Naproche: Analyzing Mathematical Language Logically and Linguistically

Marcos Cramer

TU Dresden

1 February 2020

Outline

Formal mathematics

- **Formal mathematics**: Mathematical proofs are expressed in a formal proof calculus.

Formal mathematics

- **Formal mathematics**: Mathematical proofs are expressed in a formal proof calculus.
- Each proof step can be mechanically checked.

Formal mathematics

- **Formal mathematics**: Mathematical proofs are expressed in a formal proof calculus.
- Each proof step can be mechanically checked.
- Increased trust in complex proofs

Formal mathematics

- **Formal mathematics**: Mathematical proofs are expressed in a formal proof calculus.
- Each proof step can be mechanically checked.
- Increased trust in complex proofs
- Many further advantages

Formal mathematics

- **Formal mathematics**: Mathematical proofs are expressed in a formal proof calculus.
- Each proof step can be mechanically checked.
- Increased trust in complex proofs
- Many further advantages
- Increasing interest among mathematicians

Input language

Example: Irrationality of $\sqrt{2}$

In natural mathematical language:

If $\sqrt{2}$ is rational, then the equation $a^2 = 2b^2$ is soluble in integers a, b with $(a, b) = 1$. Hence a^2 is even, and therefore a is even. If $a = 2c$, then $4c^2 = 2b^2$, $2c^2 = b^2$, and b is also even, contrary to the hypothesis that $(a, b) = 1$.

In Mizar (shortened):

```
theorem
  sqrt 2 is irrational
proof
 assume sqrt 2 is rational;
 then consider i being Integer, n being Nat such that
W1: n<>0 and
W2: sqrt 2=i/n and
W3: for i1 being Integer, n1 being Nat st n1<>0
 & sqrt 2=i1/n1 holds n<=n1 by RAT_1:25;
A5: i=sqrt 2*n by W1,XCMPLX_1:88,W2;
C: sqrt 2>=0 & n>0 by W1,NAT_1:19,SQUARE_1:93;
 then i>=0 by A5,REAL_2:121;
 then reconsider m = i as Nat by INT_1:16;
```

Naproche: Natural input language

Burali-Forti paradox in Naproche

Axiom 1: There is a set $\emptyset$ such that no y is in $\emptyset$.
Axiom 2: There is no x such that $x \in x$.

Define x to be transitive if and only if for all u, v, if $u \in v$ and $v \in x$ then $u \in x$.
Define x to be an ordinal if and only if x is transitive and for all y, if $y \in x$ then y is transitive.

Theorem: There is no x such that for all u, $u \in x$ iff u is an ordinal.
Proof:
Assume for a contradiction that there is an x such that for all u, $u \in x$ iff u is an ordinal.
Let $u \in v$ and $v \in x$. Then v is an ordinal, i.e. u is an ordinal, i.e. $u \in x$.
Thus x is transitive.
Let $v \in x$. Then v is an ordinal, i.e. v is transitive. Thus x is an ordinal.
Then $x \in x$. Contradiction by axiom 2.
Qed.

Outline

The Naproche Project

- The **Naproche** project (**Na**tural language **Pro**of **Che**cking) studies the **language and reasoning of mathematics** from the perspectives of logic and linguistics.

The Naproche Project

- The **Naproche** project (**Na**tural language **Pro**of **Che**cking) studies the **language and reasoning of mathematics** from the perspectives of logic and linguistics.
- Central goals of Naproche:
  - To develop a controlled natural language (CNL) for mathematical texts.

The Naproche Project

- The **Naproche** project (**Na**tural language **Pro**of **Che**cking) studies the **language and reasoning of mathematics** from the perspectives of logic and linguistics.
- Central goals of Naproche:
  - To develop a controlled natural language (CNL) for mathematical texts.
  - To implement a system, the **Naproche system**, which can check texts written in this CNL for logical correctness.

The Naproche Project

- The **Naproche** project (**Na**tural language **Pro**of **Che**cking) studies the **language and reasoning of mathematics** from the perspectives of logic and linguistics.
- Central goals of Naproche:
  - To develop a controlled natural language (CNL) for mathematical texts.
  - To implement a system, the **Naproche system**, which can check texts written in this CNL for logical correctness.
- Advancement and application of theoretical models from logic and linguistics

Naproche proof checking

- The proof checking algorithm keeps track of a list of first-order formulae considered true, called **premises**.

