Natural Language Proving with Naproche

by Peter Koepke

University of Bonn, Germany

EuroProofNet Summer School on *AI for Reasoning and Processing of Mathematics*

Kutaisi International University

25 June 2024

Preview

Editing, parsing, processing and proof checking of a mathematical text like the following in the $\mathbb{N}$aproche system:

Definition. $\mathbb{P}$ *is the class of prime natural numbers.*

Theorem. (Euclid) $\mathbb{P}$ *is infinite.*

Proof. Assume that r is a natural number and p is a sequence of length r and $\{p_1, \ldots, p_r\}$ is a subclass of $\mathbb{P}$.
(1) p_i is a nonzero natural number for every $i \in \mathrm{dom}(p)$.
Consider $n = p_1 \cdots p_r + 1$. $p_1 \cdots p_r$ is nonzero. Hence n is nontrivial.
Take a prime divisor q of n.
Let us show that $q \neq p_i$ for all natural numbers i such that $1 \leq i \leq r$.
Proof by contradiction. Assume the contrary. Take a natural number i such that $1 \leq i \leq r$ and $q = p_i$. q is a divisor of n and q is a divisor of $p_1 \cdots p_r$ (by Factorproperty,1). Thus q divides 1. Contradiction. qed.
Hence $\{p_1, \ldots, p_r\}$ is not the class of prime natural numbers. $\square$

Preview

- Mathematical language(s)

- (Many) informal and formal languages

- First-order logic and set theory

- ForTheL as a controlled language for mathematics

- $\mathbb{N}$aproche system for proof-checking ForTheL texts

Natural mathematical statements

- Euclid: "Prime numbers are more than any assigned multitude of prime numbers."

- Standard form today: "There are infinitely many prime numbers."

- Capturing the quantification "infinitely many" in set theory:

Definition. $\mathbb{P}$ *is the class of prime natural numbers.*

Theorem. (Euclid) $\mathbb{P}$ *is infinite.*

Mathematical symbolism and LaTeX

Definition. $\mathbb{P}$ *is the class of prime natural numbers.*

Theorem. (Euclid) $\mathbb{P}$ *is infinite.*

- Defined words and phrases like "prime number", with meanings that may differ from their natural language interpretation

- Specific (defined) symbols like $\mathbb{P}, \frac{x}{y}, \int, \ldots$

- Mathematical typesetting using LaTeX:

```
\begin{definition} $\Primes$ is the class of prime natural numbers.
\end{definition}

\begin{theorem}[Euclid]
$\Primes$ is infinite.
\end{theorem}
```

First-order formalism

Definition. $\mathbb{P}$ *is the class of prime natural numbers.*

Theorem. (Euclid) $\mathbb{P}$ *is infinite.*

- $\forall x\,(x \in \mathbb{P} \leftrightarrow \mathrm{prim}(x))$

- $\inf(\mathbb{P})$

First-order logic (FOL)

- A *language* (in the sense of FOL) is a set S of function symbols and relation symbols

- The set T^S of S-terms is the smallest set such that
  - $x \in T^S$ for all variables $x = v_0, v_1, \ldots$
  - $ft_0 \ldots t_{n-1} \in T^S$ for all $n \in \mathbb{N}$, all n-ary function symbols $f \in S$, and all $t_0, \ldots, t_{n-1} \in T^S$

- The set L^S of S-formulas is the smallest set such that
  - $t_0 \equiv t_1 \in L^S$ for all S-terms $t_0, t_1 \in T^S$
  - $Rt_0 \ldots t_{n-1} \in L^S$ for all $n \in \mathbb{N}$, all n-ary relation symbols $R \in S$, and all $t_0, \ldots, t_{n-1} \in T^S$
  - $\neg\varphi \in L^S$ for all $\varphi \in L^S$
  - $(\varphi \rightarrow \psi) \in L^S$ for all $\varphi, \psi \in L^S$
  - $\forall x \varphi \in L^S$ for all $\varphi \in L^S$ and all variables x

