Dependently-Typed Higher-Order Logic

COLIN ROTHGANG, Imdea Software Institute, Spain and Universidad Politécnica de Madrid, Spain

FLORIAN RABE, University of Erlangen-Nürnberg, Germany

CHRISTOPH BENZMÜLLER, University of Bamberg, Germany and Freie Universität Berlin, Germany

Higher-order logic HOL offers a very simple syntax and semantics for knowledge representation and reasoning in various particular domains, including in particular representing and reasoning about typed data structures. But its type system lacks advanced features where types may depend on terms. Dependent type theory offers such a rich type system, but has rather substantial conceptual differences to HOL, as well as comparatively poor proof automation support.

We introduce a dependently-typed extension DHOL of HOL that retains the style and conceptual framework of HOL. Moreover, we build a translation from DHOL to HOL and implement it as a preprocessor to HOL theorem provers able to parse TPTP, thereby making all such provers able to run on DHOL problems.

CCS Concepts: • **Theory of computation** → **Higher order logic**; *Type theory*; *Automated reasoning*.

Additional Key Words and Phrases: higher-order logic, dependent type-theory, automated theorem proving

ACM Reference Format:

Colin Rothgang, Florian Rabe, and Christoph Benzmüller. 2025. Dependently-Typed Higher-Order Logic. 1, 1 (July 2025), 61 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction and Related Work

Motivation. Theorem proving in higher-order logic (HOL) [10, 17] has been a long-running research strand producing multiple mature interactive provers [16, 19, 22] and automated provers [3, 9, 35]. Similarly, many, mostly interactive, theorem provers are available for various versions of dependent type theory (DTT) [12, 14, 23, 25]. However, it is (maybe surprisingly) difficult to develop languages and theorem provers for dependently-typed higher-order logic, which we dub DHOL in this paper.

A preliminary version of this work appeared in the proceedings of CADE 2023 [32], which in turn is based on the master thesis [33] of the first author.
Authors' Contact Information: Colin Rothgang, Imdea Software Institute, Madrid, Spain, colin.rothgang@imdea.org and Universidad Politécnica de Madrid, Madrid, Spain; Florian Rabe, University of Erlangen-Nürnberg, Erlangen, Germany; Christoph Benzmüller, University of Bamberg, Bamberg, Germany and Freie Universität Berlin, Berlin, Germany.

Manuscript submitted to ACM

Concretely, we use HOL to refer to the version of Church's *simply*-typed λ-calculus with a base type bool for Booleans, simple function types $A \to B$, and primitive equality $=_A : A \to A \to \text{bool}$ (for all types A); cf. Andrews' Q_0[2, 3]. This already suffices to define the usual logical quantifiers and connectives. We additionally also assume a choice operator. Note that typical descriptions of HOL also assume an axiom of infinity, which makes HOL appropriate as a foundational system on top of which knowledge is formalized by *conservative extensions* as done in the large libraries of the HOL family proof assistants [16, 19, 22]. We omit this here for simplicity, but our developments carry over to the extended language (we just have to add a corresponding axiom to DHOL).

Intuitively, it is straightforward to develop DHOL in the same way on top of the *dependently*-typed λ-calculus, which uses a dependent function type $\Pi x{:}A.\ B$ instead of $\to$. However, several subtleties arise that seem deceptively minor at first but end up presenting fundamental theoretical issues. They come up already in the elementary expression $x =_A y \Rightarrow f(x) =_{B(x)} f(y)$ for some dependent function $f : \Pi x{:}A.\ B(x)$. We discuss these in the sequel.

Equality of Types and Decidability. The equality $f(x) =_{B(x)} f(y)$ (or the analogous equality over $B(y)$) is not obviously well-typed because the terms $f(x) : B(x)$ and $f(y) : B(y)$ do not have the same type. This is a fundamental issue of dependent types, and languages split into very different language families depending on how this issue is resolved.

In HOL, term equality $=_A$ is a bool-valued connective. But type equality $\equiv$ is a judgment at the same level as the typing judgment $t : A$. Thus, users of HOL can add axioms that affect the equality of terms, which in particular makes term equality undecidable, but cannot do the same for equality of types. This is critical for keeping HOL type checking decidable. Adding dependent types to HOL would blur this strict distinction between types and terms and is therefore usually avoided.

In most incarnations of DTT on the other hand, term equality is split into two versions. This approach goes back to Martin-Löf type theory [20] and the calculus of constructions [13] and uses two separate equality relations, a decidable judgment-level equality for use in the type-checker and a stronger undecidable one that users can customize via axioms. This approach also favors the use of propositions-as-types and deemphasizes or omits entirely a type of classical Booleans. This approach has been followed in various interactive provers such as Rocq, Agda and Lean [12, 14, 23].

Finally, a third choice is to add a rule that the equality $s =_A y$ between terms carries over to an equality $B(x) \equiv B(y)$ between types. This makes type-checking undecidable and is therefore usually considered as an option only with extreme caution. Nonetheless some interactive provers have successfully followed this approach. In particular, our DHOL can be seen as a fragment of PVS [24]. While the latter is a much richer language, adding, e.g., polymorphism, induction, records, and refinements, that aims at being a general-purpose interactive prover, our goal is to stay as close as possible to languages for which automated proving support is available. NuPRL [11] similarly develops a very expressive type theory that essentially subsumes DHOL, and our treatment of the semantics goes back to [1]. Mizar [4], while based on first-order set theory, allows for creating dependent types, whose semantics is given by unary predicates on sets. Type-checking at these types shows a similar behavior.

In any case, once this design decision has been made, natural system evolution and optimization tend to produce wildly different languages, and that is one reason for the interoperability gap that we see between proof assistants today. Our motivation here is to follow the third approach while staying as close as possible to the HOL approach and to automated theorem provers (ATPs) for HOL. Thus, we use a single equality relation and classical Booleans. This is arguably very

intuitive to users, especially to those outside the DTT community such as typical HOL users or mathematicians, and it is certainly much closer to the logics of the strongest available ATP systems.

This means that we have to pay the price that type equality and type-checking become dependent on user-declared term equality axioms and thus undecidable. The current paper was prompted by the observation that this price may be acceptable for two reasons:

(1) If our ultimate interest is theorem proving, undecidability comes up anyway. Indeed, it is plausible that the cost of showing the well-typedness of a conjecture will often be negligible compared to the cost of proving it.
(2) As the strength of ATPs for HOL increases, the practical drawbacks of undecidable type-checking decrease, which indicates revisiting the trade-off from time to time. Indeed, if we position DHOL close to an existing HOL ATP, it is plausible that the price will, in practice, be affordable.
(3) ATP for HOL is anyway already facing an undecidable sub- or side-problem it has to deal with: higher-order (pre-)unification. The Leo-II prover, for example, did therefore explore solutions for a tight interleaving of proof search and search for higher-order pre-unifiers; see [8].

Dependent Connectives. Moreover, even if we add a rule like "if $\vdash x =_A y$, then $\vdash B(x) \equiv B(y)$" to our type system, the above expression is still not well-typed: Above, the equality $x =_A y$ on the left of $\Rightarrow$ is needed to show the well-typedness of the equality $f(x) =_{B(x)} f(y)$ on the right. This intertwines theorem proving and type-checking even further. Concretely, we need a *dependent implication*, where the first argument is assumed to hold while checking the well-typedness of the second one. Formally, this means that to show $\vdash F \Rightarrow G : \mathsf{bool}$, we require $\vdash F : \mathsf{bool}$ and $F \vdash G : \mathsf{bool}$. Similarly, we need a dependent conjunction. And if we are classical, we may also opt to add a dependent disjunction $F \vee G$, where $\neg F$ is assumed in G.

Naturally, dependent conjunction and disjunction are not commutative anymore. This may feel disruptive, but similar behavior of connectives is well-known from short-circuit evaluation in programming languages.

The meta-logical properties of dependent connectives are straightforward. However, interestingly, these connectives can no longer be defined from just equality. They can be defined from each other if we add any one of them as a primitive. At least one of them (we will choose dependent implication) must be taken as an additional primitive in DHOL along with $=_A$.

Order of Declarations. Finally, the above generalizations require a notion of DHOL-contexts that is more complex than for HOL. HOL-contexts can be stratified into (a) a set of variable declarations $x_i : A_i$, and (b) a set of (named) logical assumptions $x_i^* \triangleright F_i$ possibly using the variables x_i. Moreover, the former are often not explicitly listed at all and instead inferred from the remainder of the sequent.

But in DHOL, the well-formedness of an A_i may now depend on previous logical assumptions. To linearize this inter-dependency, DHOL contexts must consist of a single list alternating between variable declarations and assumptions.

Contribution. Our contribution is twofold. Firstly, we introduce a new logic DHOL designed along the lines described above. Secondly, we develop and implement a sound and complete translation of DHOL into HOL. This setup allows the use of DHOL as the expressive user-facing language and HOL as the internal theorem-proving language. We position our implementation close to an existing HOL ATP, namely the LEO-III system. From the LEO-III perspective, DHOL serves as an additional input language that is translated into HOL by an external logic embedding tool [34, 36] in the

LEO-III ecosystem. This embedding tool also allows translating DHOL input into HOL problems accepted by many other HOL provers. Because LEO-III already supports such embeddings and because the TPTP syntax [37, 38] foresees the use of dependent types in ATPs and provides syntax for them (albeit without a normative semantics), we were able to implement the translation with no disruptions to existing workflows. We propose our representation of DHOL as a new TPTP dialect to be endorsed by the TPTP standard. Our work was well received in the CADE community and has already been taken up in subsequent publications such as [21], which was awarded the prize for the best student paper at IJCAR 2024.

The general idea of our translation of dependent into simple type theory is not new [5]. In that work, Martin-Löf-style dependent type theory is translated into Gordon's HOL interactive theorem prover (ITP) [16]. This work differs critically from ours because it uses DTT in propositions-as-types style. Our work builds DHOL with classical Booleans and equality predicates, which makes the task of proving the translation sound and complete very different. Moreover, their work targets an interactive prover while ours targets automated ones.

Overview. The present paper is an extended version of [32] containing in particular all proofs. It also polishes and updates the text, but the substance of the technical content is unchanged with the exception that we expanded the section on extensions of DHOL. In Sect. 2 we recap the HOL logic. In Sect. 3 we extend it to DHOL and we define our translation from DHOL to HOL in Sect. 4. In Sect. 5 we prove the completeness and in Sect. 6 the soundness of the translation. In Sect. 7 and Sect. 8 we describe how to use our translation and a HOL ATP to implement a type-checker and a theorem prover for DHOL, respectively. The DHOL TPTP syntax we propose has been officially adopted as a language in the TPTP family [28].

DHOL has received quite some attention since it was introduced. An experimental extension with choice, which misses the statement of all meta-theorems, was published as [30]. The present article also adds a choice operator (aka indefinite description) to HOL and DHOL that corresponds to the one from [30]. Then we give our meta-theorems and their proofs for this extended version of DHOL. Moreover, Sect. 9 sketches a few other extensions of DHOL that are under active development. Of these, an extension with subtyping will appear as [31]. An extension with polymorphism is under review [29].

2 Preliminaries: Higher-Order Logic

Language. Our definitions are standard [2] except that we tweak a few details in order to later present the extension to DHOL more succinctly. Throughout the paper we will mark contexts, theories, terms and types, names of assumptions and axioms as well as metavariables of the grammars in the indicated colors.

We use the following grammar for HOL:

$$
\begin{array}{llll}
T & ::= & \circ \mid T, \mathsf{a} : \mathsf{tp} \mid T, \mathsf{c} : A \mid T, c \triangleright F & \text{theories} \\
\Gamma & ::= & . \mid \Gamma, x : A \mid \Gamma, x \triangleright F & \text{contexts} \\
A, B & ::= & \mathsf{a} \mid A \to B \mid \mathsf{bool} & \text{types} \\
s, t, f, F, G & ::= & \mathsf{c} \mid x \mid \lambda x{:}A.\ t \mid \varepsilon x{:}A.\ F \mid f\ t \mid s =_A t \mid F \Rightarrow G & \text{terms}
\end{array}
$$

A theory T is a list of

- base type declarations $\mathsf{a} : \mathsf{tp}$,

- typed constant declarations $\mathsf{c} : A$, and
- named axioms $c \triangleright F$ asserting the formula F.

The ability to name the axioms is occasionally convenient to refer to them in proofs. Furthermore, TPTP has named axioms and we are interested in encoding HOL in TPTP for proof automation. But we will omit the name if it is irrelevant.

A context Γ has the same form except that no type variables are allowed. We write $\circ$ and . for the empty theory and context, respectively. We use named assumptions and axioms, since we rely on TPTP for proof automation for HOL and in TPTP axioms are also named.

Types A are either user-declared base types a, the built-in base type bool, or function types $A \to B$. Terms are constants c, variables x, λ-abstractions $\lambda x{:}A.\ t$, function applications $f\ t$, choice operators $\varepsilon x{:}A.\ F$, or obtained from the built-in bool-valued connectives $=_A$ and $\Rightarrow$. The notation $E[^x/_t]$ denotes the capture-avoiding substitution of the variable x with the term t within expression E. The notation $t^{\beta\eta}$ denotes the beta-eta normal form of t.

Definition 2.1. As usual [2], equality suffices as a primitive to define all the usual quantifiers and connectives true, false, $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\forall$ and $\exists$:

$$\begin{aligned}
\mathsf{true} &:= (\lambda x{:}\mathsf{bool}.\ x) =_{\mathsf{bool}\to\mathsf{bool}} (\lambda x{:}\mathsf{bool}.\ x)\\
\mathsf{false} &:= (\lambda x{:}\mathsf{bool}.\ x) =_{\mathsf{bool}\to\mathsf{bool}} (\lambda x{:}\mathsf{bool}.\ \mathsf{true})\\
\neg &:= \lambda x{:}\mathsf{bool}.\ x =_{\mathsf{bool}} \mathsf{false}\\
\forall_A &:= \lambda F{:}A \to \mathsf{bool}.\ F =_{A\to\mathsf{bool}} \lambda x{:}A.\ \mathsf{true}\\
\exists_A &:= \lambda F{:}A \to \mathsf{bool}.\ \forall y{:}\mathsf{bool}.\ (\forall_A(\lambda x{:}A.\ F\ x \Rightarrow y)) \Rightarrow y\\
\wedge &:= \lambda x{:}\mathsf{bool}.\ \lambda y{:}\mathsf{bool}.\ \neg(x \Rightarrow \neg y)\\
\vee &:= \lambda x{:}\mathsf{bool}.\ \lambda y{:}\mathsf{bool}.\ \neg x \Rightarrow y
\end{aligned}$$

In the sequel, we adopt the usual notations for all defined connectives.

If we wanted, we could also define implication:

$$\Rightarrow\ := \lambda x, y{:}\mathsf{bool}.\ (x \wedge y) =_{\mathsf{bool}} x$$

We include $\Rightarrow$ in the grammar even though it is definable because it will not be definable anymore in DHOL. See Ex. 3.3.

Remark 1 (Differences from Standard HOL). It is not strictly necessary to use *named* axioms and assumptions, but it makes the extension to DHOL and the proof translation into HOL simpler. In particular, it allows referring to axioms used in proofs more easily.

At this point, it is possible to reorder theories and contexts by ordering all declarations according to base types (in theories), then constants/variables, then axioms. This is because terms cannot occur in type declarations, and the

well-formedness of the type A in a declaration can never depend on an axiom. Those invariants will change when transitioning to DHOL, which is why we already use a grammar in which all declarations can alternate.

Theories and contexts:

$$\frac{}{\vdash \circ\ \mathsf{Thy}}\ \mathsf{thyEmpty} \qquad \frac{\vdash T\ \mathsf{Thy}}{\vdash T,\ \mathsf{a}\ \mathsf{tp}\ \mathsf{Thy}}\ \mathsf{thyType} \qquad \frac{\vdash_T A\ \mathsf{tp}}{\vdash T,\ \mathsf{c} : A\ \mathsf{Thy}}\ \mathsf{thyConst} \qquad \frac{\vdash_T F : \mathsf{bool}}{\vdash T,\ c \triangleright F\ \mathsf{Thy}}\ \mathsf{thyAxiom}$$

$$\frac{\vdash T\ \mathsf{Thy}}{\vdash_T .\ \mathsf{Ctx}}\ \mathsf{ctxEmpty} \qquad \frac{\Gamma \vdash_T A\ \mathsf{tp}}{\vdash_T \Gamma,\ x : A\ \mathsf{Ctx}}\ \mathsf{ctxVar} \qquad \frac{\Gamma \vdash_T F : \mathsf{bool}}{\vdash_T \Gamma,\ x \triangleright F\ \mathsf{Ctx}}\ \mathsf{ctxAssume}$$

Lookup in theory and context:

$$\frac{\mathsf{a} : \mathsf{tp}\ \mathsf{in}\ T \qquad \vdash_T \Gamma\ \mathsf{Ctx}}{\Gamma \vdash_T \mathsf{a}\ \mathsf{tp}}\ \mathsf{type} \qquad \frac{\mathsf{c} : A'\ \mathsf{in}\ T \qquad \Gamma \vdash_T A' \equiv A}{\Gamma \vdash_T \mathsf{c} : A}\ \mathsf{const} \qquad \frac{c \triangleright F\ \mathsf{in}\ T \qquad \vdash_T \Gamma\ \mathsf{Ctx}}{\Gamma \vdash_T F}\ \mathsf{axiom}$$

$$\frac{x : A'\ \mathsf{in}\ \Gamma \qquad \Gamma \vdash_T A' \equiv A}{\Gamma \vdash_T x : A}\ \mathsf{var} \qquad \frac{x \triangleright F\ \mathsf{in}\ \Gamma \qquad \vdash_T \Gamma\ \mathsf{Ctx}}{\Gamma \vdash_T F}\ \mathsf{assume}$$

Well-formedness and equality of types:

$$\frac{\vdash_T \Gamma\ \mathsf{Ctx}}{\Gamma \vdash_T \mathsf{bool}\ \mathsf{tp}}\ \mathsf{bool} \qquad \frac{\Gamma \vdash_T A\ \mathsf{tp} \qquad \Gamma \vdash_T B\ \mathsf{tp}}{\Gamma \vdash_T A \to B\ \mathsf{tp}}\to \qquad \frac{\vdash_T \Gamma\ \mathsf{Ctx}}{\Gamma \vdash_T \mathsf{bool} \equiv \mathsf{bool}}\ \mathsf{congBool}$$

$$\frac{\vdash_T \Gamma\ \mathsf{Ctx} \qquad \mathsf{a} : \mathsf{tp}\ \mathsf{in}\ T}{\Gamma \vdash_T \mathsf{a} \equiv \mathsf{a}}\ \mathsf{congBase} \qquad \frac{\Gamma \vdash_T A \equiv A' \qquad \Gamma \vdash_T B \equiv B'}{\Gamma \vdash_T A \to B \equiv A' \to B'}\ \mathsf{cong}\to$$

Typing:

$$\frac{\Gamma,\ x : A \vdash_T t : B}{\Gamma \vdash_T (\lambda x{:}A.\ t) : A \to B}\lambda \qquad \frac{\Gamma,\ x : A \vdash_T F : \mathsf{bool}}{\Gamma \vdash_T (\varepsilon x{:}A.\ F) : A}\varepsilon$$

$$\frac{\Gamma \vdash_T f : A \to B \qquad \Gamma \vdash_T t : A}{\Gamma \vdash_T f\ t : B}\ \mathsf{appl} \qquad \frac{\Gamma \vdash_T s : A \qquad \Gamma \vdash_T t : A}{\Gamma \vdash_T s =_A t : \mathsf{bool}}=$$

Term equality; congruence for λs (ξ rule) and application, reflexivity, symmetry, β, η:

$$\frac{\Gamma \vdash_T A \equiv A' \qquad \Gamma,\ x : A \vdash_T t =_B t'}{\Gamma \vdash_T \lambda x{:}A.\ t =_{A\to B} \lambda x{:}A'.\ t'}\ \mathsf{cong}\lambda \qquad \frac{\Gamma \vdash_T t =_A t' \qquad \Gamma \vdash_T f =_{A\to B} f'}{\Gamma \vdash_T f\ t =_B f'\ t'}\ \mathsf{congAppl}$$

$$\frac{\Gamma \vdash_T t : A}{\Gamma \vdash_T t =_A t}\ \mathsf{refl} \qquad \frac{\Gamma \vdash_T t =_A s}{\Gamma \vdash_T s =_A t}\ \mathsf{sym} \qquad \frac{\Gamma \vdash_T (\lambda x{:}A.\ s)\ t : B}{\Gamma \vdash_T (\lambda x{:}A.\ s)\ t =_B s[x/t]}\beta \qquad \frac{\Gamma \vdash_T t : A \to B \qquad x \text{ not in } \Gamma}{\Gamma \vdash_T t =_{A\to B} \lambda x{:}A.\ t\ x}\eta$$

Rules for implication:

$$\frac{\Gamma \vdash_T F : \mathsf{bool} \qquad \Gamma \vdash_T G : \mathsf{bool}}{\Gamma \vdash_T F \Rightarrow G : \mathsf{bool}}\Rightarrow \qquad \frac{\Gamma \vdash_T F : \mathsf{bool} \qquad \Gamma,\ x \triangleright F \vdash_T G}{\Gamma \vdash_T F \Rightarrow G}\Rightarrow\mathrm{I} \qquad \frac{\Gamma \vdash_T F \Rightarrow G \qquad \Gamma \vdash_T F}{\Gamma \vdash_T G}\Rightarrow\mathrm{E}$$

Congruence for validity, Boolean extensionality, choice and non-emptiness of types:

$$\frac{\Gamma \vdash_T F =_{\mathsf{bool}} F' \qquad \Gamma \vdash_T F'}{\Gamma \vdash_T F}\ \mathsf{cong}\vdash \qquad \frac{\Gamma \vdash_T p\ \mathsf{true} \qquad \Gamma \vdash_T p\ \mathsf{false}}{\Gamma, x : \mathsf{bool} \vdash_T p\ x}\ \mathsf{boolExt}$$

$$\frac{\Gamma,\ x : A \vdash_T F : \mathsf{bool} \qquad \Gamma \vdash_T \exists x{:}A.\ F}{\Gamma \vdash_T F[x/(\varepsilon x{:}A.\ F)]}\varepsilon\mathrm{E} \qquad \frac{\Gamma \vdash_T F : \mathsf{bool} \qquad \Gamma, x : A \vdash_T F}{\Gamma \vdash_T F}\ \mathsf{nonEmpty}$$

Fig. 1. HOL Rules

Proof System. The type and proof system uses the judgments given below. Note that (as usual) we use a meta-level judgment for the equality of types because $\equiv$ is not a bool-valued connective. On the contrary, the equality of terms

$\vdash s =_A t$ is a special case of the validity judgment $\vdash F$. In HOL, the judgment $\equiv$ is trivial and holds only if types are identical. But we include it here already because it will become non-trivial in DHOL.

Name	Judgment	Intuition
theories	$\vdash T$ Thy	T is well-formed theory
contexts	$\vdash_T \Gamma$ Ctx	Γ is well-formed context
types	$\Gamma \vdash_T A$ tp	A is well-formed type
typing	$\Gamma \vdash_T t : A$	t is well-formed term of well-formed type A
validity	$\Gamma \vdash_T F$	well-formed Boolean F is derivable
equality of types	$\Gamma \vdash_T A \equiv B$	well-formed types A and B are equal

The rules are given in Figure 1. We assume that all names in a theory or a context are unique without making that explicit in the rules.

Remark 2 (Empty Types). HOL types are commonly-assumed to be non-empty, which is why we include the rule (nonEmpty). This rule is special because we will have to drop it when extending to DHOL. But we keep it here for the variant of HOL that will serve as the target of our translation because that is how HOL is commonly implemented in ATPs.

It is easy to show by induction on derivations that α-renaming in a judgment doesn't affect its derivability. Analogously, α-renaming is also valid in DHOL.

The following lemma collects a few routine meta-theorems that we make use of later on:

Lemma 2.2. *Given the inference rules for HOL (cfg. Figure 1), the following rules are admissible:*

$$\frac{\vdash_T \Gamma \text{ Ctx}}{\vdash T \text{ Thy}}\,ctxThy \qquad \frac{\Gamma \vdash_T A \text{ tp}}{\vdash_T \Gamma \text{ Ctx}}\,tpCtx \qquad \frac{\Gamma \vdash_T t : A}{\Gamma \vdash_T A \text{ tp}}\,typingTp \qquad \frac{\Gamma \vdash_T F}{\Gamma \vdash_T F : \text{bool}}\,validTyping$$

$$\frac{\Gamma \vdash_T A \equiv B}{\Gamma \vdash_T A \text{ tp}} \equiv valid \qquad \frac{\mathsf{c} : A \; in \; T}{\Gamma \vdash_T \mathsf{c} : A}\,constS \qquad \frac{x : A \; in \; \Gamma}{\Gamma \vdash_T x : A}\,varS$$

$$\frac{\Gamma \vdash_T A \text{ tp}}{\Gamma \vdash_T A \equiv A} \equiv refl \qquad \frac{\Gamma \vdash_T A \equiv A'}{\Gamma \vdash_T A' \equiv A} \equiv sym \qquad \frac{\Gamma \vdash_T A \equiv A' \quad \Gamma \vdash_T A' \equiv A''}{\Gamma \vdash_T A \equiv A''} \equiv trans$$

$$\frac{\Gamma \vdash_T s =_A t}{\Gamma \vdash_T s : A}\,eqTyping \qquad \frac{\Gamma \vdash_T F \Longrightarrow G}{\Gamma \vdash_T F : \text{bool}}\,implTypingL \qquad \frac{\Gamma \vdash_T F \Longrightarrow G}{\Gamma \vdash_T G : \text{bool}}\,implTypingR$$

$$\frac{\Gamma \vdash_T s : A \quad \Gamma \vdash_T s : A'}{\Gamma \vdash_T A \equiv A'}\,typesUnique \qquad \frac{\Gamma \vdash_T f\ t : B \quad \Gamma \vdash_T f : A \to B}{\Gamma \vdash_T t : A}\,typingWf$$

$$\frac{\Gamma \vdash_T t : A \quad \Gamma \vdash_T f\ t : B}{\Gamma \vdash_T f : A \to B}\,applType \qquad \frac{\Gamma, x : B \vdash_T s : A \quad \Gamma \vdash_T t : B}{\Gamma \vdash_T s[x/t] : A}\,rewriteTyping$$

$$\frac{\Gamma \vdash_T F : \text{bool} \quad \Gamma \vdash_T G}{\Gamma, ass \triangleright F \vdash_T G}\,as\vdash \qquad \frac{\Gamma \vdash_T A \text{ tp} \quad \Gamma \vdash_T J \qquad for\ any\ J}{\Gamma, x : A \vdash_T J}\,var\vdash$$

$$\frac{\Gamma, x : A \vdash_T F : \text{bool}}{\Gamma \vdash_T \forall x{:}A.\ F : \text{bool}}\forall \qquad \frac{\Gamma, x : A \vdash_T F}{\Gamma \vdash_T \forall x{:}A.\ F}\forall I \qquad \frac{\Gamma \vdash_T \forall x{:}A.\ F \quad \Gamma \vdash_T t : A}{\Gamma \vdash_T F[x/t]}\forall E$$

$$\frac{\Gamma \ \mathsf{Ctx} \quad F \ in \ \Gamma}{\Gamma \vdash_T F : \mathsf{bool}}\ asTyped \qquad \frac{\Gamma \vdash_T t =_A t' \quad \Gamma \vdash_T A \equiv A' \quad \Gamma \vdash_T t : A'}{\Gamma \vdash_T t' : A'}\ cong:$$

$$\frac{\Gamma \vdash_T F =_{\mathsf{bool}} \mathsf{true}}{\Gamma \vdash_T F} = \mathsf{true} \qquad \frac{\Gamma \vdash_T F}{\Gamma \vdash_T F =_{\mathsf{bool}} \mathsf{true}}\ \mathsf{true} = \qquad \frac{\Gamma, ass_F \vdash_T G \quad \Gamma, ass_G \vdash_T F}{\Gamma \vdash_T F =_{\mathsf{bool}} G}\ propExt$$

$$\frac{\Gamma, x : A \vdash_T f\ x =_B f'\ x}{\Gamma \vdash_T f =_{A \to B} f'}\ extensionality$$

$$\frac{\Gamma \vdash_T s =_A t \quad \Gamma \vdash_T t =_A u}{\Gamma \vdash_T s =_A u}\ trans \qquad \frac{\Gamma \vdash_T s =_A s' \quad \Gamma \vdash_T t =_A t'}{\Gamma \vdash_T (s =_A t) =_{\mathsf{bool}} (s' =_A t')}\ cong =$$

$$\frac{\Gamma \vdash_T A \equiv A' \quad \Gamma, x : A \vdash_T F =_{\mathsf{bool}} F'}{\Gamma \vdash_T \forall x{:}A.\ F =_{\mathsf{bool}} \forall x{:}A'.\ F'}\ cong\forall \qquad \frac{\Gamma \vdash_T F =_{\mathsf{bool}} F' \quad \Gamma \vdash_T G =_{\mathsf{bool}} G'}{\Gamma \vdash_T F \Rightarrow G =_{\mathsf{bool}} F' \Rightarrow G'}\ cong \Rightarrow$$

$$\frac{\Gamma, x : A \vdash_T F \Rightarrow G}{\Gamma \vdash_T \forall x{:}A.\ F \Rightarrow \forall x{:}A.\ G}\ \forall \Rightarrow \qquad \frac{\Gamma \vdash_T G \Rightarrow G' \quad \Gamma \vdash_T F' \Rightarrow F}{\Gamma \vdash_T (F \Rightarrow G) \Rightarrow (F' \Rightarrow G')}\ \Rightarrow Funct$$

$$\frac{\Gamma \vdash_T F =_{\mathsf{bool}} F' \quad \Gamma \vdash_T F}{\Gamma \vdash_T F'} \vdash\ cong \qquad \frac{\Gamma \vdash_T F[x/t] \quad \Gamma \vdash_T t =_A t' \quad \Gamma, x : A \vdash_T F : \mathsf{bool}}{\Gamma \vdash_T F[x/t']}\ rewrite$$

$$\frac{\Gamma \vdash_T F \wedge G}{\Gamma \vdash_T F} \wedge El \qquad \frac{\Gamma \vdash_T F \wedge G}{\Gamma \vdash_T G} \wedge Er$$

Proof. To show the rules (ctxThy), (tpCtx), (typingTp) and (validTyping) we prove by induction on derivations, the more general statement that each subderivation satisfies the following preconditions:

Judgment	Precondition
$\vdash_T \Gamma$ Ctx	$\vdash T$ Thy
$\Gamma \vdash_T A$ tp	$\vdash_T \Gamma$ Ctx
$\Gamma \vdash_T t : A$	$\Gamma \vdash_T A$ tp
$\Gamma \vdash_T A \equiv B$	$\Gamma \vdash_T A$ tp and $\Gamma \vdash_T B$ tp
$\Gamma \vdash_T F$	$\Gamma \vdash_T F : \mathsf{bool}$

For every hypothesis H of every rule R in the derivation, we assume the respective preconditions hold for the hypotheses to the left of H, and we show that it also holds for H. Finally, we show that the respective precondition holds for the conclusion of R. The induction is straightforward.

