Research Article

# A “Propositions as Types” Interpretation of Classical Logic

**Andrew Powell**[1]

1. Imperial College London, United Kingdom

**This paper presents a theorem–proof style account of a “propositions as types” interpretation of classical logic. Inhabited and uninhabited types are represented by $\top$ and $\bot$, and a small family of type-level reductions is used to evaluate logical types at the level of truth values. Witness terms are then extracted by interpreting the identity inhabitant of $\bot \to \bot$ polymorphically at a suitable inhabited type supplied by context. The resulting framework is applied to first-order classical propositional logic and to first-, second-, and higher-order classical predicate logic. The emphasis is on a uniform formal presentation while retaining the motivating idea that classical reasoning may be represented without introducing explicit control operators into the term language.**

**Corresponding author:** Andrew Powell, andrew.powell@imperial.co.uk

## 1. Introduction

The Curry–Howard correspondence identifies propositions with types and proofs with programs. In its most familiar form, a proposition is true exactly when the corresponding type is inhabited, and a proof of the proposition is represented by a term inhabiting that type. This viewpoint is especially natural in typed systems of the $\lambda$-calculus, where terms already function as programs and types constrain how those programs may be formed; see for example [1][2][3][4][5][6], and [7].

Throughout, “true” means “evaluates to $\top$” in the Boolean inhabitation semantics fixed below, rather than “derivable as a closed judgment in a particular deductive system”.

The “propositions as types” view of logic is due initially to [8][9] and [10]; for historical discussion see also [11]. A systematic modern treatment may be found in [12].

For intuitionistic logic this correspondence is standard. In particular, existential statements require explicit witnesses, and principles such as double negation elimination do not in general hold. This viewpoint is closely associated with intuitionistic type theory in the sense of Martin-Löf [13][14][15], building on the broader intuitionistic tradition discussed, for example, in [16].

Classical logic can also be related to typed computation, but the usual accounts often proceed by extending the term language with control operators or continuation-based mechanisms. The classic starting point is Griffin's continuation-based interpretation of classical logic [17], followed by a substantial literature including [18][19], and [20]. The aim of this paper is different. We remain within an ordinary typed $\lambda$-calculus and instead separate two levels of reasoning. First, we evaluate types at the level of truth values by using a small collection of reduction rules. Second, when a type is thereby shown to be inhabited, we extract witness terms by using context together with a polymorphic reading of the identity function.

The point of this approach is not to deny that continuation-based interpretations exist, but to isolate a simpler mechanism within an ordinary typed setting. The resulting account is especially suited to logically valid principles, where the central task is not to compute a specific mathematical object but to show that a type corresponding to a logical truth is inhabited.

### *1.1. Strategy of the paper*

The strategy is as follows.

- We represent inhabited and uninhabited types by $\top$ and $\bot$.
- We identify the truth values of the four basic function types
  $\bot \to \bot, \bot \to \top, \top \to \top, \top \to \bot$.
- We adopt a small family of type-level reduction principles, such as
  $i(A \to \bot) = \bot$,
  when $A$ is inhabited.
- We use the identity inhabitant of $\bot \to \bot$ polymorphically, reading it as $i_{A \to A}$ whenever an inhabitant of $A$ is available in context.
- We apply this method to produce witnesses for logically true propositions in propositional, first-order, second-order, and higher-order classical logic.

In many cases these witnesses are *contextual*: they depend on externally supplied inhabitants in a surrounding context rather than being closed canonical terms.

The novelty claimed here is modest and specific. The point is not that classical logic admits some propositions-as-types interpretation, but that many classical principles may be handled uniformly by a small collection of truth-value reductions together with polymorphic identity, without enriching the term language with dedicated control constructs.

## 2. Preliminaries

We work in a typed $\lambda$-calculus with kinds $Type$ and $Prop$, and with a Boolean type

$$Bool := \{\top, \bot\}.$$

We use the same symbols $A, B, \ldots$ for types and the corresponding propositions where there is no risk of confusion. The general background is standard; see [1], [7], and [12].

**Definition 1.** *If $A$ is a type and $a$ is a term, then $a : A$ means that $a$ inhabits $A$. We write informally*

$$A = \top$$

*to mean that $A$ is inhabited, and*

$$A = \bot$$

*to mean that $A$ is uninhabited.*

**Remark 1.** *The meta-logic is classical. Thus for each type $A$, either $A = \top$ or $A = \bot$.*

**Remark 2** (Bracketing conventions). *We adopt the standard association conventions of the typed $\lambda$-calculus. Function types associate to the right, so that*

$$A \to B \to C$$

*means*

$$A \to (B \to C).$$

*Lambda abstraction also associates to the right, so that*

$$\lambda x : A \lambda y : B t$$

*means*

$$\lambda x : A.\, (\lambda y : B.\, t).$$

*Application associates to the left, so that*

$$stu$$

*means*

$$(st)u.$$

*Thus*

$$\lambda x : A \lambda y : B f x y$$

*is parsed as*

$$\lambda x : A.\, (\lambda y : B.\, ((fx)y)).$$

*We nevertheless insert brackets explicitly whenever this improves readability.*

**Definition 2** (Primitive connectives)**.** *Implication is represented by the function type:*

$$A \Rightarrow B \quad \text{corresponds to} \quad A \to B.$$

*Negation is defined by*

$$\neg A := A \to \bot.$$

**Definition 3** (Derived connectives)**.** *We define*

$$A \vee B := (A \to \bot) \to B$$

*and*

$$A \wedge B := (A \to (B \to \bot)) \to \bot.$$

**Remark 3.** *The connectives defined here are intended to capture classical truth conditions under the Boolean inhabitation semantics, not necessarily the usual constructive introduction and elimination rules. In particular, some derived connectives admit only contextual witnesses rather than canonical internal constructors.*

