A Formalization of the Zermelo Wellordering Theorem in the Naproche-SAD system

Peter Koepke

March 10, 2019

1 Introduction

The Naproche-SAD proof-checking system checks the logical correctness of texts written in an input language which is acceptable and readable as common mathematical language. Texts should resemble the style of undergraduate textbooks. To test the viability of this approach we are currently formalizing some basic results in Zermelo-Fraenkel set theory. Naproche-SAD accepts texts in a native ForTheL (Formula Theory Language) format. We are also testing a LaTeX add-on which accepts texts in LaTeX format and transforms them into ForTheL. In this report, we present a standard proof of the Wellordering Theorem, using ordinals, recursion and choice. This requires to formalize some initial theory. We follow parts of the Set Theory course 2018/19 at the University of Bonn.
Some mathematical principles like Choice, set formation and function formation are already implemented in Naproche-SAD. Other set theoretical axioms have to be required explicitly. There are some marked differences with ZF set theory: Naproche-SAD has an abstraction term mechanisms that allow abstractions with *arbitrary* formulas. This transcends ZF towards the stronger Kelley-Morse set theory (KM). KM provides good foundations for mathematics. ZF is however more popular, perhaps for historic reasons, or since it is more amenable to the metamathematical analyses of axiomatic set theory.
The abstraction term and function mechanism of the current version of Naproche-SAD are directed towards standard mathematical fields and so abstraction terms are naively identified with "sets" instead of "classes". To make Naproche-SAD work for unrestricted set theory, we use Naproche's sets as KM classes but make some changes to the vocabulary so that we can use a natural class / object / set ontology.
Numbers of Definitions and Theorems correspond to the Set Theory scriptum when possible. There has been a difficulty with satisfying the existence task connected with the Choice operator "choose" in the proof of the Wellordering Theorem. We have therefore deactivated that task in the Naproche source file ProofTask.hs:82
We first did the formalization in a ForTheL file `ordinals07.ftl`. After this file was accepted by Naproche-SAD it was rewritten to a LaTeX file `Zermelo01.tex`.

The LaTeX file contains further text that is typeset but not checked. After typesetting the ForTheL content is indicated by a vertical line. In this way this whole document is a single document, whose ForTheL content is proofchecked.

2 The Formalization

2.1 Preliminaries on Sets and Classes

We work with the following ontology: mathematical objects are *objects*; objects are elements of *classes*; classes which are themselves objects are *sets*. Sets are the objects studied in set theory.

Let $a \neq b$ stand for $a! = b$. Let A, B, C, D stand for *classes*. Let $x \in y$ stand for x *is an element* of y. Let $x \notin y$ stand for x *is not an element* of y.

Definition 1 (7a) *A* subclass *of B is a class A such that every element of A is an element of B. Let $x \subseteq y$ stand for x is a subclass of y.*

[synonym object/objects]

Signature 1 *An* object *is a notion. Let o stand for objects.*

Axiom 1 *Any element of any class is an object.*

[synonym set/sets]

Definition 2 *A* set *is a class that is an object. Let a, b, c, x, y, z stand for sets.*

The axiom schema of *foundation* is formalized by embedding the element relation into the universal induction relation -¡- for which Naproche provides an induction scheme.

Axiom 2 (Foundation) *If $o \in x$ then $o- < -x$.*

Theorem 1 (33) $x \notin x$.

Proof By induction on x. □

The following definition includes the *set existence* axiom.

Definition 3 (4a) *The* empty set *is the set that has no elements. Let $\emptyset$ stand for the empty set.*

We use a new unordered pair instead of the version available in Naproche-SAD. Perhaps one could as well use Naproche's pair.

Definition 4 (4c) $\{a, b\} = \{u \mid u = a or u = b\}$.

Axiom 3 (Pairing) $\{a, b\}$ *is a set.*

Definition 5 (7j) $\{a\} = \{u \mid u = a\}$.

Lemma 1 $\{a\}$ *is a set.*

Definition 6 (7e) $\bigcup(x) = \{u \mid$ *there is a such that* $u \in a \in x\}$.