The rules ($\equiv$ refl), ($\equiv$ sym), ($\equiv$ trans), ($\equiv$ valid), (eqTyping), (implTypingL), (implTypingR), (typingWf), (applType), (rewriteTyping), (as $\vdash$), (var $\vdash$) and (cong:) can be proven by more straightforward induction on derivations. As an example we will present the proof of rule ($\equiv$ refl): Type well-formedness can be proven by the rules (bool), ($\to$) and (type) in HOL. If the well-formedness is proven by rule (bool) reflexivity follows by rule (congBool). If the well-formedness is proven by rule ($\to$) the reflexivity follows by rule (cong$\to$) and if the well-formedness is proven by rule (type) the reflexivity follows by rule (congBase).

Rule (typesUnique) is provable using a combination of induction on the grammar and induction on derivations. All other rules can then be directly derived. □

Not all of these rules are actually used in this paper (though many are). We nevertheless present all of these rules here for completeness and to guide intuition.

Furthermore, their proofs can be done in a way that ensures they still hold in DHOL. The proof of the (above generalization) of the first four rules is actually given in Theorem 7.1. Only proving rule (cong:) in DHOL becomes harder, the rule (rewriteTyping) needs to be slightly modified (we need to replace A with $A[^x/_t]$ in the conclusion) and the second assumption of the rules (cong $\Rightarrow$) and ($\Rightarrow$Funct) requires F resp. G as assumption.

Lemma 2.3. *Given the inference rules for HOL (cfg. Figure 1), the following rules are admissible:*

$$\frac{\Gamma,\ x : B \vdash_T A\ \mathsf{tp}}{\Gamma \vdash_T A\ \mathsf{tp}}\ varTp \qquad \frac{\Gamma,\ x : B \vdash_T A \equiv B}{\Gamma \vdash_T A \equiv B}\ varEqTp \qquad \frac{\Gamma,\ as \triangleright F \vdash_T A\ \mathsf{tp}}{\Gamma \vdash_T A\ \mathsf{tp}}\ asTp \qquad \frac{\Gamma,\ as \triangleright F \vdash_T A \equiv B}{\Gamma \vdash_T A \equiv B}\ asEqTp$$

$$\frac{\Gamma,\ as \triangleright F \vdash_T t : A}{\Gamma \vdash_T t : A}\ asTyping$$

Proof. The rules are provable by straightforward induction on derivations similar to the rules in Lemma 2.2. □

Note that these rules hold because HOL types are simple. They don't hold for dependent types as present in DHOL.

3 Dependent Types

Language. We have carefully defined HOL in such a way that only a few surgical changes are needed to define DHOL.

The **grammar** is as follows where unchanged parts are shaded out :

T	::=	$\circ \mid T,\ a : (\Pi x{:}A.\ )^* \mathsf{tp} \mid T,\ c \triangleright A \mid T,\ c \triangleright F$	theories
Γ	::=	$. \mid \Gamma, x : A \mid \Gamma, x \triangleright F$	contexts
A, B	::=	$a\ t_1 \dots t_n \mid \Pi x{:}A.\ B \mid \mathsf{bool}$	types
s, t, f, F, G	::=	$c \mid x \mid \lambda x{:}A.\ t \mid \varepsilon x{:}A.\ F \mid f\ t \mid s =_A t \mid F \Rightarrow G$	terms

Concretely, we make only three changes:

- A base type a may now take a list of term arguments, each declared as $\Pi x{:}A.$. This allows introducing *dependent* base types. As usual, we write $A \to B$ for $\Pi x{:}A.\ B$ if x does not occur in B.
- Consequently, a reference to such a base type must now provide a list $t_1 \dots t_n$ of corresponding term arguments.
- Moreover, we switch from simple function types $A \to B$ to dependent function types $\Pi x{:}A.\ B$. As usual, we retain the notation $A \to B$ if x does not occur free in B.

We define the (dependently-typed) connectives exactly as in HOL (see Definition 2.1).

Example 3.1 (Category Theory). As a running example, we formalize the theory of a category in DHOL. It declares the base type `obj` for objects and the dependent base type `mor x y` for morphisms. Further it declares the constants `id` and `comp` for identity and composition, and the neutrality axioms. We omit the associativity axiom for brevity.

`obj` :tp

`mor` :$\Pi x, y$:`obj`. tp

$$\texttt{id} : \Pi x{:}\texttt{obj}.\ \texttt{mor}\ x\ x$$

$$\texttt{comp} : \Pi x, y, z{:}\texttt{obj}.\ \texttt{mor}\ x\ y \to \texttt{mor}\ y\ z \to \texttt{mor}\ x\ z$$

$$\texttt{neutL} \triangleright \forall x, y{:}\texttt{obj}.\ \forall m{:}\texttt{mor}\ x\ y.\ m \circ \texttt{id}_x =_{\texttt{mor}\ x\ y} m$$

$$\texttt{neutR} \triangleright \forall x, y{:}\texttt{obj}.\ \forall m{:}\texttt{mor}\ x\ y.\ \texttt{id}_y \circ m =_{\texttt{mor}\ x\ y} m$$

Here we use a few intuitive notational simplifications such as writing $\Pi x, y{:}\texttt{obj}.$ for binding two variables of the same type. We also use the notations $\texttt{id}_x$ for $\texttt{id}\ x$ and $h \circ g$ for $\texttt{comp}\ \_\ \_\ \_\ g\ h$ where the _ denote inferable arguments of type $\texttt{obj}$.

Proof System. The **judgments** stay the same and we only make minor changes to the **rules**, which we explain in the sequel.

Dependent Functions We replace all rules for $\to$ (namely ($\to$), (cong$\to$), (λ), (appl), (congλ), (congAppl), (η)) with the corresponding ones for Π:

$$\frac{\Gamma \vdash_T A\ \text{tp} \quad \Gamma, x : A \vdash_T B\ \text{tp}}{\Gamma \vdash_T \Pi x{:}A.\ B\ \text{tp}}\Pi \qquad \frac{\Gamma \vdash_T A \equiv A' \quad \Gamma, x : A \vdash_T B \equiv B'}{\Gamma \vdash_T \Pi x{:}A.\ B \equiv \Pi x{:}A'.\ B'}\text{congΠ}$$

$$\frac{\Gamma, x : A \vdash_T t : B}{\Gamma \vdash_T (\lambda x{:}A.\ t) : \Pi x{:}A.\ B}\lambda' \qquad \frac{\Gamma \vdash_T f : \Pi x{:}A.\ B \quad \Gamma \vdash_T t : A}{\Gamma \vdash_T f\ t : B[x/t]}\text{appl'}$$

$$\frac{\Gamma \vdash_T A \equiv A' \quad \Gamma, x : A \vdash_T t =_B t'}{\Gamma \vdash_T \lambda x{:}A.\ t =_{\Pi x:A.\ B} \lambda x{:}A'.\ t'}\text{congλ'} \qquad \frac{\Gamma \vdash_T t =_A t' \quad \Gamma \vdash_T f =_{\Pi x:A.\ B} f'}{\Gamma \vdash_T f\ t =_B f'\ t'}\text{congAppl'}$$

$$\frac{\Gamma \vdash_T t : \Pi x{:}A.\ B \qquad x \text{ not in } \Gamma}{\Gamma \vdash_T t =_{\Pi x:A.\ B} \lambda x{:}A.\ t\ x}\eta\Pi$$

Then we replace the rules for declaring, using, and equating base types with the ones where base types take arguments:

$$\frac{\vdash_T x_1 : A_1, \ldots, x_n : A_n\ \text{Ctx}}{\vdash T,\ a : \Pi x_1{:}A_1.\ \ldots \Pi x_n{:}A_n.\ \text{tp}\ \text{Thy}}\text{thyType'}$$

$$\frac{\vdash_T \Gamma\ \text{Ctx} \quad a : \Pi x_1{:}A_1.\ \ldots \Pi x_n{:}A_n.\ \text{tp in } T \qquad \Gamma \vdash_T t_1 : A_1 \quad \ldots \quad \Gamma \vdash_T t_n : A_n[x_1/t_1] \ldots [x_{n-1}/t_{n-1}]}{\Gamma \vdash_T a\ t_1\ \ldots\ t_n\ \text{tp}}\text{type'}$$

$$\frac{\vdash_T \Gamma\ \text{Ctx} \quad a : \Pi x_1{:}A_1.\ \ldots \Pi x_n{:}A_n.\ \text{tp in } T \qquad \Gamma \vdash_T s_1 =_{A_1} t_1 \quad \ldots \quad \Gamma \vdash_T s_n =_{A_n[x_1/t_1]\ldots[x_{n-1}/t_{n-1}]} t_n}{\Gamma \vdash_T a\ s_1\ \ldots\ s_n \equiv a\ t_1\ \ldots t_n}\text{congBase'}$$

The last of these is the critical rule via which term equality $=_A$ leaks into type equality $\equiv$. Because typing depends on type equality and everything depends on typing, this makes all judgments mutually recursive. And because term equality depends on what axioms are declared in theory and context, all judgments become undecidable. This is also the reason why we must allow any alternation of declarations in theories and contexts: for example, the well-formedness of a type A in a constant declaration may depend on previously stated axioms.

Example 3.2 (Undecidability of Typing). Continuing Ex. 3.1, consider terms $\vdash f : \mathtt{mor}\ u\ v$ and $\vdash g : \mathtt{mor}\ v'\ w$ for terms $\vdash u, v, v', w : \mathtt{obj}$. Then $\vdash g \circ f : \mathtt{mor}\ u\ w$ holds iff $\vdash f : \mathtt{mor}\ u\ v'$, which holds iff $\vdash v =_{\mathtt{obj}} v'$. Depending on the axioms present, this may be arbitrarily difficult to prove.

Empty Types We remove the rule (nonEmpty) to allow for empty types. While (dis)allowing empty types in HOL is a matter of taste, allowing empty types is the only reasonable design for DHOL. Practical dependent types often require individual instances to be empty, such as the instance $\mathtt{id}_{below}\ 0$ of the commonly used base type $\mathtt{id}_{below} : \mathtt{id}_{nat} \to \mathrm{tp}$ of natural numbers below a limit.

Choice We modify the typing rule (ε) for the choice operator (while keeping its validity rule unchanged):

$$\frac{\Gamma,\ x : A \vdash_T F : \mathrm{bool} \qquad \Gamma \vdash_T \exists x{:}A.\ F}{\Gamma \vdash_T \varepsilon x{:}A.\ F : A}\ \varepsilon'$$

This choice operator corresponds to the stronger choice operator for DHOL discussed in [30]. The difference is in the elimination rule (εE) – which is unchanged compared to HOL – whose second assumption differs by a double elimination compared to the rule in [30]. As we are not interested in using the rules in an ATP, but rather in proving soundness and completeness of the translation this change simplifies our exposition.

The typing rule for the choice operator is a subtle deviation from the usual way how choice is described in HOL. In general, defining the semantics of the choice operator is difficult if the predicate is not satisfied. In HOL, empty types are usually not allowed. So the easiest way to solve the issue is to simply make a choice term always defined but with its value unspecified. This is in particular much simpler than using partial terms that are only well-formed if the predicate is satisfiable. This trick does not work in DHOL because types may be empty. However, in DHOL the price of making choice partial is low because DHOL already allows for terms whose definedness depends on the context. Therefore, we choose a choice operator that is only well-formed in contexts in which the predicate is satisfiable. This does not harm the translation to HOL because we are making fewer choice terms well-defined than in HOL.

Dependent Connectives We replace the typing rule for implication with one in which the well-formedness of G may assume the truth of F:

$$\frac{\Gamma \vdash_T F : \mathrm{bool} \qquad \Gamma,\ \triangleright F \vdash_T G : \mathrm{bool}}{\Gamma \vdash_T F \Rightarrow G : \mathrm{bool}}\ \Rightarrow'$$

The proof rules for implication are unchanged.

Example 3.3 (Dependent Connectives). Continuing Ex. 3.1, consider the formula

$$x : \mathtt{obj},\ y : \mathtt{obj} \vdash x =_{\mathtt{obj}} y \Rightarrow \mathtt{id}_x =_{\mathtt{mor}\ x\ x} \mathtt{id}_y\ :\ \mathrm{bool}$$

which expresses that equal objects have equal identity morphisms. It is easy to prove. But it is only well-typed because the typing rule for dependent implication allows using $x =_{\mathtt{obj}} y$ while type-checking $\mathtt{id}_x =_{\mathtt{mor}\ x\ x} \mathtt{id}_y : \mathrm{bool}$, which requires deriving $\mathtt{id}_y : \mathtt{mor}\ x\ x$ and thus $\mathtt{mor}\ y\ y \equiv \mathtt{mor}\ x\ x$.

We can reuse all definitions from Def. 2.1 for DHOL. However, note the definitions of $\wedge$ and $\vee$ are carefully chosen to make them dependent as well. In particular, we can derive the following dependent formation rules for them:

$$\frac{\Gamma \vdash_T F : \mathrm{bool} \qquad \Gamma,\ \triangleright F \vdash_T G : \mathrm{bool}}{\Gamma \vdash_T F \wedge G : \mathrm{bool}} \qquad \frac{\Gamma \vdash_T F : \mathrm{bool} \qquad \Gamma,\ \triangleright \neg F \vdash_T G : \mathrm{bool}}{\Gamma \vdash_T F \vee G : \mathrm{bool}}$$

Remark 3 (HOL as a Fragment of DHOL). DHOL is essentially a conservative extension of HOL. If we restrict DHOL to theories in which all base types a have arity 0, , terms never occur in types, thus the well-formedness of an expression never depends on axioms, and thus type equality and typing become decidable.

The resulting language is not quite standard HOL though. Firstly, DHOL allows for empty types, i.e., it does not retain the rule (nonEmpty). Secondly, DHOL allows choice terms only if the predicate is satisfiable, i.e., it does not retain the full power of the rule (ε). Thus, DHOL is a conservative extension of the variant of HOL without choice and with empty types.

For references' sake we list all DHOL rules again in a figure:

Theories and contexts:

$$\frac{}{\vdash \circ\ \mathsf{Thy}}\ \text{thyEmpty'} \qquad \frac{\vdash_T x_1 : A_1,\ \ldots, x_n : A_n\ \mathsf{Ctx}}{\vdash T,\ a : \Pi x_1{:}A_1.\ \ldots \Pi x_n{:}A_n.\ \mathsf{tp}\ \mathsf{Thy}}\ \text{thyType'}$$

$$\frac{\vdash_T A\ \mathsf{tp}}{\vdash T,\ \mathsf{c} : A\ \mathsf{Thy}}\ \text{thyConst'} \qquad \frac{\vdash_T F : \mathsf{bool}}{\vdash T,\ c \triangleright F\ \mathsf{Thy}}\ \text{thyAxiom'}$$

$$\frac{\vdash T\ \mathsf{Thy}}{\vdash_T .\ \mathsf{Ctx}}\ \text{ctxEmpty'} \qquad \frac{\Gamma \vdash_T A\ \mathsf{tp}}{\vdash_T \Gamma,\ x : A\ \mathsf{Ctx}}\ \text{ctxVar'} \qquad \frac{\Gamma \vdash_T F : \mathsf{bool}}{\vdash_T \Gamma,\ x \triangleright F\ \mathsf{Ctx}}\ \text{ctxAssume'}$$

Lookup in theory and context:

$$\frac{\vdash_T \Gamma\ \mathsf{Ctx} \qquad \mathsf{a} : \Pi x_1{:}A_1.\ \ldots \Pi x_n{:}A_n.\ \mathsf{tp}\ \text{in}\ T \qquad \Gamma \vdash_T t_1 : A_1 \quad \ldots \quad \Gamma \vdash_T t_n : A_n[{}^{x_1}/_{t_1}]\ldots[{}^{x_{n-1}}/_{t_{n-1}}]}{\Gamma \vdash_T \mathsf{a}\ t_1\ \ldots\ t_n\ \mathsf{tp}}\ \text{type'}$$

$$\frac{\mathsf{c} : A'\ \text{in}\ T \qquad \Gamma \vdash_T A' \equiv A}{\Gamma \vdash_T \mathsf{c} : A}\ \text{const'} \qquad \frac{c \triangleright F\ \text{in}\ T \qquad \vdash_T \Gamma\ \mathsf{Ctx}}{\Gamma \vdash_T F}\ \text{axiom'}$$

$$\frac{x : A'\ \text{in}\ \Gamma \qquad \Gamma \vdash_T A' \equiv A}{\Gamma \vdash_T x : A}\ \text{var'} \qquad \frac{x \triangleright F\ \text{in}\ \Gamma \qquad \vdash_T \Gamma\ \mathsf{Ctx}}{\Gamma \vdash_T F}\ \text{assume'}$$

Well-formedness and equality of types:

$$\frac{\vdash_T \Gamma\ \mathsf{Ctx}}{\Gamma \vdash_T \mathsf{bool}\ \mathsf{tp}}\ \text{bool'} \qquad \frac{\Gamma \vdash_T A\ \mathsf{tp} \quad \Gamma,\ x : A \vdash_T B\ \mathsf{tp}}{\Gamma \vdash_T \Pi x{:}A.\ B\ \mathsf{tp}}\ \Pi \qquad \frac{\vdash_T \Gamma\ \mathsf{Ctx}}{\Gamma \vdash_T \mathsf{bool} \equiv \mathsf{bool}}\ \text{congBool'}$$

$$\frac{\vdash_T \Gamma\ \mathsf{Ctx} \quad \mathsf{a} : \Pi x_1{:}A_1.\ \ldots \Pi x_n{:}A_n.\ \mathsf{tp}\ \text{in}\ T \qquad \Gamma \vdash_T s_1 =_{A_1} t_1\ \ldots\ \Gamma \vdash_T s_n =_{A_n[{}^{x_1}/_{t_1}]\ldots[{}^{x_{n-1}}/_{t_{n-1}}]} t_n}{\Gamma \vdash_T \mathsf{a}\ s_1\ \ldots\ s_n \equiv \mathsf{a}\ t_1\ \ldots t_n}\ \text{congBase'} \qquad \frac{\Gamma \vdash_T A \equiv A' \quad \Gamma, x : A \vdash_T B \equiv B'}{\Gamma \vdash_T \Pi x{:}A.\ B \equiv \Pi x{:}A'.\ B'}\ \text{congΠ}$$

Typing:

$$\frac{\Gamma,\ x : A \vdash_T t : B}{\Gamma \vdash_T (\lambda x{:}A.\ t) : \Pi x{:}A.\ B}\ \lambda' \qquad \frac{\Gamma,\ x : A \vdash_T F : \mathsf{bool} \qquad \Gamma \vdash_T \exists x{:}A.\ F}{\Gamma \vdash_T \varepsilon x{:}A.\ F : A}\ \varepsilon'$$

$$\frac{\Gamma \vdash_T f : \Pi x{:}A.\ B \qquad \Gamma \vdash_T t : A}{\Gamma \vdash_T f\ t : B[{}^{x}/_{t}]}\ \text{appl'} \qquad \frac{\Gamma \vdash_T s : A \qquad \Gamma \vdash_T t : A}{\Gamma \vdash_T s =_A t : \mathsf{bool}}\ ='$$

Term equality: congruence, reflexivity, symmetry, β', η'

$$\frac{\Gamma \vdash_T A \equiv A' \quad \Gamma,\ x : A \vdash_T t =_B t'}{\Gamma \vdash_T \lambda x{:}A.\ t =_{\Pi x:A.\ B} \lambda x{:}A'.\ t'}\ \text{cong}\lambda' \qquad \frac{\Gamma \vdash_T t =_A t' \quad \Gamma \vdash_T f =_{\Pi x:A.\ B} f'}{\Gamma \vdash_T f\ t =_B f'\ t'}\ \text{congAppl'}$$

$$\frac{\Gamma \vdash_T t : A}{\Gamma \vdash_T t =_A t}\ \text{refl'} \qquad \frac{\Gamma \vdash_T t =_A s}{\Gamma \vdash_T s =_A t}\ \text{sym'} \qquad \frac{\Gamma \vdash_T (\lambda x{:}A.\ s)\ t : B}{\Gamma \vdash_T (\lambda x{:}A.\ s)\ t =_B s[{}^{x}/_{t}]}\ \beta' \qquad \frac{\Gamma \vdash_T t : \Pi x{:}A.\ B \qquad x\ \text{not in}\ \Gamma}{\Gamma \vdash_T t =_{\Pi x:A.\ B} \lambda x{:}A.\ t\ x}\ \eta\Pi$$

Rules for implication:

$$\frac{\Gamma \vdash_T F : \mathsf{bool} \qquad \Gamma \vdash_T G : \mathsf{bool}}{\Gamma \vdash_T F \Rightarrow G : \mathsf{bool}}\ \Rightarrow' \qquad \frac{\Gamma \vdash_T F : \mathsf{bool} \qquad \Gamma,\ x \triangleright F \vdash_T G}{\Gamma \vdash_T F \Rightarrow G}\ \Rightarrow\text{I'} \qquad \frac{\Gamma \vdash_T F \Rightarrow G \qquad \Gamma \vdash_T F}{\Gamma \vdash_T G}\ \Rightarrow\text{E'}$$

Congruence for validity, Boolean extensionality and choice:

$$\frac{\Gamma \vdash_T F =_{\mathsf{bool}} F' \qquad \Gamma \vdash_T F'}{\Gamma \vdash_T F}\ \text{cong}\vdash' \qquad \frac{\Gamma \vdash_T p\ \mathsf{true} \qquad \Gamma \vdash_T p\ \mathsf{false}}{\Gamma, x : \mathsf{bool} \vdash_T p\ x}\ \text{boolExt'} \qquad \frac{\Gamma,\ x : A \vdash_T F : \mathsf{bool} \qquad \Gamma \vdash_T \exists x{:}A.\ F}{\Gamma \vdash_T F[{}^{x}/_{(\varepsilon x:A.\ F)}]}\ \varepsilon\text{E'}$$

Fig. 2. DHOL Rules

4 Translating DHOL to HOL

Our translation realizes some form of *dependency erasure*, translating dependent types $\mathsf{a}\ t_1\ \ldots\ t_n$ to simple types a, effectively "merging" all instances of each dependent base type into a single simple base type, and replacing every Π with $\to$. Since this translation loses type information, we recover this information by generating a partial equivalence relation (PER) for every type. Using PERs as the semantics of dependently-typed systems is well-known [27] although it has not previously been applied to HOL-style dependently-typed systems.

Formally, we define the dependency erasure as a translation function $X \mapsto \overline{X}$ that maps any DHOL-syntax X to HOL-syntax. Additionally, we define a PER A^* on $\overline{A}$ for every DHOL-type A.

In general, a PER r on HOL-type U is a symmetric and transitive binary relation on U. In a PER, if x is related to any element, it is related to itself so that a PER behaves like an equivalence relation except for some elements not related to any other element. Thus, a PER r is the same as an equivalence relation on a subtype of U (or at least if would be if HOL had subtypes).

The intuition behind our translation is that the DHOL-type A corresponds in HOL to the quotient of the appropriate subtype of $\overline{A}$ by the equivalence A^*. In particular, the predicate $A^*\ t\ t$ captures whether t represents a term of type A as made precise in the table below:

DHOL	HOL
type A	type $\overline{A}$ and PER $A^* : \overline{A} \to \overline{A} \to \mathsf{bool}$
term $t : A$	term $\overline{t} : \overline{A}$ satisfying $A^*\ \overline{t}\ \overline{t}$

For an n-ary DHOL-base type a, the translation axiomatizes an $n+2$-ary predicate a^* that returns a PER for every n-tuple of arguments. Thus, $\mathsf{a}^*\ \overline{t_1}\ \ldots\ \overline{t_n}\ u\ u$ defines the subtype of the HOL-type a corresponding to the DHOL-type $\mathsf{a}\ t_1\ \ldots\ t_n$. For Booleans, the PER $\overline{\mathsf{bool}}$ is the identity relation. For every other type constructor, a corresponding operator on PERs must be defined: for function types, this is the well-known construction from logical relations that states that two functions are related if they map related inputs to related outputs. The following definition spells out the details.

Below we will actually use a stronger axiom a_{PER} to specify the PER returned by a^*. It forces the PER to be a very special one, namely a subset of the identity relation. Our translation would not be complete if we required this special property for all types. Indeed, PERs at function types are not sub-identities even if the PERs at base types are. But we can make the restriction for base types without losing completeness. This is critical in practice because it massively simplifies equality reasoning: DHOL-equalities are translated to the corresponding PER so that HOL-ATP optimizations for equality reasoning are not immediately applicable. But in our strengthened version, the HOL-ATP can easily turn all DHOL-equality *assumptions* on base types into HOL-equalities so that only equality *conclusions* require the extra effort of working with the PER.

Definition 4.1 (Translation). We translate DHOL-syntax by induction on the grammar. Theories and contexts are translated declaration-wise:

$$\overline{\circ} := \circ \qquad \overline{T,\ D} := \overline{T},\ \overline{D} \qquad \overline{\cdot} := \cdot \qquad \overline{\Gamma,\ D} := \overline{\Gamma},\ \overline{D} \tag{T1}$$

where D is a DHOL-declaration in a theory or context and $\overline{D}$ is a list of HOL declarations.