**Remark 4.** *The chosen definition of conjunction is classically equivalent to the usual conjunction, but its introduction and elimination rules below rely on contextual reasoning and polymorphic identity rather than on an internal pairing constructor with projections.*

*The chosen definition of disjunction is classically equivalent to more familiar formulations built from negation and implication. We use the form*

$$A \vee B := (A \to \bot) \to B$$

*because, in the present framework, it leads to especially transparent type-level reductions. In particular, excluded middle takes the form*

$$A \vee \neg A = ((A \to \bot) \to (A \to \bot)).$$

*Consequently, $A \vee B$ does not carry standard left/right tags; it is tailored to make certain classical principles, such as excluded middle, reduce to identity types at the truth-value level.*

**Definition 4** (First-order quantifiers). *Let $A$ be a non-empty type and let $P : A \to Bool$ be a Boolean-valued predicate.*

*A dependent type over $A$ is a family of types indexed by elements of $A$. Thus $P$ may be viewed as assigning to each $x : A$ a truth value $P(x)$, and correspondingly a proposition depending on $x$.*

*A dependent function on such a family is a function $p$ such that, for each $x : A$, the value $p(x)$ has the type assigned to $x$. Accordingly, the universal quantifier is interpreted as the type of dependent functions:*

$$(\forall x : A)P(x)$$

*is inhabited exactly when there is a function $p$ such that, for every $x : A$, $p(x) : P(x)$.*

*We define existential quantification by classical negation:*

$$(\exists x : A)P(x) := ((\forall x : A)(P(x) \to \bot)) \to \bot.$$

**Definition 5** (Higher-order predicate types and quantifiers). *Let $A$ be a non-empty type. Define*

$$TR(1)(A) := A \to Bool, TR(n+1)(A) := TR(n)(A) \to Bool$$

*for $n \in \mathbb{N}, n > 0$.*

*An element of $TR(1)(A)$ is a Boolean-valued predicate on $A$. An element of $TR(2)(A)$ is a Boolean-valued predicate on predicates on $A$, and so on.*

*If $P : TR(n)(A)$ and $S(P)$ is a proposition depending on $P$, then the universal quantifier*

$$(\forall P : TR(n)(A))S(P)$$

*is interpreted as the type of dependent functions assigning to each $P : TR(n)(A)$ a witness of $S(P)$.*

*The corresponding existential quantifier is defined by classical negation:*

$$(\exists P : TR(n)(A))S(P) := ((\forall P : TR(n)(A))(S(P) \to \bot)) \to \bot.$$

**Definition 6** (Polymorphic identity)**.** *We assume an identity operator $i$ such that for each type $A$,*

$$i_{A\to A} : A \to A$$

*and for each $a : A$,*

$$i_{A\to A}(a) = a.$$

*In particular, there is an identity inhabitant*

$$i_{\bot\to\bot} : \bot \to \bot.$$

**Remark 5.** *We treat $i$ as a meta-theoretic polymorphic constant, available at every type $A$, without fixing a particular polymorphic calculus (such as System F) as host; only its extensional behaviour $i_{A\to A}(a) = a$ is used.*

## 3. Truth functions and type-level reductions

We begin with the four basic function types determined by $\top$ and $\bot$.

**Proposition 1.** *The following hold:*

$$\bot \to \bot = \top, \bot \to \top = \top, \top \to \top = \top, \top \to \bot = \bot.$$

*Proof.* The type $\bot \to \bot$ is inhabited by the identity function $i_{\bot\to\bot}$, hence it is inhabited.

The type $\bot \to \top$ is also inhabited. To see this, let $B$ be any inhabited type and choose some $b : B$. A term of type $\bot \to B$ may be written

$$(\lambda x : \bot)b.$$

This is well-typed because a function from $\bot$ has no actual input to process: there is no inhabitant $x : \bot$, so the requirement that the function assign an output to each input is vacuously satisfied. The term simply specifies what the output would be if there were an input, and, since there is none, no contradiction arises. Hence $\bot \to B$ is inhabited whenever $B$ is inhabited, and in particular $\bot \to \top = \top$.

The type $\top \to \top$ is inhabited, for example by the identity on an inhabited type. By contrast, $\top \to \bot$ is uninhabited, since a total function from a non-empty type to the empty type would assign an element of $\bot$ to each inhabitant of $\top$, which is impossible. □

**Remark 6.** *Proposition 1 gives the truth table used throughout the paper. At this stage, nothing specifically classical has yet occurred beyond the adoption of Boolean truth values for inhabitation.*

**Definition 7** (Type-level reduction rules)**.** *Assume $A$ is inhabited. At the truth-value level, we adopt the following recursive reduction principles:*

$$i(A \to \bot) = \bot,$$

$$i((\forall x : A)\bot) = \bot,$$

$$i((\forall P : A \to Bool)\bot) = \bot,$$

*and, more generally,*

$$i((\forall P : TR(n)(A))\bot) = \bot$$

*for all $n > 0$.*

**Remark 7.** *These are semantic rewrite rules on inhabitation truth values, not computational reductions of terms, and are used only at type level. Their purpose is to collapse negated or universally quantified types to $\bot$ whenever the relevant domain is inhabited and some contradictory instance is forced. Double negations of such types then reduce to $\bot \to \bot$, whose canonical inhabitant is identity.*

## 4. Operator view and the role of identity

The original motivation of the paper is clarified by viewing truth functions as operators on types.

