Naturalness in Formal Mathematics

by Peter Koepke

Mathematical Institute, University of Bonn, Germany

Email: koepke@math.uni-bonn.de

Abstract

Formal mathematics is often considered to be fundamentally different from common mathematics. Whereas formal mathematics requires reformulations of mathematics in strictly formal languages, the informal, intuitive, and sometimes vague and incomplete style of common mathematics is considered to be the natural way to do mathematics, adequate for human authors and readers. We argue that with the help of powerful computer tools the gap between formal and natural mathematics can be narrowed. Wysiwyg editors like $\mathrm{T_{E}X_{MACS}}$ provide natural editing facilities; techniques from computational linguistics allow to translate (controlled) natural language texts into formal logics; automatic theorem provers are able to bridge gaps in proofs. We support these points with textbook style example texts accepted by the system NaProChe (Natural Language Proof Checking): the examples appear nearly natural to human readers, but they are strictly formal texts with respect to the formal system NaProChe.

Keywords: Natural language, formal mathematics, discourse representation theory, automatic theorem proving

1 Introduction

Principia Mathematica by Russell and Whitehead [16] were, apart from their epoch making role for the foundations of mathematics, the first comprehensive attempt in *formal mathematics*. Russell and Whitehead proved numerous mathematical facts from first principles solely by use of logical derivations. *Principia* express the program and the possibility of formal mathematics: define a formal mathematical language, fix derivation rules and initial axioms, and prove *all* of mathematics within that formal system. In subsequent years the program of formal mathematics was further strengthened by better logical systems, by the Zermelo-Fraenkel axioms of set theory (see [5]), in which all mathematical notions can be expressed, and by the Gödel completeness theorem [4] which states that usual first-order proof calculi allow to derive all logical consequences of the axioms. So formal mathematics is possible *in principle*.

On the other hand *Principia* lead to the view that a complete formalisation and thus formal mathematics would be too extensive and mathematically uninteresting. It has often been pointed out, that the proof of $1 + 1 = 2$ appears only on page 379 of the first edition, and in rather unusual notation. So Russell [10], p. 155 himself writes about the enterprise:

> ... my intellect never quite recovered from the strain.

The encyclopedic *Elements of Mathematics* by N. Bourbaki [2], p. 11 stress the completeness and exactness of the presentation but a strictly formal presentation is rejected straightaway:

> If formalized mathematics were as simple as the game of chess, then once our chosen formalized language had been described there would remain only the task of writing out our proofs in this language, [...] But the matter is far from being as simple as that, and no great experience is necessary to perceive that such a project is absolutely unrealizable: the tiniest proof at the beginnings of the Theory of Sets would already require several hundreds of signs for its complete formalization. [...] formalized mathematics cannot in practice be written down in full, [...] We shall therefore very quickly abandon formalized mathematics, [...]

Until the 1960's or 1970's this view was shared by most logicians and mathematicians. Formal proofs were only accepted in principle as the ultimate criterion for mathematical correctness:

> As to precision, we have now stated an absolute standard of rigor: A Mathematical proof is rigorous when it is (or could be) written out in the first-order predicate language $L(\in)$ as a sequence of inferences from the axioms ZFC, each inference made according to one of the stated rules. [...] When a proof is in doubt, its repair is usually just a partial approximation to the fully formal version. [8], p. 377

So mathematicians continue to work successfully with "partially formal" or semi-formal methods. They argue about a universe of mathematical objects and properties in intuitive ways which (seem to) presuppose that these objects exist like atoms and waves. Mathematical formulas are used for exactness when needed, but "too much" formalism is avoided in the interest of bringing out the "ideas" of notions or proofs. Instead one uses natural language, descriptive notions, metaphors, and examples. Depending on the proficiency level of writers and readers, only a few proof steps are explicitly written down, the others are "left to the reader".

Mathematics has developed an expert language adequate for formulating, proving and communicating mathematical problems and results. This language, especially in its English variety, is universally spoken, written and read in the mathematical community. Mathematical truth can only be established by proofs. Particular argumentative styles have evolved which regulate the degree of expliciteness and exactness in texts. A first-year undergraduate textbook will contain more proof details and steps than a research paper. Hence there are strong similarities to argumentative natural languages in other domains. From a linguistic perspective, the expert language of mathematics is distinguished by the fact that its intended meaning in principle fully agrees with its standard linguistic formal semantics expressed in first-order predicate logic.