Axiom 4 (Union) $\bigcup(x)$ *is a set.*

Definition 7 (7b) $A \cup B = \{u \mid u \in A$ *or* $u \in B\}$.

Lemma 2 $a \cup b$ *is a set.*

Proof $a \cup b = \bigcup(\{a, b\})$. □

Definition 8 (7c) $A \cap B = \{u \mid u \in A$ *and* $u \in B\}$.

Axiom 5 (Separation) $x \cap A$ *is a set.*

Definition 9 (7d) $A \setminus B = \{u \mid u \in A$ *and* $u \notin B\}$.

Lemma 3 $a \setminus B$ *is a set.*

Proof Define $C = \{u \mid u \in a$ and $u \notin B\}$. $a \setminus B = a \cap C$. □

Definition 10 (30) $\mathrm{Succ}(x) = x \cup \{x\}$.

Lemma 4 $\mathrm{Succ}(x)$ *is a set.*

2.2 Preliminaries on Functions

We use the function notion and corresponding definition mechanisms built into Naproche-SAD.

Let F, G, f, g stand for *functions*.

Definition 11 (15e) *Let* F *be a function.*

$$F\text{”}A = \{F(u) \mid u \in A \text{ and } u \in \mathrm{Dom}(F)\}.$$

Definition 12 (15b) $\mathrm{range}(F) = F\text{”}\mathrm{Dom}(F)$.

The replacement schema takes the following form:

Axiom 6 (Replacement) *F"x is a set.*

Definition 13 (19a) *A* function from A to B *is a function* F *such that* $\mathrm{Dom}(F) = A$ *and* $\mathrm{range}(F) \subseteq B$. *Let* $F : A \to B$ *stand for* F *is a function from* A *to* B.

Definition 14 (19c) *A* surjective function from A to B *is a function* F *such that* $\mathrm{Dom}(F) = A$ *and* $\mathrm{range}(F) = B$.

Definition 15 (19d) *An* injective function from A to B *is a function* F *such that* $F : A \to B$ *and* $\forall u, v \in A(u \neq v \Rightarrow F(u) \neq F(v))$.

Definition 16 (19e) *A* bijective function from A *to* B *is a function* F *such that* F *is a surjective function from* A *to* B *and* F *is an injective function from* A *to* B. *Let* $F : A \leftrightarrow B$ *stand for* F *is a bijective function from* A *to* B.

Definition 17 *Let* F *be a function. Let* v *be an object.*

$$F_*(v) = \{u \mid u \in \mathrm{Dom}(F) \text{ and } F(u) = v\}.$$

Theorem 2 *Let* F *be an injective function from* A *to* x. *Then* A *is a set.*

Proof $\mathrm{range}(F) \subseteq x$. $\mathrm{range}(F)$ is a set. Indeed $\mathrm{range}(F) = x \cap \mathrm{range}(F)$. Define

$$G(v) = \text{ choose } u \in F_*(v) \text{ in } u \text{ for } v \text{ in } \mathrm{range}(F).$$

Then $\mathrm{range}(G) = A$. □

2.3 Ordinal Numbers

Let 0 stand for $\emptyset$.

Definition 18 $1 = \{0\}$.

Definition 19 (37a) *A is* transitive *iff every element of* A *is a subclass of* A. *Let* $\mathrm{Trans}(x)$ *stand for* x *is transitive.*

[synonym ordinal/ordinals]

Definition 20 (37b) *An* ordinal *is a set* x *such that* x *is transitive and every element of* x *is transitive. Let* $\mathrm{Ord}(x)$ *stand for* x *is an ordinal. Let* $\alpha, \beta, \gamma, \lambda$ *denote ordinals.*

Theorem 3 (38a) 0 *is an ordinal.*

Theorem 4 (38b) $\mathrm{Succ}(\alpha)$ *is an ordinal.*

Theorem 5 (39) *Every element of* α *is an ordinal.*

Theorem 6 $(\alpha \in \beta$ *and* $\beta \in \gamma) => \alpha \in \gamma$.