Naproche proof checking

- The proof checking algorithm keeps track of a list of first-order formulae considered true, called **premises**.
- The premise list gets continuously updated during the verification process.

Naproche proof checking

- The proof checking algorithm keeps track of a list of first-order formulae considered true, called **premises**.
- The premise list gets continuously updated during the verification process.
- Each assertion is checked by an ATP based on the currently active premises.

Example

Suppose n is even. Then there is a k such that $n = 2k$. Then $n^2 = 4k^2$, so $4|n^2$.

Naproche proof checking

- The proof checking algorithm keeps track of a list of first-order formulae considered true, called **premises**.
- The premise list gets continuously updated during the verification process.
- Each assertion is checked by an ATP based on the currently active premises.

Example

Suppose n is even. Then there is a k such that $n = 2k$. Then $n^2 = 4k^2$, so $4|n^2$.

- Sentence-by-sentence proof verification:
  - $\Gamma, even(n) \vdash^? \exists k\ n = 2 \cdot k$

Naproche proof checking

- The proof checking algorithm keeps track of a list of first-order formulae considered true, called **premises**.
- The premise list gets continuously updated during the verification process.
- Each assertion is checked by an ATP based on the currently active premises.

Example

Suppose n is even. Then there is a k such that $n = 2k$. Then $n^2 = 4k^2$, so $4|n^2$.

- Sentence-by-sentence proof verification:
  - $\Gamma, \mathit{even}(n) \vdash^? \exists k \; n = 2 \cdot k$
  - $\Gamma, \mathit{even}(n), n = 2 \cdot k \vdash^? n^2 = 4 \cdot k^2$

Naproche proof checking

- The proof checking algorithm keeps track of a list of first-order formulae considered true, called **premises**.
- The premise list gets continuously updated during the verification process.
- Each assertion is checked by an ATP based on the currently active premises.

Example

Suppose n is even. Then there is a k such that $n = 2k$. Then $n^2 = 4k^2$, so $4|n^2$.

- Sentence-by-sentence proof verification:
  - $\Gamma, even(n) \vdash^? \exists k\ n = 2 \cdot k$
  - $\Gamma, even(n), n = 2 \cdot k \vdash^? n^2 = 4 \cdot k^2$
  - $\Gamma, even(n), n = 2 \cdot k, n^2 = 4 \cdot k^2 \vdash^? 4|n^2$

Naproche proof checking (2)

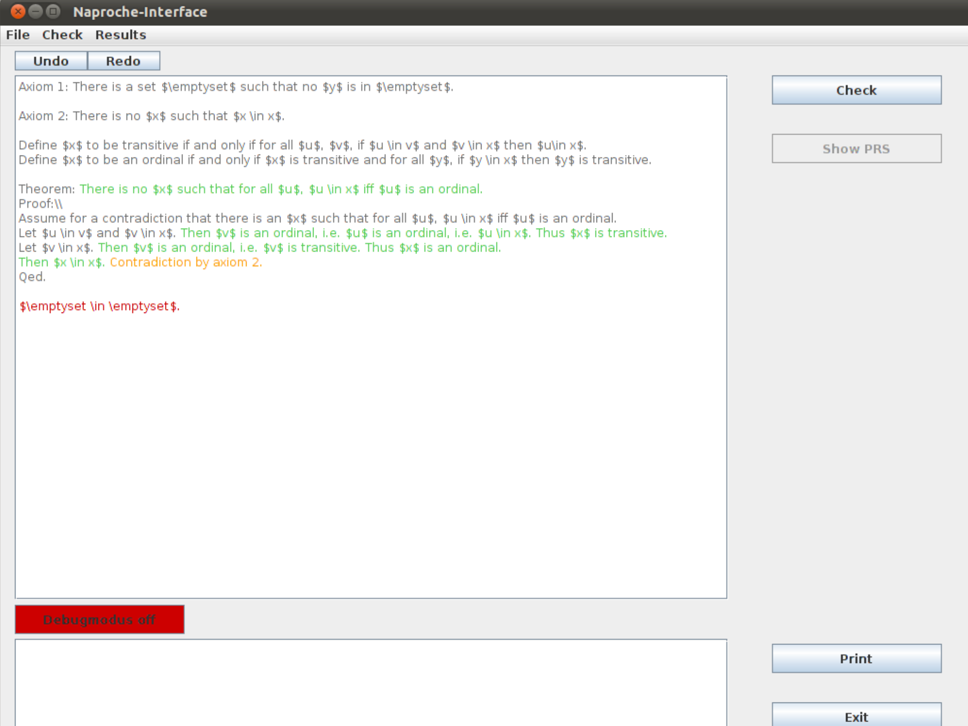
- An assumption is processed in **No-Check Mode**.

