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 California, 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.

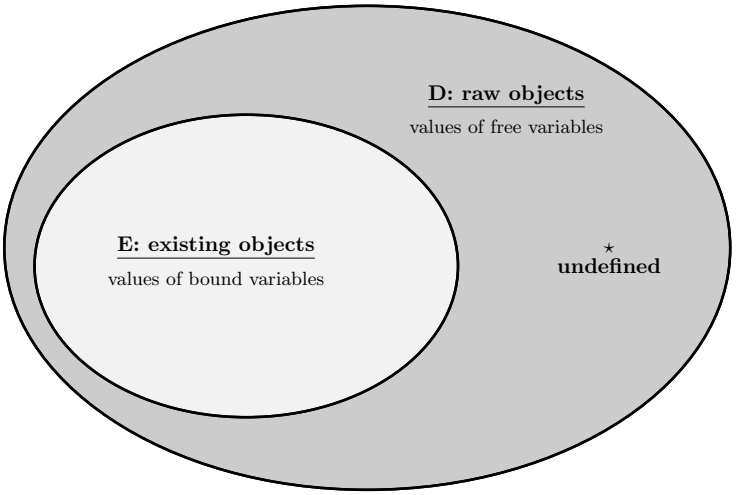
In this extended abstract we show how free logic can be "implemented" in any theorem proving system for classical higher-order logic (HOL) [1]. The proposed solution employs a semantic embedding of free (or inclusive logic) in HOL. We present an exemplary implementation of this idea in the mathematical 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.



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 ⋆, which is intended to denote a distinguished "non-existing" element of **D**.

typedecl i — the type for indiviuals
consts *fExistence*:: $i \Rightarrow bool$ (**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*: $\neg\mathbf{E}(\star)$

The two primitive logical connective we introduce for free logic are negation ($\neg$) and implication ($\rightarrow$). 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"); $\neg$ and the infix operator $\rightarrow$ are introduced as syntactical sugar.[3]

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

The main challenge is to appropriately define free logic universal quantification ($\forall$) and free logic definite description (**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 $\neg$ and $\rightarrow$ from above, their mappings are relativized in the sense that the existence predicate **E** is utilized as guard in their definitions.

The definition of the free logic universal quantifier $\forall$ thus becomes:

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

Apparently, this definitions restricts the set of objects the $\forall$-operator is ranging over to the set of existing objects **E**. Note that this set can be empty (if desired, we may of course simply postulate that the domain **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 $\forall$. With the definition below we can now use the more familiar notation $\forall x.\ \varphi(x)$ instead of writing $\forall(\lambda x.\ \varphi(x))$ or $\forall\varphi$.

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

Definite description **I** in free logic works as follows: Given an unary set $\Phi = \{a\}$, with a being an "existing" element in **E**, **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 **E**, $\mathbf{I}\Phi$ returns the distinguished "non-existing" object denoted by $\star$. With the help of Isabelle/HOL's definite description operator *THE*, **I** can thus be defined as follows:

[3] The numbers in (*infixr* $\rightarrow$ *49*) and (*binder* $\forall$ [*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*: $(\forall x.\ \Phi(x) \rightarrow \Psi(x)) \rightarrow ((\forall x.\ \Phi(x)) \rightarrow (\forall x.\ \Psi(x)))$ **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

5 Application in Category Theory

We exemplarily employ our free logic reasoning framework from above for an application in category theory. More precisely, we study some properties of the foundational axiom system of Freyd and Scedrov; see their textbook "Categories, Allegories" [3], p. 3. As expected, the composition $x{\cdot}y$, for morphisms x and y, is introduced by Freyd and Scedrov as a partial operation, cf. axiom *A1* below: the composition $x{\cdot}y$ exists if and only if the target of x coincides with the source of y. This is why free logic, as opposed to e.g. classical logic, is better suited as a starting point in this mathematical application area.[4]

In the remainder we identify the base type i of free logic with the raw type of morphisms. Moreover, we introduce constant symbols for the following operations: *source* of a morphism x, *target* of a morphism x and *composition* of morphisms x and y. These operations are denoted by Freyd and Scedrov as $\Box x$,

[4] The precise logic setting is unfortunately not discussed in the very beginning of Freyd's and Scedrov's textbook. Appendix B, however, contains a concise formal definition of the assumed logic. Note the special notion of equality used below (which is different from Kleene equality) and also remember that we postulated a 'non-existing' entity.