The translation of a (dependent) typing judgment yields a typing judgment and a validity judgment, so a type declaration corresponds to a type declaration and a declaration of a PER for that type declaration. Thus, the translation $\overline{\mathsf{a} : \Pi x_1{:}A_1.\ \ldots \Pi x_n{:}A_n.\ \mathsf{tp}}$ of a base type declaration is given by

$$\mathsf{a} : \mathsf{tp}, \tag{T2}$$

$$\mathsf{a}^* : \overline{A_1} \to \ldots \to \overline{A_n} \to \mathsf{a} \to \mathsf{a} \to \mathsf{bool} \tag{T3}$$

$$\mathsf{a}_{PER} \triangleright \forall x_1{:}\overline{A_1}.\ \ldots \forall x_n{:}\overline{A_n}.\ \forall u, v{:}\mathsf{a}.\ \mathsf{a}^*\ x_1\ \ldots\ x_n\ u\ v \Rightarrow u =_a v \tag{T4}$$

Constant and variable declarations are translated to declarations of the same name with a typing axiom that they are in the PER, and axioms and assumptions are translated straightforwardly:

$$\overline{\mathsf{c} : A} \ :=\ \mathsf{c} : \overline{A}, \tag{T5}$$

$$c^* \triangleright A^*\ c\ c \tag{T6}$$

$$\overline{x : A} \ :=\ x : \overline{A},$$

$$x^* \triangleright A^*\ x\ x \tag{T7}$$

$$\overline{c \triangleright F} \ :=\ c \triangleright \overline{F} \tag{T8}$$

$$\overline{x \triangleright F} \ :=\ x \triangleright \overline{F} \tag{T9}$$

The last two identically looking cases are for axioms and assumptions respectively. Since we use the same notations for both, and they are translated analogously, their translations look the same as well.

The cases of $\overline{A}$ and A^* for types A are:

$$\overline{\mathsf{a}\ t_1\ \ldots\ t_n} \ :=\ \mathsf{a} \tag{T10}$$

$$(\mathsf{a}\ t_1\ \ldots\ t_n)^* \ :=\ \mathsf{a}^*\ \overline{t_1}\ \ldots\ \overline{t_n} \tag{T11}$$

$$\overline{\Pi x{:}A.\ B} \ :=\ \overline{A} \to \overline{B} \tag{T12}$$

$$(\Pi x{:}A.\ B)^*\ f\ g \ :=\ \forall x, y{:}\overline{A}.\ A^*\ x\ y \Rightarrow B^*\ (f\ x)\ (g\ y) \tag{T13}$$

$$\overline{\mathsf{bool}} \ :=\ \mathsf{bool} \tag{T14}$$

$$\mathsf{bool}^*\ s\ t \ :=\ s =_{\mathsf{bool}} t \tag{T15}$$

The cases of $\overline{t}$ for terms t are straightforward except for, crucially, translating equality to the respective PER and relativizing the choice operator:

$$\overline{\mathsf{c}} \ :=\ \mathsf{c} \qquad \overline{x} \ :=\ x \qquad \overline{\lambda x{:}A.\ t} \ :=\ \lambda x{:}\overline{A}.\ \overline{t} \qquad \overline{f\ t} \ :=\ \overline{f}\ \overline{t} \qquad \overline{F \Rightarrow G} \ :=\ \overline{F} \Rightarrow \overline{G} \tag{T16}$$

$$\overline{s =_A t} \ :=\ A^*\ \overline{s}\ \overline{t} \tag{T17}$$

$$\overline{\varepsilon x{:}A.\ F} \ :=\ \varepsilon x{:}\overline{A}.\ A^*\ x\ x \wedge \overline{F} \tag{T18}$$

Example 4.2 (Translating Derived Connectives). It is straightforward to show that all defined DHOL-connectives are translated to their defined HOL-counterparts. For example, we have (up to logical equivalence in HOL) that $\overline{F \wedge G} = \overline{F} \wedge \overline{G}$.

For quantifiers, the translation yields

$$\overline{\forall\, x{:}A.\ F} \;=\; (A \to \mathsf{bool})^*\ \overline{\lambda x{:}A.\ F}\ \overline{\lambda x{:}A.\ \mathsf{true}}$$

$$= \forall\, x, y{:}\overline{A}.\ A^*\ x\ y \Rightarrow \mathsf{bool}^*\ \overline{F}\ \mathsf{true}$$

This looks clunky, but (because A^* is a PER as shown in Theorem 5.1) it is equivalent to

$$\forall\, x{:}\overline{A}.\ A^*\ x\ x \Rightarrow \overline{F}. \tag{T19}$$

Thus, DHOL-$\forall$ is translated to HOL-$\forall$ relativized using $A^*\ x\ x$.The corresponding rule

$$\overline{\exists\, x{:}A.\ F} = \exists\, x{:}\overline{A}.\ A^*\ x\ x \wedge \overline{F} \tag{T20}$$

can be shown analogously.

Example 4.3 (Categories in HOL). We give a fragment of the translation of Ex. 3.1:

$$\begin{array}{ll}
\mathtt{obj} : \mathsf{tp} & \mathtt{obj}^* : \mathtt{obj} \to \mathtt{obj} \to \mathsf{bool} \\
\mathtt{mor} : \mathsf{tp} & \mathtt{mor}^* : \mathtt{obj} \to \mathtt{obj} \to \mathtt{mor} \to \mathtt{mor} \to \mathsf{bool} \\
\mathtt{id} : \mathtt{obj} \to \mathtt{mor} & \mathtt{id}^* : \forall\, x, y{:}\mathtt{obj}.\ \mathtt{obj}^*\ x\ y \Rightarrow \mathtt{mor}^*\ x\ x\ (\mathtt{id}\ x)\ (\mathtt{id}\ y) \\
\multicolumn{2}{l}{\mathtt{comp} : \mathtt{obj} \to \mathtt{obj} \to \mathtt{obj} \to \mathtt{mor} \to \mathtt{mor} \to \mathtt{mor}} \\
\multicolumn{2}{l}{\mathtt{neutL} : \forall\, x{:}\mathtt{obj}.\ \mathtt{obj}^*\ x\ x \Rightarrow \forall\, y{:}\mathtt{obj}.\ \mathtt{obj}^*\ y\ y \Rightarrow} \\
\multicolumn{2}{l}{\qquad \forall\, m{:}\mathtt{mor}.\ \mathtt{mor}^*\ x\ y\ m\ m \Rightarrow \mathtt{mor}^*\ x\ y\ (\mathtt{comp}\ x\ x\ y\ (\mathtt{id}\ x)\ m)\ m}
\end{array}$$

Here, for brevity, we have omitted $\mathtt{obj}_{PER}$, $\mathtt{mor}_{PER}$, and $\mathtt{comp}^*$ and have already used the translation rule for $\forall$ from Ex. 4.2. The result is structurally close to what a native formalization of categories in HOL would look like, but slightly more verbose.

5 Completeness

Now we establish that our translation is sound and complete.

Remark 4 (Terminology Soundness/Completeness). Maybe surprisingly, there are two competing intuitions for the two concepts that lead to opposite namings: Firstly, in general for every language translation, soundness should refer to the preservation of judgments along the translation and completeness to their reflection. This is in line with the usual use of the words for denotational semantics, seen as a translation from syntax to semantics.

Secondly, if, as in our case, the translation is used to call a theorem prover in the target language to discharge proof obligations in the source language, we can think of the entire translation+theorem prover system as a proof calculus for the source language. If the concepts soundness and completeness are used with respect to that calculus, their roles switch: soundness is now reflection and completeness preservation of judgments.

We will (somewhat conflictedly) adopt the latter convention because it was preferred by a majority of test readers.

The completeness theorem states that our translation preserves all DHOL-judgments. Moreover, the theorem statement clarifies the intuition behind our translation:

Theorem 5.1 (Completeness). *We have*

if in DHOL	*then in HOL*
$\vdash T$ Thy	$\vdash \overline{T}$ Thy
$\vdash_T \Gamma$ Ctx	$\vdash_{\overline{T}} \overline{\Gamma}$ Ctx
$\Gamma \vdash_T A$ tp	$\overline{\Gamma} \vdash_{\overline{T}} \overline{A}$ tp *and* $\overline{\Gamma} \vdash_{\overline{T}} A^* : \overline{A} \to \overline{A} \to \mathsf{bool}$ *and* A^* *is a PER*
$\Gamma \vdash_T A \equiv B$	$\overline{\Gamma} \vdash_{\overline{T}} \overline{A} \equiv \overline{B}$ *and* $\overline{\Gamma} \vdash_{\overline{T}} A^* =_{\Pi x:\overline{A}.\ \Pi y:\overline{A}.\ \mathsf{bool}} B^*$
$\Gamma \vdash_T t : A$	$\overline{\Gamma} \vdash_{\overline{T}} \overline{t} : \overline{A}$ *and* $\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{t}\ \overline{t}$
$\Gamma \vdash_T F$	$\overline{\Gamma} \vdash_{\overline{T}} \overline{F}$

Additionally the following substitution lemmata hold:

$$\Gamma,\ x : A \vdash_T B \text{ tp } \textit{and } \Gamma \vdash_T u : A \quad \textit{implies} \quad \overline{\Gamma} \vdash_{\overline{T}} \overline{B[x/u]} \equiv \overline{B}[x/\overline{u}] \tag{1}$$

$$\Gamma,\ x : A \vdash_T t : B \ \textit{ and } \ \Gamma \vdash u : A \ \textit{ and } \ \overline{\Gamma},\ x : \overline{A},\ xRx \triangleright A^*\ x\ x \vdash_{\overline{T}} \overline{t} : \overline{B} \quad \textit{implies} \quad \overline{\Gamma} \vdash_{\overline{T}} \overline{t[x/u]} =_{\overline{B}} \overline{t}[x/\overline{u}] \tag{2}$$

$$\Gamma,\ x : A \vdash_T t : B \textit{ and } \overline{\Gamma} \vdash_{\overline{T}} B^*\ \overline{t}\ \overline{t} \quad \textit{implies} \quad \overline{\Gamma},\ x, x' : \overline{A},\ xRx' \triangleright A^*\ x\ x' \vdash_{\overline{T}} B^*\ \overline{t}\ \overline{t}[x/x'] \tag{3}$$

Moreover, all PERs are symmetric and transitive:

$$\Gamma \vdash_T t : B \textit{ and } \Gamma \vdash_T s : B \quad \textit{implies} \quad \overline{\Gamma} \vdash_{\overline{T}} B^*\ \overline{t}\ \overline{s} \Rightarrow B^*\ \overline{s}\ \overline{t} \tag{4}$$

$$\Gamma \vdash_T t : B \textit{ and } \Gamma \vdash_T s : B \textit{ and } \Gamma \vdash_T u : B \quad \textit{implies} \quad \overline{\Gamma} \vdash_{\overline{T}} B^*\ \overline{t}\ \overline{s} \Rightarrow \left(B^*\ \overline{s}\ \overline{u} \Rightarrow B^*\ \overline{t}\ \overline{u}\right) \tag{5}$$

The remainder of this section proves the substitution lemmata, symmetry and transitivity of PERs by induction on the grammar and afterwards the remaining claims by a relatively straightforward induction on derivations.

We will actually prove a slightly stronger theorem, where for term equality $\Gamma \vdash_T s =_A t$ (as a special case of validity) we not only show $\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{s}\ \overline{t}$ (the claim of the validity judgement) but also $\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{s}\ \overline{s}$, $\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{t}\ \overline{t}$, $\overline{\Gamma} \vdash_{\overline{T}} \overline{s} : \overline{A}$ and $\overline{\Gamma} \vdash_{\overline{T}} \overline{t} : \overline{A}$. This yields a more useful induction hypothesis.

Proof. Firstly, we will prove the substitution lemma, symmetry and transitivity of PERs by induction on the grammar, i.e. by induction on the shape of the terms and types.

Afterwards, we will prove completeness of the translation w.r.t. all DHOL judgments by induction on the derivations. This means that we consider the inference rules of DHOL and prove that if completeness holds for the premises of a DHOL inference rule, then it also holds for the conclusion of the rule and in particular this implies that the translation of the conclusion holds. For the inductive steps for the typing rule (=), we also require the fact that for any (well-formed) type A in DHOL we have $A^* : \overline{A} \to \overline{A} \to \mathsf{bool}$. This follows directly from how the A^* are generated/defined in the translation.

Substitution lemma and symmetry and transitivity of the typing relations.

Proof of (1): Since the translation of types commutes with the type productions of the grammar, and since the simple type $\overline{B}$ cannot depend on x, (1) is obvious.

Proof of (2): We show (2) by induction on the grammar of DHOL.

If x is not a free variable in t, then $\overline{t[x/u]} = \overline{t} = \overline{t}[x/\overline{u}]$ and the claim (2) follows by rule (refl). So assume that x is a free variable of t.

If t is a variable, then by assumption we have $t = x$ and thus $\overline{t[x/u]} = \overline{u} = \overline{t}[x/\overline{u}]$, so the claim follows by rule (refl).

If t is a λ-term $\lambda y{:}C.\ s$ (where x occurs free in s), the induction hypothesis yields $\overline{\Gamma},\ y : \overline{C} \vdash_{\overline{T}} \overline{s[x/u]} =_{\overline{B}} \overline{s}[x/\overline{u}]$, where B is the type of s. By rule (congλ), the desired result of $\overline{\Gamma} \vdash_{\overline{T}} \overline{\lambda y{:}C.\ s[x/u]} =_{\overline{A}} \overline{\lambda y{:}C.\ s}[x/\overline{u}]$ follows.

If t is a choice term $\varepsilon y{:}A.\ F$ (where x occurs free in F and x and y are distinct variables), the induction hypothesis yields $\overline{\Gamma},\ y : \overline{A} \vdash_{\overline{T}} \overline{F[x/u]} =_{\mathsf{bool}} \overline{F}[x/\overline{u}]$. It follows that $\overline{(\varepsilon y{:}A.\ F)\ [x/u]} = \overline{\varepsilon y{:}A.\ F[x/u]} = \varepsilon y{:}\overline{A}.\ (A^*\ y\ y \wedge \overline{F[x/u]}) = \varepsilon y{:}\overline{A}.\ (A^*\ y\ y \wedge \overline{F}[x/\overline{u}]) = \left(\varepsilon y{:}\overline{A}.\ \left(A^*\ y\ y \wedge \overline{F}\right)\right)[x/\overline{u}] = \overline{\varepsilon x{:}A.\ F}\ [x/\overline{u}]$ as desired.

If t is a function application $f\ s$ (where x occurs free in f and/or s), then by the induction hypothesis we have $\overline{\Gamma} \vdash_{\overline{T}} \overline{s[x/u]} =_{\overline{C}} \overline{s}[x/\overline{u}]$ and $\overline{\Gamma} \vdash_{\overline{T}} \overline{f[x/u]} =_{\overline{C}\to\overline{B}} \overline{f}[x/\overline{u}]$, where C is the type of s. By rule (congAppl), the claim of $\overline{\Gamma} \vdash_{\overline{T}} \overline{(f\ s)\ [x/u]} =_{\overline{B}} \overline{f\ s}[x/\overline{u}]$ follows.

If t is an equality $s =_A s'$ (where x occurs free in s and/or s'), then by induction hypothesis we have $\overline{\Gamma} \vdash_{\overline{T}} \overline{s[x/u]} =_{\overline{A}} \overline{s}[x/\overline{u}]$ and $\overline{\Gamma} \vdash_{\overline{T}} \overline{s'[x/u]} =_{\overline{A}} \overline{s'}[x/\overline{u}]$, where A is the type of s and s'. By rule (cong$=$), the claim of $\overline{\Gamma} \vdash_{\overline{T}} \overline{(s =_A s')\ [x/u]} =_{\mathsf{bool}} \left(\overline{s =_A s'}\right)[x/\overline{u}]$ follows.

If t is an implication $u \Rightarrow v$, then the claim follows directly from the induction hypotheses (the typing assumptions follow from (implTypingL) and (implTypingR)), as the substitution commutes with the implication.

Proof of symmetry (4) and transitivity (5) of the typing relations: Before we can show (3), we first need to prove the symmetry and transitivity of the typing relations: We can prove both by induction on the type A. Denote $\Delta := \overline{\Gamma},\ x, y : \overline{A},\ xRy \triangleright A^*\ x\ y$ and $\Theta := \overline{\Gamma},\ x, y, z : \overline{A},\ xRy \triangleright A^*\ x\ y,\ yRz \triangleright A^*\ y\ z$ respectively. If we can show $\Delta \vdash_{\overline{T}} A^*\ y\ x$ and $\Theta \vdash_{\overline{T}} A^*\ x\ z$ respectively, then the claims (4) and (5) follow using the rules ($\Rightarrow$I) (to push the hypotheses to the right of the $\vdash_{\overline{T}}$), ($\forall$I) (to move the variables from the context into universal quantifier) and ($\forall$E) (to eliminate the variables and substitute the corresponding terms from the claim). Those are therefore the claims we are going to show.

Observe that for a type A declared in the theory T, the symmetry and transitivity of A^* follow from the PER axiom generated by the translation of the type declaration declaring A. This follows directly from the symmetry and transitivity of equality and (2): We can use the rules ($\forall$I) and ($\Rightarrow$E) to derive the equalities between the terms. We obtain one of the validity premises (it doesn't matter which one as they only differ in which of the terms s, t, u is placed where and they are all equal anyway) by rule (assume). Finally, we replace as needed by rule (rewrite) (the typing premises can be shown using the rules (appl) and (const)) the terms s, t, u in it to yield the claim.

If A is bool, the typing relation is $=_{\mathsf{bool}}$ which is symmetric and transitive by the rules (sym) and (trans) respectively. In these cases the claims can follow directly by the rule (assume) and rule (sym) respectively rule (assume) and by rule (trans).

We can now prove the symmetry and transitivity of PERs for an arbitrary type A, by induction on the shape of A.

Proof of (3): This can be easily proven by induction of the structure of t. If t is a constant or variable other than x, the substitution won't affect it and the claim is trivial. If $t = x$, then rule (typesUnique) implies that $A = B$ and the claim follows by rule (assume). If t is a λ-term, equality or choice operator, the claim follows directly from the induction hypothesis and transitivity of the typing relations. Finally, if t is a function application $f\ s$ with f of type $\Pi x{:}A'.\ A$ and s of type A', then the induction hypotheses yield:

$$\overline{\Gamma},\ x, x' : \overline{A},\ xRx' \triangleright A^*\ x\ x' \vdash_{\overline{T}} \forall y, y'{:}\overline{A'}.\ A'^*\ \overline{y}\ \overline{y'} \Rightarrow A^*\ \overline{f'\ y}\ \overline{f'\ y'}$$

and

$$\overline{\Gamma},\ x, x' : \overline{A},\ xRx' \triangleright A^*\ x\ x' \vdash_{\overline{T}} A^*\ s'\ s',$$

where f' and s' denote the results of the substitutions in f and s respectively. By the rules ($\forall$E) and ($\Rightarrow$E), the claim follows.

To prove the main results of the theorem, it remains to show that all HOL rules preserve the main claim of the theorem (given in the table in the theorem). We group the rules in the following categories: well-formedness of theories, contexts, types, type equalities, well-typedness of terms, and validity, itself split between term equalities and the other validity rules.

Well-formedness of theories. Well-formedness of DHOL theories can be shown using the rules (thyEmpty'), (thyType'), (thyConst') and (thyAxiom'):

(thyEmpty'):

$$\vdash \circ\ \mathsf{Thy} \qquad \text{(thyEmpty)} \tag{6}$$
$$\vdash \overline{\circ}\ \mathsf{Thy} \qquad \text{(thyEmpty)}$$

(thyType'):

$\vdash_T x_1 : A_1, \ldots, x_n : A_n\ \mathsf{Ctx}$	By Assumption	(7)
$\vdash_{\overline{T}} x_1 : \overline{A_1}, x_1^* \triangleright A_1{}^*\ x_1\ x_1, \ldots, x_n : \overline{A_n}, x_n^* \triangleright A_n^*\ x_n\ x_n\ \mathsf{Ctx}$	Induction Hypothesis,(7)	(8)
$\vdash \overline{T}\ \mathsf{Thy}$	(ctxThy),(8)	(9)
$\vdash \overline{T},\ \mathsf{a} : \mathsf{tp}\ \mathsf{Thy}$	(thyType),(9)	(10)
$\vdash \overline{T,\ \mathsf{a} : \Pi x_1{:}A_1.\ \ldots \Pi x_n{:}A_n.\ \mathsf{tp}}\ \mathsf{Thy}$	(T2),(10)	

(thyConst'):

$\vdash_T A\ \mathsf{tp}$	By Assumption	(11)
$\vdash_{\overline{T}} \overline{A}\ \mathsf{tp}$	Induction Hypothesis,(11)	(12)
$\vdash \overline{T},\ \mathsf{c} : \overline{A}\ \mathsf{Thy}$	(thyConst),(12)	(13)
$\vdash \overline{T,\ \mathsf{c} : A}\ \mathsf{Thy}$	(T5),(13)	(14)

(thyAxiom'):

$\vdash_T F : \mathsf{bool}$	By Assumption	(15)
$\vdash_{\overline{T}} \overline{F} : \mathsf{bool}$	Induction Hypothesis,(15)	(16)
$\vdash \overline{T},\ ax \triangleright \overline{F}\ \mathsf{Thy}$	(thyAxiom),(16)	(17)
$\vdash \overline{T,\ ax \triangleright F}\ \mathsf{Thy}$	(T8),(17)	(18)

Well-formedness of contexts. Well-formedness of contexts can be concluded using the rules (ctxEmpty'), (ctxVar') and (ctxAssume'):
(ctxEmpty'):

$\vdash T\ \mathsf{Thy}$	By Assumption	(19)
$\vdash \overline{T}\ \mathsf{Thy}$	Induction Hypothesis,(19)	(20)
$\vdash_{\overline{T}} .\ \mathsf{Ctx}$	(ctxEmpty),(20)	(21)
$\vdash_{\overline{T}} \cdot\ \mathsf{Ctx}$	(T1),(21)	(22)

(ctxVar'):

$\Gamma \vdash_T A\ \mathsf{tp}$	By Assumption	(23)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{A}\ \mathsf{tp}$	Induction Hypothesis,(23)	(24)
$\overline{\Gamma} \vdash_{\overline{T}} A^* : \overline{A} \to \overline{A} \to \mathsf{bool}$	Induction Hypothesis,(23)	(25)
$\vdash_{\overline{T}} \overline{\Gamma},\ x : \overline{A}\ \mathsf{Ctx}$	(ctxVar),(24)	(26)
$\overline{\Gamma},\ x : \overline{A} \vdash_{\overline{T}} A^* : \overline{A} \to \overline{A} \to \mathsf{bool}$	(var $\vdash$),(24),(25)	(27)
$\overline{\Gamma},\ x : \overline{A} \vdash_{\overline{T}} A^*\ x\ x : \mathsf{bool}$	(appl),(appl),(27),(varS),(varS)	(28)
$\vdash_{\overline{T}} \overline{\Gamma},\ x : \overline{A},\ A^*\ x\ x\ \mathsf{Ctx}$	(ctxAssume),(28)	(29)
$\vdash_{\overline{T}} \overline{\Gamma,\ x : A}\ \mathsf{Ctx}$	(T7),(29)	(30)

(ctxAssume'):

$\Gamma \vdash_T F : \mathsf{bool}$	By Assumption	(31)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{F} : \mathsf{bool}$	Induction Hypothesis,(31)	(32)
$\vdash_{\overline{T}} \overline{\Gamma},\ ass \triangleright \overline{F}\ \mathsf{Ctx}$	(ctxAssume),(32)	(33)
$\vdash_{\overline{T}} \overline{\Gamma,\ ass \triangleright F}\ \mathsf{Ctx}$	(T9),(33)	(34)

Well-formedness of types. Since we already showed symmetry and transitivity of PERs (in general) in 4 and 5, we no longer have to prove it here. Well-formedness of types can be shown in DHOL using the rules (type'), (Π) and (bool'):
(type'):

$\mathsf{a} : \Pi x_1{:}A_1.\ \ldots \Pi x_n{:}A_n.\ \mathsf{tp}\ \mathsf{in}\ T$	By Assumption	(35)

$\Gamma \vdash_T t_1 : A_1$	By Assumption	(36)
$\vdots$		
$\Gamma \vdash_T t_n : A_n[{}^{x_1}/_{t_1}] \ldots [{}^{x_{n-1}}/_{t_{n-1}}]$	By Assumption	(37)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{t_1} : \overline{A_1}$	Induction Hypothesis, (36)	(38)
$\ldots$		
$\overline{\Gamma} \vdash_{\overline{T}} \overline{t_n} : \overline{A_n}$	Induction Hypothesis, (37)	(39)
$\vdash_T \Gamma$ Ctx	By Assumption	(40)
$\vdash_{\overline{T}} \overline{\Gamma}$ Ctx	Induction Hypothesis,(40)	(41)
$\mathsf{a} : \mathsf{tp}$ in $\overline{T}$	(T2),(35)	(42)
$\mathsf{a}^* : \overline{A_1} \to \ldots \overline{A_n} \to \mathsf{a} \to \mathsf{a} \to \mathsf{bool}$ in $\overline{T}$	(T3),(35)	(43)
$\overline{\Gamma} \vdash_{\overline{T}} a$ tp	(type),(42),(41)	
$\overline{\Gamma} \vdash_{\overline{T}} \mathsf{a}^* : \overline{A_1} \to \ldots \overline{A_n} \to \mathsf{a} \to \mathsf{a} \to \mathsf{bool}$	(constS),(43)	(44)
$\overline{\Gamma} \vdash_{\overline{T}} \mathsf{a}^* \ \overline{t_1} \ \ldots \ \overline{t_n} : \overline{\mathsf{a}} \to \overline{\mathsf{a}} \to \mathsf{bool}$	(appl),...,(appl),(44),(38),...,(39)	(45)
$\overline{\Gamma} \vdash_{\overline{T}} (a\ t_1 \ldots t_n)^* : \overline{\mathsf{a}} \to \overline{\mathsf{a}} \to \mathsf{bool}$	(T11),(45)	

(Π):

$\Gamma \vdash_T A$ tp	By Assumption	(46)
$\Gamma, x : A \vdash_T B$ tp	By Assumption	(47)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{A}$ tp	Induction Hypothesis,(46)	(48)
$\overline{\Gamma}, x : \overline{A}, x^* \triangleright A^*\ x\ x \vdash_{\overline{T}} \overline{B}$ tp	Induction Hypothesis,(T7),(47)	(49)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{B}$ tp	(varTp),(asTp),(49)	(50)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{A} \to \overline{B}$ tp	(→),(48),(50)	(51)
$\overline{\Gamma} \vdash_{\overline{T}} A^* : \overline{A} \to \overline{A} \to \mathsf{bool}$	Induction Hypothesis,(46)	(52)
$\overline{\Gamma}, x : \overline{A}, x^* \triangleright A^*\ x\ x \vdash_{\overline{T}} B^* : \overline{B} \to \overline{B} \to \mathsf{bool}$	Induction Hypothesis,(47)	(53)
$\overline{\Gamma}, x, y : \overline{A}, xRy \triangleright A^*\ x\ y \vdash_{\overline{T}} B^* : \overline{B} \to \overline{B} \to \mathsf{bool}$	(⇒E),(as ⊢),(var ⊢),(⇒I),(53),transitivity and symmetry of A^*	(54)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{\Pi x{:}A.\ B}$ tp	(T12),(51)	
$\overline{\Gamma} \vdash_{\overline{T}} (\Pi x{:}A.\ B)^* : (\overline{\Pi x{:}A.\ B}) \to (\overline{\Pi x{:}A.\ B}) \to \mathsf{bool}$	(T13),(∀),(⇒),(appl),(varS),(52),(54)	

(bool'):

$\vdash_T \Gamma$ Ctx	By Assumption	(55)
$\vdash_{\overline{T}} \overline{\Gamma}$ Ctx	Induction Hypothesis,(55)	(56)
$\overline{\Gamma} \vdash_{\overline{T}} \mathsf{bool}$ tp	(bool),(56)	

Type-equality. Type-equality can be shown using the rules (congBool'), (congBase') and (congΠ):

(congBool'):

$\vdash_T \Gamma$ Ctx	By Assumption	(57)
$\vdash_{\overline{T}} \overline{\Gamma}$ Ctx	Induction Hypothesis,(57)	(58)
$\overline{\Gamma} \vdash_{\overline{T}} \mathsf{bool} \equiv \mathsf{bool}$	(congBool),(58)	

(congBase'):