Naproche proof checking (2)

- An assumption is processed in **No-Check Mode**.
- The No-Check Mode is also used for φ and ψ in $\neg\varphi$, $\exists x\ \varphi$, $\varphi \vee \psi$ and $\varphi \rightarrow \chi$.

Naproche proof checking (2)

- An assumption is processed in **No-Check Mode**.
- The No-Check Mode is also used for φ and ψ in $\neg\varphi$, $\exists x\ \varphi$, $\varphi \vee \psi$ and $\varphi \rightarrow \chi$.
- We have proved soundness and completeness theorems for the proof checking algorithm.


Naproche-Interface
File
Check
Results
Undo
Redo
Axiom 1: There is a set $\emptyset$ such that no y is in $\emptyset$.
Axiom 2: There is no x such that $x \in x$.
Define x to be transitive if and only if for all u, v, if $u \in v$ and $v \in x$ then $u\in x$.
Define x to be an ordinal if and only if x is transitive and for all y, if $y \in x$ then y is transitive.
Theorem: There is no x such that for all u, $u \in x$ iff u is an ordinal.
Proof:\\
Assume for a contradiction that there is an x such that for all u, $u \in x$ iff u is an ordinal.
Let $u \in v$ and $v \in x$. Then v is an ordinal, i.e. u is an ordinal, i.e. $u \in x$. Thus x is transitive.
Let $v \in x$. Then v is an ordinal, i.e. v is transitive. Thus x is an ordinal.
Then $x \in x$. Contradiction by axiom 2.
Qed.
$\emptyset \in \emptyset$.
Check
Show PRS
Debugmodus off
Print
Exit

Outline

Dynamic Quantification

Example

Suppose n is even. Then there is a k such that $n = 2k$. Then $n^2 = 4k^2$, so $4|n^2$.

Dynamic Quantification

Example

Suppose n is even. Then there is a k such that $n = 2k$. Then $n^2 = 4k^2$, so $4|n^2$.

Example

If a space X retracts onto a subspace A, then the homomorphism $i_* : \pi_1(A, x_0) \to \pi_1(X, x_0)$ induced by the inclusion $i : A \hookrightarrow X$ is injective.

A. Hatcher: Algebraic topology (2002)

Dynamic Quantification (2)

Solution: **Dynamic Predicate Logic** (**DPL**) by Groenendijk and Stokhof

Dynamic Quantification (2)

Solution: **Dynamic Predicate Logic** (**DPL**) by Groenendijk and Stokhof

Example

If a farmer owns a donkey, he beats it.

PL: $\forall x \forall y\ (farmer(x) \wedge donkey(y) \wedge owns(x, y) \rightarrow beats(x, y))$

DPL: $\exists x\ (farmer(x) \wedge \exists y\ (donkey(y) \wedge owns(x, y))) \rightarrow beats(x, y)$

Implicit dynamic function introduction

Suppose that, for each vertex v of K, there is a vertex $g(v)$ of L such that $f(st_K(v)) \subset st_L(g(v))$. Then g is a simplicial map $V(K) \to V(L)$, and $|g| \simeq f$.

M. Lackenby: Topology and groups (2008)

Implicit dynamic function introduction

Suppose that, for each vertex v of K, there is a vertex $g(v)$ of L such that $f(st_K(v)) \subset st_L(g(v))$. Then g is a simplicial map $V(K) \to V(L)$, and $|g| \simeq f$.