The language of mathematics is considered "natural" by its practitioners. Indeed there are strong intuitions about naturalness in many parts of mathematics: natural numbers, natural deductions; proofs are considered to be natural or "unnatural", and mathematicians are striving for naturality. Usually formal mathematics is considered "unnatural", since formal texts cannot readily be authored, read and checked by human readers.

The practical side of formal mathematics changed drastically with the arrival of electronic computers. The tedious task of checking formal rules in statements and derivations can easily be taken over by the text processing capabilities of computers. MCCARTHY [9] wrote in 1962:

> Checking mathematical proofs is potentially one of the most interesting and useful applications of automatic computers. Computers can check not only the proofs of new mathematical theorems but also proofs that complex engineering systems and computer programs meet their specifications. Proofs to be checked by computer may be briefer and easier to write than the informal proofs acceptable to mathematicians. This is because the computer can be asked to do much more work to check each step than a human is willing to do, and this permits longer and fewer steps. . . . The combination of proof-checking techniques with proof-finding heuristics will permit mathematicians to try out ideas for proofs that are still quite vague and may speed up mathematical research.

Development in the 1960's first focussed on provers instead of checkers but it was soon realised that automatic proving ran into serious complexity problems. As a consequence N. G. DE BRUIJN developed the pioneering *Automath* system to assist mathematicians in the authoring and checking of formal proofs. The approach was intended to be "natural" as DE BRUIJN explained in his *Reflections on Automath* [3], p. 215:

So I got to studying the structure of mathematics by starting from the existing mathematical language and from the need to make such language understandable for machines. I think we might call that appoach "natural". "Natural deduction" is a part of it. At once I have to stress that the use of the word "natural" has little to do with nature, or with the true nature of things. It refers to the reasoning habits of many centuries, and [...] the Automath project tries to bring communication with machines in harmony with the usual communication between people.

The general applicability of Automath was demonstrated by L. S. VAN BENTHAM JUTTING [13] by translating the *Grundlagen der Analysis* of E. LANDAU [7].

Automath employed a difficult to read and write, LISP-like input language. The problem of "readability" of the language of formal mathematics was addressed by the MIZAR system of A. TRYBULEC [1]. The MIZAR language resembles an ALGOL like computer language and captures various features of the common mathematical language. Moreover MIZAR allows a more natural style of proving by bridging "obvious" proof steps using an inbuilt automatic prover. The system accepts simple transformations and deductions which in ordinary proofs are used without further justification.

The following is an excerpt of the definition of an *ordinal number*, taken from the MIZAR article `ORDINAL1` which is part of the MIZAR mathematical library:

```
The Mizar article:
    The Ordinal Numbers
by
Grzegorz Bancerek
[...]
 definition let X;
  attr X is epsilon-transitive means
:Def2: for x st x in X holds x c= X;
  attr X is epsilon-connected means
:Def3: for x,y st x in X & y in X holds x in y or x = y or y in x;
 end;
[...]
 definition let IT be set;
  attr IT is ordinal means
:Def4:  IT is epsilon-transitive epsilon-connected;
 end;
[...]
```

The example shows that MIZAR has an input language which is "readable" to some degree. The automatic prover allows to reduce the blowup from ordinary to formal proof to a reasonable size. MIZAR constitutes a large formal system consisting of a language and a proof calculus based on the inbuilt proof checker. Every deduction provable by the proof checker can be seen as a rule of the MIZAR proof calculus. We are thus considering a formal system with extremely many deduction rules which can only be handled by computer. MIZAR goes some way towards natural mathematics which partly explains its success.

But one would hardly call the above text "natural". A natural paraphrasing of the definition of an ordinal would, e.g., read like:

"A set is called an *ordinal* if it is $\in$-transitive and $\in$-connected."

The MIZAR notation does not use standard mathematical symbols like $\in$ for elementhood; the MIZAR language is grammatically incorrect English; proofs in MIZAR still involve a great number of trivial intermediate steps.

