Automating Free Logic in Isabelle/HOL

Christoph Benzmüller[1] and Dana Scott[2]

[1] Freie Universität Berlin, Germany & Visiting Scholar at Stanford University, USA
c.benzmueller@fu-berlin.de,
http://www.christoph-benzmueller.de
[2] Visiting Scholar at University of Califormia, Berkeley, USA
dana.scott@cs.cmu.edu ,
http://www.cs.cmu.edu/ scott/

Abstract. We present an interactive and automated theorem prover for free higher-order logic. Our implementation on top of the Isabelle/HOL framework utilizes a semantic embedding of free logic in classical higher-order logic. The capabilities of our tool are demonstrated with first experiments in category theory.

Keywords: Free logic, interactive and automated theorem proving, model finding, application to category theory

1 Introduction

Partiality and undefinedness are core concepts in various areas of mathematics. Modern mathematical proof assistants and theorem proving systems are often based on traditional classical or intuitionistic logics and provide rather inadequate support for these challenge concepts. Free logic [5,6], in contrast, offers a theoretically and practically appealing solution. Unfortunately, however, we are not aware of any implemented and available theorem proving system for free logic.

In this extended abstract we show how free logic can be "implemented" in any theorem proving system for classical higher-order logic (HOL) [1]. The proposed solution employs a semantic embedding of free (or inclusive logic) in HOL. We present an exemplary implementation of this idea in the mathematical proof assistant Isabelle/HOL [4]. Various state-of-the-art first-order and higher-order automated theorem provers and model finders are integrated (modulo suitable logic translations) with Isabelle via the Sledgehammer tool [2], so that our solution can be utilized, via Isabelle as foreground system, with a whole range of other background reasoners. As a result we obtain an elegant and powerful implementation of an interactive and automated theorem proving (and model finding) system for free logic.

To demonstrate the practical relevance of our new system, we report on first experiments in category theory. In these experiments, theorem provers were able to detect a (presumably unknown) redundancy in the foundational axiom system of the category theory textbook by Freyd and Scedrov [3].

2 Free Logic

Terms in classical logic denote, without exceptions, entities in a non-empty domain of (existing) objects **D**, and it are these objects of **D** the universal and existential quantifiers do range over. Unfortunately, however, these conditions may render classical logic unsuited for handling mathematically relevant issues such as undefinedness and partiality. For example in category theory composition of maps is not always defined.

Free logic (and inclusive logic) has been proposed as an alternative to remedy these shortcomings. It distinguishes between a raw domain of possibly non-existing objects **D** and a particular subdomain **E** of **D**, containing only the "existing" entities. Free variables range over **D** and quantified variables only over **E**. Each term denotes in **D** but not necessarily in **E**. The particular notion of free logic as exploited below has been introduced by Scott [6]. A graphical illustration of this notion of free logic is presented in Fig. 1.

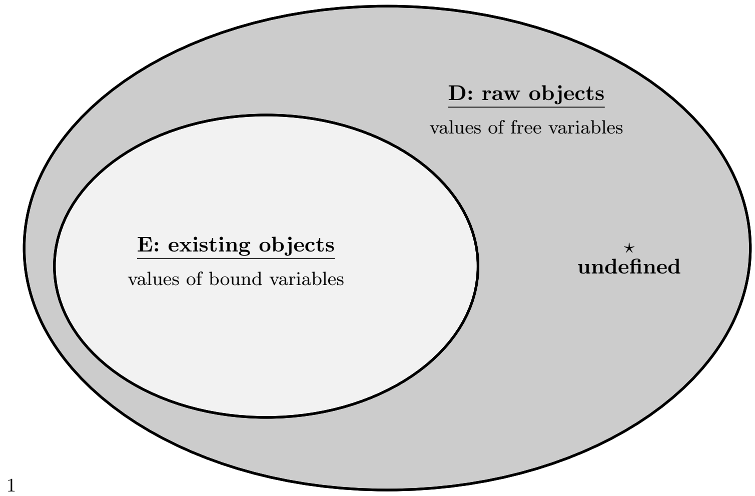

Fig. 1. Illustration of the Semantical Domains of Free Logic

3 Implementing Free Logic in Isabell/HOL

We start out with introducing a type i of individuals. The domain of objects associated with this this type will serve as the domain of raw objects **D**, cf. Fig 1. Moreover, we introduce an existence predicate **E** on type i. As mentioned, **E** characterises the subset of existing objects in **D**. Next, we declare a special constant symbol star $\star$, which is intended to denote a distinguished "non-existing" element of **D**.