M. Lackenby: Topology and groups (2008)

- Solution: **Typed Higher-Order Dynamic Predicate Logic (THODPL)**

THODPL

- There can be a complex term after a quantifier:
  1. $\forall x \, \exists f(x) \, R(x, f(x))$

THODPL

- There can be a complex term after a quantifier:
  1. $\forall x\ \exists f(x)\ R(x, f(x))$
  2. $\forall x\ \exists y\ R(x, y)$
  3. $\exists f\ \forall x\ R(x, f(x))$
- 1 has the same truth conditions as 2.

THODPL

- There can be a complex term after a quantifier:
  1. $\forall x\ \exists f(x)\ R(x, f(x))$
  2. $\forall x\ \exists y\ R(x, y)$
  3. $\exists f\ \forall x\ R(x, f(x))$
- 1 has the same truth conditions as 2.
- But unlike 2, 1 dynamically introduces the function symbol f, and hence turns out to be equivalent to 3.

THODPL in proof checking

- Quantification over a complex term is checked in the same way as quantification over a variable:

For each vertex v of K, there is a vertex $g(v)$ of L such that $f(st_K(v)) \subset st_L(g(v))$. Then g is a simplicial map $V(K) \to V(L)$.

$\Gamma, \text{vertex}(v, K) \vdash^? \exists w\ (\text{vertex}(w, K) \wedge f(st_K(v)) \subset st_L(w))$

THODPL in proof checking

- Quantification over a complex term is checked in the same way as quantification over a variable:

For each vertex v of K, there is a vertex $g(v)$ of L such that $f(st_K(v)) \subset st_L(g(v))$. Then g is a simplicial map $V(K) \to V(L)$.

$\Gamma, \text{vertex}(v, K) \vdash^? \exists w \, (\text{vertex}(w, K) \wedge f(st_K(v)) \subset st_L(w))$

- However, it dynamically introduces a new function symbol.
- The premise corresponding to this quantification gets skolemized with this new function symbol:

$\Gamma, \forall v \, (\text{vertex}(v, K) \to (\text{vertex}(g(v), K) \wedge f(st_K(v)) \subset st_L(g(v))))$
$\vdash^? \text{simplicial\_map}(g, V(K), V(L))$

THODPL in proof checking (2)

- The theorem prover does not need to prove the existence of a function, but its existence may nevertheless be assumed as a premise.

THODPL in proof checking (2)

- The theorem prover does not need to prove the existence of a function, but its existence may nevertheless be assumed as a premise.
- Similarly, $\forall x\ \exists f(x)\ R(x, f(x))$ is proof-checked in the same way as $\forall x\ \exists y\ R(x, y)$, but as a premise it has the force of $\exists f\ \forall x\ R(x, f(x))$.

Definitions

- Definitions dynamically expand the language.

Definitions

- Definitions dynamically expand the language.
- Examples:

Definitions

- Definitions dynamically expand the language.
- Examples:
  - Define c to be $\sqrt{\frac{\pi}{6}}$.

Definitions

- Definitions dynamically expand the language.
- Examples:
  - Define c to be $\sqrt{\frac{\pi}{6}}$.
  - Define $f(x)$ to be x^2.

Definitions

- Definitions dynamically expand the language.
- Examples:
  - Define c to be $\sqrt{\frac{\pi}{6}}$.
  - Define $f(x)$ to be x^2.
  - Define $f_x(y, z)$ to be $x(2y - 5z)$.

Definitions

- Definitions dynamically expand the language.
- Examples:
  - Define c to be $\sqrt{\frac{\pi}{6}}$.
  - Define $f(x)$ to be x^2.
  - Define $f_x(y,z)$ to be $x(2y-5z)$.
  - Define n to be even if and only if there is some k such that $n=2k$.

Definitions

- Definitions dynamically expand the language.
- Examples:
  - Define c to be $\sqrt{\frac{\pi}{6}}$.
  - Define $f(x)$ to be x^2.
  - Define $f_x(y,z)$ to be $x(2y-5z)$.
  - Define n to be even if and only if there is some k such that $n=2k$.
  - Define $m|_k n$ iff there is an $l<k$ such that $m \cdot l = n$.