$x\Box$ and $x{\cdot}y$, respectively. We adopt their notation as syntactic sugar below, even though we are not particularly fond of the use of $\Box$ in this context.

consts *source*:: $i{\Rightarrow}i$ ($\Box$- [*108*] *109*)
target:: $i{\Rightarrow}i$ (-$\Box$ [*110*] *111*)
composition:: $i{\Rightarrow}i{\Rightarrow}i$ (**infix** $\cdot$ *110*)

Ordinary equality on morphisms is defined as follows:

abbreviation *OrdinaryEquality*:: $i{\Rightarrow}i{\Rightarrow}bool$ (**infix** $\approx$ *60*)
where $x \approx y \equiv ((\mathbf{E}\ x) \leftrightarrow (\mathbf{E}\ y)) \wedge x = y$

We are now in the position to model the category theory axiom system of Freyd and Scedrov.

axiomatization *FreydsAxiomSystem* **where**
A1: $\mathbf{E}(x{\cdot}y) \leftrightarrow ((x\Box) \approx (\Box y))$ **and**
A2a: $((\Box x)\Box) \approx \Box x$ **and**
A2b: $\Box(x\Box) \approx \Box x$ **and**
A3a: $(\Box x){\cdot}x \approx x$ **and**
A3b: $x{\cdot}(x\Box) \approx x$ **and**
A4a: $\Box(x{\cdot}y) \approx \Box(x{\cdot}(\Box y))$ **and**
A4b: $(x{\cdot}y)\Box \approx ((x\Box){\cdot}y)\Box$ **and**
A5: $x{\cdot}(y{\cdot}z) \approx (x{\cdot}y){\cdot}z$

Experiments with our new reasoning framework for free logic quickly showed that axiom *A2a* is redundant. For example, as Isabelle's internal prover metis[5] confirms, *A2a* is implied by *A2b*, *A3a*, *A3b* and *A4a*.

lemma *A2aIsRedundant-1*: $(\Box x)\Box \approx \Box x$ **by** (*metis A2b A3a A3b A4a*)

A human readable and comprehensible reconstruction of this redundancy is presented below. Our handmade proof employs axioms *A2b*, *A3a*, *A3b*, *A4a* and *A5*, that is, this proof could be further optimized by eleminating the dependency on *A5*.

lemma *A2aIsRedundant-2*: $(\Box x)\Box \approx \Box x$
proof −
have *L1*: $\forall x.\ (\Box\Box x){\cdot}((\Box x){\cdot}x) \approx ((\Box\Box x){\cdot}(\Box x)){\cdot}x$ **using** *A5* **by** *metis*
hence *L2*: $\forall x.\ (\Box\Box x){\cdot}x \approx ((\Box\Box x){\cdot}(\Box x)){\cdot}x$ **using** *A3a* **by** *metis*
hence *L3*: $\forall x.\ (\Box\Box x){\cdot}x \approx (\Box x){\cdot}x$ **using** *A3a* **by** *metis*
hence *L4*: $\forall x.\ (\Box\Box x){\cdot}x \approx x$ **using** *A3a* **by** *metis*
have *L5*: $\forall x.\ \Box((\Box\Box x){\cdot}x) \approx \Box((\Box\Box x){\cdot}(\Box x))$ **using** *A4a* **by** *auto*
hence *L6*: $\forall x\ .\Box((\Box\Box x){\cdot}x) \approx \Box\Box x$ **using** *A3a* **by** *metis*
hence *L7*: $\forall x.\ \Box\Box(x\Box) \approx \Box(\Box\Box(x\Box)){\cdot}(x\Box)$ **by** *auto*
hence *L8*: $\forall x.\ \Box\Box(x\Box) \approx \Box(x\Box)$ **using** *L4* **by** *metis*
hence *L9*: $\forall x.\ \Box\Box(x\Box) \approx \Box x$ **using** *A2b* **by** *metis*

[5] Metis is a trusted prover of Isabelle, since it returns proofs in Isabelle's trusted proof kernel. Initially, however, we have worked with Isabelle's Sledgehammer tool in our experiments, which in turn performs calls to several integrated first-order theorem provers. These calls then return valuable information on the particular proof dependencies, which in turn suggest the successful calls with metis as presented here.

hence *L10*: $\forall x.\ \Box\Box x \approx \Box x$ **using** *A2b* **by** *metis*
hence *L11*: $\forall x.\ \Box\Box((\Box x)\Box) \approx \Box\Box(x\Box)$ **using** *A2b* **by** *metis*
hence *L12*: $\forall x.\ \Box\Box((\Box x)\Box) \approx \Box x$ **using** *L9* **by** *metis*
have *L13*: $\forall x.\ (\Box\Box((\Box x)\Box))\cdot((\Box x)\Box) \approx ((\Box x)\Box)$ **using** *L4* **by** *auto*
hence *L14*: $\forall x.\ (\Box x)\cdot((\Box x)\Box) \approx (\Box x)\Box$ **using** *L12* **by** *metis*
hence *L15*: $\forall x.\ (\Box x)\Box \approx (\Box x)\cdot((\Box x)\Box)$ **using** *L14* **by** *auto*
then show *?thesis* **using** *A3b* **by** *metis*
qed