- Further notations like $(\varphi \wedge \psi)$ or $(\varphi \vee \psi)$ can be introduced as abbreviations

First-order logic

- $\forall x \forall \varepsilon > 0 \exists \delta > 0 \forall x' \left(|x' - x| < \delta \rightarrow |f(x') - f(x)| < \varepsilon \right)$

- Mathematics can be carried out in set theory which can be axiomatized in first-order logic

- First-order Logic satisfies the Gödel completeness theorem (theoretical basis for formal mathematics)

- First-order Logic is the strongest logic which satisfies the compactness theorem and the Löwenheim-Skolem theorem (Lindström)

- Herbrand's theorem leads to practically efficient formal proving: *automated* theorem proving (ATP) and *interactive* theorem proving (ITP)

Defining FOL in Backus-Naur form

- *var* $\longrightarrow$ "v_0" | "v_1" | $\ldots$

- *term* $\longrightarrow$ *var* | "f" {*term*}

- *formula* $\longrightarrow$ *term* "$=$" *term* | "R" {*term*} | "$\neg$" *formula*
  | "(" *formula* "$\longrightarrow$" *formula* ")" | "$\forall$" *var* *formula*

FOL within $\mathbb{N}$aproche 2023 (Haskell)

- *term* $\longrightarrow$ *var* | "f" {*term*}

- *formula* $\longrightarrow$ *term* "$=$" *term* | "R" {*term*} | "$\neg$" *formula* | "(" *formula* "$\longrightarrow$" *formula* ")" |
"$\forall$" *var formula*

corresponds to the following Haskell data type:

```
data Formula =
  All Decl Formula          | Exi Decl Formula |
  Iff Formula Formula       | Imp Formula Formula      |
  Or  Formula Formula       | And Formula Formula      |
  Tag Tag Formula           | Not Formula              |
  Top                       | Bot                      |
  Trm { trmName :: TermName, trmArgs :: [Formula]}          |
  Var { varName :: VariableName, . . . , varPosition ::
Position.T }
```

Symbolic FO statements in ForTheL

in Andrei Paskevich: *The syntax and semantics of the ForTheL language*, `http://nevidal.org/download/forthel.pdf`

symbStatement → `forall` *classRelation symbStatement*
| `exists` *classRelation symbStatement*
| *symbStatement* <=> *symbStatement*
| *symbStatement* => *symbStatement*
| *symbStatement* \/ *symbStatement*
| *symbStatement* /\ *symbStatement*
| `not` *symbStatement*
| (*statement*)
| *primRelation*

Parsing symbolic formulas in $\mathbb{N}$aproche 2023

A symbolic formula like

$$\neg\varphi \longrightarrow \psi$$

is parsed, from left to right, by the parser `symbolicFormula` which is defined as a deeply nested combination of other parsers:

```
symbolicFormula :: FTL Formula
symbolicFormula  = biimplication
  where
    biimplication = implication >>= binary Iff (symbolicIff >> implication)
    implication   = disjunction >>= binary Imp (symbolicImp >> implication)
    ...
    nonbinary     = universal -|- existential -|- negation -|- separated -|- atomic
    negation      = Not <$> (symbolicNot >> nonbinary)
    ...
    binary op p f = optLL1 f $ fmap (op f) p

    symbolicIff = symbol "<=>" <|> token "\\iff"
    symbolicImp = symbol "=>" <|> token "\\implies"
    ...
    atomic = relation -|- parenthesised (optInText statement)
    ...
```

English as a formal language

- proposed by Richard Montague 1970

- formal grammars and parsers for (a subset) of English

- influential and much disputed in linguistics and philosophy

- controlled natural language (CNL), defined by grammatical rules for natural english phrases

Mathematical English as a formal language

- mathematical texts are distinctively more formal than ordinary texts

- a mathematical CNL may be acceptable to mathematicians

- the controlled natural language ForTheL by A. Paskevich and others and developed further by the $\mathbb{N}$aproche project tries to approximate mathematical English