Theorem 7 (40b) *For all* α $\alpha \notin \alpha$.

Definition 21 (41) $\alpha < \beta$ *iff* $\alpha \in \beta$.

The following linearity property can be proved by a double induction. For the proof, α and β have to be universally quantified.

Axiom 7 (40c) $\alpha < \beta$ *or* $\alpha = \beta$ *or* $\beta < \alpha$.

Theorem 8 (44) *If* $\alpha < \beta$ *then* $\alpha- < -\beta$.

Theorem 9 (42a) $\alpha < \mathrm{Succ}(\alpha)$.

Theorem 10 (42b) *If* $\beta < \mathrm{Succ}(\alpha)$ *then* $\beta = \alpha$ *or* $\beta < \alpha$.

Definition 22 (43a) *A* successor ordinal *is an ordinal* α *such that* $\exists\beta\alpha = \mathrm{Succ}(\beta)$.

Definition 23 (43b) *A* limit ordinal *is an ordinal* α *such that* $\alpha \neq 0$ *and* α *is not a successor ordinal.*

Definition 24 (37c) $\mathrm{Ord} = \{u \mid u$ *is an ordinal*$\}$.

The following is the well-known Burali-Forti paradox.

Theorem 11 (Exercise15) Ord *is not a set.*

Proof Assume the contrary. Ord is transitive and every element of Ord is transitive. Ord $\in$ Ord. Contradiction. □

2.4 The Well-Ordering Theorem

Theorem 12 (72c) *For every set* x *there exists ordinal* α*, function* f *such that*

$$f : \alpha \leftrightarrow x.$$

Proof Let x be a set. Define
$F(\alpha) =$ case $x \setminus (F"\alpha) = \emptyset \rightarrow x$, case $(x \setminus (F"alpha)) \neq \emptyset \rightarrow$ choose an element v of $x \setminus (F"alpha)$ in v for α in Ord.
(1) There exists α such that $F(\alpha) = x$.

Proof. Assume the contrary.
(1a) For all $\alpha \in \mathrm{Ord} F(\alpha) \in x \setminus (F"\alpha)$.
(1b) $F : \mathrm{Ord} \to x$.
(1c) F is an injective function from Ord to x.
Proof. Let $\beta, \gamma \in \mathrm{Ord}$. Assume $\beta \neq \gamma$. Then $F(\beta) \neq F(\gamma)$.
Proof. Case $\beta < \gamma$. $F(\gamma) \in x \setminus (F"\gamma)$. $F(\beta) \neq F(\gamma)$. end.
Case $\gamma < \beta$. $F(\beta) \in x \setminus (F"\beta)$. $F(\beta) \neq F(\gamma)$. end. end. qed. qed.
Take an ordinal α such that $F(\alpha) = x$ and for all $\beta(\beta < \alpha => F(\beta) \neq x)$.
Proof. Assume the contrary. Let us show that for all ordinals α $F(\alpha) \neq x$.
Proof by induction on alpha. qed. qed.
Define $f(\beta) = F(\beta)$ for β in α.
(2a) For all $\beta \in \alpha f(\beta) \in x \setminus (F"\beta)$.
(2b) $f : \alpha \to x$.
(2c) f is an injective function from α to x.
Proof. Let $\beta, \gamma \in \alpha$. Assume $\beta \neq \gamma$. Then $f(\beta) \neq f(\gamma)$.
Proof. Case $\beta < \gamma$. $f(\gamma) \in x \setminus (F"\gamma)$. $f(\beta) \neq f(\gamma)$. end.
Case $\gamma < \beta$. $F(\beta) \in x \setminus (F"\beta)$. $f(\beta) \neq f(\gamma)$. end. end. qed.
(2d) f is a surjective function from α to x.
Proof. Assume the contrary. $\mathrm{range}(f) \neq x$. $x \setminus (F"\alpha) \neq \emptyset$. $F(\alpha) \in x$. $x \in x$.
Contradiction. qed. □

3 Remarks

3.1 Axiomatics

The present text uses only a part of the set theoretic axioms. Infinity and Powerset are not used, so that the text in principle could be interpreted within finitary set theory. Eventually, the axiomatic assumptions in the axioms and mechanisms of Naproche-SAD have to be fixed and determined.
Although using KM set theory appears more elegant than ZF, if may be better to restrict to ZF to be in line with other systems.