This poses the

Challenge. *To devise a strictly formal system for mathematics, implemented by computer, whose input language is an extensive part of the common mathematical language, and whose proof style is close to proof styles found in the mathematical literature.*

There are several motivations for this goal.

–   Mathematical logic claims to model the axiomatic method of modern mathematics. After successfully mathematicallly but it does not fully reflect the actual languages and arguments of mathematicians.

–   The challenge is an extrapolation of a general trend in formal mathematics towards naturalness.

–   The gap between natural and formal mathematical proofs is a topic in the philosophy of mathematics. Bridging the gap may influence that discussion.

–   The language of mathematics as an expert language stands out by the fact that its intended semantics is in principle fully captured by a translation into first-order formulas. This makes the language of mathematics a paradigm for studies in theoretical and computational linguistics.

–   Meeting the challenge will have applications for mathematical authoring and tutoring systems.

This paper claims that by an intelligent combination of techniques naturalness of formalisations may be achieved at least for limited areas of mathematics. It substantiates some of the goals formulated in an earlier joint paper with BERNHARD SCHRÖDER [6]. We discuss aspects of mathematical texts and authoring and indicate how these are modeled in the system NaProChe (Natural Language Proof Checking). The NaProChe project is under development at the University of Bonn (`www.naproche.net`). We then discuss some of the factors involved with respect to their influence on naturalness.

There have been other developments of computer systems for "understanding" natural mathematical proofs. The PhD projects of DONALD SIMON [12] and CLAUS ZINN [18] studied the analysis of natural texts in number theory and were able to verify a limited range of proofs from textbooks (see also SIMON [11] and ZINN [17]. The project VeriMathDoc (User-Oriented Interactive Generation of Verifiable Mathematical Documents) by SERGE AUTEXIER and STEFAN BUSEMANN is currently developing a mathematical assistant system that naturally supports mathematicians in the development, authoring and publication of mathematical work.

2 Authoring mathematical texts

The standard textbook *Set Theory* by JECH [5] introduces ordinal numbers, CANTOR's generalisation of the integer numbers, on p. 19:

1. [...] Informally, an ordinal number is the order-type of a well-ordered set.
2. We shall now give a formal definition of ordinal numbers.
3.   **Ordinal Numbers**
4. The idea is to define ordinal numbers so that
5. $\alpha < \beta$ iff $\alpha \in \beta$,   and $\alpha = \{\beta : \beta < \alpha\}$
6. **Definition 2.9.** A set T is *transitive* if every element of T is a subset of T.
7. (Equivalently, $\bigcup T \subseteq T$, or $T \subseteq P(T)$.)
8. **Definition 2.10.** A set is an *ordinal number* (an *ordinal*) if it is transitive

9. and well-ordered by $\in$.

Let us discuss some features of this "natural" piece of mathematics.

a) The text is set by a mathematical typesetting system like L^AT_EX. The structure of the text is mirrored in the layout, in particular by environments like **Definition** (lines 6, 8).

b) The text is semi-formal, with more natural language phrases than mathematical formulas. Natural language is mixed with symbolic notation (5, 7).

c) The text uses a narrative style which resembles a talk or a tutorial explanation. It explicitly stresses informal mathematical ideas (1, 4) and intentions (2). Definitions are explained by equivalent properties (7).

d) A definition is viewed as a convention of language. In (8), an abbreviated usage ("ordinal" instead of "ordinal number") is introduced as part of the definition.

e) The text is partially incomplete: in (5) one has to infer from the context that the variables α and β range over ordinal numbers.

f) The text does not provide all details of the argument. The equivalences in line (7) are not proved at all.

g) The text fragment assumes a general background theory introduced before.

Authoring such texts involves:

A) General architecture of the background theory, choice of conventions, notations and symbols.

B) Structuring of the intended text, often in the definition-theorem-proof style.

C) Finding proofs of theorems.

D) Formulating mathematical statements with a view towards readability and informativeness.

E) Typesetting the text.

