S_0]\mathcal{C}(\boldsymbol{r})) \Rightarrow [\ \{\Theta \mid b = \mathtt{tt}\}](\mathcal{O}(S_0) \wedge [S_0]\mathcal{C}(\boldsymbol{r}))$, for any term b. Recall that the constructor $[S]$ on the top level should always be understood as pushed in front of atomic formulas, so that what we actually need to prove is

$$(\mathcal{O}(S_0) \wedge [S_0]\mathcal{C}(\boldsymbol{r})) \Rightarrow [\ \{\Theta \mid b = \mathtt{tt}\}]\mathcal{O}(S_0) \wedge [S_0 \cap \{\Theta \mid b = \mathtt{tt}\}]\mathcal{C}(\boldsymbol{r}))$$

This follows from the axioms (Orestr) and (CSubsets), using the axiom $X \cap Y \subseteq X$ (which we assume as a defining axiom of the predicate $\subseteq$). To conclude, we prove

$$(\phi_b \wedge \phi_1 \wedge \phi_0) \Rightarrow \mathcal{O}((S_0 \cap R_0) \cup (S_1 \cap R_1) \wedge [(S_0 \cap R_0) \cup (S_1 \cap R_1)]\mathcal{C}(\boldsymbol{r}),$$

where ϕ_b is the formula $\mathcal{O}(R_0) \wedge \mathcal{O}(R_1) \wedge (R_0 \subseteq \{\Theta \mid b = \mathtt{ff}\}) \wedge (R_1 \subseteq \{\Theta \mid b = \mathtt{tt}\})$. This is a consequence of the axioms in Figure 8: openness of $(S_0 \cap R_0) \cup (S_1 \cap R_1)$ comes from (Ostab), whereas continuity of $\boldsymbol{r}$ follows from (CSubsets) and (COpenUnion). ◀

Armed with Proposition 15, let us come back to our main example. Assuming an appropriate axiomatization of the relations $<, >, \leq, \geq$, we define $R_1 = \{\Theta \mid (x < 3) = \mathtt{tt}\}$, $R_0 = \{\Theta \mid (x > 3) = \mathtt{tt}\}$, and derive the judgment

$$\cdot \mid \cdot \vdash^{\Theta} (x < 3) : \mathtt{Bool}\ \{\mathcal{O}(R_0) \wedge \mathcal{O}(R_1) \wedge (R_0 \subseteq \{\Theta \mid (x < 3) = \mathtt{ff}\}) \wedge (R_1 \subseteq \{\Theta \mid (x < 3) = \mathtt{tt}\})\}.$$

Then, using the rule for conditionals, we infer the possible discontinuity points of t, as well as those of u_1 (resp. u_0) smaller (resp. greater) than 3, and the point $\boldsymbol{x} = 3$ itself.

6 Conclusion

We have introduced several systems of *open* higher-order logic, i.e. extensions of HOL capable of dealing with first-order properties natively. Even if our logics do not increase the expressive power of HOL, they considerably improve its effectiveness, allowing for compositional proofs of nontrivial program properties, such as (local) continuity and differentiability. We have tested our formalism on several nontrivial examples, obtaining compositional proofs of state-of-the-art results (such as correctness of automatic differentiation) in a formal framework.

The number of extensions of OHOL defined and the heterogeneity of the examples analyzed suggest that OHOL (and variations thereof) may play an important role in the formal analysis of higher-order programming languages. Prompted by that observation, the authors plan to investigate applications of the logical systems to programs with effects (e.g. probabilistic programs) and to intensional program analyses (such as quantitative reasoning and program complexity).

References

1 Alejandro Aguirre, Gilles Barthe, Lars Birkedal, Aleš Bizjak, Marco Gaboardi, and Deepak Garg. Relational reasoning for markov chains in a probabilistic guarded lambda calculus. In *Proc. of ESOP 2018*, volume 10801 of *LNCS*, pages 214–241. Springer, 2018. doi:10.1007/978-3-319-89884-1_8.

2 Alejandro Aguirre, Gilles Barthe, Marco Gaboardi, Deepak Garg, and Pierre-Yves Strub. A relational logic for higher-order programs. *Proc. ACM Program. Lang.*, 1(ICFP):21:1–21:29, 2017. doi:10.1145/3110265.