$\mathsf{a} : \Pi x_1{:}A_1. \ldots \Pi x_n{:}A_n.$ tp in T	By Assumption	(59)
$\Gamma \vdash_T s_1 =_{A_1} t_1$	By Assumption	(60)
$\vdots$		
$\Gamma \vdash_T s_n =_{A_n[x_1/t_1]\ldots[x_{n-1}/t_{n-1}]} t_n$	By Assumption	(61)
$\vdash_T \Gamma$ Ctx	By Assumption	(62)
$\mathsf{a} :$ tp in $\overline{T}$	(T2),(59)	(63)
$\mathsf{a}^* : \overline{A_1} \to \ldots \to \overline{A_n} \to \bar{\mathsf{a}} \to \bar{\mathsf{a}} \to \mathsf{bool}$ in $\overline{T}$	(T3),(59)	(64)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{s_1} =_{\overline{A_1}} \overline{t_1}$	Induction Hypothesis,(60)	(65)
$\vdots$		
$\overline{\Gamma} \vdash_{\overline{T}} \overline{s_n} =_{\overline{A_n}} \overline{t_n}$	Induction Hypothesis,(61)	(66)
$\vdash_{\overline{T}} \overline{\Gamma}$ Ctx	Induction Hypothesis,(62)	(67)
$\overline{\Gamma} \vdash_{\overline{T}} \mathsf{a} :$ tp	(type),(63),(67)	(68)
$\overline{\Gamma} \vdash_{\overline{T}} \mathsf{a} \equiv \mathsf{a}$	(congBase),(67),(68)	(69)
$\overline{\Gamma} \vdash_{\overline{T}} \mathsf{a}^* =_{\overline{A_1}\to\ldots\to\overline{A_n}\to\bar{\mathsf{a}}\to\bar{\mathsf{a}}\to\mathsf{bool}} \mathsf{a}^*$	(refl),(constS),(64),(67)	(70)
$\overline{\Gamma} \vdash_{\overline{T}} \mathsf{a}^* \overline{s_1} =_{\overline{A_2}\to\ldots\to\overline{A_n}\to\bar{\mathsf{a}}\to\bar{\mathsf{a}}\to\mathsf{bool}} \mathsf{a}^* \overline{t_1}$	(congAppl),(65),(70)	(71)
$\vdots$		
$\overline{\Gamma} \vdash_{\overline{T}} \mathsf{a}^* \overline{s_1} \ \ldots \ \overline{s_n} =_{\bar{\mathsf{a}}\to\bar{\mathsf{a}}\to\mathsf{bool}} \mathsf{a}^* \overline{t_1} \ \ldots \ \overline{t_n}$	(congAppl),(66),Previous line	(72)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{\mathsf{a}\ s_1\ \ldots\ s_n} \equiv \overline{\mathsf{a}\ t_1\ \ldots\ t_n}$	(T10),(69)	
$\overline{\Gamma} \vdash_{\overline{T}} (\mathsf{a}\ s_1\ \ldots\ s_n)^* =_{\overline{\mathsf{a}\ s_1\ \ldots\ s_n}\to\overline{\mathsf{a}\ s_1\ \ldots\ s_n}\to\mathsf{bool}} (\mathsf{a}\ t_1\ \ldots\ t_n)^*$	(T11),(T10),(72)	

(congΠ):

$\Gamma \vdash_T A \equiv A'$	By Assumption	(73)
$\Gamma, x : A \vdash_T B \equiv B'$	By Assumption	(74)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{A} \equiv \overline{A'}$	Induction Hypothesis,(73)	(75)
$\overline{\Gamma} \vdash_{\overline{T}} A^* =_{\overline{A}\to\overline{A}\to\mathsf{bool}} A'^*$	Induction Hypothesis,(73)	(76)

$$\overline{\Gamma},\ x : \overline{A},\ x^* \triangleright A^*\ x\ x \vdash_{\overline{T}} \overline{B} \equiv \overline{B'} \qquad \text{Induction Hypothesis,(74)} \qquad (77)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{B} \equiv \overline{B'} \qquad \text{(varEqTp),(asEqTp),(77)} \qquad (78)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{A} \to \overline{B} \equiv \overline{A'} \to \overline{B'} \qquad \text{(cong}\to\text{),(75),(78)} \qquad (79)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{\Pi x{:}A.\ B} \equiv \overline{\Pi x{:}A'.\ B'} \qquad \text{(T12),(79)}$$

$$\overline{\Gamma},\ x : \overline{A},\ x^* \triangleright A^*\ x\ x \vdash_{\overline{T}} B^* =_{\overline{B}\to\overline{B}\to\text{bool}} B'^* \qquad \text{Induction Hypothesis,(74)} \qquad (80)$$

$$\overline{\Gamma},\ x,y : \overline{A},\ xRy \triangleright A^*\ x\ y \vdash_{\overline{T}} A^*\ x\ x \qquad \text{(5),(var),(var),(var),(assume),(sym),(assume)} \qquad (81)$$

$$\overline{\Gamma},\ x,y : \overline{A},\ xRy \triangleright A^*\ x\ y \vdash_{\overline{T}} B^* =_{\overline{B}\to\overline{B}\to\text{bool}} B'^* \qquad (\Rightarrow\text{E}),(\text{as}\vdash\ ),(\text{var}\vdash\ ),(\Rightarrow\text{I}),(80),(81) \qquad (82)$$

$$\overline{\Gamma},\ x,y : \overline{A},\ xRy \triangleright A^*\ x\ y \vdash_{\overline{T}} B^*\ (\overline{f}\ x)\ (\overline{f}\ y) =_{\text{bool}} B'^*\ (\overline{f}\ x)\ (\overline{f}\ y) \qquad \text{(congAppl),(congAppl),(82),(refl),(refl)} \qquad (83)$$

$$\overline{\Gamma},\ f : \overline{A} \to \overline{B},\ x,y : \overline{A} \vdash_{\overline{T}} A^*\ x\ y =_{\text{bool}} A'^*\ x\ y \qquad \text{(congAppl),(congAppl),(76),(refl),(refl)} \qquad (84)$$

$$\overline{\Gamma},\ f : \overline{A} \to \overline{B},\ x,y : \overline{A} \vdash_{\overline{T}} A^*\ x\ y \Rightarrow B^*\ (f\ x)\ (f\ y) =_{\text{bool}} A'^*\ x\ y \Rightarrow B'^*\ (f\ x)\ (f\ y) \qquad \text{(cong}\Rightarrow\text{),(84),(83)} \qquad (85)$$

$$\overline{\Gamma},\ f : \overline{A} \to \overline{B} \vdash_{\overline{T}} \forall x,y{:}\overline{A}.\ A^*\ x\ y \Rightarrow (B^*\ (f\ x)\ (f\ y)) =_{\text{bool}} \forall x,y{:}\overline{A'}.\ A'^*\ x\ y \Rightarrow (B'^*\ (f\ x)\ (f\ y)) \qquad \text{(cong}\forall\text{),(75),(85)} \qquad (86)$$

$$\overline{\Gamma} \vdash_{\overline{T}} (\Pi x{:}A.\ B)^* =_{\overline{A}\to\overline{A}\to\text{bool}} (\Pi x{:}A'.\ B')^* \qquad \text{(T13),(cong}\lambda\text{),(86)}$$

Typing. Typing can be shown using the rules (λ'), (ε'), (appl'), ($\Rightarrow$'), (const'), (var'), (='):

(λ'):

$$\Gamma,\ x : A \vdash_T t : B \qquad \text{By Assumption} \qquad (87)$$

$$\Gamma,\ x : \overline{A},\ x^* \triangleright A^*\ x\ x \vdash_{\overline{T}} \overline{t} : \overline{B} \qquad \text{Induction Hypothesis,(T7),(87)} \qquad (88)$$

$$\Gamma,\ x : \overline{A},\ x^* \triangleright A^*\ x\ x \vdash_{\overline{T}} B^*\ \overline{t}\ \overline{t} \qquad \text{Induction Hypothesis,(T7),(87)} \qquad (89)$$

$$\Gamma,\ x,y : \overline{A},\ xRy \triangleright A^*\ x\ y \vdash_{\overline{T}} B^*\ \overline{t}\ \overline{t} \qquad (\Rightarrow\text{E}),(\text{as}\vdash\ ),(\text{var}\vdash\ ),(\Rightarrow\text{I}),(89),(4),(\text{assume}) \qquad (90)$$

$$\Gamma,\ x,y : \overline{A},\ xRy \triangleright A^*\ x\ y \vdash_{\overline{T}} B^*\ \overline{t}\ \overline{t}[x/y] \qquad (3),(87),(90) \qquad (91)$$

$$\Gamma \vdash_T \forall x,y{:}\overline{A}.\ A^*\ x\ y \Rightarrow B^*\ \overline{t}\ \overline{t}[x/y] \qquad (\forall\text{I}),(\forall\text{I}),(\Rightarrow\text{I}),(91) \qquad (92)$$

$$\Gamma,\ x : \overline{A} \vdash_{\overline{T}} \overline{t} : \overline{B} \qquad \text{(asTyping),(88)} \qquad (93)$$

$$\overline{\Gamma} \vdash_{\overline{T}} (\lambda x{:}\overline{A}.\ \overline{t}) : \overline{A} \to \overline{B} \qquad (\lambda)\text{,(93)} \qquad (94)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{\lambda x{:}A.\ t} : \overline{\Pi x{:}A.\ B} \qquad \text{(T12),(94)}$$

By applying the case (T13) to (92) and β-normalizing we yield the required result. From now we will do β-normalizations after applying definitions without comment.

$$\overline{\Gamma} \vdash_{\overline{T}} (\Pi x{:}A.\ B)^* \quad \overline{\lambda x{:}A.\ t}\ \overline{\lambda x{:}A.\ t} \qquad \text{explanation}$$

(ε'):

$\Gamma, x : A \vdash_T F : \mathsf{bool}$	By Assumption	(95)
$\Gamma \vdash_T \exists x{:}A.\ F$	By Assumption	(96)
$\overline{\Gamma} \vdash_{\overline{T}} \exists x{:}\overline{A}.\ A^*\ x\ x \wedge \overline{F}$	Induction Hypothesis,(T18),(96)	(97)
$\overline{\Gamma}, x : \overline{A}, x^* \triangleright A^*\ x\ x \vdash_{\overline{T}} \overline{F} : \mathsf{bool}$	Induction Hypothesis, (95)	(98)
$\overline{\Gamma}, x : \overline{A} \vdash_{\overline{T}} \overline{F} : \mathsf{bool}$	(asTyping), (98)	(99)
$\Gamma, x : A \vdash_T A\ \mathsf{tp}$	(typingTp),(var),(95)	(100)

Since we are proving completeness w.r.t. the individual judgements one by one, we may assume here that the translation is complete w.r.t. those judgements, for which we already proved completeness. We are therefore allowed to use completeness w.r.t. well-formedness of types here.

$\overline{\Gamma}, x : \overline{A} \vdash_{\overline{T}} A^* : \overline{A} \to \overline{A} \to \mathsf{bool}$	Explanation,(100)	(101)
$\overline{\Gamma}, x : \overline{A} \vdash_{\overline{T}} A^*\ x\ x : \mathsf{bool}$	(appl),(appl),(101)	(102)
$\overline{\Gamma}, x : \overline{A} \vdash_{\overline{T}} A^*\ x\ x \wedge \overline{F} : \mathsf{bool}$	Typing rule for $\wedge$,(99),(102)	(103)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{\varepsilon x{:}A.\ F} : \overline{A}$	(ε),(T18),(103)	(104)
$\overline{\Gamma} \vdash_{\overline{T}} A^*\ \left(\varepsilon x{:}\overline{A}.\ A^*\ x\ x \wedge \overline{F}\right)\ \left(\varepsilon x{:}\overline{A}.\ A^*\ x\ x \wedge \overline{F}\right) \wedge \overline{F}$	(εE), (103),(98)	(105)
$\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{\varepsilon x{:}A.\ F}\ \overline{\varepsilon x{:}A.\ F}$	(T18),($\wedge$El), (105)	

(appl'):

$\Gamma \vdash_T f : \Pi x{:}A.\ B$	By Assumption	(106)
$\Gamma \vdash_T t : A$	By Assumption	(107)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{f} : \overline{A} \to \overline{B}$	Induction Hypothesis,(T12),(106)	(108)
$\overline{\Gamma} \vdash_{\overline{T}} (\Pi x{:}A.\ B)^*\ \overline{f}\ \overline{f}$	Induction Hypothesis,(106)	(109)
$\overline{\Gamma} \vdash_{\overline{T}} \forall x{:}\overline{A}.\ \forall y{:}\overline{A}.$ $A^*\ x\ y \Rightarrow B^*\ (\overline{f}\ x)\ (\overline{f}\ y)$	(T13),(109)	(110)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{t} : \overline{A}$	Induction Hypothesis,(107)	(111)
$\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{t}\ \overline{t}$	Induction Hypothesis,(107)	(112)
$\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{t}\ \overline{t} \Rightarrow B^*\ (\overline{f}\ \overline{t})\ (\overline{f}\ \overline{t})$	($\forall$E),($\forall$E),(110),(111),(111)	(113)
$\overline{\Gamma} \vdash_{\overline{T}} B^*\ (\overline{f}\ \overline{t})\ (\overline{f}\ \overline{t})$	($\Rightarrow$E),(113),(112)	(114)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{f}\ \overline{t} : \overline{B}$	(appl),(108),(111)	(115)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{f\ t} : \overline{B}$	(T16),(115)	
$\overline{\Gamma} \vdash_{\overline{T}} B^*\ \overline{f\ t}\ \overline{f\ t}$	(T16),(114)	

(⇒'):

$\Gamma \vdash_T F : \mathsf{bool}$	By Assumption	(116)
$\Gamma,\ ass \triangleright F \vdash_T G : \mathsf{bool}$	By Assumption	(117)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{F} : \mathsf{bool}$	Induction Hypothesis,(116)	(118)
$\overline{\Gamma},\ \overline{F} \vdash_{\overline{T}} \overline{G} : \mathsf{bool}$	Induction Hypothesis,(117)	(119)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{G} : \mathsf{bool}$	(asTyping),(119)	(120)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{F} \Rightarrow \overline{G} : \mathsf{bool}$	(⇒),(118),(120)	(121)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{F \Rightarrow G} : \mathsf{bool}$	(T16),(121)	(122)
$\overline{\Gamma} \vdash_{\overline{T}} \mathsf{bool}^*\ \overline{F \Rightarrow G}\ \overline{F \Rightarrow G}$	(T17),(T16),(refl),(122)	

(const'):

$\mathsf{c} : A' \text{ in } T$	By Assumption	(123)
$\Gamma \vdash_T A' \equiv A$	By Assumption	(124)
$\mathsf{c} : \overline{A'} \text{ in } \overline{T}$	(T5),(123)	(125)
$c^* \triangleright A'^*\ c\ c \text{ in } \overline{T}$	(T6),(123)	(126)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{A'} \equiv \overline{A}$	Induction Hypothesis,(124)	(127)
$\overline{\Gamma},\ x : \overline{A'} \vdash_{\overline{T}} A'^* =_{\overline{A} \to \overline{A} \to \mathsf{bool}} A^*$	(var ⊢),Induction Hypothesis,(124)	(128)
$\overline{\Gamma},\ x : \overline{A'} \vdash_{\overline{T}} A'^*\ x\ x =_{\mathsf{bool}} A^*\ x\ x$	(congAppl),(congAppl),(128),(refl),(refl)	(129)
$\overline{\Gamma} \vdash_{\overline{T}} \forall\, x{:}\overline{A}.\ A'^*\ x\ x =_{\mathsf{bool}} A^*\ x\ x$	(∀I),(129)	(130)
$\overline{\Gamma} \vdash_{\overline{T}} \mathsf{c} : \overline{A}$	(const),(125),(127)	(131)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{\mathsf{c}} : \overline{A}$	(T16),(131)	
$\overline{\Gamma} \vdash_{\overline{T}} A'^*\ \overline{\mathsf{c}}\ \overline{\mathsf{c}}$	(T16),(axiom),(126),(typingTp),(131)	(132)
$\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{\mathsf{c}}\ \overline{\mathsf{c}}$	(⊢ cong),(∀E),(130),(131),(132)	

(var'):

$x : A' \text{ in } \Gamma$	By Assumption	(133)
$\Gamma \vdash_T A' \equiv A$	By Assumption	(134)
$x : \overline{A'} \text{ in } \overline{\Gamma}$	(T7),(133)	(135)
$x^* \triangleright A'^*\ x\ x \text{ in } \overline{\Gamma}$	(T7),(133)	(136)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{A'} \equiv \overline{A}$	Induction Hypothesis,(134)	(137)
$\overline{\Gamma},\ x : \overline{A'} \vdash_{\overline{T}} A'^* =_{\overline{A} \to \overline{A} \to \mathsf{bool}} A^*$	(var ⊢),Induction Hypothesis,(134)	(138)
$\overline{\Gamma},\ x : \overline{A'} \vdash_{\overline{T}} A'^*\ x\ x =_{\mathsf{bool}} A^*\ x\ x$	(congAppl),(congAppl),(138),(refl),(refl)	(139)
$\overline{\Gamma} \vdash_{\overline{T}} \forall\, x{:}\overline{A}.\ A'^*\ x\ x =_{\mathsf{bool}} A^*\ x\ x$	(∀I),(139)	(140)

$\overline{\Gamma} \vdash_{\overline{T}} x : \overline{A}$	(var),(135),(137)	(141)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{x} : \overline{A}$	(T16),(141)	
$\overline{\Gamma} \vdash_{\overline{T}} A'^{*}\ \overline{x}\ \overline{x}$	(T16),(assume),(136)	(142)
$\overline{\Gamma} \vdash_{\overline{T}} A^{*}\ \overline{x}\ \overline{x}$	($\vdash$ cong),($\forall$E),(140),(141),(142)	

(='):

$\Gamma \vdash_T s : A$	By Assumption	(143)
$\Gamma \vdash_T t : A$	By Assumption	(144)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{s} : \overline{A}$	Induction Hypothesis,(143)	(145)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{t} : \overline{A}$	Induction Hypothesis,(144)	(146)
$\overline{\Gamma} \vdash_{\overline{T}} A^{*}\ \overline{s}\ \overline{s}$	Induction Hypothesis,(143)	(147)
$\Gamma \vdash_T A$ tp	(typingTp),(143)	(148)

Since we are proving completeness w.r.t. the individual judgements one by one, we may assume here that the translation is complete w.r.t. those judgements, for which we already proved completeness. We are therefore allowed to use completeness w.r.t. well-formedness of types here.

$\overline{\Gamma} \vdash_{\overline{T}} A^{*} : \overline{A} \to \overline{A} \to \text{bool}$	Completeness w.r.t. well-formedness of A,(148)	(149)
$\overline{\Gamma} \vdash_{\overline{T}} A^{*}\ \overline{s}\ \overline{t} : \text{bool}$	(appl),(appl),(149),(145),(146)	(150)
$\overline{\Gamma} \vdash_{\overline{T}} \text{bool}^{*}\ (A^{*}\ \overline{s}\ \overline{t})\ (A^{*}\ \overline{s}\ \overline{t})$	(T15),(refl),(150)	

Term equality. Fix a context. Consider two DHOL terms $s, t : A$. By rule (eqTyping) and rule (sym) $\overline{s} =_{\overline{A}} \overline{t}$ implies $\overline{s} : \overline{A}$ and $\overline{t} : \overline{A}$. By rule (rewrite), $\overline{s} =_{\overline{A}} \overline{t}$ and $A^{*}\ \overline{s}\ \overline{s}$ then jointly imply $A^{*}\ \overline{t}\ \overline{t}$ and $A^{*}\ \overline{s}\ \overline{t}$ (the type of A^{*} follows by rule (applType), so applying rule (appl) yields the other typing assumption of the rule (rewrite)). In the completeness cases for term equality we need to show $A^{*}\ \overline{s}\ \overline{t}$ and $A^{*}\ \overline{s}\ \overline{s}$, $A^{*}\ \overline{t}\ \overline{t}$, $\overline{s} : \overline{A}$ and $\overline{t} : \overline{A}$ (as we are proving a slightly stronger result for term equality than for validity in general). As these all follow from $\overline{s} =_{\overline{A}} \overline{t}$ and $A^{*}\ \overline{s}\ \overline{s}$, we will usually show these two slightly easier statements instead (or alternatively and analogously the equivalent statement with $A^{*}\ \overline{t}\ \overline{t}$ instead of $A^{*}\ \overline{s}\ \overline{s}$). On the other hand by symmetry and transitivity $A^{*}\ \overline{s}\ \overline{t}$ implies the other needed statements, so that too is sufficient to prove. This reduces the completeness claim for a term-equality $s =_A t$ to showing $A^{*}\ \overline{s}\ \overline{s}$ as well as $\overline{s} =_{\overline{A}} \overline{t}$ or $\overline{s} : \overline{A}$ or $\overline{t} : \overline{A}$.

Term equality can be shown using the rules (congAppl'), (congλ'), ($\eta\Pi$), (refl'), (sym') and (β') in DHOL (the cases for the rules (boolExt') and (εE') which can be used to prove formulae (including potentially equalities) are treated later).

(congAppl'):

$\Gamma \vdash_T t =_A t'$	By Assumption	(151)
$\Gamma \vdash_T f =_{\Pi x:A.\ B} f'$	By Assumption	(152)
$\overline{\Gamma} \vdash_{\overline{T}} A^{*}\ \overline{t}\ \overline{t'}$	Induction Hypothesis,(151)	(153)

$\overline{\Gamma} \vdash_{\overline{T}} \forall\, x{:}\overline{A}.\ \forall\, y{:}\overline{A}.\ A^*\ x\ y \Rightarrow$		
$\quad (\Pi z{:}A.\ B)^*\ \overline{f}\ x\ \overline{f'}\ y$	Induction Hypothesis,(T13),(152)	(154)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{t} : \overline{A}$	Induction Hypothesis,(151)	(155)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{t'} : \overline{A}$	Induction Hypothesis,(151)	(156)
$\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{t}\ \overline{t'} \Rightarrow (\Pi z{:}A.\ B)^*\ \overline{f}\ \overline{t}\ \overline{f'}\ \overline{t'}$	(∀E),(∀E),(154),(155),(156)	(157)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{f} : \overline{A} \to \overline{B}$	Induction Hypothesis,(T12),(152)	(158)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{f'} : \overline{A} \to \overline{B}$	Induction Hypothesis,(T12),(152)	(159)
$\overline{\Gamma} \vdash_{\overline{T}} (\Pi z{:}A.\ B)^*\ \overline{f}\ \overline{t}\ \overline{f'}\ \overline{t'}$	(⇒E),(157),(153)	(160)
$\overline{\Gamma} \vdash_{\overline{T}} (\Pi z{:}A.\ B)^*\ \overline{f\ t}\ \overline{f'\ t'}$	(T16),(160)	
$\overline{\Gamma} \vdash_{\overline{T}} \overline{f\ t} : \overline{B}$	(T16),(appl),(158),(155)	
$\overline{\Gamma} \vdash_{\overline{T}} \overline{f'\ t'} : \overline{B}$	(T16),(appl),(159),(156)	

(congλ'):

$\Gamma \vdash_T A \equiv A'$	By Assumption	(161)
$\Gamma,\ x : A \vdash_T t =_B t'$	By Assumption	(162)
$\Gamma,\ x : A \vdash_T t : B$	(eqTyping),(162)	(163)
$\Gamma,\ x : A \vdash_T t' : B$	(eqTyping),(sym),(162)	(164)
$\overline{\Gamma} \vdash_{\overline{T}} \overline{A} \equiv \overline{A'}$	Induction Hypothesis,(161)	(165)
$\overline{\Gamma} \vdash_{\overline{T}} A^* : \overline{A} \to \overline{A} \to \text{bool}$	(eqTyping),Induction Hypothesis,(161)	(166)
$\overline{\Gamma},\ x : \overline{A},\ x^* \triangleright A^*\ x\ x \vdash_{\overline{T}} B^*\ \overline{t}\ \overline{t'}$	Induction Hypothesis,(162)	(167)
$\overline{\Gamma},\ z : \overline{A},\ z^* \triangleright A^*\ z\ z \vdash_{\overline{T}} B^*\ \overline{t}[{}^{x}/_{z}]\ \overline{t'}[{}^{x}/_{z}]$	α-renaming,(167)	(168)
$\overline{\Gamma},\ z : \overline{A} \vdash_{\overline{T}} A^*\ z\ z \Rightarrow$		
$\quad B^*\ \overline{t}[{}^{x}/_{z}]\ \overline{t'}[{}^{x}/_{z}]$	(⇒I),(168),(appl),(appl),(166),(var),(var)	(169)
$\overline{\Gamma} \vdash_{\overline{T}} \forall\, z{:}\overline{A}.\ A^*\ z\ z \Rightarrow$		
$\quad B^*\ \overline{t}[{}^{x}/_{z}]\ \overline{t'}[{}^{x}/_{z}]$	(∀I),(169)	(170)
$\overline{\Gamma},\ x, y : \overline{A} \vdash_{\overline{T}} \forall\, z{:}\overline{A}.\ A^*\ z\ z \Rightarrow$		
$\quad B^*\ \overline{t}[{}^{x}/_{z}]\ \overline{t'}[{}^{x}/_{z}]$	(var ⊢),(var ⊢),(170)	(171)
$\overline{\Gamma},\ x, y : \overline{A} \vdash_{\overline{T}} A^*\ x\ x \Rightarrow B^*\ \overline{t}\ \overline{t'}$	(∀E),(171),(varS)	(172)
$\overline{\Gamma},\ x, y : \overline{A},\ xRy \triangleright A^*\ x\ y \vdash_{\overline{T}} A^*\ x\ x \Rightarrow B^*\ \overline{t}\ \overline{t'}$	(as ⊢),(appl),(appl),(166),(varS),(var),(172)	(173)
$\overline{\Gamma},\ x, y : \overline{A},\ xRy \triangleright A^*\ x\ y \vdash_{\overline{T}} A^*\ x\ x$	(5),(assume),(4),(assume)	(174)
$\overline{\Gamma},\ x, y : \overline{A},\ xRy \triangleright A^*\ x\ y \vdash_{\overline{T}} B^*\ \overline{t}\ \overline{t'}$	(⇒E),(173),(174)	(175)
$\overline{\Gamma},\ x, y : \overline{A},\ xRy \triangleright A^*\ x\ y \vdash_{\overline{T}} B^*\ \overline{t'}\ \overline{t'}[{}^{x}/_{y}]$	(3),(164),(5),(4),(175),(175)	(176)
$\overline{\Gamma},\ x, y : \overline{A},\ xRy \triangleright A^*\ x\ y \vdash_{\overline{T}} B^*\ \overline{t}\ \overline{t'}[{}^{x}/_{y}]$	(5),(175),(176)	(177)

$$\overline{\Gamma},\ x, y : \overline{A} \vdash_{\overline{T}} A^*\ x\ y \Rightarrow B^*\ \overline{t}\ \overline{t'}[^x/_y] \qquad (\Rightarrow\text{I}),(\text{appl}),(\text{appl}),(166),(\text{varS}),(\text{var}),(177) \qquad (178)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \forall\, x{:}\overline{A}.\ \forall\, y{:}\overline{A}.\ A^*\ x\ y \Rightarrow B^*\ \overline{t}\ \overline{t'}[^x/_y] \qquad (\forall\text{I}),(\forall\text{I}),(178) \qquad (179)$$

$$\overline{\Gamma},\ x : \overline{A},\ x^* \triangleright A^*\ x\ x \vdash_{\overline{T}} \overline{t} : \overline{B} \qquad \text{Induction Hypothesis},(162) \qquad (180)$$

$$\overline{\Gamma},\ x : \overline{A},\ x^* \triangleright A^*\ x\ x \vdash_{\overline{T}} \overline{t'} : \overline{B} \qquad \text{Induction Hypothesis},(162) \qquad (181)$$

$$\overline{\Gamma},\ x : \overline{A} \vdash_{\overline{T}} \overline{t} : \overline{B} \qquad (\text{asTyping}),(180) \qquad (182)$$

$$\overline{\Gamma},\ x : \overline{A} \vdash_{\overline{T}} \overline{t'} : \overline{B} \qquad \text{Explanation},(181) \qquad (183)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \lambda x{:}\overline{A}.\ \overline{t} : \overline{A} \to \overline{B} \qquad (\lambda),(182) \qquad (184)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \lambda x{:}\overline{A}.\ \overline{t'} : \overline{A} \to \overline{B} \qquad (\lambda),(183) \qquad (185)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{\lambda x{:}A.\ t =_{\Pi x:A.\ B} \lambda x{:}A'.\ t'} \qquad (\text{T17}),(179)$$

($\eta\Pi$):

$$\Gamma \vdash_T t : \Pi x{:}A.\ B \qquad \text{By Assumption} \qquad (186)$$

$$\Gamma \vdash_T x \text{ not in } \Gamma \qquad \text{By Assumption} \qquad (187)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{t} : \overline{A} \to \overline{B} \qquad (\text{T12}),\text{Induction Hypothesis},(186) \qquad (188)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{t} =_{\overline{A} \to \overline{B}} \lambda x{:}\overline{A}.\ \overline{t}\ x \qquad (\eta),(188),(187) \qquad (189)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{t} =_{\overline{\Pi x:A.\ B}} \overline{\lambda x{:}A.\ t\ x} \qquad (\text{T12}),(189)$$

$$\overline{\Gamma} \vdash_{\overline{T}} (\Pi x{:}A.\ B)^*\ \overline{t}\ \overline{t} \qquad \text{Induction Hypothesis},(186)$$

(refl'):

$$\Gamma \vdash_T t : A \qquad \text{By Assumption} \qquad (190)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{t} : \overline{A} \qquad \text{Induction Hypothesis},(190) \qquad (191)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{t} =_{\overline{A}} \overline{t} \qquad (\text{refl}),(191)$$

$$\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{t}\ \overline{t} \qquad \text{Induction Hypothesis},(190)$$

(sym'):

$$\Gamma \vdash_T s =_A t \qquad \text{By Assumption} \qquad (192)$$

$$\Gamma \vdash_T s : A \qquad (\text{eqTyping}),(192) \qquad (193)$$

$$\Gamma \vdash_T t : A \qquad (\text{eqTyping}),(\text{sym}),(192) \qquad (194)$$

$$\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{s}\ \overline{t} \qquad \text{Induction Hypothesis},(192) \qquad (195)$$

$$\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{t}\ \overline{s} \qquad (\Rightarrow\text{E}),(4),(193),(194),(195)$$

(β'):

$$\Gamma \vdash_T (\lambda x{:}A.\ s)\ t : B \qquad \text{By Assumption} \qquad (196)$$

$\overline{\Gamma} \vdash_{\overline{T}} (\lambda x{:}\overline{A}.\ \overline{s})\ \overline{t} : \overline{B}$ Induction Hypothesis,(T16),(196) (197)

$\overline{\Gamma} \vdash_{\overline{T}} (\lambda x{:}\overline{A}.\ \overline{s})\ \overline{t} =_{\overline{B}} \overline{s}[^{x}/_{\overline{t}}]$ (β),(197) (198)

$\overline{\Gamma} \vdash_{\overline{T}} \overline{s}[^{x}/_{\overline{t}}] : \overline{B}$ (eqTyping),(sym),(198) (199)

Since (appl) is a necessary condition, for some HOL type A' we have $\overline{\Gamma} \vdash_{\overline{T}} \overline{t} : A'$ and $(\lambda x{:}\overline{A}.\ \overline{s}) : A' \to \overline{B}$. Since ($\lambda$) is also a necessary condition, by rule (typesUnique) the latter implies $\overline{\Gamma} \vdash_{\overline{T}} \overline{s} : \overline{A}$ and $\overline{A'} = \overline{A}$.