F) Critical checking of the evolving text for stylistic, grammatical and mathematical correctness, and corrections when necessary.
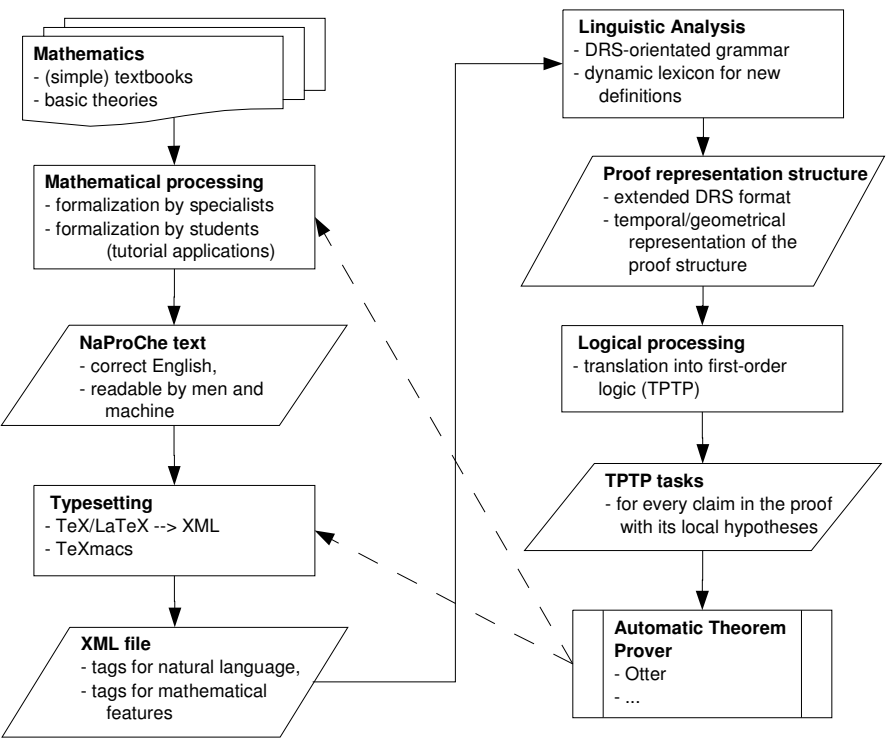
Ideally, a formal system meeting the above Challenge should incorporate A), should provide a rich language for B) and D), should be compatible with E), and should only accept texts satisfying F). The design of such a system will combine work in mathematics, linguistics, mathematical typesetting, logic, and automatic theorem proving.

3 The NaProChe project

The NaProChe project (Natural language Proof Checking) was initiated by Bernhard Schröder and the present author at the University of Bonn to focusses on an interdisciplinary study of the semi-formal language of mathematics. A central goal of NaProChe is to develop a controlled natural language (CNL) for mathematical texts and adapted proof checking software which checks the CNL for syntactical and mathematical correctness. The project is still at a prototypical stage, further information is available at `www.naproche.net`.

The NaProChe system accepts LᴬTₑX-style texts, consisting of mathematical formulas and connecting natural language text from a controlled natural language. The natural language is parsed using techniques from computational linguistics and transformed into first-order formulas. The formulas are given to an automatic theorem prover who checks whether each formula of an argument is a logical consequence of preceding formulas or axioms.

The flow of information in the NaProChe system is depicted in the following diagram where the dashed lines indicate feedback paths:

Mathematics
- (simple) textbooks
- basic theories

Mathematical processing
- formalization by specialists
- formalization by students
  (tutorial applications)

NaProChe text
- correct English,
- readable by men and
  machine

Typesetting
- TeX/LaTeX --> XML
- TeXmacs

XML file
- tags for natural language,
- tags for mathematical
  features

Linguistic Analysis
- DRS-orientated grammar
- dynamic lexicon for new
  definitions

Proof representation structure
- extended DRS format
- temporal/geometrical
  representation of the
  proof structure

Logical processing
- translation into first-order
  logic (TPTP)

TPTP tasks
- for every claim in the proof
  with its local hypotheses

Automatic Theorem Prover
- Otter
- ...

The development of NaProChe is driven by examples like the following where a very simple grammar allows to formulate the introduction of ordinal numbers in a nearly natural way.

1. **The Theory of Ordinals**

2. We make some set theoretic assumptions. The *empty set* $\emptyset$ is characterised by:
3.   Assume that $\neg\exists x\, x \in \emptyset$.
4.   Assume that for all x not $x \in x$.
5. We define *ordinals* according to John von Neumann:
6.   Define for all x Trans(x) if and only if $\forall u \forall v\, (u \in v \wedge v \in x \rightarrow u \in x)$.
7.   Define for all x x is an ordinal iff Trans$(x) \wedge \forall y\, (y \in x \rightarrow$ Trans$(y))$.