3.2 Natural and Formal Text

The text of the proof of the well-ordering theorem in the lecture notes is of similar structure and length as the formal proof, although the formal proof misses some features of the informal proof like directly taking a minimal ordinal with some property. Instead, induction is used to show the existence of a minimal ordinal.
A marked weakness of Naproche-SAD are the term-rewriting facilities. Fortunately in set theory, this issue hardly arises.
We conjecture that most of the set theory lecture notes can eventually be reformulated in ForTheL.

3.3 Possible Improvements

Future versions of Naproche-SAD should allow more chaining of statements. In Definitions and Theorems, there could be chains of definitions or conclusions instead of single statements. One could allow listing and enumeration environments of LaTeX. Then theorems could have been formalized as one theorem with parts $(a), (b)$ etc.
Argumentative connectives like "since" should be included in extensions of ForTheL. Presently they have to be modelled by the "Indeed" construct or by listing of premises.
Some more constructions, like choosing minimal witnesses of properties.

4 From LaTeX to ForTheL

The LaTeX-input to Naproche-SAD is achieved by a simple-minded translation which transforms LaTeX-text into corresponding ForTheL-text. A lot of typesetting information is simply filtered out and not given to Naproche-SAD. In particular the $-symbols are filtered out since in ForTheL there is no real difference between symbolic and textual input. Style information like \emph{word} becomes simply `word`.
LaTeX-environments like the definition, lemma, theorem or proof environments are modified into the corresponding ForTheL top level sections. E.g., the text
\begin{theorem}
theoremtext
\end{theorem}
becomes
`Theorem. (the translation of) theoremtext`
Actually only text within Forthel environments of the form
\begin{forthel}
theoremtext
\end{forthel}
is transmitted to Naproche-SAD. Everything outside those environments is filtered away so that one can write free LaTeX outside the ForTheL content. Actually this document is a LaTeX-document containing several ForTheL-environments. These are typeset with a vertical marginal line.
All this filtering is coded into the Haskell file `LaTeX.hs` which exports the translation function `fromLaTeX :: String -> String`. This function is employed in the source code of SAD3 in the file `Reader.hs` where `fromLaTeX` is applied to the input string within the `reader`-function.
Eventually a simple LaTeX-format should become the standard input format for Naproche-SAD. This could be integrated into the standard workflow of mathematicians. If parts of a text should be checked by Naproche-SAD put them into ForTheL-environments and feed the text to Naproche-SAD.

5 Discussion

Formalizing in SAD3 is not straightforward. The ForTheL language is a relatively rich language which appears like a natural language with a rather complicated (formal) grammar. This makes it at first hard to predict which constructs will be accepted.
Also specific forms of constructs, although logically and intuitively equivalent, may greatly influence checking times or the checkability at all. Probably, some logical reformulations, however simple, may lead to further resolution steps and expand the search space of the ATP. Perhaps some obvious logical equivalences could be handled by simplifications in the reasoner instead of by the ATP?
The user of Naproche-sAD seems to play something like a cooperative trial-and-error game with the ATP. There are cases, where the system is unable to do trivial steps like deducing A from a long conjunction like $(A \wedge B \wedge C \wedge D \wedge E)$ whereas sometimes impressive argumentative chains of lemmas are found immediately. It seems that the heuristics of the ATP may give a "long" formula like $(A \wedge B \wedge C \wedge D \wedge E)$ low priority, whereas the powerful resolution algorithm will find non-trivial lemma applications through unification. It would be desirable to have more intuitions about the abilities of the ATP. Perhaps one could even tune the ATP towards the kind of tasks that Naproche-SAD produces.