HOL Constant Definition Done Right

Rob Arthan

Lemma 1 Ltd./ School of Electronic Engineering and Computer Science, Queen Mary, University of London, UK

Abstract. This note gives a proposal for a simpler and more powerful replacement for the mechanisms currently provided in the various HOL implementations for defining new constants.

1 Introduction

The design of the HOL logic and of its definitional principles [7] evolved in the late 80s and early 90s. Some form of this design has been implemented in HOL4 [8], HOL Light [3], HOL Zero [1], Isabelle/HOL [6] and ProofPower [2]. While the definitional principles have stood the test of time in many practical applications, we believe there is still some room for improvement. This note discusses issues with the mechanisms for introducing new constants and proposes a new and more general mechanism that addresses these issues.

2 The Problem

The original Classic HOL provided a mechanism for defining new constants known as `new_definition`. This worked as follows: given a possibly empty list of variables $x_1, \ldots, x_n$ and a term t whose free variables are contained in the x_i, it introduced a new constant[1] c of the appropriate type and the axiom:

$$\vdash \forall x_1 \ \ldots \ x_n .\ c\ x_1 \ \ldots \ x_n = t.$$

This simple mechanism is remarkably powerful but suffered from two significant shortcomings, both pointed out by Roger Jones[2]:

RJ1 The mechanism does not support implicit definitions. As one example, it is pleasant to define the destructors of a data type as the left inverses of the constructors. Thus one wants to define Pre in terms of Suc by:

$$\mathsf{Pre}(\mathsf{Suc}(n)) = n.$$

[1] The details of the mechanism for specifying the names of new constants are not important for present purposes.

[2] At various places in this note, I sketch observations made by other people. The wording used is mine and not theirs and any misrepresentation is my responsibility.

As another example, the exponential function is naturally defined by a differential equation:

$$\mathsf{exp}(0) = 1$$
$$(\mathsf{D}\,\mathsf{exp})(x) = \mathsf{exp}(x).$$

In such cases, the mechanism can be used to define constants having the desired properties, but one has to use the Hilbert choice operator to give witnesses and then derive the implicit definitions as theorems. This results in a loss of abstraction and unintended identities, e.g., the naive way of defining two constants c_1 and c_2 both with the loose defining property $c_i \leq 10$ will result in an extension in which $c_1 = c_2$ is provable.

RJ2 The mechanism is unsound. The condition on the free variables of t is certainly necessary. Without it, we could take t to be a variable, $y : \mathbb{N}$, and define a new constant c satisfying $\vdash \forall y : \mathbb{N} \cdot c = y$. Specialising this in two different ways, we could prove both $c = 1$ and $c = 2$. However, the condition is not sufficient. If $\#$ is a polymorphic function such that $\#X$ is the size of X when X is a finite set, then we can use the mechanism to define a constant $c : \mathbb{N}$ satisfying the axiom $c = \#\{x : \alpha \mid x = x\}$, where α is a type variable. But then if **1** and **2** denote types with 1 and 2 members respectively, we can instantiate α to prove both $c = \#\{x : \mathbf{1} \mid x = x\} = 1$ and $c = \#\{x : \mathbf{2} \mid x = x\} = 2$.

The fix for **RJ2** was to change `new_definition` so as to check that all type variables appearing anywhere in the term t also appear in the type of the constant c that is being defined. HOL Light, HOL Zero, Isabelle/HOL and ProofPower were all implemented after the problem was known, so they incorporated this solution from scratch. The fix in Classic HOL was carried forward into HOL4.