**Definition 8.** *We interpret the four basic function types as follows:*

- *$\bot \to \bot$ as an identity type, inhabited by $i_{\bot\to\bot}$;*
- *$\bot \to \top$ as a vacuous function type, inhabited whenever the codomain is inhabited;*
- *$\top \to \top$ as the ordinary case of a type inhabited by a function between inhabited types, with identity as the canonical example;*
- *$\top \to \bot$ as uninhabited at term level, though it may be viewed at type level as a destructor sending an inhabited type to $\bot$.*

**Remark 8.** *The terminology "instantiator" and "destructor" is intended only as an informal guide to the type-level behaviour of these function types. The mathematically important point is simply the inhabitation behaviour described in Proposition 1.*

**Theorem 2** (Double negation at type level). *If $A$ is inhabited, then*

$$(A \to \bot) \to \bot$$

*reduces at type level to*

$$\bot \to \bot.$$

*If $A$ is uninhabited, then*

$$(A \to \bot) \to \bot = \bot.$$

*Proof.*

Assume first that $A$ is inhabited. By Definition 7,

$$i(A \to \bot) = \bot.$$

Hence

$$(A \to \bot) \to \bot$$

reduces to

$$\bot \to \bot.$$

By Proposition 1, this type is inhabited.

Now assume that $A$ is uninhabited. Then $A = \bot$, so $A \to \bot$ is inhabited, hence has truth value $\top$. Therefore

$$(A \to \bot) \to \bot$$

has truth value

$$\top \to \bot = \bot$$

by Proposition 1. □

**Corollary 3.** *A type $A$ is inhabited if and only if $\neg\neg A$ is inhabited.*

*Proof.* Immediate from Theorem 2. □

**Proposition 4** (Contextual witness extraction). *Let $T$ be a type expression occurring in a derivation, and suppose that, under assumptions collected in a context $\Gamma$, the type $T$ reduces at truth-value level to $\bot \to \bot$.*

*Assume further that the same context $\Gamma$ contains an inhabitant $a : A$ of a type $A$ relevant to the construction represented by $T$. Then the identity inhabitant*

$$i_{\bot\to\bot} : \bot \to \bot$$

*may be instantiated polymorphically as*

$$i_{A\to A} : A \to A,$$

*and this instantiated identity may be used in the construction of a witness for $T$ relative to the context $\Gamma$.*

*Proof.* By hypothesis, the type $T$ has the same truth value as $\bot \to \bot$ under the assumptions in $\Gamma$. Hence $T$ is inhabited at the truth-value level.

Now the context $\Gamma$ also contains an inhabitant $a : A$. By the polymorphic identity principle, identity may be instantiated at $A$ to give

$$i_{A\to A} : A \to A.$$

Thus, while the reduction of $T$ to $\bot \to \bot$ does not by itself determine a unique canonical term of type $T$, it shows that the target type has reduced to identity form. The context then supplies the specific inhabited type at which this identity is to be read. Hence a witness for $T$ is obtained relative to $\Gamma$. In particular, the witness for $T$ is *relative* to $\Gamma$; no claim is made that there is a closed canonical inhabitant of $T$ independent of context. □

**Remark 9.** *The truth-value computation alone shows only that a type is inhabited in a coarse Boolean sense. A concrete witness is obtained only relative to a context supplying the inhabited type at which the generic identity witness is to be instantiated.*

## 5. First-order classical propositional logic

The presentation here follows the general spirit of classical logical systems in the tradition of [21], while also using introduction and elimination rules familiar from natural deduction; see [22] and [23].

**Theorem 5** (Implication introduction)**.** *The type*

$$A \to (B \to A)$$

*is inhabited.*

*Proof.* The term

$$(\lambda x : A)(\lambda y : B)x$$

has type

$$A \to (B \to A).$$

□

**Theorem 6** (Modus ponens)**.** *The type*

$$A \to ((A \to B) \to B)$$

*is inhabited.*

*Proof.* The term

$$(\lambda x : A)(\lambda y : A \to B)yx$$

has the required type. □

**Theorem 7** (Falsum introduction)**.** *The type*

$$A \to ((A \to \bot) \to \bot)$$

*is inhabited.*

*Proof.* Suppose first that $A$ is inhabited. Then by Definition 7,

$$A \to \bot$$

reduces to $\bot$, so

$$(A \to \bot) \to \bot$$

reduces to $\bot \to \bot$, which is inhabited by identity. Hence the whole type is inhabited.

A witness may be written explicitly as

$$(\lambda x : A)(\lambda f : A \to \bot)fx.$$

If $A$ is uninhabited, then the domain of the outer implication is empty, so the type is inhabited by ex falso reasoning. □

**Theorem 8** (Contextual double negation elimination)**.** *Let* $\Gamma$ *be a context containing an inhabitant* $a : A$*. Then, relative to* $\Gamma$*, the type*

$$((A \to \bot) \to \bot) \to A$$

*is inhabited.*

*Proof.* Assume $\Gamma \vdash a : A$. Then $A$ is inhabited in the context $\Gamma$. By Definition 7,

$$i(A \to \bot) = \bot.$$

Hence, at truth-value level,

$$(A \to \bot) \to \bot$$

reduces to

$$\bot \to \bot.$$

By Proposition 1, the latter type is inhabited by identity.