$\overline{\Gamma} \vdash_{\overline{T}} (\lambda x{:}\overline{A}.\ \overline{s})\ \overline{t} =_{\overline{B}} \overline{s[^{x}/_{t}]}$ (trans),(198),(2),above explanation, (199) (200)

$\overline{\Gamma} \vdash_{\overline{T}} \overline{(\lambda x{:}A.\ s)\ t} =_{\overline{B}} \overline{s[^{x}/_{t}]}$ (T16),(200)

$\overline{\Gamma} \vdash_{\overline{T}} (\Pi x{:}A.\ B)^{*}\ \overline{((\lambda x{:}A.\ s)\ t)}\ \overline{((\lambda x{:}A.\ s)\ t)}$ Induction Hypothesis,(196)

Validity. Validity can be shown using the rules (axiom'), (assume'), (cong $\vdash$ '), ($\Rightarrow$I'), ($\Rightarrow$E'), (εE') and (boolExt'). In case the formula whose validity is shown happens to be an equality we need to additionally show that the terms on both sides of the equality are well-typed and bool* holds on them. Since bool* is defined to be just $=_{\text{bool}}$ the first two claims follow from the rule (eqTyping) and the rules (eqTyping) and (sym), respectively. The other claims for term equality then directly follow by rule (refl).

(axiom'):

$ax \triangleright F$ in T By Assumption (201)

$\vdash_{T} \Gamma$ Ctx By Assumption (202)

$ax \triangleright \overline{F}$ in $\overline{T}$ (T8),(201) (203)

$\vdash_{\overline{T}} \overline{\Gamma}$ Ctx Induction Hypothesis,(202) (204)

$\overline{\Gamma} \vdash_{\overline{T}} \overline{F}$ (axiom),(203),(204)

(assume'), (cong $\vdash$ '), ($\Rightarrow$I') and ($\Rightarrow$E'): These cases straightforwardly follows from the induction hypotheses and the corresponding rules in HOL (namely (assume), (cong $\vdash$), ($\Rightarrow$I) and ($\Rightarrow$E)), similar to the proof in the case for rule (axiom').

(εE'):

$\overline{\Gamma},\ x : A \vdash_{T} F : \text{bool}$ By Assumption (205)

$\Gamma \vdash_{T} \exists x{:}A.\ F$ By Assumption (206)

By the same arguments as in the case of rule (ε'):

$\overline{\Gamma},\ x : \overline{A} \vdash_{\overline{T}} A^{*}\ x\ x \wedge \overline{F} : \text{bool}$ See above (207)

$\overline{\Gamma} \vdash_{\overline{T}} \exists x{:}\overline{A}.\ A^{*}\ x\ x \wedge \overline{F}$ Induction Hypothesis,(T18),(206) (208)

$\overline{\Gamma} \vdash_{\overline{T}} \left(A^{*}\ x\ x \wedge \overline{F}\right)[^{x}/_{(\varepsilon x:\overline{A}.\ A^{*}\ x\ x \wedge \overline{F})}]$ (εE),(207),(208) (209)

$\overline{\Gamma} \vdash_{\overline{T}} \overline{F}[^{x}/_{(\varepsilon x:\overline{A}.\ A^{*}\ x\ x \wedge \overline{F})}]$ ($\wedge$Er),(209) (210)

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{F[x/\epsilon x{:}A.\ F]} \qquad \text{(T7),(210)}$$

(boolExt'):

$$\Gamma \vdash_T p\ \mathsf{true} \qquad \text{By Assumption} \qquad (211)$$

$$\Gamma \vdash_T p\ \mathsf{false} \qquad \text{By Assumption} \qquad (212)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{p}\ \mathsf{true} \qquad \text{Induction Hypothesis,(T16),(211)} \qquad (213)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \overline{p}\ \mathsf{false} \qquad \text{Induction Hypothesis,(T16),(212)} \qquad (214)$$

$$\overline{\Gamma} \vdash_{\overline{T}} \forall z{:}\mathsf{bool}.\ \overline{p}\ z \qquad \text{(boolExt),(213),(214)} \qquad (215)$$

$$\overline{\Gamma},\ x : \mathsf{bool} \vdash_{\overline{T}} \forall z{:}\mathsf{bool}.\ \overline{p}\ z \qquad \text{(var}\vdash\text{),(215)} \qquad (216)$$

$$\overline{\Gamma},\ x : \mathsf{bool} \vdash_{\overline{T}} \overline{p}\ x \qquad \text{(}\forall\text{E),(216),(assume)} \qquad (217)$$

$$\overline{\Gamma},\ x : \mathsf{bool},\ x^* \triangleright \mathsf{bool}^*\ x\ x \vdash_{\overline{T}} \overline{p}\ x \qquad \text{(as}\vdash\text{),(217)} \qquad (218)$$

$$\overline{\Gamma,\ x : \mathsf{bool}} \vdash_{\overline{T}} \overline{p\ x} \qquad \text{(T7),(218)}$$

□

6 Soundness

The reverse direction of Theorem 5.1 is much trickier. To understand why, we look at two canaries in the coal mine that we have used to reject multiple intuitive but untrue conjectures:

Example 6.1 (Non-Injectivity of the Translation). Continuing Ex. 3.1, assume terms $u, v : \mathtt{obj}$ and consider the identify functions $I_u := \lambda f{:}\mathtt{mor}\ u\ u.\ f$ and $I_v := \lambda f{:}\mathtt{mor}\ v\ v.\ f$. Both are translated to the same HOL-term $\overline{I_u} = \overline{I_v} = \lambda f{:}\mathtt{mor}.\ f$ (because I_u and I_v only differ in the types, which are erased by our translation).

Consequently, the ill-typed DHOL-Boolean $F := I_u =_{\mathtt{mor}\ u\ u \to \mathtt{mor}\ u\ u} I_v$ is translated to the HOL-Boolean $(\lambda f{:}\mathtt{mor}.\ f) =_{\mathtt{mor}\to\mathtt{mor}} (\lambda f{:}\mathtt{mor}.\ f)$, which is not only well-typed in HOL but also a theorem.

To better understand the underlying issue we introduce the notion of *spurious* terms.

If the translation $\overline{t}$ of an *ill*-typed DHOL-term t is *well*-typed in HOL, we call it a **spurious** term (w.r.t. its preimage t) if t is ill-typed (otherwise, we call it **proper**). Intuitively, we should be able to use the PERs A^* to sort out the spurious terms, e.g., by using the predicate $A^*\ t\ t$ in HOL to ensure that $t : A$ in DHOL. That is in fact possible and commonly done for existing PER-based translations of dependent type theories. But it fails in the presence of classical Booleans:

Example 6.2 (Trivial PERs for Classical Booleans). Consider the property $\mathsf{bool}^*\ x\ x$. Thm. 5.1 guarantees $\mathsf{bool}^*\ \mathsf{true}\ \mathsf{true}$ and $\mathsf{bool}^*\ \mathsf{false}\ \mathsf{false}$. Thus, we can use Boolean extensionality in HOL to prove that $\forall x{:}\mathsf{bool}.\ \mathsf{bool}^*\ x\ x$, making the property trivial. In particular, we can prove $\mathsf{bool}^*\ \overline{F}\ \overline{F}$ for the spurious Boolean F from Ex. 6.1.

More generally, the property $(\Pi x{:}A.\ B)^*\ x\ x$ is trivial in this way whenever it is trivial for B and thus for all n-ary bool-valued function types.

We have conducted some experiments using intuitionistic HOL, undefinedness, or tree-valued logic, but all variants had similar issues. In any case, even if some variant of HOL could circumvent the issue, it would be of limited value because the main purpose of the translation is to use a theorem prover in the target language. Thus, we are tied to the variants of HOL for which strong ATP support is readily available.

Remark 5 (Trivial PERs for Built-in Base Types). Ex. 6.2 is just a special case of a more general effect. It occurs for every base type that is built into both DHOL and HOL and translated to itself. bool is the simplest example of that kind and the only one in our setting. But reasonable language extensions could feature built-in base types a for numbers, strings, etc., and all of those would suffer from the same issue if they were translated to a built-in analogue in HOL. Intuitively, this is because the translation is surjective and the HOL type has no space to accommodate additional spurious terms.

More formally, such built-in base types usually come with built-in induction principles that derive a universal property from all its ground instances (such as Boolean extensionality does for Booleans). In that case, $\mathsf{a}^*\ x\ x$ becomes trivial.

Note, however, that this effect only occurs for *built-in* base types that are translated to built-in analogues. It does *not* occur for *user-declared* base types. For example, consider a theory that declares a base type N for the natural numbers and an induction axiom for it. N is not translated to itself but to a fresh HOL-type, and the translation of the induction axiom relativizes the quantifier $\forall$ by $N^*\ x\ x$. Because of this relativization, the translated induction axiom cannot be used to prove $\forall\, x{:}\overline{\mathsf{N}}.\ N^*\ x\ x$ and thus the type $\overline{\mathsf{N}}$ has space to accommodate spurious terms.

Therefore, we cannot expect the reverse directions of the statements in Thm. 5.1 to hold in general. We will therefore restrict our considerations to only validity judgments. Additionally, we need the following technical condition.

Definition 6.3. We call a type symbol $\mathsf{a} : \Pi x_1{:}A_1.\ \ldots \Pi x_n{:}A_n.\ \mathsf{tp}$ in theory T partially inhabited if $\vdash_T \exists\, y{:}\mathsf{a}\ t_1\ \ldots\ t_n.\ \mathsf{true}$ is derivable (for some $t_1, \ldots, t_n$). We call a theory partially inhabited if every type symbol in it is.

Assuming that the DHOL theory is both well-typed and partially inhabited, we can show the following property that is sufficient to make our translation well-behaved:

Theorem 6.4 (Soundness). *Assume a well-formed DHOL-theory* $\vdash\ T\ \mathsf{Thy}$ *which is partially inhabited.*

$$\textit{If}\ \ \Gamma \vdash_T F : \mathsf{bool}\ \ \textit{and}\ \ \overline{\Gamma} \vdash_{\overline{T}} \overline{F},\ \ \textit{then}\ \ \Gamma \vdash_T F$$

In particular, if $\Gamma \vdash_T s : A$ *and* $\Gamma \vdash_T t : A$ *and* $\overline{\Gamma} \vdash_{\overline{T}} A^*\ \overline{s}\ \overline{t}$, *then* $\Gamma \vdash s =_A t$.

We expect the partial inhabitation requirement to be redundant, but have not been able to complete the proof without it yet. In any case, the requirement is extremely mild: it is already satisfied if every type symbol has some inhabited instance. That is virtually always the case in practical theories.

Theorem 5.1 thus ensures that the reverse direction does hold for the validity judgment at least — if we have already established that all involved expressions are well-typed in DHOL. Consequently, we *can* use a HOL-ATP to prove DHOL-conjectures *if* we validate independently that the conjecture is well-typed to begin with. A respective procedure is developed and described in Section 7.

The remainder of this section develops a number of lemmas and then carries out the proof of Thm. 6.4. The key idea is to transform a HOL-proof of $\overline{F}$ into one that is in the well-typed image of the translation, at which point we can read off a (partial) DHOL-proof of F. This DHOL proof will be missing sub-proofs to show the well-typedness of the terms occurring in the proof. Using the (assumed) well-typedness of the DHOL problem and induction, we can show that all terms occurring in the partial DHOL proof are in fact well-typed, yielding the existence of a complete DHOL proof. As we will see below, the technical details are more complicated, and our proof will actually have to introduce some subtle variations of this basic idea.

6.1 Type-wise injectivity of the translation

Lemma 6.5. *Let Δ be a DHOL context and let Γ denote its translation. Given two DHOL terms s, t of type A and assuming s and t are not identical, it follows that $\overline{s}$ and $\overline{t}$ are not identical.*

Proof. We prove this by induction on the shape of the types both equalities are over — in case both terms are equalities — and by subinduction on the shape of the two translated terms otherwise. We observe that terms created using a different top-level production are non-identical and will remain that way in the image. So we can go over the productions one by one and assuming type-wise injectivity for subterms show injectivity of applying them. Different constants are mapped to different constants and different variables to different variables, so in those cases there is nothing to prove. If two function applications or implications differ in DHOL then one of the two pairs of corresponding arguments must differ as well. By the induction hypothesis so will the images of the terms in that pair. Since function application and implication both commute with the translation, it follows that the images of the function applications or implications also differ. Since the translations of the terms on both sides of an equality also show up in the translation, the same argument also works for two equalities over the same type. Similarly for lambda functions and choice terms of same type (s, t have same type so cannot be lambda function or choice terms over different types).

Consider now two equalities over different types that get identified by dependency-erasure.

In case of equalities over different base types, the typing relations that are applied in the images are different, so the images of the equalities differ. For equalities $f =_{\Pi x:A.\ B} g$, $f' =_{\Pi x:A'.\ B'} g'$ over different Π-types either the domain type or the codomain type must differ by rule (congΠ). If the domain types A and A' differ, then the typing assumption after the two universal quantifiers of the translated equalities will differ. The applications $B^*\ (\overline{f}\ x)\ (\overline{f}\ y)$ and $B'^*\ (\overline{f'}\ x)\ (\overline{f'}\ y)$ of the typing relations on the right-hand side of the $\Rightarrow$ of the translated equalities will differ if the codomains B and B' (of the types the equalities are over) are different. The claim then follows from the induction hypothesis. □

This lemma implies that identical HOL terms have identical preimages (if any) of a given DHOL type, this property is required for reconstructing (partial) DHOL proofs from (partial) HOL proofs.

6.2 Quasi-preimages for terms and validity statements in admissible HOL derivations

Definition 6.6. Let t be an ill-typed DHOL term with well-typed image $\overline{t}$ in HOL. In this case we will say that $\overline{t}$ is a *spurious* term w.rt. its preimage t. If the preimage is unique or clear from the context we will simply say that $\overline{t}$ is spurious. Similarly, a term $\overline{s}$ in HOL that is the image of a well-typed term s, will be called *proper* w.r.t its preimage s. A term t in HOL that is not the image of any (well-typed or not) term is said to be *improper*.

Example 6.7 (Proper and spurious terms). Given the DHOL theory $\mathfrak{M}$:

$$\mathsf{a} : \Pi x{:}\mathsf{bool}.\ \mathsf{tp}$$
$$\mathsf{d} : \mathsf{a}\ \mathsf{false}$$
$$\mathsf{c} : \Pi x{:}\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}$$

The HOL term $\mathsf{c}\ \mathsf{d}$ is spurious, the HOL term $\lambda x{:}\mathsf{a}.\ \mathsf{true} =_{\mathsf{a}\to\mathsf{bool}} \mathsf{c}$ is proper w.r.t. its preimage $\lambda x{:}\mathsf{a}\ \mathsf{true}.\ \mathsf{true} =_{\Pi x:\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}} \mathsf{c}$ but spurious w.r.t. its other preimage $\lambda x{:}\mathsf{a}\ \mathsf{false}.\ \mathsf{true} =_{\Pi x:\mathsf{a}\ \mathsf{false}.\ \mathsf{bool}} \mathsf{c}$ and the HOL term $\mathsf{c} =_{\mathsf{a}\to\mathsf{bool}} \mathsf{c}$ is improper.

We will use this DHOL theory $\mathfrak{M}$ and these three HOL terms as a running example to illustrate the definitions throughout this section.

Firstly, we will consider the preimage of a typing relation A^* to be the equality symbol $\lambda x{:}A.\ \lambda y{:}A.\ x =_A y$ (if equality is treated as a (parametric) binary predicate rather than a production of the grammar this eta reduces to the symbol $=_A$).

Using this convention, we define the replacement of an improper HOL term, which is either a proper term or a spurious term. The replacement of an improper HOL term is defined by:

Definition 6.8. Let t be a HOL term. Then we define the replacement $\mathsf{repl}\ [t]$ of t by first matching line below:

$$\mathsf{repl}\ [\bar{t}] := \bar{t} \tag{PT1}$$
$$\mathsf{repl}\ [\mathsf{repl}\ [s]] := \mathsf{repl}\ [s] \tag{PT2}$$
$$\mathsf{repl}\ [A^*\ s] := \lambda y{:}\overline{A}.\ A^*\ s\ \mathsf{repl}\ [y] \tag{PT3}$$
$$\mathsf{repl}\ [A^*] := \lambda x{:}\overline{A}.\ \lambda y{:}\overline{A}.\ A^*\ x\ y \tag{PT4}$$
$$\mathsf{repl}\ [c] := \mathsf{c} \tag{PT5}$$
$$\mathsf{repl}\ [x] := x \tag{PT6}$$
$$\mathsf{repl}\ [f\ t] := \mathsf{repl}\ [f]\ \ \mathsf{repl}\ [t] \tag{PT7}$$
$$\mathsf{repl}\ [\lambda x{:}C.\ t] := \lambda x{:}C.\ \mathsf{repl}\ [t] \tag{PT8}$$

Consider $\forall x{:}\overline{A}.\ G$. In the two lines below, we require that F (as it occurs in these lines) is not of shape $A^* x'\ x \Rightarrow \_$ (for x' a variable bound in a universal quantifier whose body is this term) or of shape $\forall x'{:}\overline{A}.\ A^*\ x\ x' \Rightarrow \_$:

$$\mathsf{repl}\ \left[\forall x{:}\overline{A}.\ A^*\ x\ x \Rightarrow F\right] := \forall x, x'{:}\overline{A}.\ A^*\ x\ x' \Rightarrow \mathsf{repl}\ [F] \tag{PT9}$$
$$\mathsf{repl}\ \left[\forall x{:}\overline{A}.\ F\right] := \forall x, x'{:}\overline{A}.\ A^*\ x\ x' \Rightarrow \mathsf{repl}\ [F] \tag{PT10}$$

Otherwise $\forall x{:}\overline{A}.\ F$ is the translation of a universal quantifier and (PT1) applies.

The next line will depend on a choice of preimage type A for s and t :

$$\mathsf{repl}\,[s =_{\overline{A}} t] := A^*\ \mathsf{repl}\,[s]\ \mathsf{repl}\,[t] \tag{PT11}$$

$$\mathsf{repl}\,[s \Rightarrow t] := \mathsf{repl}\,[s] \Rightarrow \mathsf{repl}\,[t] \tag{PT12}$$

If F is not of the form $A^* x\ x \wedge G$ for some G then:

$$\mathsf{repl}\,[\varepsilon x{:}\overline{A}.\ F] := \varepsilon x{:}\overline{A}.\ \mathsf{repl}\,[F] \tag{PT13}$$

For terms t in the image of the translation, we define the replacement of t to be t itself.

Example 6.9 (Replacements of various terms). Consider again the DHOL theory $\mathfrak{M}$ from Example 6.7. The HOL terms $\mathsf{c}\ \mathsf{d}$ is its own replacement, the (beta-reduced) replacement of $\lambda x{:}\mathsf{a}.\ \mathsf{true} =_{\mathsf{a}\to\mathsf{bool}} \mathsf{c}$ is $\forall x, y{:}\mathsf{a}.\ (\mathsf{a}\ \mathsf{true})^*\ x\ y \Rightarrow \mathsf{bool}^*\ \mathsf{true}\ (\mathsf{c}\ y)$ (if $\Pi x{:}\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}$ is chosen as preimage type for the type the equality is over) and the HOL term $\mathsf{c} =_{\mathsf{a}\to\mathsf{bool}} \mathsf{c}$ has the replacement $\forall x, y{:}\mathsf{a}.\ (\mathsf{a}\ \mathsf{true})^*\ x\ y \Rightarrow \mathsf{bool}^*\ (\mathsf{c}\ x)\ (\mathsf{c}\ y)$ (again if $\Pi x{:}\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}$ is chosen as preimage type for the type the equality is over).

Definition 6.10. Assume a well-formed DHOL theory T.

We say that an HOL context Δ is *proper* (relative to $\overline{T}$) iff there exists a well-formed HOL context Θ (relative to $\overline{T}$) s.t. there is a well-formed DHOL context Γ (relative to T) with $\overline{\Gamma} = \Theta$ and Θ can be obtained from Δ by adding well-typed typing assumptions. In this case, Γ is called a *quasi-preimage* of Δ. Inspecting the translation, it becomes clear that Γ is uniquely determined by the choices of the preimages of the types of variables without a typing assumption in Δ.

Given a proper HOL context Δ and a well-typed HOL formula φ over Δ, we say that φ is *quasi-proper* iff $\mathsf{repl}\,[\varphi] = \overline{F}$ for $\Gamma \vdash_T F : \mathsf{bool}$ and Γ is a quasi-preimage of Δ. In that case, we call F a *quasi-preimage* of φ.

Finally, we call a validity judgement $\Delta \vdash_{\overline{T}} \varphi$ in HOL *proper* iff

(1) Δ is proper,

(2) φ is quasi-proper in context Δ

In this case, we will call $\overline{\Gamma} \vdash_{\overline{T}} \overline{F}$ a relativization of $\Delta \vdash_{\overline{T}} \varphi$ and $\Gamma \vdash_T F$ a *quasi-preimage* of the statement $\Delta \vdash_{\overline{T}} \varphi$, where Γ is a quasi-preimage of Δ and F a quasi-preimage of φ. Additionally, for HOL terms with preimages we consider these preimages to be quasi-preimages of the HOL term as well.

Example 6.11 (Quasi-preimages of various terms). Consider again the DHOL theory $\mathfrak{M}$ from Example 6.7. The HOL term $\mathsf{c}\ \mathsf{d}$ has the unique (quasi)-preimage $\mathsf{c}\ \mathsf{d}$ (which is however ill-typed in DHOL) and the HOL term $\lambda x{:}\mathsf{a}.\ \mathsf{true} =_{\mathsf{a}\to\mathsf{bool}} \mathsf{c}$ has the quasi-preimages $\lambda x{:}\mathsf{a}\ \mathsf{true}.\ \mathsf{true} =_{\Pi x:\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}} \mathsf{c}$ and $\lambda x{:}\mathsf{a}\ \mathsf{false}.\ \mathsf{true} =_{\Pi x:\mathsf{a}\ \mathsf{false}.\ \mathsf{bool}} \mathsf{c}$, though the latter is ill-typed in DHOL. The well-typed one of the quasi-preimages for $\mathsf{c} =_{\mathsf{a}\to\mathsf{bool}} \mathsf{c}$ is $\mathsf{c} =_{\Pi x:\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}} \mathsf{c}$, the ill-typed one is $\mathsf{c} =_{\Pi x:\mathsf{a}\ \mathsf{false}.\ \mathsf{bool}} \mathsf{c}$.

The HOL context $\Gamma := .,\ x : \mathsf{a}$ is not proper but adding the typing assumption $x^* \triangleright (\mathsf{a}\ \mathsf{true})^*\ x\ x$ turns it into the proper HOL context $\Delta := .,\ x : \mathsf{a},\ x^* \triangleright (\mathsf{a}\ \mathsf{true})^*\ x\ x$. It follows that the preimage $\Gamma,\ x : \mathsf{a}\ \mathsf{true}$ of the context Δ is a

quasi-preimage of Γ. Since $\mathsf{c} =_{\mathsf{a}\to\mathsf{bool}} \mathsf{c}$ is quasi-proper (in the empty context) and Δ is a proper context, it follows that $\Delta \vdash_{\overline{T}} \mathsf{c} =_{\mathsf{a}\to\mathsf{bool}} \mathsf{c}$ is a proper validity judgment.

6.3 Transforming HOL derivations into admissible HOL derivations

It will be useful to distinguish between two different kinds of improper terms.

Definition 6.12. An improper term is called *almost proper* iff its replacement isn't spurious (w.r.t. a given quasi-preimage) and contains no spurious subterms, otherwise it is called *unnormalizably spurious*. Thus, improper terms are almost proper iff their quasi-preimage is well-typed. Furthermore, proper terms are almost proper by definition.

Example 6.13 (Almost proper terms). Consider again the DHOL theory:

$$\mathsf{a} : \Pi x{:}\mathsf{bool}.\ \mathsf{tp}$$
$$\mathsf{d} : \mathsf{a}\ \mathsf{false}$$
$$\mathsf{c} : \Pi x{:}\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}$$

The HOL term $\mathsf{c}\ \mathsf{d}$ is unnormalizably spurious w.r.t. its unique preimage $\mathsf{c}\ \mathsf{d}$. The HOL term $\lambda x{:}\mathsf{a}.\ \mathsf{true} =_{\mathsf{bool}} \mathsf{c}$ is almost proper w.r.t. its quasi-preimage $\lambda x{:}\mathsf{a}\ \mathsf{true}.\ \mathsf{true} =_{\mathsf{bool}} \mathsf{c}$ and unnormalizably spurious w.r.t. its quasi-preimage $\lambda x{:}\mathsf{a}\ \mathsf{false}.\ \mathsf{true} =_{\mathsf{bool}} \mathsf{c}$. The HOL term $\mathsf{c} =_{\mathsf{a}\to\mathsf{bool}} \mathsf{c}$ is almost proper w.r.t. its quasi-preimage $\mathsf{c} =_{\Pi x:\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}} \mathsf{c}$ and unnormalizably spurious w.r.t. its quasi-preimage $\mathsf{c} =_{\Pi x:\mathsf{a}\ \mathsf{false}.\ \mathsf{bool}} \mathsf{c}$.

Choosing (quasi-)preimage types for a HOL derivation.