typedecl *i* — the type for indiviuals
consts *fExistence*:: $i{\Rightarrow}bool$ ($\mathbf{E}$) — Existence predicate
consts *fStar*:: i ($\star$) — Distinguished symbol for undefinedness

We postulate that $\star$ is a "non-existing" object in D.

axiomatization where *fStarAxiom*: $\boldsymbol{\neg}\mathbf{E}(\star)$

The two primitive logical connective we introduce for free logic are negation ($\boldsymbol{\neg}$) and implication ($\boldsymbol{\rightarrow}$). They are identified with negation ($\neg$) and implication ($\longrightarrow$) in the underlying Isabelle/HOL logic. The internal names in Isabelle/HOL of the new logical connectives are *fNot* and *fImplies* (the prefix *f* stands for "free"); $\boldsymbol{\neg}$ and the infix operator $\boldsymbol{\rightarrow}$ are introduced as syntactical sugar.[3]

abbreviation *fNot*:: $bool{\Rightarrow}bool$ ($\boldsymbol{\neg}$)
 where $\boldsymbol{\neg}\varphi \equiv \neg\varphi$
abbreviation *fImplies*:: $bool{\Rightarrow}bool{\Rightarrow}bool$ (**infixr** $\boldsymbol{\rightarrow}$ *49*)
 where $\varphi\boldsymbol{\rightarrow}\psi \equiv \varphi{\longrightarrow}\psi$

The main challenge is to appropriately define free logic universal quantification ($\boldsymbol{\forall}$) and free logic definite description ($\mathbf{I}$). Again, we are interested to relate these logical operators to the respective operators $\forall$ and *THE* in the Isabelle/HOL logic. Different to the trivial maps for $\boldsymbol{\neg}$ and $\boldsymbol{\rightarrow}$ from above, their mappings are relativized in the sense that the existence predicate $\mathbf{E}$ is utilized as guard in their definitions.
The definition of the free logic universal quantifier $\boldsymbol{\forall}$ thus becomes:

abbreviation *fForall*:: $(i{\Rightarrow}bool){\Rightarrow}bool$ ($\boldsymbol{\forall}$)
 where $\boldsymbol{\forall}\Phi \equiv \forall x.\ \mathbf{E}(x){\longrightarrow}\Phi(x)$

Apparently, this definitions restricts the set of objects the $\boldsymbol{\forall}$-operator is ranging over to the set of existing objects $\mathbf{E}$. Note that this set can be empty (if desired, we may of course simply postulate that the domain $\mathbf{E}$ is non-empty: $\exists x.\ \mathbf{E}(x)$).
The Isabelle framework supports the introduction of syntactic sugar for binding notations. Here we make use of this option to introduce binding notation for $\boldsymbol{\forall}$. With the definition below we can now use the more familiar notation $\boldsymbol{\forall} x.\ \varphi(x)$ instead of writing $\boldsymbol{\forall}(\lambda x.\ \varphi(x))$ or $\boldsymbol{\forall}\varphi$.

abbreviation *fForallBinder*:: $(i{\Rightarrow}bool){\Rightarrow}bool$ (**binder** $\boldsymbol{\forall}$ [*8*] *9*)
 where $\boldsymbol{\forall} x.\ \varphi(x) \equiv \boldsymbol{\forall}\varphi$

Definite description $\mathbf{I}$ in free logic works as follows: Given an unary set $\Phi = \{a\}$, with a being an "existing" element in $\mathbf{E}$, $\mathbf{I}$ returns the single element a of Φ. In all other cases, that is, if Φ is not unary or a is not an element of $\mathbf{E}$, $\mathbf{I}\Phi$ returns the distinguished "non-existing" object denoted by $\star$. With the help of Isabelle/HOL's definite description operator *THE*, $\mathbf{I}$ can thus be defined as follows:

[3] The numbers in (*infixr* $\boldsymbol{\rightarrow}$ *49*) and (*binder* $\boldsymbol{\forall}$ [*8*] *9*) (see below) specify structural priorities and thus help to avoid brackets in formula representations.