Definitions

- Definitions dynamically expand the language.
- Examples:
  - Define c to be $\sqrt{\frac{\pi}{6}}$.
  - Define $f(x)$ to be x^2.
  - Define $f_x(y, z)$ to be $x(2y - 5z)$.
  - Define n to be even if and only if there is some k such that $n = 2k$.
  - Define $m|_k n$ iff there is an $l < k$ such that $m \cdot l = n$.
- Definitions are treated like dynamic existential quantifiers.

Definitions

- Definitions dynamically expand the language.
- Examples:
  - Define c to be $\sqrt{\frac{\pi}{6}}$.
  - Define $f(x)$ to be x^2.
  - Define $f_x(y,z)$ to be $x(2y-5z)$.
  - Define n to be even if and only if there is some k such that $n = 2k$.
  - Define $m|_k n$ iff there is an $l < k$ such that $m \cdot l = n$.
- Definitions are treated like dynamic existential quantifiers.
- The existential proof obligation is trivial.

Outline

Undefined terms

- Mathematical texts can involve **potentially undefined terms** like $\frac{1}{x}$.

Undefined terms

- Mathematical texts can involve **potentially undefined terms** like $\frac{1}{x}$.
- Such terms arise by applying **partial functions** to ungrounded terms.

Undefined terms

- Mathematical texts can involve **potentially undefined terms** like $\frac{1}{x}$.
- Such terms arise by applying **partial functions** to ungrounded terms.
- First-order logic has no means for handling partial functions and potentially undefined terms.

Undefined terms

- Mathematical texts can involve **potentially undefined terms** like $\frac{1}{x}$.
- Such terms arise by applying **partial functions** to ungrounded terms.
- First-order logic has no means for handling partial functions and potentially undefined terms.
- We make use of **presupposition theory** from formal linguistics for solving this problem.

Presuppositions

- A **presupposition** of some utterance is an implicit assumption that is taken for granted when making the utterance and needed for its interpretation.

Presuppositions

- A **presupposition** of some utterance is an implicit assumption that is taken for granted when making the utterance and needed for its interpretation.
- Presuppositions are triggered by certain lexical items called **presupposition triggers**, e.g. “the”, “to know”, “to stop”, “still”.

Presuppositions

- A **presupposition** of some utterance is an implicit assumption that is taken for granted when making the utterance and needed for its interpretation.
- Presuppositions are triggered by certain lexical items called **presupposition triggers**, e.g. “the”, “to know”, “to stop”, “still”.

Example

He stopped beating his wife.

Presupposition in mathematical texts

- Most presupposistion triggers are rare or absent in mathematical texts, e.g. "to know", "to stop" and "still".

Presupposition in mathematical texts

- Most presupposistion triggers are rare or absent in mathematical texts, e.g. “to know”, “to stop” and “still”.
- Definite descriptions do appear, e.g. “the smallest natural number n such that $n^2 - 1$ is prime”.

Presupposition in mathematical texts

- Most presupposistion triggers are rare or absent in mathematical texts, e.g. "to know", "to stop" and "still".
- Definite descriptions do appear, e.g. "the smallest natural number n such that $n^2 - 1$ is prime".
- A special mathematical presupposition trigger: Expressions denoting partial functions, e.g. "/" and "$\sqrt{}$"

Proof checking algorithm with presuppositions

Presuppositions also have to be checked in No-Check Mode.

Proof checking algorithm with presuppositions

Presuppositions also have to be checked in No-Check Mode.

Example 1

Assume that B contains $\sqrt{y}$.

Proof checking algorithm with presuppositions

Presuppositions also have to be checked in No-Check Mode.

Example 1

Assume that B contains $\sqrt{y}$.

$\Gamma \vdash^? y \geq 0$

Proof checking algorithm with presuppositions

Presuppositions also have to be checked in No-Check Mode.

Example 1

Assume that B contains $\sqrt{y}$.

$\Gamma \vdash^{?} y \geq 0$

Example 2

B does not contain $\sqrt{y}$.