Lemma 6.14 (Indexing lemma). *Assume that $\Gamma \vdash_T F : \mathsf{bool}$ holds in DHOL (relative to a partially inhabited theory). Given a valid HOL derivation D of the statement $\overline{\Gamma} \vdash_{\overline{T}} \overline{F}$, we can choose a DHOL type $T(t)$ (called type index) for each occurrence of a HOL term t in D, s.t. the following properties hold:*

(1) $T(\overline{t}) = A$ for any DHOL term t satisfying $\Gamma \vdash_T t : A$,

(2) $T(\mathsf{c}) = A$ if $\mathsf{c} : A$ is a constant in T,

(3) $T(x) = A$ if $x : A$ is a variable declaration in Γ,

(4) $T(s) = T(t)$ for s, t within an equality of the form $s =_A t$ for some HOL type A,

(5) $T(s) = T(t) = \mathsf{bool}$ for s, t within an implication of the form $s \Rightarrow t$,

(6) $T(x) = A$ for x in $\left(\lambda x{:}\overline{B}.\ s\right)\ t$ if $T(t) = A$,

(7) $T(s =_A t) = \mathsf{bool}$,

(8) when variables are moved from the context into a λ-binder or vice versa the index of said variable is preserved

(9) whenever a term t occurs both in the assumptions and conclusions of a step S in D, the index of t is the same in all those occurrences of t in S,

(10) *if* x, t *in* $\lambda x{:}\overline{B}.\ t$ *satisfy* $T(x) = A$ *and* $T(t) = B$, *then* $T(\lambda x{:}B.\ t) = \Pi x{:}A.\ B$,

(11) *if* x, F *in* $\varepsilon x{:}\overline{B}.\ F$ *satisfy* $T(x) = A$ *and* $T(F) = \mathsf{bool}$, *then* $T(\varepsilon x{:}B.\ F) = B$,

(12) $T(t) = A$ *implies* t *has type* $\overline{A}$ *and* A *is non-empty.*

Proof. This lemma only holds for well-formed derivations of translations of well-typed conjectures over well-formed theories. It will not hold for arbitrary formulae (as can be seen by considering equalities between constants of equal HOL but different DHOL types). The proof of the lemma will therefore use the fact that the final statement in the derivation is the translation of a well-typed DHOL statement (which already determines the "correct" indices for the terms within that statement) and then show that for each step in a well-formed HOL proof concluding a statement that we can correctly index, the assumptions of that step can also be indexed correctly. We will thus proceed by "backwards induction" on the shape of the derivation D.

The assumptions of the theory and contexthood rules only contain terms already contained in the conclusions, so the associated cases in the proof are all trivial.

Similarly the assumptions of lookup rules and type well-formedness rules contain no additional terms, so those cases are also trivial.

The typing rules are about forming larger terms from subterms, so if those larger terms can be consistently (i.e. according to the claim of the lemma) indexed, then the same is necessarily also true for the subterms. Thus the typing rules also have only trivial cases. By the same arguments the cases for the congruence rules (congλ), (congAppl), the symmetry and transitivity rules for term equality (sym) and (refl) and the rules (β), (η), ($\Rightarrow$), (εE) and ($\Rightarrow$I) are also all trivial.

It remains to consider the cases for the rules ($\Rightarrow$E), (cong $\vdash$), (boolExt) and (nonEmpty).

($\Rightarrow$E): Here, the assumptions of the rule contain the additional terms F and $F \Rightarrow G$. However as both terms are of type bool all their (quasi)-preimages have type bool as well and picking indices according to any of the quasi-preimages of $F \Rightarrow G$ will work.

(cong $\vdash$): Here, the assumptions of the rule contain the additional terms F' and $F =_{\mathsf{bool}} F'$. Both terms are of type bool, so by the same argument as in the previous case, we can index them consistently with the claim of this lemma.

(boolExt): Here, the assumptions of the rule contain the additional terms p true and p false and true and false. All these terms are of type bool, so by the same argument as in the previous two cases, we can index them consistently with the claim of this lemma.

(nonEmpty): Here, the second assumption contains an additional variable of type A. As this variable doesn't occur in F, we can index it by the type of any of its (quasi)-preimages.

By assumption, the DHOL theory is partially inhabited, so we can chose this type index to be inhabited as well. □

Remark 6. This lemma is the only place where we need the partial inhabitation assumption. In order to remove this assumption from the soundness proof, one would need to define a further proof transformation that ensures that all terms in the HOL proof taken by this lemma are of HOL types that have inhabited DHOL preimages.

Remark 7. In the following, we will use the phrase *type index* to refer to a choice of DHOL types for each HOL term in a derivation satisfying the properties of the previous lemma.

Definition 6.15. A valid HOL derivation is called *admissible* iff we can choose quasi-preimages for all terms occurring in it s.t. all terms in the derivation are almost proper w.r.t. their chosen quasi-preimages.

This definition is useful, since admissible derivations are precisely those HOL derivations that allow us to consistently lift the terms occurring in them to well-typed DHOL terms.

In the following, we describe a proof transformation which maps HOL derivations of translations of well-typed validity statements to admissible HOL derivations.

Definition 6.16. A *statement transformation* in a given logic is a map that maps statements in the logic to statements in the logic. Similarly an *indexed statement transformation* is a map that maps HOL statements with indexed terms to HOL statements.

Definition 6.17. A *macro-step M for* an (indexed) statement transformation T *replacing* a step S in a derivation is a sequence of steps $S_1, \ldots, S_n$ (called *micro-steps* of M) s.t. the assumptions of the S_i that are not concluded by S_j with $j < i$ are results of applying T to assumptions of step S and furthermore the conclusion of step S_n is the result of applying T to the conclusion of S. The assumptions of those S_j that are not concluded by previous micro-steps of M are called the *assumptions of macro-step M* and the conclusion of the last micro-step S_n of M is called the *conclusion of macro-step M*.

Thus, we can replace each step in a derivation by a macro step, and we can transform that derivation to a derivation in which the given indexed statement transformation is applied to all statements. This is useful to simplify and normalize derivations to derivations with certain additional properties. In our case, we want to normalize a given HOL derivation into an admissible HOL derivation. Thus, we need to define an indexed statement transformation for which all terms in the image of the transformation are almost proper and then replace all steps in the derivation by macro steps for that statement transformation.

Since the notions of the replacement and a quasi-proper term only makes sense once we fix a choice of type indices (in the sense of Lemma 6.14), the indexed statement transformation will actually depend on the choice of type indices.

Definition 6.18. A *normalizing statement transformation* sRed $(\cdot)$ is defined to be an indexed statement transformation that replaces terms in statements as described below. The definition of the transformation of a term depends on its type index — a DHOL-type A (called *preimage type*) — for each term t. We will write those types as indices to the HOL terms, so for instance t_A indicates a HOL term t of type $\overline{A}$ and preimage type A.

These preimage types are used to effectively associate to each term a type of a possible quasi-preimage (hence their name), which is useful as for λ-functions there are quasi-preimages of potentially many different types. We require that for an indexed term t_A, term t has type $\overline{A}$ and that for almost proper terms t_A with unique quasi-preimage the quasi-preimage has type A.

Since variables and lambda binders are the sole cause for HOL terms having multiple (quasi-)preimages, choosing indices for variables in HOL terms induces unique quasi-preimages (respecting those type indices). This uniqueness is a direct consequence of Lemma 6.5.

We will consider only those quasi-preimages that respect type indices, for the notions of unnormalizably spurious and almost proper terms.

With respect to these choices, the transformation will do the following two things (in this order) in order to "normalize" unnormalizably spurious terms to almost proper ones:

(1) apply beta- and eta reductions (yielding its beta-eta normal form) and in case this doesn't yield almost proper terms,

(2) replace unnormalizably spurious function applications of type $\overline{B}$ by the "default terms" $w_{\overline{B}}$ of type $\overline{B}$, defined as the translation of a fixed DHOL term of type B (whose existence follows from the last property of Lemma 6.14).

As we are assuming a valid HOL derivation indexed according to Lemma 6.14, we will only define this transformation on well-typed HOL terms with preimage types consistent with the indexing lemma. We can then define the transformation of t_A (denoted by $\mathsf{sRed}\ (t_A)$) by the first matching line for t_A as follows:

$$\mathsf{sRed}\ (t_A) \quad := t \quad \text{if } t \text{ has quasi-preimage of type } A \tag{SR1}$$

$$\mathsf{sRed}\ (f_{\Pi x{:}A.\ B}\ t_A) \quad := \mathsf{sRed}\ (f_{\Pi x{:}A.\ B})\ \mathsf{sRed}\ (t_A) \quad \text{if } f_{\Pi x{:}A.\ B}\ t_A \text{ not beta or eta reducible} \tag{SR2}$$

In the following cases, we assume that the term t_A in $\mathsf{sRed}\ (\cdot)$ on the left of := isn't an almost proper term w.r.t. the type index A:

$$\mathsf{sRed}\ (t_A) \quad := \mathsf{sRed}\ \left(t_A{}^{\beta\eta}\right) \quad \text{if } t \text{ is beta or eta reducible} \tag{SR3}$$

$$\mathsf{sRed}\ \left(s_A =_{\overline{A}} t_A\right) \quad := \mathsf{sRed}\ (s_A) =_{\overline{A}} \mathsf{sRed}\ (t_A) \tag{SR4}$$

$$\mathsf{sRed}\ (F_{\mathsf{bool}} \Rightarrow G_{\mathsf{bool}}) \quad := \mathsf{sRed}\ (F_{\mathsf{bool}}) \Rightarrow \mathsf{sRed}\ (G_{\mathsf{bool}}) \tag{SR5}$$

$$\mathsf{sRed}\ ((\varepsilon x{:}A.\ F_{\mathsf{bool}})_A) \quad := \varepsilon x{:}A.\ \mathsf{sRed}\ (F_{\mathsf{bool}}) \tag{SR6}$$

$$\mathsf{sRed}\ (\lambda x{:}A.\ s_B) \quad := \lambda x{:}A.\ \mathsf{sRed}\ (s_B) \tag{SR7}$$

$$\mathsf{sRed}\ \left((f_{\Pi x{:}A.\ B}\ t_{A'})_{B'}\right) \quad := w_{\overline{B}} \quad \text{if } A \neq A' \text{ or } B \neq B' \tag{SR8}$$

$$\mathsf{sRed}\ (t_A[{}^{x_B}/_{s_B}]) \quad := \mathsf{sRed}\ (t_A)\ [{}^{x}/_{\mathsf{sRed}\ (s_B)}] \tag{SR9}$$

Observe that $\mathsf{sRed}\ (\cdot)$ of a term is always almost proper.

Example 6.19 (Transformations of various HOL terms). Consider again the DHOL theory:

$$\mathsf{a} : \Pi x{:}\mathsf{bool}.\ \mathsf{tp}$$
$$\mathsf{d} : \mathsf{a}\ \mathsf{false}$$
$$\mathsf{c} : \Pi x{:}\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}$$

The HOL term $\mathsf{c}_{\Pi x:\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}}\ \mathsf{d}_{\mathsf{a}\ \mathsf{false}}$ has the unique ill-typed (quasi)-preimage $\mathsf{c}\ \mathsf{d}$ in DHOL, so by (SR8) its transformation is w_{a}. The HOL term $\lambda x_{\mathsf{a}\ \mathsf{true}}{:}\mathsf{a}.\ \mathsf{true} =_{\mathsf{a}\to\mathsf{bool}} \mathsf{c}_{\Pi x:\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}}$ is almost proper and thus is its own transformation. The HOL term $\mathsf{c}_{\Pi x:\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}} =_{\mathsf{a}\to\mathsf{bool}} \mathsf{c}_{\Pi x:\mathsf{a}\ \mathsf{true}.\ \mathsf{bool}}$ is also almost proper, so it too is its own transformation. Note

that for all the mentioned HOL terms in this example the type indices are uniquely determined by Lemma 6.14(1). So sRed $(\cdot)$ left the terms that have direct well-typed DHOL analogues unchanged but replaced the one term that isn't almost proper w.r.t. any quasi-preimage with a proper term.

Lemma 6.20. *Assume a well-typed DHOL theory T and a conjecture $\Gamma \vdash_T \varphi$ with Γ well-formed and φ well-typed. Assume a valid HOL derivation D of $\overline{\Gamma} \vdash_{\overline{T}} \overline{\varphi}$. Assume type indices for the terms in D according to the properties of the indexing lemma (Lemma 6.14). Then, for any steps S in D we can construct a macro-step for the normalizing statement transformation replacing step S s.t. after replacing all steps by their macro-steps:*

- *the resulting derivation is valid,*
- *all terms occurring in the derivation are almost proper (wr.t. the quasi-preimages determined by the type indices).*

Proof. We will show this by induction on the shape of D.

Firstly, we observe that there are no dependent types in HOL and the context and axioms contain no spurious subterms. Hence, well-formedness (of theories, contexts, types) and type-equality judgements are unaffected by the transformation. So there is nothing to prove for the well-formedness and type-equality rules (those steps can be replaced by a macro step containing exactly this single step).

Type indices: We observe that the properties of the type indices provided by Lemma 6.14 ensure that the smallest unnormalizably spurious terms (i.e. without unnormalizably spurious proper subterms) are function applications in which function and argument are both almost proper. Furthermore, in such a case the function is not a λ-function.

Now that the typing indices are chosen for all terms in the derivation, the notion of almost proper becomes well-defined for them and we can address the problem of replacing the steps in the derivation with suitable macro steps. It remains to consider the typing and validity rules (see Figure 1) and to construct macro steps for the steps in the derivation using them for the normalizing statement transformation.

Since terms indexed by a type A have type $\overline{A}$ it is easy to see from Definition 6.18 that the normalizing statement transformation replaces terms of type $\overline{A}$ by terms of type $\overline{A}$.

(const): Since constants are proper terms, there is nothing to prove.

(var): Since context variables are proper terms, there is nothing to prove.

(=):

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (s)_A : \overline{A} \qquad \text{By Assumption} \qquad (219)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (t)_A : \overline{A} \qquad \text{By Assumption} \qquad (220)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (s)_A =_{\overline{A}} \mathsf{sRed}\ (t)_A : \mathsf{bool} \qquad (=),(219),(220) \qquad (221)$$

If $s_A =_{\overline{A}} t_A$ is almost proper the following line follows by (SR1), otherwise it follows by (SR4).

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(s_A =_{\overline{A}} t_A\right) : \mathsf{bool} \qquad \text{explanation}$$

(λ):

$$\Delta,\ x_A : \overline{A} \vdash_{\overline{T}} \mathsf{sRed}\ (t_B)_B : \overline{B} \qquad \text{By Assumption} \qquad (222)$$

$$\Delta \vdash_{\overline{T}} \left(\lambda x_A{:}\overline{A}.\ \mathsf{sRed}\ (t_B)_B\right) : \overline{A} \to \overline{B} \qquad (\lambda),(222) \qquad (223)$$

If $\mathsf{sRed}\ (t)_B$ isn't an unnormalizably spurious function application $\mathsf{sRed}\left(f_{\Pi y:A'.\ B}\right)\ x_A$ for which x doesn't appear in f:

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(\lambda x_A{:}\overline{A}.\ t_B\right) : \overline{A} \to \overline{B} \qquad (\mathrm{SR7}),(223)$$

Else by (SR3) we have $\mathsf{sRed}\left(\lambda x_A{:}\overline{A}.\ t_B\right) = \mathsf{sRed}\left(\lambda x_A{:}\overline{A}.\ \mathsf{sRed}\ (t_B)_B\right) = \mathsf{sRed}\left(f_{\Pi y:A.\ B}\right)$. By the remark about the type of $\mathsf{sRed}\ (\cdot)$ it follows that $\mathsf{sRed}\left(f_{\Pi y:A.\ B}\right)$ has type $\overline{\Pi y{:}A.\ B} = \overline{A} \to \overline{B}$.

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(f_{\Pi y:A'.\ B}\right) : \overline{A} \to \overline{B} \qquad \text{see above} \qquad (224)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(\lambda x_A{:}\overline{A}.\ t_B\right) : \overline{A} \to \overline{B} \qquad (\mathrm{SR3}),(224)$$

(ε):

$$\Gamma,\ x_A : \overline{A} \vdash_T \mathsf{sRed}\ (F_{\mathsf{bool}}) : \mathsf{bool} \qquad \text{By Assumption} \qquad (225)$$

$$\Gamma \vdash_T \left(\varepsilon x_A{:}\overline{A}.\ \mathsf{sRed}\ (F_{\mathsf{bool}})\right)_A : \overline{A} \qquad (\varepsilon),(225) \qquad (226)$$

$$\Gamma \vdash_T \mathsf{sRed}\left(\left(\varepsilon x_A{:}\overline{A}.\ F_{\mathsf{bool}}\right)_A\right) : \overline{A} \qquad (\mathrm{SR6}),(226)$$

(appl):

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (f_{\Pi x:A.\ B}) : \overline{A} \to \overline{B} \qquad \text{By Assumption} \qquad (227)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (t_{A'}) : \overline{A} \qquad \text{By Assumption} \qquad (228)$$

If $A \equiv A'$:

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (f_{\Pi x:A.\ B}\ t_{A'}) : \overline{B} \qquad (\mathrm{SR2}),(\mathrm{appl}),(227),(228)$$

If $A \not\equiv A'$ then Lemma 6.14 (6) implies that f is not a lambda function (otherwise the type indices would have been chosen to ensure $A \equiv A'$) and thus $\mathsf{sRed}\ (f_{\Pi x:A.\ B})\ \mathsf{sRed}\ (t_{A'})$ and $f_{\Pi x:A.\ B}\ t_{A'}$ are not beta reducible. Thus by (SR8) we have

$$\mathsf{sRed}\ (f_{\Pi x:A.\ B}\ t_{A'}) = w_B.$$

By construction we have $w_B : \overline{B}$:

$$\Delta \vdash_{\overline{T}} w_B : \overline{B} \qquad \text{By construction} \qquad (229)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (f_{\Pi x:A.\ B}\ t_{A'}) : \overline{B} \qquad \text{Above explanation},(229)$$

(⇒):

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (F)_{\mathsf{bool}} : \mathsf{bool} \qquad \text{By Assumption} \qquad (230)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (G)_{\mathsf{bool}} : \mathsf{bool} \qquad \text{By Assumption} \qquad (231)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (F)_{\mathsf{bool}} \Rightarrow \mathsf{sRed}\ (G)_{\mathsf{bool}} : \mathsf{bool} \qquad (\Rightarrow),(230),(231) \qquad (232)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (F_{\mathsf{bool}} \Rightarrow F_{\mathsf{bool}}) : \mathsf{bool} \qquad \mathsf{SR5},(232)$$

(axiom): Since translations of axioms to HOL are always proper terms and the additionally generated axioms are almost proper, there is nothing to prove here.

(assume): If the axiom is a typing axiom generated by the translation, it follows that it is almost proper. Similarly, if it is an axiom for a base type. Otherwise:

$$ass \triangleright \mathsf{sRed}\ (F_{\mathsf{bool}})\ \mathsf{in}\ \Delta \qquad \text{By Assumption} \qquad (233)$$

$$\vdash_{\overline{T}} \Delta\ \mathsf{Ctx} \qquad \text{By Assumption} \qquad (234)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (F_{\mathsf{bool}}) \qquad (\mathsf{assume}),(233),(234)$$

By assumption sRed (F) is almost proper (with a quasi-preimage of type bool), so the conclusion of the rule is almost proper and there is nothing to prove here.

(congλ):

$$\Delta \vdash_{\overline{T}} \overline{A} \equiv \overline{A'} \qquad \text{By Assumption} \qquad (235)$$

$$\Delta,\ x_A : \overline{A} \vdash_{\overline{T}} \mathsf{sRed}\ \left(t_B =_{\overline{B}} {t'}_B\right)_{\mathsf{bool}} \qquad \text{By Assumption} \qquad (236)$$

$$\Delta,\ x_A : \overline{A} \vdash_{\overline{T}} \mathsf{sRed}\ (t_B)_B =_{\overline{B}} \mathsf{sRed}\ \left(t'_B\right)_B \qquad (\mathsf{SR4}),(236) \qquad (237)$$

$$\Delta \vdash_{\overline{T}} \lambda x_A{:}\overline{A}.\ \mathsf{sRed}\ (t_B)_B =_{\overline{A}\to\overline{B}} \lambda x_A{:}\overline{A}.\ \mathsf{sRed}\ \left({t'}_B\right)_B \qquad (\mathsf{cong}\lambda),(235),(237) \qquad (238)$$

By assumption $\mathsf{sRed}\ (t)_B =_{\overline{B}} \mathsf{sRed}\ (t')_B$ is almost proper with quasi-preimage consistent with type indices and $\overline{A} \equiv \overline{A'}$, thus also $\lambda x_A{:}\overline{A}.\ \mathsf{sRed}\ (t)_B =_{\overline{A}\to\overline{B}} \lambda x_A{:}\overline{A}.\ \mathsf{sRed}\ (t')_B$ almost proper with quasi-preimage consistent with type indices.

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ \left(\lambda x_A{:}\overline{A}.\ t_B =_{\overline{A}\to\overline{B}} \lambda x_A{:}\overline{A}.\ {t'}_B\right) \qquad (\mathsf{SR7}),(\mathsf{SR4}),(238)$$

(congAppl):

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ \left(t_A =_{\overline{A}} {t'}_A\right) \qquad \text{By Assumption} \qquad (239)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (t_A) =_{\overline{A}} \mathsf{sRed}\ \left({t'}_A\right) \qquad (\mathsf{SR4}),(239) \qquad (240)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ \left(f_{\Pi x:A'.\ B} =_{\overline{A}\to\overline{B}} {f'}_{\Pi x:A'.\ B}\right) \qquad \text{By Assumption} \qquad (241)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (f_{\Pi x:A'.\ B}) =_{\overline{A}\to\overline{B}} \mathsf{sRed}\ \left({f'}_{\Pi x:A'.\ B}\right) \qquad \text{either (SR1) or (SR4)},(241) \qquad (242)$$

Assume that $A \not\equiv A'$. By Lemma 6.14 (6) the type indices of λ-function and the arguments they are applied to are chosen consistently. Since the indices for f and f' aren't chosen consistently with those of t and t' ($A \not\equiv A'$), it follows that sRed (f) and sRed (f') are both not λ-functions. Consequently, the applications sRed $(f\ s)$ and sRed (f') sRed (s') (and thus also sRed $(f'\ s')$) are not beta- or eta reducible. Thus, sRed $(f_{\Pi x:A'.\ B}\ t_A) = w_{\overline{B}}$ and sRed $(f'_{\Pi x:A'.\ B}\ t'_A) = w_{\overline{B}}$ and we yield:

$$\Delta \vdash_{\overline{T}} \text{sRed}\ (f_{\Pi x:A'.\ B}\ t_A) =_{\overline{B}} \text{sRed}\ \left(f'_{\Pi x:A'.\ B}\ t'_A\right) \qquad \text{(refl)} \tag{243}$$

Otherwise (if $A \equiv A'$) the terms sRed $(f)_{\Pi x:A.\ B}$ sRed $(t_A)_A$ and sRed $(f')_{\Pi x:A.\ B}$ sRed $(t'_A)_A$ are almost proper with quasi-preimages consistent with type indices. By (SR1) or (SR2) (depending on whether $f_{\Pi x:A.\ B}$ and t_A themselves are proper) it follows:

$$\text{sRed}\ (f_{\Pi x:A.\ B}\ t_A) = \text{sRed}\ (f)_{\Pi x:A.\ B}\ \text{sRed}\ (t_A)_A$$

and

$$\text{sRed}\ \left(f'_{\Pi x:A.\ B}\ t'_A\right) = \text{sRed}\ \left(f'\right)_{\Pi x:A.\ B}\ \text{sRed}\ \left(t'_A\right)_A$$

and thus:

$$\Delta \vdash_{\overline{T}} \text{sRed}\ (f_{\Pi x:A.\ B}\ t_A) =_{\overline{B}} \text{sRed}\ \left(f'_{\Pi x:A.\ B}\ t'_A\right) \qquad \text{(congAppl),(240),(242)} \tag{244}$$

In either case, the desired result of

$$\Delta \vdash_{\overline{T}} \text{sRed}\ \left(f_{\Pi x:A'.\ B}\ t_A =_{\overline{B}} f'_{\Pi x:A'.\ B}\ t'_A\right)$$

follows by (SR4) or (SR1).

(refl):

$$\Delta \vdash_{\overline{T}} \text{sRed}\ (t_A)_A : \overline{A} \qquad \text{By Assumption} \tag{245}$$

$$\Delta \vdash_{\overline{T}} \text{sRed}\ (t_A)_A =_{\overline{A}} \text{sRed}\ (t_A)_A \qquad \text{(refl),(245)} \tag{246}$$

$$\Delta \vdash_{\overline{T}} \text{sRed}\ \left(t_A =_{\overline{A}} t_A\right) \qquad \text{SR4,(246)}$$

(sym):

$$\Delta \vdash_{\overline{T}} \text{sRed}\ \left(t_A =_{\overline{A}} s_A\right) \qquad \text{By Assumption} \tag{247}$$

$$\Delta \vdash_{\overline{T}} \text{sRed}\ (t_A)_A =_{\overline{A}} \text{sRed}\ (s_A)_A \qquad \text{(SR4),(247)} \tag{248}$$

$$\Delta \vdash_{\overline{T}} \text{sRed}\ (s_A)_A =_{\overline{A}} \text{sRed}\ (t_A)_A \qquad \text{(sym),(248)} \tag{249}$$

$$\Delta \vdash_{\overline{T}} \text{sRed}\ \left(s_A =_{\overline{A}} t_A\right) \qquad \text{(SR4),(249)}$$

(β):

$$\Delta \vdash_{\overline{T}} \text{sRed}\ \left(\left(\lambda x_A{:}\overline{A}.\ s_B\right)\ t_{A'}\right) : \overline{B} \qquad \text{By Assumption} \tag{250}$$

By Lemma 6.14 (6), it follows that $A = A'$. If $\mathsf{sRed}\left(\left(\lambda x_A{:}\overline{A}.\ s_B\right)\ t_A\right)$ is almost proper with quasi-preimage of type $B \equiv B'$, then $\mathsf{sRed}\left(\left(\lambda x_A{:}\overline{A}.\ s_B\right)\ t_A\right)_{B'} = \left(\lambda x_A{:}\overline{A}.\ s_B\right)\ t_A$ and thus:

$$\Delta \vdash_{\overline{T}} \left(\lambda x_A{:}\overline{A}.\ s_B\right)\ t_A : \overline{B} \qquad \text{(SR1),(250)} \qquad (251)$$

$$\Delta \vdash_{\overline{T}} \left(\lambda x_A{:}\overline{A}.\ s_B\right)\ t_A =_{\overline{B}} s_B[x_A/t_A] \qquad (\beta)\text{,(251)} \qquad (252)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(\left(\lambda x_A{:}\overline{A}.\ s_B\right)\ t_A =_{\overline{B}} s_B[x_A/t_A]\right) \qquad \text{(SR4),(SR1),(252)}$$

Otherwise (if $\mathsf{sRed}\left(\left(\lambda x_A{:}\overline{A}.\ s_B\right)\ t_A\right)$ is not almost proper with quasi-preimage of type B) by (SR3),

$$\mathsf{sRed}\left(\left(\lambda x_A{:}\overline{A}.\ s_B\right)\ t_{A'}\right) = \mathsf{sRed}\left(s_B[x_A/t_{A'}]\right)$$

and we yield:

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(\left(\lambda x_A{:}\overline{A}.\ s_B\right)\ t_{A'}\right) =_{\overline{B}} \mathsf{sRed}\left(s_B[x_A/t_{A'}]\right) \qquad \text{(refl),above observation} \qquad (253)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(\left(\lambda x_A{:}\overline{A}.\ s_B\right)\ t_{A'} =_{\overline{B}} s_B[x_A/t_{A'}]\right) \qquad \text{(SR4),(253)}$$

(η):

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(t_{\Pi x:A.\ B}\right) : \overline{A} \to \overline{B} \qquad \text{By Assumption} \qquad (254)$$

$$x \text{ not in } \Delta \qquad \text{By Assumption} \qquad (255)$$

Since $\mathsf{sRed}\left(t_{\Pi x:A.\ B}\right)$ is almost proper with quasi-preimage of type $\Pi x{:}A.\ B$, there exists some DHOL quasi-preimage t' for t of DHOL type $\Pi x{:}A.\ B$. Its eta-expansion $\lambda x{:}A.\ t'\ x$ is therefore also a DHOL term of type $\Pi x{:}A.\ B$ — and a quasi-preimage of $\lambda x_A{:}\overline{A}.\ \mathsf{sRed}\left(t_{\Pi x:A.\ B}\right)\ x_A$. Thus, by definition of almost proper $\lambda x_A{:}\overline{A}.\ \mathsf{sRed}\left(t_{\Pi x:A.\ B}\right)\ x_A$ is also almost proper with quasi-preimage of type $\Pi x{:}A.\ B$. It follows:

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(t_{\Pi x:A.\ B}\right) =_{\overline{A}\to\overline{B}} \lambda x_A{:}\overline{A}.\ \mathsf{sRed}\left(t_{\Pi x:A.\ B}\right)\ x_A \qquad (\eta)\text{,(254),(255)} \qquad (256)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(t_{\Pi x:A.\ B} =_{\overline{A}\to\overline{B}} \lambda x_A{:}\overline{A}.\ t_{\Pi x:A.\ B}\ x_A\right) \qquad \text{(SR4),(SR2),(SR7),(256)}$$

(cong $\vdash$):