abbreviation *fThat*:: $(i{\Rightarrow}bool){\Rightarrow}i$ (**I**)
 where $\mathbf{I}\Phi \equiv$ *if* $\exists\, x.\ \mathbf{E}(x) \wedge \Phi(x) \wedge (\forall\, y.\ (\mathbf{E}(y) \wedge \Phi(y)) \longrightarrow (y = x))$
 then $THE\ x.\ \mathbf{E}(x) \wedge \Phi(x)$
 else $\star$

Analogous to above we introduce binder notation for **I**, so that we can write $\mathbf{I}x.\ \varphi(x)$ instead of $\mathbf{I}(\lambda x.\ \varphi(x))$ or $\mathbf{I}\varphi$.

abbreviation *fThatBinder*:: $(i{\Rightarrow}bool){\Rightarrow}i$ (**binder I** [*8*] *9*)
 where $\mathbf{I}x.\ \varphi(x) \equiv \mathbf{I}(\varphi)$

Further logical connectives of free can now be defined in the usual way (and for $\exists$ we again introduce binder notation).

abbreviation *fOr* (**infixr** $\vee$ *51*) **where** $\varphi{\vee}\psi \equiv (\neg\varphi){\rightarrow}\psi$
abbreviation *fAnd* (**infixr** $\wedge$ *52*) **where** $\varphi{\wedge}\psi \equiv \neg(\neg\varphi{\vee}\neg\psi)$
abbreviation *fEquiv* (**infixr** $\leftrightarrow$ *50*) **where** $\varphi{\leftrightarrow}\psi \equiv (\varphi{\rightarrow}\psi){\wedge}(\psi{\rightarrow}\varphi)$
abbreviation *fEquals* (**infixr** $=$ *56*) **where** $x{=}y \equiv x{=}y$
abbreviation *fExists* ($\exists$) **where** $\exists\,\Phi \equiv \neg(\forall\,(\lambda y.\neg(\Phi\ y)))$
abbreviation *fExistsBinder* (**binder** $\exists$ [*8*]*9*) **where** $\exists\,x.\ \varphi(x) \equiv \exists\,\varphi$

4 Functionality Tests

We exemplarily investigate some example proof problems from Scott's paper [6], pp. 183-184, where a free logic with a single relation symbol **r** is discussed.

consts *r*:: $i{\Rightarrow}i{\Rightarrow}bool$ (**infixr r** *70*)

The implication $x\ \mathbf{r}\ x \rightarrow x\ \mathbf{r}\ x$, where x is a free variable, is valid independently whether x is defined (i.e. "exists") or not. In Isabelle/HOL this quickly confirmed by the simplification procedure simp.

lemma $x\ \mathbf{r}\ x \rightarrow x\ \mathbf{r}\ x$ **by** *simp*

However, as intended, the formula $\exists\, y.\ y\ \mathbf{r}\ y \rightarrow y\ \mathbf{r}\ y$ is not valid, since set of existing objects **E** could be empty. Nitpick quickly presents a respective countermodel.

lemma $\exists\, y.\ y\ \mathbf{r}\ y \rightarrow y\ \mathbf{r}\ y$ **nitpick** [*user-axioms*] **oops**

Consequently, also the implication $(x\ \mathbf{r}\ x \rightarrow x\ \mathbf{r}\ x) \rightarrow (\exists\, y.\ y\ \mathbf{r}\ y \rightarrow y\ \mathbf{r}\ y)$ has a countermodel, where **E** is empty.

lemma $(x\ \mathbf{r}\ x \rightarrow x\ \mathbf{r}\ x) \rightarrow (\exists\, y.\ y\ \mathbf{r}\ y \rightarrow y\ \mathbf{r}\ y)$ **nitpick** [*user-axioms*] **oops**

If we rule out that **E** is empty, e.g. with additional condition $(\exists\, y.\ y = y)$ in the antecedent of the above formula, then we obtain a valid implication. Isabelle trivially proves this with procedure simp.

lemma $((x\ \mathbf{r}\ x \rightarrow x\ \mathbf{r}\ x) \wedge (\exists\, y.\ y = y)) \rightarrow (\exists\, y.\ y\ \mathbf{r}\ y \rightarrow y\ \mathbf{r}\ y)$ **by** *simp*