Proof checking algorithm with presuppositions

Presuppositions also have to be checked in No-Check Mode.

Example 1

Assume that B contains $\sqrt{y}$.

$\Gamma \vdash^{?} y \geq 0$

Example 2

B does not contain $\sqrt{y}$.

$\Gamma \vdash^{?} y \geq 0$
$\Gamma, y \geq 0 \vdash^{?} \neg\sqrt{y} \in B$

Interpretation of plural expressions

- **Distributive** and **collective** interpretation

Examples

2 and 3 are prime numbers.

12 and 25 are coprime.

Interpretation of plural expressions

- **Distributive** and **collective** interpretation

Examples

2 and 3 are prime numbers.

12 and 25 are coprime.

x and y are distinct integers such that some odd prime number divides $x + y$.

Interpretation of plural expressions

- **Distributive** and **collective** interpretation

Examples

2 and 3 are prime numbers.

12 and 25 are coprime.

x and y are distinct integers such that some odd prime number divides $x + y$.

x and y are prime numbers p such that some odd prime number q divides $p + 1$.

Interpretation of plural expressions

- **Distributive** and **collective** interpretation

Examples

2 and 3 are prime numbers.

12 and 25 are coprime.

x and y are distinct integers such that some odd prime number divides $x + y$.

x and y are prime numbers p such that some odd prime number q divides $p + 1$.

- The **plural interpretation algorithm** of the Naproche system interprets all of these sentences correctly.

Symbolic mathematics

- The symbolic part of mathematical language has a rich variety of syntactic forms.

Symbolic mathematics

- The symbolic part of mathematical language has a rich variety of syntactic forms.
- Syntactic forms of function application:

Symbolic mathematics

- The symbolic part of mathematical language has a rich variety of syntactic forms.
- Syntactic forms of function application:

$f(x)$	$m+n$	mn	$n!$
$\sin x$	$[K : k]$	$\frac{m}{n}$	$a *_G b$

Symbolic mathematics

- The symbolic part of mathematical language has a rich variety of syntactic forms.
- Syntactic forms of function application:

$f(x)$	$m+n$	mn	$n!$
$\sin x$	$[K:k]$	$\frac{m}{n}$	$a *_G b$

- All of these are supported by Naproche.

Symbolic mathematics

- The symbolic part of mathematical language has a rich variety of syntactic forms.
- Syntactic forms of function application:

$f(x)$	$m+n$	mn	$n!$
$\sin x$	$[K:k]$	$\frac{m}{n}$	$a *_G b$

- All of these are supported by Naproche.
- A definition of a function introduces the syntax for that function.

Symbolic mathematics

- The symbolic part of mathematical language has a rich variety of syntactic forms.
- Syntactic forms of function application:

$f(x)$	$m+n$	mn	$n!$
$\sin x$	$[K:k]$	$\frac{m}{n}$	$a *_G b$

- All of these are supported by Naproche.
- A definition of a function introduces the syntax for that function.
- Symbolic expressions are also a potential source of ambiguities:

Symbolic mathematics

- The symbolic part of mathematical language has a rich variety of syntactic forms.
- Syntactic forms of function application:

$f(x)$	$m+n$	mn	$n!$
$\sin x$	$[K:k]$	$\frac{m}{n}$	$a *_G b$

- All of these are supported by Naproche.
- A definition of a function introduces the syntax for that function.
- Symbolic expressions are also a potential source of ambiguities:
  - $a(b+c)$

Symbolic mathematics

- The symbolic part of mathematical language has a rich variety of syntactic forms.
- Syntactic forms of function application:

$f(x)$	$m+n$	mn	$n!$
$\sin x$	$[K:k]$	$\frac{m}{n}$	$a *_G b$

- All of these are supported by Naproche.
- A definition of a function introduces the syntax for that function.
- Symbolic expressions are also a potential source of ambiguities:
  - $a(b+c)$
- Naproche disambiguates symbolic expressions by a combination of a type system, presupposition checking and preference for more recently introduced notation.

Outline

Main text case

- The main test case was the book **Einführung in die Analysis** by Edmund Landau.