A new mechanism, `new_specification`, was introduced to address **RJ1**. `new_specification` takes as input a theorem of the form $\vdash \exists v_1 \ldots v_n \cdot p$ and introduces a list of new constants $c_1, \ldots, c_n$ and the axiom

$$\vdash p[c_1/v_1, \ldots, c_n/v_n].$$

`new_specification` requires that the free variables of p be contained in the v_i and that every type variable appearing anywhere in p also appear in the type of each new constant c_i, thus avoiding reintroducing the problem of **RJ2** under a different guise. The result is conservative and hence sound. It also supports a very useful range of implicit definitions. However, there are two issues that I noted during the ProofPower implementation:

RA1 Given `new_specification`, `new_definition` is redundant: what it does can easily be realised by a derived mechanism that given the list of variables $x_1, \ldots, x_n$ and the term t, automatically proves:

$$\vdash \exists y \cdot \forall x_1 \ \ldots \ x_n \cdot y \ x_1 \ \ldots \ x_n = t$$

and then applies `new_specification`. Unfortunately, in order to prove existentially quantified statements, one needs a definition of the existential quantifier, and so `new_definition` seems necessary to avoid a bootstrapping problem. (Since it is only required for bootstrapping, the ProofPower implementation of `new_definition` only covers the simple case where the axiom has the form $\vdash c = t$.)

RA2 The condition on type variables imposed by `new_specification` is stronger than one would like. It is natural for certain "concrete" structures to be characterized by more "abstract" properties such as universal mapping properties. For example, data types can be characterized as initial algebras:

$$\forall(z : \alpha)(s : \alpha \to \alpha)\cdot\ \exists! f : \mathbb{N} \to \alpha\cdot\ f(0) = z \wedge \forall n\cdot\ f(\mathsf{Suc}(n)) = s(f(n)).$$

However, the above characterization cannot be used as a defining property for the successor function with `new_specification`. Characterizing objects by universal properties is endemic in modern mathematics and computer science, so it is irritating to be compelled to resort to circumlocutions.

In HOL4, ProofPower and HOL Zero, `new_specification` is implemented as a primitive operation. However, in HOL Light, it is derived. I believe this was primarily a consequence of the following design goal for HOL Light:

JH1 The primitive inference system for HOL Light should be defined in terms of language primitives and equality alone and should not depend on the axiomatization of the logical connectives.

A form of `new_specification` that does not involve existential quantification was implemented in early versions of HOL Light. This took as input a theorem of the form $\vdash p\ t$. Later, to simplify the correctness argument for the system, `new_specification` was re-implemented as a derived operation that uses the Hilbert choice operator to translate its inputs into a form suitable for `new_definition`, applies `new_definition`, then derives the desired axiom to be passed back to the user from the stronger axiom returned by `new_definition`. Thus HOL Light bypasses **RA1**, but at the price of a certain inelegance, since we have to trust the derived rule to discard the axiom returned by `new_definition`. This became worse when HOL Light was enhanced to address the following observation of Mark Adams:

MA1 If an LCF style system does not record all the axioms and definitions that have been introduced, the correctness claim for the system has to be defined in terms of a state **and** the sequence of operations which produced that state. This makes it impossible to implement a proof auditing procedure that works by analysing the current state of the system.

As a result of **MA1** axioms and definitions in HOL Light are now recorded. The current HOL Light implementation uses a trick to prevent two constants with the same loose defining property being provably equal. The trick is based on the following idea: to define c_1 and c_2 such that $c_1, c_2 \leq 10$, say, define

$c_1 = (\varepsilon f\cdot \forall n\cdot f(n) \leq 10)\ 1$ and $c_2 = (\varepsilon f\cdot \forall n\cdot f(n) \leq 10)\ 2$; then c_1 and c_2 have the desired property, but $c_1 = c_2$ is not provable. Nonetheless some unintended identities are still provable that would not be provable if `new_specification` were implemented as a primitive as in HOL4 or ProofPower.

The equivalent of `new_specification` in Isabelle/HOL is its **specification** command. This is implemented using an equational definition and the choice function, but that definition only exists in a private namespace. Some aspects of the abstraction offered by `new_specification` are provided by the very popular locale mechanism in Isabelle.

Quantification over type variables as implemented in HOL-Omega [4] obviates many of the problems discussed here. However, our present concern is with improvements that preserve the delightful simplicity of the Classic HOL logic.