We analyse some further statements (respectively statement instances) from Scott's paper [6], p. 185. Because of space restrictions we do not further comment these statements here. Altogether they provide further evidence that our implementation of free logic in fact obeys the intended properties.

lemma *S1*: $(\forall x.\ \Phi(x) \rightarrow \Psi(x)) \longrightarrow ((\forall x.\ \Phi(x)) \rightarrow (\forall x.\ \Psi(x)))$ **by** *auto*

lemma *S2*: $\forall y.\ \exists x.\ x = y$ **by** *auto*

lemma *S3*: $\alpha = \alpha$ **by** *auto*

lemma *S4*: $(\Phi(\alpha) \wedge (\alpha = \beta)) \rightarrow \Phi(\beta)$ **by** *auto*

lemma *UI-1*: $((\forall x.\ \Phi(x)) \wedge (\exists x.\ x = \alpha)) \rightarrow \Phi(\alpha)$ **by** *auto*

lemma *UI-2*: $(\forall x.\ \Phi(x)) \rightarrow \Phi(\alpha)$ **nitpick** [*user-axioms*] **oops** — Countermodel by Nitpick

lemma *UI-cor1*: $\forall y.((\forall x.\ \Phi(x)) \rightarrow \Phi(y))$ **by** *auto*

lemma *UI-cor2*: $\forall y.((\forall x.\ \neg(x = y)) \rightarrow \neg(y = y))$ **by** *auto*

lemma *UI-cor3*: $\forall y.((y = y) \rightarrow (\exists x.\ x = y))$ **by** *auto*

lemma *UI-cor4*: $(\forall y.\ y = y) \rightarrow (\forall y. \exists x.\ x = y)$ **by** *simp*

lemma *Existence*: $(\exists x.\ x = \alpha) \longrightarrow \mathbf{E}(\alpha)$ **by** *simp*

lemma *I1*: $\forall y.\ ((y = (\mathbf{I}x.\ \Phi(x))) \leftrightarrow (\forall x.\ ((x = y) \leftrightarrow \Phi(x))))$ **by** (*smt fStarAxiom the-equality*)

abbreviation *Star* ($\bigotimes$) **where** $\bigotimes \equiv \mathbf{I}y.\ \neg\ (y = y)$

lemma *StarTest*: $\bigotimes = \star$ **by** *simp*

lemma *I2*: $\neg(\exists y.\ y = (\mathbf{I}x.\ \Phi(x))) \rightarrow (\bigotimes = (\mathbf{I}x.\ \Phi(x)))$ **by** (*metis* (*no-types*, *lifting*) *the-equality*)

lemma *ExtI*: $(\forall x.\ \Phi(x) \leftrightarrow \Psi(x)) \rightarrow ((\mathbf{I}x.\ \Phi(x)) = (\mathbf{I}x.\ \Psi(x)))$ **by** (*smt the1-equality*)

lemma *I3*: $(\bigotimes = \alpha \vee \bigotimes = \beta) \rightarrow \neg(\alpha\ \mathbf{r}\ \beta)$ **nitpick** [*user-axioms*] **oops** — Countermodel by Nitpick

References

1. C. Benzmüller and D. Miller. Automation of higher-order logic. In J. Siekmann, D. Gabbay, and J. Woods, editors, *Handbook of the History of Logic, Volume 9 — Logic and Computation*. Elsevier, 2014.
2. J. Blanchette, S. Böhme, and L. Paulson. Extending Sledgehammer with SMT solvers. *J. of Automated Reasoning*, 51(1):109–128, 2013.
3. P. J. Freyd and A. Scedrov. *Categories, Allegories*. North Holland, 1990.
4. T. Nipkow, L. Paulson, and M. Wenzel. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*. Number 2283 in LNCS. Springer, 2002.
5. J. Nolt. Free logic. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Winter 2014 edition, 2014.
6. D. Scott. Existence and description in formal logic. In R. Schoenman, editor, *Bertrand Russell: Philosopher of the Century*, pages 181–200. George Allen & Unwin, London, 1967. (Reprinted with additions in: Philosophical Application of Free Logic, edited by K. Lambert. Oxford Universitry Press, 1991, pp. 28 - 48).