Now apply Proposition 4. Since the context $\Gamma$ contains $a : A$, the generic identity inhabitant

$$i_{\bot \to \bot}$$

may be read as

$$i_{A \to A} : A \to A.$$

Accordingly, the type

$$((A \to \bot) \to \bot) \to A$$

is inhabited relative to $\Gamma$.

A corresponding witness term may be written schematically as

$$(\lambda k : (A \to \bot) \to \bot) a,$$

where the term $a : A$ is supplied by the context. □

**Remark 10.** *Theorem 8 does not assert that double negation elimination is witnessed by a closed term constructed independently of assumptions. Rather, once a context already fixes an inhabitant of $A$, the double negation type reduces to identity form and is therefore witnessed schematically by polymorphic identity at $A$.*

**Theorem 9** (Excluded middle)**.** *The type*

$$A \vee \neg A$$

*is inhabited.*

*Proof.* By definition,

$$A \vee \neg A = ((A \to \bot) \to (A \to \bot)).$$

This is an identity type, hence inhabited by

$$i_{(A\to\bot)\to(A\to\bot)}.$$

□

**Remark 11.** *With the present definition of disjunction, excluded middle reduces to a particularly simple identity type. More generally, disjunction is treated here as a classically behaved derived connective whose type-level behaviour is tailored to the reduction rules used throughout the paper.*

**Theorem 10** (Conjunction introduction)**.** *If $A$ and $B$ are inhabited, then*

$$A \wedge B$$

*is inhabited.*

*Proof.* By definition,

$$A \wedge B = (A \to (B \to \bot)) \to \bot.$$

If $A$ and $B$ are inhabited, then $B \to \bot$ reduces to $\bot$, hence

$$A \to (B \to \bot)$$

reduces to $A \to \bot$, which in turn reduces to $\bot$. Therefore

$$(A \to (B \to \bot)) \to \bot$$

reduces to $\bot \to \bot$, hence is inhabited. □

**Theorem 11** (Conjunction elimination, contextual form)**.** *If $A$ and $B$ are inhabited relative to the context supplying $A$ and $B$, then both*

$$A \wedge B \to A \qquad \text{and} \qquad A \wedge B \to B$$

*are inhabited.*

*Proof.* By definition,


doi.org/10.32388/IY3BKH.3

$$A \wedge B = (A \to (B \to \bot)) \to \bot.$$

If $A$ and $B$ are inhabited, then, as in the proof of conjunction introduction,

$$A \wedge B$$

reduces to

$$\bot \to \bot.$$

Hence $A \wedge B$ is inhabited.

Now let $a : A$ and $b : B$. Since $A$ and $B$ are inhabited, the identity inhabitant of

$$\bot \to \bot$$

may be instantiated polymorphically at $A$ and at $B$. Thus

$$A \wedge B \to A$$

and

$$A \wedge B \to B$$

are inhabited relative to the contexts supplying $a : A$ and $b : B$, respectively.

Schematically, witnesses may be written as

$$(\lambda z : A \wedge B)a \qquad \text{and} \qquad (\lambda z : A \wedge B)b.$$

□

**Remark 12.** *Unlike in constructive treatments, we do not define projections $\pi_1, \pi_2 : A \wedge B \to A, B$ that extract components from a canonical pair. Instead, elimination proceeds by reducing $A \wedge B$ to an identity type at the truth-value level and reading identity at contextually supplied inhabitants.*

## 6. First-order predicate logic

We now pass to first-order quantification. The main novelty here is not the witness terms for universal quantification, which are standard, but the treatment of existential quantification by reduction to double negation. General background on such type-theoretic treatments may be found in [12].

**Theorem 12** (Universal introduction)**.** *Let $P : A \to Bool$. If, under the assumption $x : A$, one has a witness*

$$p(x) : P(x),$$

*then*

$$(\forall x : A)P(x)$$

*is inhabited.*

*Proof.* The dependent function

$$(\lambda x : A)p(x)$$

witnesses

$$(\forall x : A)P(x).$$

□

**Theorem 13** (Universal elimination). *If*

$$p : (\forall x : A)P(x)$$

*and* $a : A$*, then*

$$p(a) : P(a).$$

*Equivalently, the type*

$$(\forall x : A)P(x) \to P(a)$$

*is inhabited.*

*Proof.* Assume that $a : A$. Since a witness of

$$(\forall x : A)P(x)$$

is, by definition, a dependent function assigning to each $x : A$ a witness of $P(x)$, application at $a$ yields

$$p(a) : P(a).$$

□

**Theorem 14** (Existential introduction). *If* $\Gamma \vdash a : A$ *and* $\Gamma \vdash p : (\forall x : A)P(x)$*, then, relative to* $\Gamma$*, the type*

$$(\exists x : A)P(x)$$

*is inhabited by a contextual witness depending on the chosen $a$.*

*Proof.* By definition,

$$(\exists x : A)P(x) = ((\forall x : A)(P(x) \to \bot)) \to \bot.$$

Now assume $\Gamma \vdash p : (\forall x : A)P(x)$. By Theorem 13,

$$p(a) : P(a).$$

Then the type

$$P(a) \to \bot$$

is uninhabited in the context $\Gamma$, since any inhabitant of it would send $p(a) : P(a)$ to an inhabitant of $\bot$.