Some comments:

a) The actual mathematical text is written in a special indented environment (3, 4, 6, 7). Unindented text is interpreted as a commentary (1, 2, 5).

b) Input to the NaProChe system takes place via a LaTeX-quality text editor. All TeX symbols are available.

c) The mathematical text is written in the NaProChe language in which mathematical formulas and connecting prose are interspersed. Due to limitations in the current implementation some features are not really natural like the outer brackets in lines 6, 7. It is clear that this can be smoothed out by a better grammar for the natural language phrases.

4 Enhancing the naturalness of formal systems

To show that the gap between natural and formal mathematics can in principle be narrowed one may search for favourable mathematical contexts which lend themselves to uncomplicated formalisations. Experiences from those contexts may yield information about the general project to "naturalise" formal mathematics.

The naturalness of mathematical texts depends on many factors which are related to human expectations and abilities in various areas. Fields of mathematics have developed their own sublanguage of the mathematical language with specific symbols, methods and implicit background assumptions. A text may be directed at an audience with a specific previous knowledge. These factors will also be appreciated differently by different individuals. So we can only discuss some general aspects of a formal system which affect naturalness.

4.1 Mathematical aspects

Mathematical theories strongly influence their style of presentation. Obviously a theory is more adequate for a natural formalisation if it is highly formal anyway.

If a theory is based on intuitively well-understood concepts from, e.g., geometry, physics, or social interaction, then the presentation tends to appeal to those intuitions in plain but linguisticly involved natural language which may be difficult to analyse by methods of computational linguistics. If a theory is built up axiomatically or algebraically the development is usually more formal.

In the course of unfolding a theory new intuitions evolve and are employed. So the beginnings of a theory will be more adequate for natural formalisations than advanced parts.

Mathematical texts combine logical arguments with numerical and symbolic computations. Up to now the techniques of formal mathematics have emphasised logical arguments, so one should prefer "logical" theories.

Set theory in some appropriate axiomatisation appears to be a feasable system for the general formalisation of mathematics, and has been used in several formalisation projects, e.g., by MIZAR.

4.2 Linguistic aspects

The language of mathematics combines natural language with mathematical formulas. Most natural language words and constructs retain their original meanings, but there are some exceptions and extensions. Through definitions, a word may get a new mathematical meaning: after the definition of an algebraic "ring" or "field", these words retain their standard grammar ("rings" is also the plural of an algebraic ring) but the meaning is completely determined by the definition. Also new words may be introduced. Concerning the meaning of grammatical constructs, the standard mathematical language tries to be complete and disambiguous. Whereas the coordination with "or" is often understood as "either-or", the usual mathematical interpretation is the inclusive "or"; an exclusive "or" has to be made explicit by "either-or" or other means. The tendency to avoid ambiguity helps the linguistic analysis of the mathematical language.

On the other hand mathematical exactness requires an analysis of *every* sentence of a text. The analysis must be intelligable for a human author so that the author can keep control over the process. This motivates the use of a grammar based deep linguistic anas.

A mathematical text is a discourse in the language of mathematics, i.e., a structured sequence of sentences. Linguistic has developed theories and methods for dealing with discourses. Discourse representation theory provides means to transform a given discourse into a logic representation which retains important structural elements of the discourse like the scopes of certain constructs or the linkages of sentences through pronouns and other anaphora.

Altogether one is lead to the definition of a *controlled natural language* (CNL) which is a subset of the natural language of mathematics but with a strict formal grammar and formal semantics. There are several powerful controlled languages with an associated computer implementation. The language *Attempto Controlled English* (ACE) combines a rich "natural" language with several interesting mechanisms for mathematical applications. In the NaProChe project we are developing a controlled mathematical language along the lines of ACE with modifications necessary for mathematics.

4.3 Proof representation structures

ACE translates input texts into discourse representation structures as an intermediate layer between natural input and its first-order equivalent. There are, however, aspects of proofs which standard discourse representation theory does not model properly, like the order of statements or the scope of assumptions. This necessitates the introduction of proof representation structures (PRS) which are enriched discourse representation structures able to represent the procedural aspects of a proof. PRS seem to be the crucial data structures to connect natural and formal proofs.