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(F_{\mathsf{bool}} =_{\mathsf{bool}} {F'}_{\mathsf{bool}}\right) \qquad \text{By Assumption} \qquad (257)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left({F'}_{\mathsf{bool}}\right) \qquad \text{By Assumption} \qquad (258)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(F_{\mathsf{bool}}\right) =_{\mathsf{bool}} \mathsf{sRed}\left({F'}_{\mathsf{bool}}\right) \qquad \text{(SR4),(257)} \qquad (259)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(F_{\mathsf{bool}}\right) \qquad \text{(cong } \vdash \text{),(259),(258)}$$

($\Rightarrow$I):

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(F_{\mathsf{bool}}\right) : \mathsf{bool} \qquad \text{By Assumption} \qquad (260)$$

$$\Delta,\ ass_F \triangleright \mathsf{sRed}\left(F_{\mathsf{bool}}\right) \vdash_{\overline{T}} \mathsf{sRed}\left(G_{\mathsf{bool}}\right) \qquad \text{By Assumption} \qquad (261)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\left(F_{\mathsf{bool}}\right) \Rightarrow \mathsf{sRed}\left(G_{\mathsf{bool}}\right) \qquad (\Rightarrow\text{I}),(260),(261) \qquad (262)$$

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (F_{\mathsf{bool}} \Rightarrow G_{\mathsf{bool}}) \qquad (\mathrm{SR5}),(262)$$

($\Rightarrow$E):

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (F_{\mathsf{bool}} \Rightarrow G_{\mathsf{bool}}) \qquad \text{By Assumption} \qquad (263)$$
$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (F_{\mathsf{bool}}) \qquad \text{By Assumption} \qquad (264)$$
$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (F_{\mathsf{bool}}) \Rightarrow \mathsf{sRed}\ (G_{\mathsf{bool}}) \qquad (\mathrm{SR5}),(263) \qquad (265)$$
$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (G_{\mathsf{bool}}) \qquad (\Rightarrow\mathrm{E}),(265),(264)$$

(εE):

$$\Gamma,\ x_A : \overline{A} \vdash_T \mathsf{sRed}\ (F_{\mathsf{bool}}) : \mathsf{bool} \qquad \text{By Assumption} \qquad (266)$$
$$\Gamma \vdash_T \mathsf{sRed}\ \left(\exists x{:}\overline{A}.\ F_{\mathsf{bool}}\right) \qquad \text{By Assumption} \qquad (267)$$

By Lemma 6.14, (SR1), (SR7) it follows $\mathsf{sRed}\ \left(\forall y_A{:}\overline{A}.\ G_{\mathsf{bool}}\right) = \forall y_A{:}\overline{A}.\ \mathsf{sRed}\ (G_{\mathsf{bool}})$ for all A, G and thus by definition of $\exists$ we yield $\mathsf{sRed}\ \left(\exists x{:}\overline{A}.\ F_{\mathsf{bool}}\right) = \exists x{:}\overline{A}.\ \mathsf{sRed}\ (F_{\mathsf{bool}})$:

$$\Gamma \vdash_T \exists x{:}\overline{A}.\ \mathsf{sRed}\ (F_{\mathsf{bool}}) \qquad \text{explanation, } (267) \qquad (268)$$
$$\Gamma \vdash_T \mathsf{sRed}\ (F_{\mathsf{bool}})\ [x/\left(\varepsilon x_A{:}\overline{A}.\ \mathsf{sRed}\ (F_{\mathsf{bool}})\right)_A] \qquad (\varepsilon\mathrm{E}),(266),(268) \qquad (269)$$
$$\Gamma \vdash_T \mathsf{sRed}\ (F_{\mathsf{bool}})\ [x/\mathsf{sRed}\left(\left(\varepsilon x_A{:}\overline{A}.\ F_{\mathsf{bool}}\right)_A\right)] \qquad (\mathrm{SR6}),(269) \qquad (270)$$
$$\Gamma \vdash_T \mathsf{sRed}\ \left(F_{\mathsf{bool}}[x/\left(\varepsilon x_A{:}\overline{A}.\ F_{\mathsf{bool}}\right)_A]\right) \qquad (\mathrm{SR9}),(270)$$

(boolExt):

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (p_{\mathsf{bool}\to\mathsf{bool}}\ \mathsf{true}_{\mathsf{bool}}) \qquad \text{By Assumption} \qquad (271)$$
$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (p_{\mathsf{bool}\to\mathsf{bool}}\ F_{\mathsf{bool}}) \qquad \text{By Assumption} \qquad (272)$$
$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (p_{\mathsf{bool}\to\mathsf{bool}})\ \mathsf{true} \qquad (\mathrm{SR2}),(\mathrm{SR1}),(271) \qquad (273)$$
$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (p_{\mathsf{bool}\to\mathsf{bool}})\ \mathsf{false} \qquad (\mathrm{SR2}),(\mathrm{SR1}),(272) \qquad (274)$$
$$\Delta,\ x : \mathsf{bool} \vdash_{\overline{T}} \mathsf{sRed}\ (p_{\mathsf{bool}\to\mathsf{bool}})\ x \qquad (\mathrm{boolExt}),(273),(274) \qquad (275)$$
$$\Delta,\ x : \mathsf{bool} \vdash_{\overline{T}} \mathsf{sRed}\ (p_{\mathsf{bool}\to\mathsf{bool}}\ x) \qquad (\mathrm{SR2}),(\mathrm{SR1}),(275)$$

(nonEmpty):

$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (F_{\mathsf{bool}}) : \mathsf{bool} \qquad \text{By Assumption} \qquad (276)$$
$$\Delta,\ x_A : \overline{A} \vdash_{\overline{T}} \mathsf{sRed}\ (F_{\mathsf{bool}}) \qquad \text{By Assumption} \qquad (277)$$
$$\Delta \vdash_{\overline{T}} \mathsf{sRed}\ (F_{\mathsf{bool}}) \qquad (\mathrm{nonEmpty}),(276),(277)$$

□

6.4 Lifting admissible HOL derivations of validity statements to DHOL

We finally have all required results to prove the soundness of the translation from DHOL to HOL.

Proof of Theorem 6.4. As shown in Lemma 6.20, we may assume that the proof of $\overline{\Gamma} \vdash_{\overline{T}} \overline{F}$ is admissible, so it only contains almost-proper terms. Consequently, whenever an equality $s =_{\overline{A}} t$ is derivable in HOL and s', t' are the quasi-preimages of s, t respectively, it follows that its quasi-preimage $s' =_A t'$ is well-typed in DHOL, and thus $s' : A$ and $t' : A$. Without loss of generality (adding extra assumptions throughout the proof) we may assume that the context of the (final) conclusion is the translation of a DHOL context. By Lemma 6.5 the translation is term-wise injective.

Therefore, the translated conjecture is a proper validity statement with unique (quasi)-preimage in DHOL. If we can lift a derivation of the translated conjecture to a valid DHOL derivation of its quasi-preimage, the resulting derivation is a valid derivation of the original conjecture. This means, that it suffices to prove that we can lift admissible derivations of a proper validity statement S in HOL to a derivation of a quasi-preimage of S.

We prove this claim by induction on the validity rules of HOL as follows:

Given a validity rule R with assumptions $A_1, \ldots, A_n$, validity assumptions (assumptions that are validity statements) $V_1, \ldots, V_m$, non-judgement assumptions (meaning assumptions that something occurs in a context or theory) $N_1, \ldots, N_p$ and conclusion C we will show the following:

Claim. *Assuming that the A_i and the N_j hold.*

(1) *Assume that the conclusion C is proper with quasi-preimage C^{-1}. Then the contexts G_i of the V_i are proper and the quasi-preimages of the V_i are well-formed.*

(2) *Assume that whenever an V_i is proper its quasi-preimage (where we choose the same preimages for identical terms and types with several possible preimages) holds in DHOL and that the conclusion C is proper with quasi-preimage C^{-1}. Then, C^{-1} holds in DHOL.*

Consider the first part of this claim, namely that if C is proper then the V_i are proper. Since all formulae appearing in the derivation are almost proper, this implies that the V_i themselves are proper and by construction (choice of quasi-preimage) the contexts of their quasi-preimages fit together with the context of C^{-1}.

The translation clearly implies that if an N_j holds in HOL, the corresponding non-judgement assumption ${N_j}^{-1}$ holds in DHOL (e.g. if $\overline{F}$ is an axiom in $\overline{T}$, then F must be an axiom in T).

Since the validity judgement being derived is proper, it follows from this first part of the claim that the validity assumptions of all validity rules in the derivation are proper.

By induction on the validity rules, if given an arbitrary validity rule R whose assumptions hold and whose validity assumptions all satisfy a property P we can show that P holds on the conclusion of R, then all derivable validity judgments have property P. Since all the validity assumptions and conclusions of validity rules in the derivation are proper, the property of having a derivable quasi-preimage is well-defined and can be used as such a property. By this induction principle, it follows that the $\overline{F}$ has a derivable quasi-preimage, i.e. $\Gamma \vdash_T F$.

Thus, it suffices to prove that the claim holds for the validity rules in HOL.

We will now consider the validity rules one by one. For each rule we first prove the first part of the claim. Sometimes we also need that the quasi-preimages of some non-validity (typically typing) assumptions hold, so we will prove that this also follows from the conclusion being proper. Then the assumption of the second part, combined with the first part implies that the quasi-preimages of the V_i hold in DHOL and it is easy to prove that also C^{-1} holds in DHOL.

Throughout this proof we will use the notation $\tilde{t}$ to denote that t is the quasi-preimage of $\tilde{t}$. Since the translation is surjective on type-level we will only need this notation on term-level.

Validity in HOL can be shown using the rules (congλ), (η), (congAppl), (cong $\vdash$), (β), (refl), (sym), (assume), (axiom), ($\Rightarrow$I), ($\Rightarrow$E), (boolExt), (εE) and (nonEmpty).

(congλ):

Since the conclusion is proper, it follows that the preimage

$$\Gamma \vdash_T \lambda x{:}A.\ t =_{\Pi x:A.\ B} \lambda x{:}A.\ t'$$

of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} \forall\, x, y{:}\overline{A}.\ A^*\ x\ y \Rightarrow B^*\ (\tilde{t}\ x)\ (\widetilde{t'}\ y)$$

of the conclusion is well-formed. By rule (eqTyping) and rule (sym') we obtain $\Gamma \vdash_T \lambda x{:}A.\ t : \Pi x{:}A.\ B$ and $\Gamma \vdash_T \lambda x{:}A.\ t' : \Pi x{:}A.\ B$ in DHOL.

$\Gamma \vdash_T \lambda x{:}A.\ t : \Pi x{:}A.\ B$	see above	(278)
$\Gamma \vdash_T \lambda x{:}A.\ t' : \Pi x{:}A.\ B$	see above	(279)
$\Gamma, y : A \vdash_T (\lambda x{:}A.\ t)\ y : B[{}^x/_y]$	(appl'),(var $\vdash$),(278),(assume')	(280)
$\Gamma, y : A \vdash_T (\lambda x{:}A.\ t')\ y : B[{}^x/_y]$	(appl'),(var $\vdash$),(279),(assume')	(281)
$\Gamma, y : A \vdash_T (\lambda x{:}A.\ t)\ y =_{B[{}^x/_y]} t[{}^x/_y]$	(β'),(280)	(282)
$\Gamma, y : A \vdash_T (\lambda x{:}A.\ t')\ y =_{B[{}^x/_y]} t'[{}^x/_y]$	(β'),(281)	(283)
$\Gamma, x : A \vdash_T t : B$	α-renaming,(cong:),(282),($\equiv$ refl),(280)	(284)
$\Gamma, x : A \vdash_T t' : B$	α-renaming,(cong:),(283),($\equiv$ refl),(281)	(285)
$\Gamma, x : A \vdash_T t =_B t' : \mathsf{bool}$	(='),(284),(285)	

Clearly, $\Gamma, x : A \vdash_T t =_B t'$ is a quasi-preimage of the validity assumption, so this proves the first part of the claim.

Regarding the second part:

$\Gamma, x : A \vdash_T t =_B t'$	By Assumption	(286)

By rule (typingTp) applied to (278) we yield $\Gamma \vdash_T \Pi x{:}A.\ B$ tp. Since rule (Π) is a necessary condition, it follows that also $\Gamma \vdash_T A$ tp.

$\Gamma \vdash_T A \equiv A$	($\equiv$ refl),explanation	(287)
$\Gamma \vdash_T \lambda x{:}A.\ t =_{\Pi x:A.\ B} \lambda x{:}A.\ t'$	(congλ'),(287)	(288)

(η): Since the rule has no validity assumption, the first part of the claim holds.

For the second part, we still need the quasi-preimage of the assumption to hold, so we will show that it follows from the conclusion being proper.

Since the conclusion is proper, it follows that the preimage

$$\Gamma \vdash_T t =_{\Pi x{:}A.\ B} \lambda x{:}A.\ t\ x$$

of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} \forall\, x{:}\overline{A}.\ \forall\, y{:}\overline{A}.\ A^*\ x\ y \Rightarrow B^*\ (\widetilde{t}\ x)\ \left(\lambda x{:}\overline{A}.\ \widetilde{t}\ x\right)\ y$$

of the conclusion is well-formed. By rule (eqTyping) and rule (sym) we obtain $\Gamma \vdash_T t : \Pi x{:}A.\ B$ and $\Gamma \vdash_T \lambda x{:}A.\ t\ x : \Pi x{:}A.\ B$ in DHOL. Clearly, $\Gamma \vdash_T t : \Pi x{:}A.\ B$ is a quasi-preimage of the validity assumption, so this proves the quasi-preimage of the assumption of the rule.

Regarding the second part:

$\Gamma \vdash_T t : \Pi x{:}A.\ B$	see above	(289)
$\Gamma \vdash_T t =_{\Pi x{:}A.\ B} \lambda x{:}A.\ t\ x$	($\eta\Pi$),(289),wlog. (renaming) x not in Γ	

(congAppl): Since the conclusion is proper, it follows that the preimage

$$\Gamma \vdash_T f\ t =_B f'\ t'$$

of the replacement

$$B^*\ (\widetilde{f}\ \widetilde{t})\ (\widetilde{f'}\ \widetilde{t'})$$

of the conclusion is well-formed. By rule (eqTyping) and rule (sym) we obtain $\Gamma \vdash_T f\ t : B$ and $\Gamma \vdash_T f'\ t' : B$ in DHOL. Obviously, $\Gamma \vdash_T t =_A t'$ and $\Gamma \vdash_T f =_{\Pi x{:}A.\ B} f'$ are quasi-preimages of the validity assumptions.

Since the validity assumptions use the same context as the conclusion, it follows that they are both proper with uniquely determined context. As observed in the beginning of the proof if a proper assumption of a rule is an equality over a type $\overline{A}$, the induction hypothesis implies that the quasi-preimage of that assumption in which the equality is over type A must be well-formed. Hence both $\Gamma \vdash_T t =_A t'$ and $\Gamma \vdash_T f =_{\Pi x{:}A.\ B} f'$ are well-formed in DHOL, so we have proven the first part of the claim.

Regarding the second part of the claim:

$\Gamma \vdash_T t =_A t'$	By Assumption	(290)
$\Gamma \vdash_T f =_{\Pi x{:}A.\ B} f'$	By Assumption	(291)
$\Gamma \vdash_T f\ t =_B f'\ t'$	(congAppl'),(290),(291)	(292)

This is what we had to show.

(cong $\vdash$): Since the conclusion is proper, it follows that the preimage

$$\Gamma \vdash_T F$$

of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} \widetilde{F}$$

of the conclusion is well-formed. Thus we have $\Gamma \vdash_T F : \mathsf{bool}$. Since the validity assumptions use the same context as the conclusion, it follows that they are both proper with uniquely determined context. As observed in the beginning of the proof if a proper assumption of a rule is an equality over a type $\overline{A}$ (here $A = \overline{A} = \mathsf{bool}$), the induction hypothesis implies that the quasi-preimage of that assumption in which the equality is over type bool must be well-formed. Clearly, $\Gamma \vdash_T F' =_{\mathsf{bool}} F$ and $\Gamma \vdash_T F$ are the quasi-preimages of the two validity assumptions. Since the former is a validity statement about the quasi-preimage of an equality, it follows that $\Gamma \vdash_T F' =_{\mathsf{bool}} F$ is well-formed. We have already seen that $\Gamma \vdash_T F$ is well-typed. This shows the first part of the claim.

Regarding the second part:

$\Gamma \vdash_T F' =_{\mathsf{bool}} F$	By Assumption	(293)
$\Gamma \vdash_T F'$	By Assumption	(294)
$\Gamma \vdash_T F$	(cong $\vdash$ '),(293),(294)	

(β): Since the rule has no validity assumptions, the first part of the claim trivially holds.

Since the conclusion is proper, it follows that the preimage

$$\Gamma \vdash_T (\lambda x{:}A.\ s)\ t =_{B[x/t]} s[x/t]$$

of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} B^*\ (\lambda x{:}\overline{A}.\ \widetilde{s})\ \widetilde{t}\ \widetilde{s}[x/\widetilde{t}]$$

of the conclusion is well-formed. By rule (eqTyping), we obtain $\Gamma \vdash_T (\lambda x{:}A.\ s)\ t : B[x/t]$ in DHOL. Clearly, $\Gamma \vdash_T (\lambda x{:}A.\ s)\ t : B[x/t]$ is a quasi-preimage of the assumption of the rule, so we have proven that the quasi-preimage of the assumption of the rule holds in DHOL.

Regarding the second part:

$\Gamma \vdash_T \Gamma \vdash_T (\lambda x{:}A.\ s)\ t : B[x/t]$	see above	(295)
$\Gamma \vdash_T \Gamma \vdash_T (\lambda x{:}A.\ s)\ t =_{B[x/t]} s[x/t]$	(β'),(295)	

(refl): Once again the rule has no validity assumptions, so the first part of the claim trivially holds.

Since the conclusion is proper, it follows that the preimage

$$\Gamma \vdash_T t =_A t'$$

of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} A^*\ \widetilde{t}\ \widetilde{t}$$

of the conclusion is well-formed. By Lemma 6.5 it follows that t and t' are identical so the quasi-preimage is $\Gamma \vdash_T t =_A t$. By rule (eqTyping), we obtain $\Gamma \vdash_T t : A$ in DHOL, the quasi-preimage of the assumption of the rule.

Regarding the second part of the claim:

$$\Gamma \vdash_T t : A \qquad \text{see above} \qquad (296)$$
$$\Gamma \vdash_T t =_A t \qquad (\text{refl'}),(296)$$

(sym): Since the conclusion is proper, it follows that the preimage

$$\Gamma \vdash_T t =_A s$$

of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} A^* \; \widetilde{t} \; \widetilde{s}$$

of the conclusion is well-formed. By the rules (eqTyping) and (sym) both $\Gamma \vdash_T t : A$ and $\Gamma \vdash_T s : A$ follow. By rule (=) it follows that $\Gamma \vdash_T s =_A t$ is well-formed. Clearly, $\Gamma \vdash_T s =_A t$ is a quasi-preimage of the validity assumption, so we have proven the first part of the claim.

Regarding the second part:

$$\Gamma \vdash_T t =_A s \qquad \text{By Assumption} \qquad (297)$$
$$\Gamma \vdash_T s =_A t \qquad (\text{sym'}),(297) \qquad (298)$$

(assume): Once again, there are no validity assumption, so the first part of the claim is trivial.

Since the conclusion is proper, it follows that the preimage

$$\Gamma \vdash_T F$$

of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} \widetilde{F}$$

of the conclusion is well-formed and thus $\Gamma \vdash_T F : \mathsf{bool}$.

$$\Gamma \vdash_T F : \mathsf{bool} \qquad \text{see above} \qquad (299)$$
$$\Gamma \vdash_T \mathsf{bool}\ \mathsf{tp} \qquad (\text{typingTp}),(299) \qquad (300)$$
$$\vdash_T \Gamma\ \mathsf{Ctx} \qquad (\text{tpCtx}),(300) \qquad (301)$$

The context assumption may be the translation of a context assumption in DHOL or a typing assumption added by the translation. In the latter case, $\widetilde{F}$ is of the form $\widetilde{F} = A^*\ x\ x$ for $x : A$ in Γ. In that case, the second part of the claim $\Gamma \vdash_T F$ can be concluded as follows:

$$\Gamma \vdash_T x : A \qquad (\text{var'}),(\text{congBase'}) \qquad (302)$$
$$\Gamma \vdash_T x =_A x \qquad (\text{refl'}),(302)$$

Otherwise:

$$ass \triangleright F \text{ in } \Gamma \qquad \text{By Assumption} \qquad (303)$$
$$\Gamma \vdash_T F \qquad (\text{assume'}),(303),(301)$$

(axiom): Once again, there are no validity assumption, so the first part of the claim is trivial.

Since the conclusion is proper, it follows that the preimage

$$\Gamma \vdash_T F$$

of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} \widetilde{F}$$

of the conclusion is well-formed and thus $\Gamma \vdash_T F : \mathsf{bool}$.

$\Gamma \vdash_T F : \mathsf{bool}$	see above	(304)
$\Gamma \vdash_T \mathsf{bool}\ \mathsf{tp}$	(typingTp),(304)	(305)
$\vdash_T \Gamma\ \mathsf{Ctx}$	(tpCtx),(305)	(306)

The axiom may be the translation of an axiom in T, a typing axiom added by the translation or an axiom added for some base type A. In the first case, the second part of the claim follows by:

$ax \triangleright F$ in T	By Assumption	(307)
$\Gamma \vdash_T F$	(axiom'),(307),(306)	

If the axiom is a typing axiom then its preimage states that some constant c of type A satisfies $\mathsf{c} =_A \mathsf{c}$ which follows by rule (refl).

If the axiom is the PER axiom generated for some A type declared in T, then it's quasi-preimage states that equality on A implies itself which is obviously true.

($\Rightarrow$I): Since the conclusion is proper, it follows that the preimage

$$\Gamma \vdash_T F \Rightarrow G$$

of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} \widetilde{F} \Rightarrow \widetilde{G}$$

of the conclusion is well-formed and thus $\Gamma \vdash_T F \Rightarrow G : \mathsf{bool}$.

$\Gamma \vdash_T F \Rightarrow G : \mathsf{bool}$	see above	(308)
$\Gamma \vdash_T F : \mathsf{bool}$	(implTypingL),(308)	(309)
$\Gamma \vdash_T G : \mathsf{bool}$	(implTypingR),(308)	(310)
$\Gamma, ass \triangleright F \vdash_T G : \mathsf{bool}$	(as $\vdash$),(309),(310)	

Obviously $\Gamma, ass \triangleright F \vdash_T G$ is a quasi-preimage of the validity assumption of the rule, so the first part of the claim is proven.

Regarding the second part:

$\Gamma, ass \triangleright F \vdash_T G$	By Assumption	(311)
$\Gamma \vdash_T F \Rightarrow G$	($\Rightarrow$I'),(309),(311)	

(⇒E): Since the conclusion is proper, it follows that the preimage

$$\Gamma \vdash_T G$$

of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} \widetilde{G}$$

of the conclusion is well-formed and thus $\Gamma \vdash_T G : \mathsf{bool}$.

Since the validity assumptions use the same context as the conclusion, it follows that they are both proper and uniquely determined.

Since the formula $\widetilde{F}$ (where $\overline{\Gamma} \vdash_{\overline{T}} \widetilde{F}$ is the second validity assumption) must be almost proper, it follows that its preimage F is well-typed i.e. $\Gamma \vdash_T F : \mathsf{bool}$.

$\Gamma \vdash_T F : \mathsf{bool}$	$\widetilde{F}$ almost proper	(312)
$\Gamma \vdash_T G : \mathsf{bool}$	see above	(313)
$\Gamma \vdash_T F \Rightarrow G : \mathsf{bool}$	(⇒'),(312),(313)	

Clearly, $\Gamma \vdash_T F \Rightarrow G$ and $\Gamma \vdash_T F$ are quasi-preimages of the two validity assumptions of the rule, so we have proven the first part of the claim.

Regarding the second part:

$\Gamma \vdash_T F \Rightarrow G$	By Assumption	(314)
$\Gamma \vdash_T F$	By Assumption	(315)
$\Gamma \vdash_T G$	(⇒E'),(314),(315)	

(boolExt): Since the conclusion is proper, it follows that the preimage

$$\Gamma \vdash_T \forall x{:}\mathsf{bool}.\ p\ x$$

of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} \forall x{:}\mathsf{bool}.\ \forall y{:}\mathsf{bool}.\ \mathsf{bool}^*\ x\ y \Rightarrow \mathsf{bool}^*\ ((\lambda x{:}\mathsf{bool}.\ \mathsf{true})\ x)\ (\widetilde{p}\ y)$$

of the conclusion is well-formed and thus $\Gamma \vdash_T (\forall x{:}\mathsf{bool}.\ p\ x) : \mathsf{bool}$. Expanding the definition of ∀ yields:

$\Gamma \vdash_T (\lambda x{:}\mathsf{bool}.\ \mathsf{true} =_{\Pi x{:}\mathsf{bool}.\ \mathsf{bool}} \lambda x{:}\mathsf{bool}.\ p\ x) : \mathsf{bool}$	see above	(316)
$\Gamma \vdash_T \lambda x{:}\mathsf{bool}.\ p\ x : \Pi x{:}\mathsf{bool}.\ \mathsf{bool}$	(eqTyping),(sym'),(316)	(317)

Clearly we have $\mathsf{true} : \mathsf{bool}$ and $\mathsf{false} : \mathsf{bool}$:

$\Gamma \vdash_T (\lambda x{:}\mathsf{bool}.\ p\ x)\ \mathsf{true} : \mathsf{bool}$	(appl'),(317),explanation	(318)
$\Gamma \vdash_T (\lambda x{:}\mathsf{bool}.\ p\ x)\ \mathsf{false} : \mathsf{bool}$	(appl'),(317),explanation	(319)
$\Gamma \vdash_T p\ \mathsf{true} =_{\mathsf{bool}} (\lambda x{:}\mathsf{bool}.\ p\ x)\ \mathsf{true}$	(sym'),(β'),(318)	(320)
$\Gamma \vdash_T p\ \mathsf{false} =_{\mathsf{bool}} (\lambda x{:}\mathsf{bool}.\ p\ x)\ \mathsf{false}$	(sym'),(β'),(319)	(321)

$\Gamma \vdash_T p\ \mathsf{true} : \mathsf{bool}$ (eqTyping),(320)

$\Gamma \vdash_T p\ \mathsf{false} : \mathsf{bool}$ (eqTyping),(321)

Since $\Gamma \vdash_T p\ \mathsf{true}$ and $\Gamma \vdash_T p\ \mathsf{false}$ are clearly quasi-preimages of the two validity assumptions of the rule, we have proven the first part of the claim.

Regarding the second part:

$\Gamma \vdash_T p\ \mathsf{true}$	By Assumption	(322)
$\Gamma \vdash_T p\ \mathsf{false}$	By Assumption	(323)
$\Gamma \vdash_T \forall x{:}\mathsf{bool}.\ p\ x$	(boolExt'),(322),(323)	(324)

(εE): Since the conclusion is well-formed, it follows that the preimage $\Gamma \vdash_T F[^{x}/_{\varepsilon x:A.\ F}]$ of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} \widetilde{F}[^{x}/_{\varepsilon x:\overline{A}.\ A^{*}\ x\ x\ \wedge\ \widetilde{F}}]$$

of the conclusion is well-formed. If x is a free variable of F, the well-formedness of that preimage and the fact that typing rule (ε') is the only rule that can show well-typedness of choice terms, imply that both $\Gamma,\ x : A \vdash_T F : \mathsf{bool}$ and $\Gamma \vdash_T \exists x{:}A.\ F$ hold. If x doesn't occur in F, then the well-formedness of the preimage yields $\Gamma \vdash_T F : \mathsf{bool}$ and $\Gamma \vdash_T \exists x{:}A.\ F : \mathsf{bool}$ follows. This completes the proof for the first part of the claim in this case.

Regarding the second part of the claim for this rule:

$\Gamma \vdash_T \exists x{:}A.\ F$	By Assumption	(325)
$\Gamma,\ x : A \vdash_T F : \mathsf{bool}$	see above	(326)
$\Gamma \vdash_T F[^{x}/_{\varepsilon x:A.\ F}]$	(εE'),(326),(325)	(327)

(nonEmpty): Since the conclusion is well-formed, it follows that the preimage $\Gamma \vdash_T F$ of the replacement

$$\overline{\Gamma} \vdash_{\overline{T}} \widetilde{F}$$

of the conclusion is well-formed. Since the type-indices must be well-formed DHOL types, it follows that A is well-formed. According to rule (var$\vdash$), we then yield $\Gamma,\ x : A \vdash_T F : \mathsf{bool}$. Regarding the second part: By choice of the type indices (last property in Lemma 6.14) we know that the type A is non-empty in DHOL, so there is some term t with $\Gamma \vdash_T t : A$.