3 Proposed Alternative

The proposed alternative is to discard `new_definition` and to adapt and generalise `new_specification` so that it does not depend on the meaning of the existential quantifier. The generalised `new_specification`, which we will call `gen_new_specification`, takes as input a theorem of the following form

$$v_1 = t_1, \ldots, v_n = t_n \vdash p$$

where the v_i are variables. If all is well, `gen_new_specification` will introduce new constants $c_1, \ldots, c_n$ and the following axiom:

$$\vdash p[c_1/v_1, \ldots, c_n/v_n].$$

`gen_new_specification` imposes the following restrictions:

- the v_i must be pairwise distinct;
- the terms t_i must have no free variables;
- the free variables of p must be contained in the v_i;
- any type variable occurring in the type of any subterm of a t_i must occur in the type of the corresponding v_i.

There is no restriction on the type variables appearing in p.

Claim 1 `gen_new_specification` *is conservative and hence sound.*

Proof: Assume that a sequent $\Gamma \vdash q$ containing no instances of the c_i is provable using the axiom $\vdash p[c_1/v_1, \ldots, c_n/v_n]$ introduced using `gen_new_specification`. We will show how to transform a proof tree with conclusion $\Gamma \vdash q$ into a proof tree with the same conclusion that does not use the new axiom. First, by simple equality reasoning, derive from the theorem $v_1 = t_1, \ldots, v_n = t_n \vdash p$ that was passed to `new_specification`, the theorem $\vdash p[t_1/v_1, \ldots, t_n/v_n]$. Now replace each type instance of a c_i in the proof tree with the corresponding type instance of t_i and wherever a type instance of the axiom $\vdash p[c_1/v_1, \ldots, c_n/v_n]$ is used in

the proof tree, replace it with the corresponding type instance of a proof tree for $\vdash p[t_1/v_1, \ldots, t_n/v_n]$. By inspection of the primitive inference rules in [3], if one replaces instances of constants in a correct inference by closed terms of the same type in such a way that assumptions or conclusions of the sequents involved that were syntactically identical before the replacement remain syntactically identical, then the result is also a correct inference. As the condition on type variables imposed by `gen_new_specification` guarantees that two instances of a c_i are syntactically identical iff the corresponding instances of t_i are syntactically identical, we have constructed a correct proof tree whose conclusion is $\Gamma \vdash q$. ■

Claim 2 `gen_new_specification` *subsumes* `new_definition`.

Proof: In the simplest case, to define c with axiom $\vdash c = t$, where t has no free variables and contains no type variables that do not appear in its type, apply `gen_new_specification` to the axiom $v = t \vdash v = t$. This is all we need to define the logical connectives [3].

For the general case, to define c with axiom $\vdash \forall x_1 \ldots x_n \cdot c\ x_1 \ldots\ x_n = t$, take the axiom $v = (\lambda x_1 \ldots x_n \cdot t) \vdash v = (\lambda x_1 \ldots x_n \cdot t)$, derive $v = (\lambda x_1 \ldots x_n \cdot t) \vdash \forall x_1 \ldots x_n \cdot v\ x_1 \ldots\ x_n = t$ from it and then apply `gen_new_specification`. ■

Claim 3 `gen_new_specification` *subsumes* `new_specification`.

Proof: Given the theorem $\vdash \exists v_1 \ldots v_n \cdot p$, we can derive from it the theorem $v_1 = \varepsilon v_1 \cdot \exists v_2 \ldots v_n \cdot p \vdash \exists v_2 \ldots v_n \cdot p$ and apply `gen_new_specification` to define a constant c_1 with defining axiom $\vdash \exists v_2 \ldots v_n \cdot p[c_1/v_1]$. Iterating this process we can define $c_2, \ldots, c_n$ such that the defining axiom of c_n is $\vdash p[c_1/v_1, \ldots, c_n/v_n]$. Thus we can achieve the same effect as `new_specification` at the expense of additional intermediate definitions. This is sufficient to define the constructor and destructors for binary products.