In the NaProChe system the PRS contains information on the visibility of relevant assumptions for every statement in the proof. Immediately preceding statements or distinguished main lemmas or theorems are the most probable and "visible" preconditions for a statement so that these should be attempted with higher priority for the proof of the statement. A good design of visibility criteria can help the automatic prover and make proofs more natural in the sense that "obvious" potential preconditions are selected by system in a way similar to the tactics a human prover.

4.4 Logical aspects

In principle all mathematical statements can be unravelled into first-order statements about sets and the membership-relation. Often such an unravelling has an exponential growth and is not feasable for practical purposes. Therefore intermediate logics have to be used which are close to the "natural logic" of the mathematical input text. This requires an efficient (weak) type system so that complex objects and notions can be atomic objects and notions of a higher level in the type system. This was already described by BOURBAKI [2], p. 10:

> [...] it is imperative to condense the formalized text by the introduction of a fairly large number of new words (called *abbreaviating symbols*) and additional rules of syntax (called *deductive criteria*). By doing this we obtain languages which are much more manageable than the formalized language in its strict sense. Any mathematician will agree that these condensed languages can be considered as merely shorthand transcriptions of the original formalized language.

4.5 Employing automatic theorem proving

Proofs come with a certain step size or granularity depending on the style of proof. The steps of the proof could be proved formally but this is not done if the author thinks that it would be mathematically irrelevant. Checking a proof requires justification of the proof steps, and automatic checking requires automatic proving of the steps. Ideally an automatic theorem prover (ATP) like Otter or Vampire would be able to prove steps in proofs of a natural granularity. Experiments with systems like SAD [15] indicate that at least in certain mathematical contexts, automatic provers can interpolate natural granularities in non-trivial proofs.

For certain applications it may on the contrary be necessary to weaken the automatic prover. When using the NaProChe system as a tutorial system for introductory logic the prover was only allowed to carry out single rules of the logic calculus under consideration.

4.6 Typesetting

Mathematical texts stand out by the elaborate typography for printing formulas. Systems like TeX and LaTeX enable mathematicians to do mathematical typesetting without expert help. These systems have become defacto standards in mathematical publishing and can be considered the "natural" format for communicating mathematics. Naturalised formal mathematics should accept text in those formats.

The NaProChe system uses the editor TeX$_{MACS}$ [14] which can be approximately described as a wysiwyg LaTeX editor. TeX$_{MACS}$ is extensible and can easily be modified to be able to control the NaProChe system. Using a wysiwyg editor for writing mathematical texts has certain advantages over writing LaTeX in a text editor: whereas LaTeX can produce the same graphical result in many ways, especially through macros, the user of a wysiwyg editor will tend to produce the result in ways built into the editor. The latter increases the chances that an internal representation is uniquely determined by the graphical result.

5 Conclusion

In light of the preceding discussion we conjecture that a combination of adaptations of the following systems may yield a system in which natural proofs for extensive areas of mathematics can be authored and checked for correctness:

— background theory: set theory with a rich language and flexible weak typing

- – input format and editor: L^AT_EX or T_EX_{MACS}

- – input language and linguistic analysis: controlled natural language inspired by Attempto Controlled English, with mathematical modifications and enrichments

- – proof representations: enriched discourse representation structures, with further features from the SAD system

- – automatic theorem provers: Otter, Vampire, ...

As a further indication for the feasability of this project we conclude with a fundamental result from the beginnings of set theory formalized and checked in the current version of NaProChe. The language of the example is grammatical and stylistically acceptable though inelegant.

The Burali-Forti paradoxon was one of the first of the famous paradoxes in set theory: the class Ord of all ordinals is *not* a set. We write $\mathrm{Ord}(u)$ for "x is an ordinal":

Theorem. There is no x such that $\forall u\,(u \in x \leftrightarrow \mathrm{Ord}(u))$.

Proof. Assume for a contradiction that there is x such that $\forall u\,(u \in x \leftrightarrow \mathrm{Ord}(u))$. Take x such that $\forall u\,(u \in x \leftrightarrow \mathrm{Ord}(u))$.