Thus, axiom *A2a* can be removed from the theory. Alternatively, we could also eliminate *A2b* which is implied by *A1*, *A2a* and *A3a*:

lemma *A2bIsRedundant*: $\Box(x\Box) \approx \Box x$ **by** (*metis A1 A2a A3a*)

In fact, by a systematic experimentation within our free logic theorem proving framework, we can show that Freyd's and Scedroc's axiomatic theory can be reduced to just the following five axioms:

axiomatization *FreydsAxiomSystemReduced* **where**
B1: $\mathbf{E}(x\cdot y) \leftrightarrow ((x\Box) \approx (\Box y))$ **and**
B2a: $((\Box x)\Box) \approx \Box x$ **and**
B3a: $(\Box x)\cdot x \approx x$ **and**
B3b: $x\cdot(x\Box) \approx x$ **and**
B5: $x\cdot(y\cdot z) \approx (x\cdot y)\cdot z$

The dropped axioms can then be introduced as lemmas.

lemma *B2b*: $\Box(x\Box) \approx \Box x$ **by** (*metis B1 B2a B3a*)
lemma *B4a*: $\Box(x\cdot y) \approx \Box(x\cdot(\Box y))$ **by** (*metis B1 B2a B3a*)
lemma *B4b*: $(x\cdot y)\Box \approx ((x\Box)\cdot y)\Box$ **by** (*metis B1 B2a B3a*)

6 Summary of Technical Contribution and Further Work

We have presented a new reasoning framework for free logic, and we have exemplary applied it for some first experiments in category theory. We have shown that, in our free logic setting, the category theory axiom system of Freyd and Scedrov is redundant and that three axioms can be dropped.

Our free logic reasoning framework is publicly available for reuse: Simply download Isabelle from https://isabelle.in.tum.de and initialize it (respectively import) the file `FreeFOL.thy` from our sources available at www.christoph-benzmueller.de/papers/2016-ICMS.zip. Our category theory experiments are contained in the file `FreydScedrov.thy`.

Comparisons with other theorem provers for free logic are not possible at this stage, since we are not aware of any other existing systems.

We also want to emphasize that this paper has been written entirely within the Isabelle framework by utilizing the Isabelle "build" tool; cf. [8], section 2. It is thus an example of a formally verified mathematical document, where the pdf

document as presented here has been generated directly from the verified source files mentioned above.[6]

Further work includes the continuation of our formalization studies in category theory. It seems plausible that substantial parts of the textbook of Freyd and Scedrov can now be formalised in our framework. An interesting question clearly is how far automation scales and whether some further (previously unknown) insights can eventually be contributed by the theorem provers. Moreover, we have already started to compare the axiom system by Freyd and Scedrov with a more elegant set of self-dual axioms developed by Scott. Furthermore, we plan to extend our studies to projective geometry, which is another area where free logic may serve as a suitable starting point for formalisation.

In addition to our implementation of free logic as a theory in Isabelle/HOL, we plan to support an analogous logic embedding in the new Leo-III theorem prover [9]. The idea is that Leo-III can then be envoked with a specific flag telling it to automatically switch its underlying logic setting from higher-order classical logic to first-order and higher-order free logic, while retaining TPTP TH0 [7] as the common input syntax.

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).
7. G. Sutcliffe and C. Benzmüller. Automated reasoning in higher-order logic using the TPTP THF infrastructure. *J. of Formalized Reasoning*, 3(1):1–27, 2010.
8. M. Wenzel. The isabelle system manual. https://www.cl.cam.ac.uk/research/hvg/Isabelle/dist/Isabelle2016/doc/system.pdf, February 2016.
9. M. Wisniewski, A. Steen, and C. Benzmüller. Leopard - A generic platform for the implementation of higher-order reasoners. In M. Kerber, J. Carette, C. Kaliszyk, F. Rabe, and V. Sorge, editors, *Intelligent Computer Mathematics - International Conference, CICM 2015, Washington, DC, USA, July 13-17, 2015, Proceedings*, volume 9150 of *LNCS*, pages 325–330. Springer, 2015.

[6] By suitably adapting the Isabelle call as contained in file `runIsabelle.sh` in our zip-package, the verification and generation process can be easily reproduced by the reader.