Once we have binary products, we can simulate n-tuples by iterated pairing. This means that given the theorem $\vdash \exists v_1 \ldots v_n \cdot p$, we can derive the theorem $\vdash \exists z \cdot p[\pi_1(z)/v_1, \ldots, \pi_n(z)/v_n]$ in which the n bound variables $v_1, \ldots, v_n$ have been collected into a single n-tuple denoted by the fresh variable z (here π_i denotes the projection onto the i-th factor). Now we can derive from that the theorem $v_1 = t_1, \ldots, v_n = t_n \vdash p$ where t_i is $\pi_i(\varepsilon z \cdot p[\pi_1(z)/v_1, \ldots, \pi_n(z)/v_n])$. Given this theorem as input, `gen_new_specification` has exactly the same effect as `new_specification` given the input theorem $\vdash \exists v_1 \ldots, v_n \cdot p$. ■

4 Conclusion

Let me assess the proposed new definitional mechanism `gen_new_specification` against the observations that led to it:

RJ1 By claim 3, the support for implicit definitions is at least as good with `gen_new_specification` as with `new_specification`. In fact it is better: `new_specification` cannot define new constants $f : \alpha \to \mathbb{N}$ and $n : \mathbb{N}$ with defining property $\forall x \cdot \neg f\ x = n$, but `gen_new_specification` can.

RJ2 By claim 1, the proposed alternative is sound. What is more, this proof has been formalised in HOL4: Ramana Kumar, Scott Owens and Magnus Myreen have recently completed a formal proof of soundness for the HOL logic and its definitional principles including `gen_new_specification` [5].

RA1 By claim 2, `new_definition` is no longer required. (As seen in the proof of this claim, the special case needed to define the logical connectives does not involve any reasoning about them, so there is no bootstrapping issue.)

RA2 The restriction on type variables now applies only to the equations that give the witnesses to the consistency of the definition. Defining properties such as initial algebra conditions are supported.

JH1 `gen_new_specification` is defined solely in terms of equality and primitive language constructs.

MA1 The unintended identities arising as a result of recording definitions in HOL Light will not occur if `gen_new_specification` is adopted as the primitive mechanism for defining constants.

My conclusion when I wrote the first draft of this note was that the proposal was well worth adopting. It has recently been implemented in HOL4 and ProofPower. In both cases it is a replacement for `new_definition` and the existing `new_specification` has been retained for pragmatic reasons. The ProofPower implementation includes an implementation of the proof of claim 3 above and this completely replaces `new_specification` in the development of many of the theories supplied with the system, including all the "pervasive" theories such as the theories of pairs and natural numbers that form part of the logical kernel.

Acknowledgments: I would like to thank the ITP 2014 Programme Chairs, the referees, Mark Adams, John Harrison, Roger Jones, Ramana Kumar, Magnus Myreen, Scott Owens, Konrad Slind and Makarius Wenzel for their kind assistance in divers ways in the preparation and publication of this note.

References

1. Adams, M.: HOL Zero, `http://www.proof-technologies.com/holzero/`
2. Arthan, R., Jones, R.B.: Z in HOL in ProofPower. BCS FACS FACTS (2005-1), `http://www.lemma-one.com/ProofPower/index/`
3. Harrison, J.: HOL Light: an overview. In: TPHOLs. LNCS, vol. 5674. Springer (2009), `http://www.cl.cam.ac.uk/~jrh13/hol-light/`
4. Homeier, P.V.: The HOL-Omega logic. In: Berghofer, S., Nipkow, T., Urban, C., Wenzel, M. (eds.) TPHOLs. LNCS, vol. 5674, pp. 244–259. Springer (2009)
5. Kumar, R., Arthan, R., Myreen, M.O., Owens, S.: HOL with definitions: semantics, soundness, and a verified implementation. In: Interactive Theorem Proving. These proceedings, Springer (2014)
6. Makarius Wenzel and many others: The Isabelle/Isar Reference Manual, `http://isabelle.in.tum.de/dist/Isabelle2013-2/doc/isar-ref.pdf`
7. Michael Norrish and many others: The HOL System: Logic, 3rd edn., `http://hol.sourceforge.net/documentation.html`
8. Slind, K., Norrish, M.: A brief overview of HOL4. In: Theorem Proving in Higher Order Logics (TPHOLs). LNCS, vol. 5170. Springer (2008)