Therefore the universally quantified type

$$(\forall x : A)(P(x) \to \bot)$$

is uninhabited in $\Gamma$, because it would in particular provide an inhabitant of $P(a) \to \bot$ when evaluated at $a$.

By Definition 7, this universal type therefore reduces to $\bot$, and so the existential type reduces to

$$\bot \to \bot.$$

Hence the type is inhabited at the truth-value level.

Finally, since the context $\Gamma$ already contains the witnessing instance $a : A$ and, by Theorem 13, the witness

$$p(a) : P(a),$$

Proposition 4 allows the identity inhabitant of $\bot \to \bot$ to be read at the specific inhabited type determined by $P(a)$. Thus $(\exists x : A)P(x)$ is inhabited relative to $\Gamma$. □

**Theorem 15** (Existential elimination). *Let $A$ be a non-empty type and let $P : A \to Bool$. Assume that*

$$\Gamma \vdash e : (\exists x : A)P(x),$$

*and suppose that for an arbitrary $a : A$, whenever*

$$\Gamma, a : A, p(a) : P(a) \vdash q : Q,$$

*one can derive a witness of $Q$, where $a$ is not free in $Q$. Then $Q$ is inhabited relative to $\Gamma$.*

*Proof.* By definition,

$$(\exists x : A)P(x) = ((\forall x : A)(P(x) \to \bot)) \to \bot.$$

Thus the assumption

$$\Gamma \vdash e : (\exists x : A)P(x)$$

means that the type

$$(\forall x : A)(P(x) \to \bot)$$

cannot be inhabited in the context $\Gamma$.

Now let $a : A$ be arbitrary. If we have

$$p(a) : P(a),$$

then the type

$$P(a) \to \bot$$

is uninhabited, since any inhabitant of it would map the witness $p(a) : P(a)$ to an inhabitant of $\bot$.

Hence the universally quantified type

$$(\forall x : A)(P(x) \to \bot)$$

is uninhabited, because by Theorem 13 any inhabitant of it would in particular yield an inhabitant of

$$P(a) \to \bot$$

when evaluated at $a$.

Therefore, by Definition 7, the type

$$(\exists x : A)P(x) = ((\forall x : A)(P(x) \to \bot)) \to \bot$$

reduces to

$$\bot \to \bot.$$

So any witnessing instance $p(a) : P(a)$ yields the identity type as a witness to

$$(\exists x : A)P(x)$$

by polymorphic identity.

By hypothesis, from such a witnessing instance one may derive a witness of $Q$. Hence $Q$ is inhabited relative to $\Gamma$. □

**Remark 13.** *Again, the conclusion that $Q$ is inhabited is relative to $\Gamma$ and to the witnessing instance(s) used in the derivation of $q : Q$; the framework does not provide a closed eliminator term for $(\exists x : A)P(x)$.*

*Theorem 15 expresses existential elimination in the same two-tier manner as existential introduction: a witnessing instance forces the existential type to reduce to the identity type $\bot \to \bot$, and polymorphic identity then provides the corresponding witness relative to context.*

## 7. Second-order predicate logic

The second-order case follows the first-order pattern closely. The main difference is that quantification now ranges over predicates rather than elements.

**Definition 9.** *Let $A$ be a non-empty type. A second-order predicate variable has type*

$$P : A \to Bool.$$

*If $S$ is a proposition depending on such a predicate, define*

$$(\exists P : A \to Bool)S(P) := ((\forall P : A \to Bool)(S(P) \to \bot)) \to \bot.$$

**Theorem 16** (Second-order universal introduction). *If, under the assumption $P : A \to Bool$, one has a witness of $S(P)$, then*

$$(\forall P : A \to Bool)S(P)$$

*is inhabited.*

*Proof.* Exactly as in the first-order case, by predicate abstraction. □

**Theorem 17** (Second-order universal elimination)**.**

*If*

$$u : (\forall P : A \to Bool)S(P)$$

*and $Q : A \to Bool$, then*

$$u(Q) : S(Q).$$

*Equivalently, the type*

$$(\forall P : A \to Bool) S(P) \to S(Q)$$

*is inhabited.*

*Proof.* A witness is

$$(\lambda u : (\forall P : A \to Bool) S(P)) u(Q).$$

Since a witness of

$$(\forall P : A \to Bool) S(P)$$

assigns to each predicate $P : A \to Bool$ a witness of $S(P)$, application at $Q$ yields

$$u(Q) : S(Q).$$

□

**Theorem 18** (Second-order existential introduction). *If* $\Gamma \vdash Q : A \to Bool$ *and* $\Gamma \vdash s : (\forall P : A \to Bool) S(P)$*, then, relative to* $\Gamma$*,*

$$(\exists P : A \to Bool) S(P)$$

*is inhabited.*

*Proof.* This is the direct analogue of first-order existential introduction, with the quantified variable now ranging over predicates rather than elements.