3 Gilles Barthe, Raphaëlle Crubillé, Ugo Dal Lago, and Francesco Gavazzo. On the versatility of open logical relations: Continuity, automatic differentiation, and a containment theorem. In *Proc. of ESOP 2020*, volume 12075 of *LNCS*. Springer, 2020. doi:10.1007/978-3-030-44914-8_3.

4 Edwin Brady. Idris, a general-purpose dependently typed programming language: Design and implementation. *Journal of Functional Programming*, 23(5):552–593, 2013. doi:10.1017/S095679681300018X.

5 Stephen A. Cook. Soundness and completeness of an axiom system for program verification. *SIAM Journal on Computing*, 7(1):70–90, 1978. doi:10.1137/0207005.

6 Mario Coppo and Mariangiola Dezani-Ciancaglini. A new type assignment for lambda-terms. *Archiv fur mathematische Logik und Grundlagenforschung*, 19(1):139–156, 1978. doi:10.1007/BF02011875.

7 Ugo Dal Lago and Marco Gaboardi. Linear dependent types and relative completeness. In *Proc. of LICS 2011*, pages 133–142, 2011. doi:10.1109/LICS.2011.22.

8 Ugo Dal Lago, Francesco Gavazzo, and Alexis Ghyselen. Open higher-order logic (long version), 2022. arXiv:2211.06671.

9 Rowan Davies and Frank Pfenning. Intersection types and computational effects. In *Proc. of ICFP 2000*, pages 198–208, 2000. doi:10.1145/351240.351259.

10 Robert W Floyd. Assigning meanings to programs. In *Program Verification*, pages 65–81. Springer, 1993.

11 Alejandro Aguirre Galindo. *Relational Logics for Higher-Order Effectful Programs*. Ph.D Thesis, 2020.

12 C. A. R. Hoare. An axiomatic basis for computer programming. *Commun. ACM*, 12(10):576–580, 1969. doi:10.1145/363235.363259.

13 Martin Hofmann. Linear types and non-size-increasing polynomial time computation. *Information and Computation*, 183(1):57–85, 2003. doi:10.1016/S0890-5401(03)00009-9.

14 Mathieu Huot, Sam Staton, and Matthijs Vákár. Correctness of automatic differentiation via diffeologies and categorical gluing. In *Proc. of FoSSACS 2020*, volume 12077 of *LNCS*, pages 319–338. Springer, 2020.

15 Achim Jung and Jerzy Tiuryn. A new characterization of lambda definability. In *Proc. of TLCA 1993*, volume 664 of *LNCS*, pages 245–257. Springer, 1993. doi:10.1007/BFb0037110.

16 Damiano Mazza and Michele Pagani. Automatic differentiation in PCF. *Proc. ACM Program. Lang.*, 5(POPL):1–27, 2021. doi:10.1145/3434309.

17 Gordon Plotkin. Lambda-definability and logical relations. Memorandum SAI-RM-4, University of Edinburgh, 1973.

18 Francois Pottier. A simple view of type-secure information flow in the pi-calculus. In *Proc. of CSFW 2002*, pages 320–330, 2002. doi:10.1109/CSFW.2002.1021826.

19 Tetsuya Sato, Alejandro Aguirre, Gilles Barthe, Marco Gaboardi, Deepak Garg, and Justin Hsu. Formal verification of higher-order probabilistic programs: Reasoning about approximation, convergence, bayesian inference, and optimization. *Proc. ACM Program. Lang.*, 3(POPL):38:1–38:30, 2019. doi:10.1145/3290351.

20 Amir Shaikhha, Andrew Fitzgibbon, Dimitrios Vytiniotis, and Simon Peyton Jones. Efficient differentiable programming in a functional array-processing language. *Proc. ACM Program. Lang.*, 3(ICFP):97:1–97:30, 2019. doi:10.1145/3341701.

21 Richard Statman. Logical relations and the typed lambda-calculus. *Information and Control*, 65(2):85–97, 1985. doi:10.1016/S0019-9958(85)80001-2.

22 Sam Staton, Frank Wood, Hongseok Yang, Chris Heunen, and Ohad Kammar. Semantics for probabilistic programming: higher-order functions, continuous distributions, and soft constraints. In *Proc. of LICS 2016*, pages 525–534, 2016. doi:10.1145/2933575.2935313.