Main text case

- The main test case was the book **Einführung in die Analysis** by Edmund Landau.
- We translated the first chapter of this book to the Naproche CNL.

Main text case

- The main test case was the book **Einführung in die Analysis** by Edmund Landau.
- We translated the first chapter of this book to the Naproche CNL.
- We left the same proof gaps as in the original text.

Main text case

- The main test case was the book **Einführung in die Analysis** by Edmund Landau.
- We translated the first chapter of this book to the Naproche CNL.
- We left the same proof gaps as in the original text.
- Most proof steps could be automatically verified by ATPs.

Landau's axioms

Assume that there is a set of objects called natural numbers.
Small latin letters will stand throughout for natural numbers.
Axiom 1: 1 is a natural number.
Axiom 2: For every x, there is a natural number x'.
Axiom 3: For every x, $x' \neq 1$.
Axiom 4: If $x' = y'$, then $x = y$.
Axiom 5: Suppose $\mathfrak{M}$ is a set of natural numbers satisfying the following properties:
Property 1: 1 belongs to $\mathfrak{M}$.
Property 2: If x belongs to $\mathfrak{M}$, then x' belongs to $\mathfrak{M}$.
Then $\mathfrak{M}$ contains all natural numbers.

Existence of addition function

Theorem 4: There is precisely one function $x, y \mapsto x + y$ such that for all x, y, $x + y$ is a natural number and $x + 1 = x'$ and $x + y' = (x + y)'$.
Proof:
A) Fix x. Suppose that there are functions $y \mapsto a_y$ and $y \mapsto b_y$ such that $a_1 = x'$ and $b_1 = x'$ and for all y, $a_{y'} = (a_y)'$ and $b_{y'} = (b_y)'$. Let $\mathfrak{M}$ be the set of y such that $a_y = b_y$. $a_1 = x' = b_1$, so 1 belongs to $\mathfrak{M}$. If y belongs to $\mathfrak{M}$, then $a_y = b_y$, i.e. by axiom 2 $(a_y)' = (b_y)'$, i.e. $a_{y'} = (a_y)' = (b_y)' = b_{y'}$, i.e. y' belongs to $\mathfrak{M}$. So $\mathfrak{M}$ contains all natural numbers. Thus for all y, $a_y = b_y$. Thus there is at most one function $y \mapsto x + y$ such that $x + 1 = x'$ and for all y, $x + y' = (x + y)'$.
B) Now let $\mathfrak{M}$ be the set of x such that there is a function $y \mapsto x + y$ such that for all y, $x + y$ is a natural number and $x + 1 = x'$ and $x + y' = (x + y)'$. Suppose $x = 1$. Define $x + y$ to be y'. Then $x + 1 = 1' = x'$, and for all y,$x + y' = (y')' = (x + y)'$. Thus 1 belongs to $\mathfrak{M}$. Let x belong to $\mathfrak{M}$. Then there is a function $y \mapsto x + y$ such that for all y, $x + y$ is a natural number and $x + 1 = x'$ and $x + y' = (x + y)'$. For defining $+$ at x', define $x' + y$ to be $(x + y)'$. Then $x' + 1 = (x + 1)' = (x')'$ and for all y, $x' + y' = (x + y')' = ((x + y)')' = (x' + y)'$. So x' belongs to $\mathfrak{M}$.
Thus $\mathfrak{M}$ contains all x. So for every x, there is a function $y \mapsto x + y$ such that for all y, $x + y$ is a natural number and $x + 1 = x'$ and $x + y' = (x + y)'$. Qed.

Outline

Conclusion

- We have developed a controlled natural language for mathematical texts.

Conclusion

- We have developed a controlled natural language for mathematical texts.
- The Naproche system can check the correctness of texts written in this language.

Conclusion

- We have developed a controlled natural language for mathematical texts.
- The Naproche system can check the correctness of texts written in this language.
- Bridge between formal proofs and informal proofs

Conclusion

- We have developed a controlled natural language for mathematical texts.
- The Naproche system can check the correctness of texts written in this language.
- Bridge between formal proofs and informal proofs
- Interesting theoretical work linking mathematical logic and formal linguistics