By definition,

$$(\exists P : A \to Bool) S(P) = ((\forall P : A \to Bool)(S(P) \to \bot)) \to \bot.$$

Since $\Gamma \vdash s : (\forall P : A \to Bool) S(P)$, Theorem 17 yields

$$s(Q) : S(Q).$$

Therefore the type

$$S(Q) \to \bot$$

is uninhabited in context. Hence the universal type

$$(\forall P : A \to Bool)(S(P) \to \bot)$$

is uninhabited and reduces to $\bot$ by Definition 7. Therefore the existential type reduces to $\bot \to \bot$, which is inhabited at the truth-value level. By the same contextual use of polymorphic identity as in the first-order case, the existential type is inhabited relative to $\Gamma$. $\square$

**Theorem 19** (Second-order existential elimination). *Let $A$ be a non-empty type. Assume that*

$$\Gamma \vdash e : (\exists P : A \to Bool) S(P),$$

*and suppose that for an arbitrary predicate $Q : A \to Bool$, whenever*

$$\Gamma, Q : A \to Bool, s(Q) : S(Q) \vdash r : R,$$

*one can derive a witness of $R$, where $Q$ is not free in $R$. Then $R$ is inhabited relative to $\Gamma$.*

*Proof.* By definition,

$$(\exists P : A \to Bool) S(P) = ((\forall P : A \to Bool)(S(P) \to \bot)) \to \bot.$$

Thus the assumption

$$\Gamma \vdash e : (\exists P : A \to Bool) S(P)$$

means that

$$(\forall P : A \to Bool)(S(P) \to \bot)$$

cannot be inhabited in the context $\Gamma$.

Now let $Q : A \to Bool$ be arbitrary. If we have

$$s(Q) : S(Q),$$

then the type

$$S(Q) \to \bot$$

is uninhabited.

Hence the universal type

$$(\forall P : A \to Bool)(S(P) \to \bot)$$

is uninhabited, because by Theorem 17 any inhabitant of it would yield an inhabitant of

$$S(Q) \to \bot$$

when evaluated at $Q$.

Therefore

$$(\exists P : A \to Bool) S(P)$$

reduces to

$$\bot \to \bot.$$

So any witnessing instance $s(Q) : S(Q)$ yields the identity type as a witness to

$$(\exists P : A \to Bool) S(P)$$

by polymorphic identity.

By the elimination hypothesis, such a witnessing instance suffices to derive $R$. Hence $R$ is inhabited relative to $\Gamma$. □

**Remark 14.** *Second-order existential reasoning is completely parallel to the first-order case: the only difference is that the singular witness is now a predicate $Q : A \to Bool$ rather than an element $a : A$. As in the first-order case, the resulting inhabitant of $R$ is contextual and arises via instantiation of identity at a specific witnessing predicate $Q$.*

## 8. Higher-order predicate logic

The higher-order case is a finite iteration of the second-order case. The only change is that the quantified variable ranges over predicates of finite type above $A$.

**Remark 15.** *We continue to use the higher-order predicate hierarchy from Definition 5. Thus a higher-order predicate variable of order $n$ is a variable of type $TR(n)(A)$.*

**Theorem 20** (Higher-order universal introduction)**.** *If, under the assumption $P : TR(n)(A)$, one has a witness of $S(P)$, then*

$$(\forall P : TR(n)(A)) S(P)$$

*is inhabited.*

*Proof.* Exactly as before, by abstraction on the higher-order predicate variable. A witness is of the form


doi.org/10.32388/IY3BKH.3

$$(\lambda P : TR(n)(A))s(P),$$

where $s(P) : S(P)$. □

**Theorem 21** (Higher-order universal elimination). *If*

$$u : (\forall P : TR(n)(A))S(P)$$

*and* $Q : TR(n)(A)$*, then*

$$u(Q) : S(Q).$$

*Equivalently, the type*

$$(\forall P : TR(n)(A))S(P) \to S(Q)$$

*is inhabited.*

*Proof.* A witness is

$$(\lambda u : (\forall P : TR(n)(A))S(P))u(Q).$$

Since a witness of

$$(\forall P : TR(n)(A))S(P)$$

assigns to each $P : TR(n)(A)$ a witness of $S(P)$, application at $Q$ yields

$$u(Q) : S(Q).$$

□

**Theorem 22** (Higher-order existential introduction). *Let* $\Gamma \vdash Q : TR(n)(A)$ *and suppose* $\Gamma \vdash s : (\forall P : TR(n)(A))S(P)$*. Then, relative to* $\Gamma$*,*

$$(\exists P : TR(n)(A))S(P)$$

*is inhabited.*

*Proof.* This is the same argument as in the first-order and second-order cases. The only difference is that the singular witness now has the higher-order predicate type $Q : TR(n)(A)$.