$\Gamma,\ x : A \vdash_T F$	By Assumption	(328)
$\Gamma \vdash_T t : A$	By Assumption	(329)
$\Gamma \vdash_T \forall x{:}A.\ F$	($\forall$I),(328)	(330)

Since $\Gamma \vdash_T F : \mathsf{bool}$, it follows that x doesn't occur in F:

$\Gamma \vdash_T F$	Explanation,($\forall$E),(330),(329)

□

7 Type-Checking

Theorem 6.4 allows for lifting only validity judgments from HOL to DHOL. There is no corresponding theorem for the other judgments, in particular for the type-checking judgment, which occurs as a prerequisite of Thm. 6.4. Consequently, we need a type-checker for DHOL before we can use the translation to obtain a theorem prover. But because type-checking is also undecidable, Thm. 6.4 by itself is not sufficient to obtain a theorem prover for DHOL.

We need to close this gap by reducing all judgments that occur during type-checking to instances of the validity judgment $\Gamma \vdash_T F$ in such a way that $\Gamma \vdash_T F : \mathsf{bool}$ is guaranteed. Inspecting the rules of DHOL, we observe that all DHOL-judgments would be decidable if we had an oracle $\Gamma \vdash_T F$. Indeed, our DHOL-rules are already written in a way that essentially allows reading off this reduction algorithm. It only remains to show that this algorithm only calls the oracle on well-formed formulae so that Thm. 6.4 is applicable. Formally, we show the following:

Theorem 7.1. *Consider a DHOL-derivation that is partial in such a way that all subderivations for subgoals of the form $\Gamma \vdash_T F$ are still missing. If these subgoals are discharged in left-to-right depth-first order (meaning the subderivations of each node are visited in the left-to-right order of the rule's hypotheses and all subsubderivations are (recursively) visited before moving on to the next subderivation), then each such subgoal satisfies $\Gamma \vdash_T F : \mathsf{bool}$.*

Proof. We prove, by induction on derivations, the more general statement that each subderivation satisfies the following preconditions:

Judgment	Precondition
$\vdash_T \Gamma\ \mathsf{Ctx}$	$\vdash T\ \mathsf{Thy}$
$\Gamma \vdash_T A\,\mathsf{tp}$	$\vdash_T \Gamma\ \mathsf{Ctx}$
$\Gamma \vdash_T t : A$	$\Gamma \vdash_T A\,\mathsf{tp}$
$\Gamma \vdash_T A \equiv B$	$\Gamma \vdash_T A\,\mathsf{tp}$ and $\Gamma \vdash_T B\,\mathsf{tp}$
$\Gamma \vdash_T F$	$\Gamma \vdash_T F : \mathsf{bool}$

This is similar to the proof of the first four rules in Lemma 2.2, except that we are proving it by induction on DHOL (not HOL) derivations. The induction is straightforward.

The most interesting case is the one for rule (congBase'). It derives $\Gamma \vdash_T \mathsf{a}\ s_1\ \ldots\ s_n \equiv \mathsf{a}\ t_1\ \ldots t_n$ from $\Gamma \vdash_T s_i =_{A_i} t_i$ for each i. Here, the left-to-right order of hypotheses is critical: $\Gamma \vdash_T s_1 =_{A_1} t_1$ is needed to show that $\Gamma \vdash_T s_2 =_{A_2[x_1/t_1]} t_2 : \mathsf{bool}$, and so on. □

Our rules for DHOL are optimized for simplicity. To obtain a practical type-checker, we need to employ a bidirectional algorithm, where the typing judgment $\Gamma \vdash_T t : A$ is split into a type-inference judgment $\Gamma \vdash_T t :^i A$ (which returns A) and a type-checking judgment $\Gamma \vdash_T t :^c A$ (which receives A as input and returns yes/no). The details are routine. For example, the rules corresponding to the typing rules (λ'), (appl'), (const') and (var') become

$$\frac{\Gamma,\ x : A \vdash_T t :^i B}{\Gamma \vdash_T (\lambda x{:}A.\ t) :^i \Pi x{:}A.\ B} \qquad \frac{\Gamma \vdash_T f :^i \Pi x{:}A.\ B \qquad \Gamma \vdash_T t :^c A}{\Gamma \vdash_T f\ t :^i B[x/t]} \qquad \frac{\mathsf{c} : A \text{ in } T}{\Gamma \vdash_T \mathsf{c} :^i A} \qquad \frac{x : A \text{ in } \Gamma}{\Gamma \vdash_T x :^i A}$$

And we add the type-checking rule:

$$\frac{\Gamma \vdash_T t :^i A' \qquad \Gamma \vdash_T A' \equiv A}{\Gamma \vdash_T t :^c A}$$

We have implemented the resulting algorithm in our MMT/LF logical framework [18, 26]. As the oracle for the validity judgment, it uses Thm. 6.4 in combination with a theorem prover for HOL. We describe the details in Sect. 8.

8 Theorem Proving

TPTP representation. To obtain a theorem prover for DHOL, we apply our translation and run a HOL theorem prover. There are two ways to run the translation: within our type checker for DHOL, thus effectively using the HOL ATP as a hammer tool, or as a preprocessor of a HOL ATP, thus effectively extending it to a DHOL ATP. We have implemented both approaches.

In both cases, we used TPTP as the interface language for the ATP system. Although TPTP has never standardized a *semantics* for dependent types, the TPTP *syntax* anticipates such an extension of HOL and already supports them. Concretely, TPTP represents

- a declaration of a dependent type symbol $a : \Pi x_1{:}A_1. \ \ldots \Pi x_n{:}A_n.$ tp as the TPTP typed symbol declaration `a:!>[X1:A1,...,Xn:An]:$tType`,
- a base type `a` $t_1 \ldots t_n$ as `a @ t1 ... @ tn`, and
- a function type $\Pi x{:}A.\ B$ as `!>[X:A]:B`.

To our knowledge, no ATP system had made use of this aspect of the syntax before we suggested the above representation as a new TPTP dialect when we introduced DHOL [32]. Since then it has been used in theorem provers for DHOL [21], and our suggestion has been expanded into and adopted as a formal TPTP standard [28].

Implementations. We have implemented a preprocessor for the Leo-III [35] ATP for HOL that accepts DHOL in the above TPTP encoding and translates it to HOL (also in TPTP encoding) by applying our translation. We chose it because its existing preprocessor infrastructure [36] made this step easier.

Since the preprocessor emits TPTP output, and all major HOL ATPs support TPTP input, we can use use the same preprocessor to extend any other HOL ATP as well. [1]

Additionally, we implemented DHOL and the type-checking algorithm from Sect. 7 in the Mmt logical framework [26]. It uses Leo-III as backend and oracle for discharging proof obligations, both the ones collected during type-checking and the ultimate theorem proving goal. Because Mmt and LEO-III are implemented in the same programming language (Scala), the integration does not use TPTP and operates directly in memory. But any other HOL ATP can be used instead by utilizing a TPTP export.[2] [3]

Example 8.1. The example theories given throughout this paper and a few other example conjectures have been formalized in native TPTP and/or in Mmt. [4] In particular, the TPTP representation of the conjecture given in Ex. 3.3 as well

[1] The implementation of this preprocessor can be found at https://github.com/leoprover/logic-embedding/blob/master/embedding-runtime/src/main/scala/leo/modules/embeddings/DHOLEmbedding.scala.
[2] The formalization of DHOL in Mmt is given in https://gl.mathhub.info/MMT/LATIN2/-/blob/devel/source/logic/hol_like/dhol.mmt
[3] The implementation of the translation in Mmt is given at https://gl.mathhub.info/MMT/LATIN2/-/tree/devel/scala/latin2/tptp.
[4] Those examples can be found at https://gl.mathhub.info/MMT/LATIN2/-/blob/devel/source/casestudies/2023-cade

as the result of translating into the HOL dialect of TPTP [6] are given in CategoryTheory/category-theory-lemmas-dhol.p and CategoryTheory/category-theory-lemmas-hol.p in that folder. Unsurprisingly, Leo-III can prove this simple theorem easily.

Using Mmt as framework has the added benefit that we can apply Mmt's notations, type inference, and module system out of the box for DHOL. However, the interconnection between type inference and theorem proving can still be improved: Mmt's type inference algorithm often encounters proof obligations of the form $\vdash_T s =_A t$ where s and t contain meta-variables, whose values are to be inferred. But proof obligations can only be translated to HOL once all meta-variables have been instantiated. Thus, our implementation in Mmt must choose heuristically whether it tries to use a proof obligation to solve a meta-variable or whether it waits for the meta-variable to be solved and then sends the proof obligation to the ATP system. Currently, the former option is chosen if straightforward congruence reasoning can be used to solve the variable.

Finally, following the original publication of DHOL in [32], an alternative theorem prover for DHOL [21] was build. In this paper, Lash (a fork of Satallax with limited features) is extended from HOL to DHOL. There is also a recent paper [30], extending DHOL with choice operators.

Evaluation. In order to evaluate the practical usefulness of the translation we studied various example problems about set theory and category theory. These examples are written in Mmt and take advantage of abbreviation and user-defined notations; this allows using implicit arguments, which are inferred by Mmt and significantly improve readability. [5] Our Mmt implementation of DHOL successfully type-checks all problems and translates them into TPTP problems to be solved by HOL ATPs. We used multiple HOL ATPs for the resulting set of HOL TPTP problems.

For set theory, we used the followings problems about function composition:

(1) associativity of function composition (applied to a given argument)
(2) associativity of function composition
(3) g is a left inverse of f iff f is a right inverse of g
(4) functions with left-inverses are injective
(5) functions with right-inverses are surjective
(6) functions with (left and right) inverses are bijective

Running all provers (with 60s timeout) supported at https://www.tptp.org/cgi-bin/SystemOnTPTP yields that many provers can solve 3 of the problems, Vampire can solve 4 of them and 5 out of the 6 conjectures can be solved by at least 1 system. The detailed results are given in Table 3. A short synopsis of these theorem provers is provided in [7]. Here, no means no proof was found (timeout) and yes means a proof was found within 60s.

For category theory, we used the following problems:

(1) transitivity of being an isomorphism
(2) any object that satisfies the predicate of being the (binary) product is isomorphic to the binary product
(3) commutativity of predicate of being a (binary) product
(4) uniqueness of predicate for being a (binary) product up to isomorphism

[5] They can be found at https://gl.mathhub.info/MMT/LATIN2/-/blob/devel/source/casestudies/2023-cade.

HOL ATP	problem 1	problem 2	problem 3	problem 4	problem 5	problem 6
agsyHOL	no	no	yes	no	no	no
cvc5	yes	no	yes	no	no	yes
E	yes	no	no	yes	no	yes
HOLyHammer	yes	no	no	no	no	yes
Lash	no	no	yes	no	no	yes
LEO-II	yes	no	no	no	no	yes
Satallax	yes	no	yes	no	no	yes
Vampire	yes	no	yes	no	yes	yes
Zipperpin	yes	no	no	no	yes	yes
total	7	0	6	1	2	8

Fig. 3. Prover results for Function composition theorems

(5) commutativity of the predicate of the (binary) product up to isomorphism
(6) associativity of (binary) products

As suggested by an anonymous IJCAR reviewer, we formalized these problems twice: once in DHOL and automatically translated to HOL, and once natively in HOL. We did this to evaluate whether artefacts introduced by the translation, which is defined for the general case, yield a significant performance penalty compared to a manual formalization in HOL, which can be simpler in special cases. For category theory, for example, the translated DHOL problems use PERs to track which morphisms (with what domain and codomain) are composable whereas the native HOL problems use slightly simpler custom function symbols. Running all provers (with a 60 second timeout) yields the results summarized in Fig. 4 (where we omit the provers that proved none of the problems).

As one can see in the tables, there is no large performance difference between the translated DHOL and the native HOL formalizations. However, both are difficult for current HOL ATPs. In any case, the DHOL formalizations are significantly easier to write and more readable: the native HOL formalizations must use predicates encoding the domain and codomain of morphisms, making the formalization more cumbersome, whereas the more expressive type system makes DHOL formalization simpler and more readable.

Overall more problems generated from the native HOL formalization can be solved by some HOL ATP (5/6 compared to 3/6 for the DHOL formalization). But analyzing the test results for individual provers reveals that in 8 cases a DHOL conjecture but not its native HOL version can be proven by a HOL ATP and in 13 cases the converse, indicating that both DHOL and native HOL formalizations have different advantages. Overall the HOL ATPs found 25 successful proofs for the native HOL problems and 20 for the DHOL problems. This suggests that current HOL ATPs can prove native HOL problems somewhat better than their translated DHOL counterparts, but not much better.

Furthermore, our translation has so far been engineered for generality and soundness/completeness and not for ATP efficiency. Indeed, future work has multiple options to boost the ATP performance on translated DHOL, e.g., by

HOL ATP	problem 1 proved		problem 2 proved		problem 3 proved	
	DHOL	native HOL	DHOL	native HOL	DHOL	native HOL
agsyHOL	yes	no	no	no	yes	no
cocATP	yes	no	no	no	no	no
cvc5	yes	yes	no	no	yes	no
cvc5-SAT	yes	no	no	no	no	no
E	yes	yes	no	no	no	yes
HOLyHammer	yes	yes	no	no	yes	yes
Lash	yes	yes	no	no	no	no
LEO-II	yes	no	no	no	no	no
Leo-III	yes	yes	no	no	no	no
Leo-III-SAT	yes	yes	no	no	no	no
Satallax	yes	yes	no	no	yes	no
Vampire	yes	yes	no	no	no	yes
Zipperpin	yes	yes	no	no	yes	yes
total	13	9	0	0	5	4

HOL ATP	problem 4 proved		problem 5 proved		problem 6 proved	
	DHOL	native HOL	DHOL	native HOL	DHOL	native HOL
agsyHOL	no	no	no	no	no	no
cocATP	no	no	no	no	no	no
cvc5	no	yes	no	no	no	no
cvc5-SAT	no	no	no	no	no	no
E	no	yes	no	yes	no	yes
HOLyHammer	no	yes	no	no	no	yes
Lash	no	no	no	no	no	no
LEO-II	no	no	no	no	no	no
Leo-III	no	no	no	no	no	no
Leo-III-SAT	no	no	no	no	no	no
Satallax	no	no	yes	no	no	no
Vampire	no	yes	no	yes	no	yes
Zipperpin	no	yes	yes	yes	no	yes
total	0	5	2	3	0	4

Fig. 4. Prover Results for Category Theory

- developing sufficient criteria for when simpler HOL theories can be produced,[6]
- inserting lemmas into the translated theories that guide proof search in ATPs, e.g., to speed up equality reasoning, and
- adding definitions to translated DHOL problems and developing better criteria when to expand them.

Thus, we consider the test results to be promising. In particular, the translation could serve as a useful basis for type-checkers and hammer tools for DHOL ITPs.

9 Future Extensions

We sketch a few language extensions that we believe are desirable and feasible. In particular, because DHOL requires undecidable typing, it is attractive to consider other language features that require undecidable typing because, relative to that price already being paid, they become relatively cheap.

Subtying. We can extend the grammar with productions

$$A ::= A|p \mid A/r$$

for a unary predicate $p : A \to \mathsf{bool}$ and an equivalence relation $r : A \to A \to \mathsf{bool}$. The former represents the subtype of A containing the elements $x : A$ that satisfy $p\,x$. The latter represents the quotient of A by r.

Both types are inherently difficult to realize in standard HOL because they have terms occurring in types and their membership resp. equality judgments are undecidable. But they are easy to add to DHOL. Indeed, our dependency erasure translation can be extended by dropping p and r; then we recover the type information by using p and r to define the PERs on the translated types. See [31] for details.

Partial Functions and Undefined Terms. As pointed out by Bill Farmer, we can consider a variant of DHOL in which all functions are partial instead of total. This is inspired by variants of HOL with partial functions such as [15]. One of the biggest arguments against partial functions is that they require a language in which terms can be undefined in an undecidable way. But again this is easy to realize in DHOL.

An additional advantage of using partial functions is that the HOL translation would not need to use PERs: instead, it would suffice to use unary refinement predicates. PERs are needed for total functions because refinements are not closed under function formation: the type of functions between refinements of A and B cannot be a refinement of $A \to B$, it must be a quotient. But the type of *partial* functions between refinements of A and B can be written as a refinement of the type of partial functions from A to B. This would make the translation simpler and might improve the performance of theorem provers. However, we have not investigated yet how to handle *dependently* typed partial functions.

Polymorphism. Finally, shallow polymorphism (where all declarations in theories may take type arguments $\alpha : \mathsf{tp}$) is a well-known extension of HOL that is already supported by TPTP and some theorem provers. We expect it to be straightforward to extend the DHOL syntax and type system with polymorphism.

[6] In [21] it is shown that for simple types in DHOL the translation can be simplified. This is already a first simplifying result. We expect that even more simplifications are possible.

An open question is what kind of type-like arguments to allow. The easiest case considers only type arguments $\alpha : \mathsf{tp}$. But we could also allow type operator arguments $\alpha : \mathsf{tp} \to \ldots \to \mathsf{tp} \to \mathsf{tp}$. Moreover, we could combine polymorphism with dependent types and allow for arguments $\alpha : (\Pi x : A.)^*\mathsf{tp}$.

The translation would translate every DHOL-type variable α to, in HOL, a type variable α, a term-variable $\alpha^* : \alpha \to \alpha \to \mathsf{bool}$, and an assumption that α^* is a PER. We are currently investigating whether the soundness and completeness proofs carry over to this setting. See [29] for details.

10 Conclusion

We have combined two common languages used as basis of proof assistants: higher-order logic HOL and dependent type theory DTT, thereby obtaining the new dependently-typed higher-order logic DHOL. Contrary to HOL, DHOL allows for *dependent* function types. Contrary to DTT, DHOL retains the simplicity of classical Booleans and standard equality.

On the downside, we have to accept that DHOL, unlike both HOL and DTT, has an undecidable type system. Preliminary testing suggests that this disadvantage is not too large in practical theorem proving applications if the DHOL implementation integrates a strong ATP system for HOL. This design is enabled by a sound and complete translation from DHOL into HOL that allows discharging DHOL proof obligations generated during type-checking by existing HOL ATPs. We have implemented this translation as a TPTP-to-TPTP preprocessor for HOL ATP systems and outlined the implementation of a type-checker and hammer tool for DHOL based on the resulting prover. Additionally, we built a minimal DHOL system that implements type-checking and theorem proving.

Following the same general design it is possible to add various other type constructors to DHOL that are often requested by users but usually difficult to provide for system developers as they make typing undecidable. Predicate subtypes, quotient types, partial functions and description operators are examples of such features.

We expect but have not proved that our translation remains sound and complete if DHOL is extended with other features underlying common HOL systems such as built-in types for numbers, the axiom of infinity, polymorphism, or the subtype definition principle.

Acknowledgments

We thank the (anonymous) reviewers for their very valuable feedback.

Partially funded by the European Union (GA 101039196). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the European Research Council can be held responsible for them.

References

[1] Allen, S.: A Non-type-theoretic Semantics for Type-theoretic Language. Ph.D. thesis, Cornell University (1987)

[2] Andrews, P.: An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof. Academic Press (1986), https://doi.org/10.1007/978-94-015-9934-4

[3] Andrews, P., Bishop, M., Issar, S., Nesmith, D., Pfenning, F., Xi, H.: TPS: A Theorem-Proving System for Classical Type Theory. Journal of Automated Reasoning **16**(3), 321–353 (1996), https://doi.org/10.1007/BF00252180

[4] Bancerek, G., Byliński, C., Grabowski, A., Korniłowicz, A., R. Matuszewski, A.N., Pąk, K., Urban, J.: Mizar: State-of-the-art and beyond. In: Kerber, M., Carette, J., Kaliszyk, C., Rabe, F., Sorge, V. (eds.) Intelligent Computer Mathematics. p. 261–279. Springer International Publishing, Cham (2015)

[5] Bart, J., Melham, T.: Translating dependent type theory into higher order logic. In: Bezem, M., Groote, J.F. (eds.) Typed Lambda Calculi and Applications. p. 209–229. Springer Berlin, Heidelberg (1993), https://doi.org/10.1007/BFb0037108

[6] Benzmüller, C., Rabe, F., Sutcliffe, G.: THF0 – The core of the TPTP Language for Higher-Order Logic. In: Armando, A., Baumgartner, P., Dowek, G. (eds.) 4th International Joint Conference on Automated Reasoning. pp. 491–506. Springer (2008), https://doi.org/10.1007/978-3-540-71070-7_41

[7] Benzmüller, C., Andrews, P.: Church's Type Theory. In: Zalta, E.N., Nodelman, U. (eds.) The Stanford Encyclopedia of Philosophy. Metaphysics Research Lab, Stanford University, Spring 2024 edn. (2024), https://plato.stanford.edu/archives/spr2024/entries/type-theory-church/

[8] Benzmüller, C., Sultana, N., Paulson, L.C., Theiß, F.: The Higher-Order Prover Leo-II. Journal of Automated Reasoning **55**, 389–404 (2015), https://doi.org/10.1007/s10817-015-9348-y

[9] Brown, C.E.: Satallax: An automatic higher-order prover. In: Gramlich, B., Miller, D., Sattler, U. (eds.) Automated Reasoning. p. 111–117. Springer Berlin, Heidelberg (2012), https://doi.org/10.1007/978-3-642-31365-3_11

[10] Church, A.: A Formulation of the Simple Theory of Types. Journal of Symbolic Logic **5**(1), 56–68 (1940), https://doi.org/10.2307/2266170

[11] Constable, R., Allen, S., Bromley, H., Cleaveland, W., Cremer, J., Harper, R., Howe, D., Knoblock, T., Mendler, N., Panangaden, P., Sasaki, J., Smith, S.: Implementing Mathematics with the Nuprl Development System. Prentice-Hall (1986)

[12] Coq Development Team: The Coq Proof Assistant: Reference Manual. Tech. rep., INRIA (2015)

[13] Coquand, T., Huet, G.: The Calculus of Constructions. Information and Computation **76**(2/3), 95–120 (1988), https://doi.org/10.1016/0890-5401(88)90005-3

[14] de Moura, L., Kong, S., Avigad, J., van Doorn, F., von Raumer, J.: The Lean Theorem Prover (System Description). In: Felty, A., Middeldorp, A. (eds.) Automated Deduction. p. 378–388. Springer International Publishing, Cham (2015), https://doi.org/10.1007/978-3-319-21401-6_26

[15] Farmer, W.: A simple type theory with partial functions and subtypes. Annals of Pure and Applied Logic **64**(3), 211–240 (1993), https://doi.org/10.1016/0168-0072(93)90144-3

[16] Gordon, M.: HOL: A Proof Generating System for Higher-Order Logic. In: Birtwistle, G., Subrahmanyam, P. (eds.) VLSI Specification, Verification and Synthesis, p. 73–128. Kluwer-Academic Publishers (1988), https://doi.org/10.1007/978-1-4613-2007-4_3

[17] Gordon, M., Pitts, A.: The HOL Logic. In: Gordon, M., Melham, T. (eds.) Introduction to HOL, Part III, p. 191–232. Cambridge University Press (1993)

[18] Harper, R., Honsell, F., Plotkin, G.: A framework for defining logics. Journal of the Association for Computing Machinery **40**(1), 143–184 (1993), https://doi.org/10.1145/138027.138060

[19] Harrison, J.: HOL Light: A Tutorial Introduction. In: Proceedings of the First International Conference on Formal Methods in Computer-Aided Design. p. 265–269. Springer Berlin, Heidelberg (1996), https://doi.org/10.1007/BFb0031814

[20] Martin-Löf, P.: An Intuitionistic Theory of Types: Predicative Part. In: Proceedings of the '73 Logic Colloquium, pp. 73–118. North-Holland (1974), https://doi.org/10.2307/2274116

[21] Niederhauser, J., Brown, C., Kaliszyk, C.: Tableaux for automated reasoning in dependently-typed higher-order logic. In: Benzmüller, C., Heule, M., Schmidt, R. (eds.) 12th International Joint Conference on Automated Reasoning. pp. 86–104. Springer (2024), https://doi.org/10.1007/978-3-031-63498-7_6

[22] Nipkow, T., Paulson, L.C., Wenzel, M.: Isabelle/HOL - A Proof Assistant for Higher-Order Logic, Lecture Notes in Computer Science, vol. 2283. Springer (2002), https://doi.org/10.1007/3-540-45949-9

[23] Norell, U.: The Agda WiKi (2005), http://wiki.portal.chalmers.se/agda

[24] Owre, S., Rushby, J., Shankar, N.: PVS: A Prototype Verification System. In: Kapur, D. (ed.) 11th International Conference on Automated Deduction (CADE). pp. 748–752. Springer (1992), https://doi.org/10.1007/3-540-55602-8_217

[25] Pfenning, F., Schürmann, C.: System description: Twelf — a meta-logical framework for deductive systems. In: Ganzinger, H. (ed.) Automated Deduction. p. 202–206 (1999), https://doi.org/10.1007/3-540-48660-7_14

[26] Rabe, F.: A Modular Type Reconstruction Algorithm. ACM Transactions on Computational Logic **19**(4), 1–43 (2018), https://doi.org/10.1145/3234693

[27] Rahli, V., Bickford, M., Anand, A.: Formal program optimization in nuprl using computational equivalence and partial types. In: Blazy, S., Paulin-Mohring, C., Pichardie, D. (eds.) Interactive Theorem Proving - 4th International Conference, ITP 2013, Rennes, France, July 22-26, 2013. Proceedings. Lecture Notes in Computer Science, vol. 7998, pp. 261–278. Springer (2013), https://doi.org/10.1007/978-3-642-39634-2_20

[28] Ranalter, D., Kaliszyk, C., Rabe, F., Sutcliffe, G.: The Dependently Typed Higher-Order Form for the TPTP World. In: Thiemann, R., Weidenbach, C. (eds.) Frontiers of Combining Systems. Springer (2025), https://link.springer.com/chapter/10.1007/978-3-642-28717-6_32

[29] Ranalter, D., Rabe, F., Kaliszyk, C.: Polymorphic Theorem Proving for DHOL (2025), under review

[30] Ranalter, D., Brown, C., Kaliszyk, C.: Experiments with choice in dependently-typed higher-order logic. In: Bjørner, N., Heule, M., Voronkov, A. (eds.) Proceedings of 25th Conference on Logic for Programming, Artificial Intelligence and Reasoning. EPiC Series in Computing, vol. 100, pp. 311–320. EasyChair (2024), https://doi.org/10.29007/2v8h

[31] Rothgang, C., Rabe, F.: Subtyping in Dependently-Typed Higher-Order Logic. In: Thiemann, R., Weidenbach, C. (eds.) Frontiers of Combining Systems. Springer (2025), https://doi.org/10.1007/978-3-031-38499-8_25

[32] Rothgang, C., Rabe, F., Benzmüller, C.: Theorem Proving in Dependently Typed Higher-Order Logic. In: Pientka, B., Tinelli, C. (eds.) Automated Deduction. pp. 438–455. Springer (2023), https://doi.org/10.1007/978-3-031-38499-8_25

[33] Rothgang, C.: Automated theorem proving for dependent typed theories via a translation to higher-order logic. Masters thesis, FU Berlin (2023), http://dx.doi.org/10.17169/refubium-44093

[34] Steen, A.: An extensible logic embedding tool for lightweight non-classical reasoning (2022), https://doi.org/10.48550/ARXIV.2203.12352, arXiv:2203.12352

[35] Steen, A., Benzmüller, C.: Extensional higher-order paramodulation in Leo-III. Journal of Automated Reasoning **65**(6), 775–807 (2021), https://doi.org/10.1007/s10817-021-09588-x

[36] Steen, A., Sutcliffe, G., Benzmüller, C.: Solving quantified modal logic problems by translation to classical logics. Journal of Logic and Computation pp. 1–23 (2025), https://doi.org/10.1093/logcom/exaf006

[37] Sutcliffe, G.: The TPTP Problem Library and Associated Infrastructure: The FOF and CNF Parts, v3.5.0. Journal of Automated Reasoning **43**(4), 337–362 (2009), https://doi.org/10.1007/s10817-009-9143-8

[38] Sutcliffe, G., Benzmüller, C.: Automated Reasoning in Higher-Order Logic using the TPTP THF Infrastructure. Journal of Formalized Reasoning **3**(1), 1–27 (2010), https://doi.org/10.6092/issn.1972-5787/1710