Lemma. $\mathrm{Ord}(x)$.

Proof. Let $u \in v$ and $v \in x$. Then $u \in t$. $v \in x$. $\mathrm{Ord}(v)$. Together we have $u \in v$ and $\mathrm{Ord}(v)$. So $\mathrm{Ord}(u)$. $u \in x$. Thus $\forall u \forall v\,(u \in v \wedge v \in x \rightarrow u \in x)$. Hence $\mathrm{Trans}(x)$.

Consider $y \in x$. Then $\mathrm{Ord}(y)$. $\mathrm{Trans}(y) \wedge \forall z\,(z \in y \rightarrow \mathrm{Trans}(z))$. In particular $\mathrm{Trans}(y)$. Thus $\forall y\,(y \in x \rightarrow \mathrm{Trans}(y))$.

Together we have $\mathrm{Trans}(x) \wedge \forall y\,(y \in x \rightarrow \mathrm{Trans}(y))$. Hence x is an ordinal. Qed.

Then $x \in x$. But $\neg x \in x$. Contradiction. Thus we get a contradiction. Qed.

Bibliography

[1] Howard Blair and Andrzej Trybulec. Computer assisted reasoning with mizar. In *Proceedings of IJCAI'85*, pages 26–28. Morgan Kaufmann, 1985.

[2] Nicolas Bourbaki. *Theory of Sets*. Springer, Berlin, 2004.

[3] Nicolaas Govert de Bruijn. Reflections on automath. In Rob P. Nederpelt et al, editor, *Selected Papers on Automath*, volume 133 of *Studies in Logic*, pages 201–228. Elsevier, 1994.

[4] Kurt Gödel. Die Vollständigkeit der Axiome des logischen Funktionenkalküls. *Monatshefte für Mathematik und Physik*, 37:349–360, 1930.

[5] Thomas Jech. *Set Theory*. Monographs in Mathematics. Springer, Berlin, 2002.

[6] Peter Koepke and Bernhard Schröder. Natürlich formal. In Gerd Willee et al, editor, *Computational Linguistics - Achievements and Perspectives*. Gardez!-Verlag, St. Augustin, 2002.

[7] Edmund Landau. *Grundlagen der Analysis*. Akademische Verlagsgesellschaft, Leipzig, 1930.

[8] Saunders Mac Lane. *Mathematics: Form and Function*. Springer, Heidelberg, 1986.

[9] John McCarthy. Computer Programs for Checking Mathematical Proofs. In J. C. E. Decker, editor, *Recursive Function Theory: Proceedings of the Fifth Symposium in Pure Mathematics of the American Mathematical Society*, volume V of *Proceedings of Symposia in Pure Mathematics*, pages 219–227. American Mathematical Society, 1962.

[10] Bertrand Russell. *Autobiography*. Routledge, 1998.

[11] Donald Simon. Checking Natural Language Proofs. In Ewing Lusk and Ross Overbeek, editors, *9th International Conference on Automated Deduction*, volume 310 of *Lecture Notes in Computer Science*, pages 141–150. Springer Verlag, 1988.

[12] Donald Simon. *Checking Number Theory Proofs in Natural Language*. PhD thesis, UT Austin, 1990.

[13] L. S. van Benthem Jutting. *Checking Landau's "Grundlagen" in the Automath system*. PhD thesis, Eindhoven University of Technology, 1977.

[14] Joris van der Hoeven. Gnu texmacs: A free, structured, wysiwyg and technical text editor. In Daniel Flipo, editor, *Le document au XXI-ieme siecle*, volume 39-40 of *Actes du congres Gutenberg*, pages 39–50, 2001.

[15] K. Verchine, A. Lyaletski, and A. Paskevich. System for automated deduction (sad): a toll for proof verification.

[16] Alfred North Whitehead and Bertrand Russell. *Principia Mathematica*. Cambridge University Press, 1910, 1912, 1913.

[17] Claus Zinn. A Computational Framework for Understanding Mathematical Discourse. *Logic Journal of IGPL*, 11(4):457–484, 2003.

[18] Claus Zinn. *Understanding Mathematical Discourse*. PhD thesis, Universität Erlangen-Nürnberg, 2003.