By definition,

$$(\exists P : TR(n)(A))S(P) = ((\forall P : TR(n)(A))(S(P) \to \bot)) \to \bot.$$

Since $\Gamma \vdash s : (\forall P : TR(n)(A))S(P)$, Theorem 21 yields

$$s(Q) : S(Q).$$

Hence the type

$$S(Q) \to \bot$$

is uninhabited. Therefore

$$(\forall P : TR(n)(A))(S(P) \to \bot)$$

is uninhabited and reduces to $\bot$ by Definition 7. Therefore

$$((\forall P : TR(n)(A))(S(P) \to \bot)) \to \bot$$

reduces to $\bot \to \bot$, which is inhabited at the truth-value level. The witness is then obtained by the same polymorphic instantiation of identity as before. □

**Theorem 23** (Higher-order existential elimination)**.** *Let $A$ be a non-empty type and let $n > 0$. Assume that*

$$\Gamma \vdash e : (\exists P : TR(n)(A))S(P),$$

*and suppose that for an arbitrary $Q : TR(n)(A)$, whenever*

$$\Gamma, Q : TR(n)(A), s(Q) : S(Q) \vdash r : R,$$

*one can derive a witness of $R$, where $Q$ is not free in $R$. Then $R$ is inhabited relative to $\Gamma$.*

*Proof.* By definition,

$$(\exists P : TR(n)(A))S(P) = ((\forall P : TR(n)(A))(S(P) \to \bot)) \to \bot.$$

Thus the assumption

$$\Gamma \vdash e : (\exists P : TR(n)(A))S(P)$$

means that the type

$$(\forall P : TR(n)(A))(S(P) \to \bot)$$

cannot be inhabited in the context $\Gamma$.

Now let $Q : TR(n)(A)$ be arbitrary. If we have

$$s(Q) : S(Q),$$

then

$$S(Q) \to \bot$$

is uninhabited.

Hence

$$(\forall P : TR(n)(A))(S(P) \to \bot)$$

is uninhabited, because by Theorem 21 any inhabitant of it would yield an inhabitant of

$$S(Q) \to \bot$$

when evaluated at $Q$.

Therefore

$$(\exists P : TR(n)(A))S(P)$$

reduces to

$$\bot \to \bot.$$

So any witnessing instance $s(Q) : S(Q)$ yields the identity type as a witness to

$$(\exists P : TR(n)(A))S(P)$$

by polymorphic identity.

By the elimination hypothesis, such a witnessing instance suffices to derive $R$. Hence $R$ is inhabited relative to $\Gamma$. □

**Remark 16.** *At higher order the witness pattern is unchanged: a singular witness $Q : TR(n)(A)$ and an instantiated witness $s(Q) : S(Q)$ force the existential type to reduce to the identity type. As in the first-order case, the resulting inhabitant of $R$ is contextual and arises via instantiation of identity at a specific witnessing predicate $Q$.*

## 9. Discussion

The formal development above relies on a distinction between truth-value inhabitation and contextual witnessing. The ambient reasoning is carried out in a classical meta-logic, and all uses of “true”, “logical truth”, and “inhabited” are to be read in this meta-theoretic, Boolean sense. A type-level reduction to $\bot \to \bot$ shows that a target type is inhabited in the coarse Boolean sense relevant to classical logic. A witness term, however, is obtained only relative to a context supplying the inhabitant at which the generic identity witness is to be instantiated. Thus the method yields schematic or contextual witnesses rather than, in general, closed canonical terms.

This explains why the present account is best suited to logical truths. For many such principles, the essential point is that a type is inhabited uniformly in virtue of its logical form. The witness may then be represented schematically by identity or by a corresponding context-sensitive instantiation. The method is not intended to replace more informative constructive proofs in settings where the goal is to compute a specific mathematical object.

The relation with continuation-based interpretations is now also clearer. Those approaches enrich the term language to encode classical reasoning operationally [17], [18], [19], [20]. The present account instead keeps the term language ordinary and shifts the classical component into type-level reductions and classical meta-logic. Whether one regards this as a genuine semantic alternative or chiefly as an expository reorganisation, it isolates the main mechanism in a simple and uniform way.

## 10. Conclusion

We have presented a theorem–proof style account of a propositions-as-types interpretation of classical logic. The basic ingredients are a Boolean view of inhabitation, the truth values of the four basic function types, a small family of type-level reduction rules, and a polymorphic identity operator.

With these ingredients in place, many classical principles can be treated uniformly. Propositional principles such as excluded middle and double negation elimination, first-order existential reasoning, and their second- and higher-order analogues all reduce to the same core pattern: a type is shown to reduce to $\bot \to \bot$, and identity is then used as the generic inhabitant, instantiated at a suitable type when the context provides a witness. This “reduction” is always at the level of inhabitation truth values; the underlying term language remains an ordinary simply typed $\lambda$-calculus equipped with a polymorphic identity operator.

## Appendix A. Further examples of witness construction

This appendix records several additional examples illustrating the methods developed in the main text.

**Proposition 24** (Curry's rule)**.** *The type*

$$(A \to (B \to C)) \to ((A \to B) \to (A \to C))$$

*is inhabited.*

*Proof.* A witness is

$$(\lambda f : A \to (B \to C))(\lambda g : A \to B)(\lambda a : A)(f(a))(g(a)).$$

Indeed, if

$$f : A \to (B \to C), g : A \to B, a : A,$$

then

$$f(a) : B \to C \qquad \text{and} \qquad g(a) : B,$$

so

$$(f(a))(g(a)) : C.$$

□

**Proposition 25** (Peirce's law with contextual witness)**.** *The proposition*

$$((A \to B) \to A) \to A$$

*is logically true. Here "logically true" means that for every assignment of $\top/\bot$ to $A$ and $B$ consistent with the Boolean inhabitation semantics, the resulting type evaluates to $\top$. Moreover, if $\Gamma$ is a context containing an inhabitant*

$$b : B,$$

*then the type*

$$((A \to B) \to A) \to A$$

*is inhabited relative to* $\Gamma$.

*Proof.* First we verify logical truth. If $A = \top$, then the codomain is inhabited, so the whole implication is inhabited. If $A = \bot$, then

$$A \to B = \top,$$

and therefore

$$(A \to B) \to A = \top \to \bot = \bot.$$

Hence

$$((A \to B) \to A) \to A = \bot \to \bot = \top.$$

Thus the proposition is logically true in all cases.

Now assume

$$\Gamma \vdash b : B.$$

Define

$$g := (\lambda x : A)b.$$

Then

$$g : A \to B.$$

If

$$h : (A \to B) \to A,$$

then

$$h(g) : A.$$

Therefore

$$(\lambda h : (A \to B) \to A)h(g)$$

is a witness of

$$((A \to B) \to A) \to A$$

relative to the context $\Gamma$. $\square$

**Proposition 26** (A contextual existential–universal example). *Let* $\Gamma$ *be a context containing*

$$a : A.$$

*Suppose further that there is a function*

$$p : (\forall y : A)(P(a) \to P(y)).$$

*Then, relative to* $\Gamma$*, the type*

$$P(a) \to (\exists x : A)(\forall y : A)(P(x) \to P(y))$$

*is inhabited.*

*Proof.* Assume

$$\Gamma \vdash a : A.$$

The assumed function

$$p : (\forall y : A)(P(a) \to P(y))$$

already has the required universal form. For any fixed $w : A$, Theorem 13 yields

$$p(w) : P(a) \to P(w).$$

Now define

$$u := p : (\forall y : A)(P(a) \to P(y)).$$

Then $u$ is a witness of the universal statement

$$(\forall y : A)(P(a) \to P(y)).$$

By Theorem 14, using the singular witness $a : A$ together with the universal witness

$$u : (\forall y : A)(P(a) \to P(y)),$$

we obtain

$$e : (\exists x : A)(\forall y : A)(P(x) \to P(y)).$$

Therefore

$$(\lambda r : P(a))e$$


doi.org/10.32388/IY3BKH.3

witnesses

$$P(a) \to (\exists x : A)(\forall y : A)(P(x) \to P(y)).$$

□

**Remark 17.** *In the preceding proposition, the quantified steps are not primitive. The universal statement is of the form treated by Theorem 12, the instance $p(w)$ is obtained by Theorem 13, and the passage from the singular witness $a : A$ together with the universal witness $u$ to the existential statement uses Theorem 14. This keeps the appendix proof internal to the derived rules of the typed $\lambda$-calculus developed in the main text.*

## References


1. [a], [b] *Church A (1940). "A Formulation of the Simple Theory of Types." J Symbol Log. **5**(2):56–68.*
2. [^] *Girard J-Y (1972). "Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supéri eur" [Functional Interpretation and Cut Elimination of Higher-Order Arithmetic].*
3. [^] *Girard J-Y (1973). Quelques Résultats Sur Les Interpretations Fonctionnelles [Some Results on Functional In terpretations]. Springer.*
4. [^] *Girard J-Y (1989). Proofs and Types. Cambridge University Press.*
5. [^] *Girard J-Y (1986). "The System F of Variable Types, Fifteen Years Later." Theor Comput Sci. **45**:159–192.*
6. [^] *Coquand T, Huet G (1988). "The Calculus of Constructions." Inf Comput. **76**:95–120.*
7. [a], [b] *Barendregt H (1991). "Introduction to Generalized Type Systems." J Funct Program. **1**(2):125–154.*
8. [^] *Curry HB (1934). "Functionality in Combinatory Logic." Proc Natl Acad Sci U S A. **20**(11):584–90.*
9. [^] *Curry HB, Feys R (1958). Combinatory Logic. North-Holland.*
10. [^] *Howard W (1980). The Formulae-As-Types Notion of Construction. Academic Press.*
11. [^] *Wadler P (2015). "Propositions As Types." Commun ACM. **58**(12):75–84.*
12. [a], [b], [c] *Geuvers H, Nederpelt R (2014). Type Theory and Formal Proof. Cambridge University Press.*
13. [^] *Martin-Löf P (1982). Constructive Mathematics and Computer Programming. North-Holland.*
14. [^] *Martin-Löf P (1984). Intuitionistic Type Theory. Bibliopolis.*
15. [^] *Dybjer P, Palmgren E (2024). Intuitionistic Type Theory. Metaphysics Research Lab, Stanford University.*
16. [^] *Troelstra AS (2011). History of Constructivism in the 20th Century. Cambridge University Press.*
17. [a], [b] *Griffin TG (1989). "A Formulae-As-Type Notion of Control."*
18. [a], [b] *Murthy C (1991). "An Evaluation Semantics for Classical Proofs." IEEE Press.*

19. a, b *Parigot M (1992). "$\&lambda\mu$-Calculus: An Algorithmic Interpretation of Classical Natural Deduct ion."*

20. a, b *van Bakel S (2023). "Adding Negation to Lambda Mu." Log Methods Comput Sci. **19**(2):12:1–12:41.*

21. ^ *Hilbert D, Ackermann W (1950). Principles of Mathematical Logic. AMS Chelsea.*

22. ^ *Gentzen G (1969). New Version of the Consistency Proof for Elementary Number Theory. North-Holland.*

23. ^ *Prawitz D (2006). Natural Deduction: A Proof-Theoretical Study. Dover.*

## Declarations

**Funding:** No specific funding was received for this work.

**Potential competing interests:** No potential competing interests to declare.