A Backus-Naur grammar for ForTheL

The statement "$\mathbb{P}$ is infinite" can be parsed by a the following rules

simpleStatement $\longrightarrow$ *terms doesPredicate* { and *doesPredicate* }
terms $\longrightarrow$ *term* { (, | and) *term* }
term $\longrightarrow$ [( ] *quantifiedNotion* [ )] | *definiteTerm*

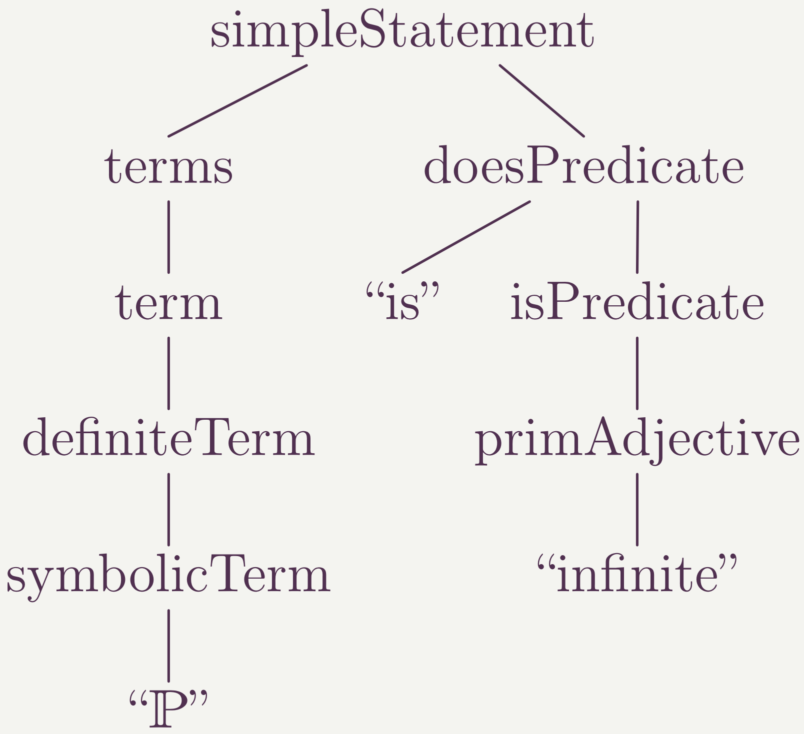
definiteTerm $\longrightarrow$ [( ] [the] *primDefiniteNoun* [ )] | *symbTerm* ...

doesPredicate $\longrightarrow$ [does | do] [not] *primVerb*
| [does | do] [not] [pairwise] *primVerbM*
| (has | have) *hasPredicate*
| (is | are | be) *isPredicate* { and *isPredicate* }
| (is | are | be) *is_aPredicate* { and *is_aPredicate* }

isPredicate $\longrightarrow$ [not] *primAdjective* | ...

primAdjective $\longrightarrow$ infinite

A parse tree for “$\mathbb{P}$ is infinite”



First-order translations of "$\mathbb{P}$ is infinite" by $\mathbb{N}$aproche

Pretty-printed first-order translation

```
isInfinite(\Primes)
```

Translation into TPTP format for further processing by ATPs, in particular by E:

```
fof(m__,conjecture, . . . => isInfinite(sbszPzrzizmzezs))).
```

DEMO:
Andrei Paskevich's The syntax and semantics of the ForTheL language

http://nevidal.org/download/forthel.pdf

DEMO:
Installing Isabelle/Naproche

`https://isabelle.in.tum.de/`

DEMO:
The $\mathbb{N}$aproche tutorial

via Isabelle Documentation (left sidebar) and `$ISABELLE_NAPROCHE/Intro.thy`

Takeaways

- It is in principle possible to combine natural language processing (NLP) with strong automated theorem proving (ATP) to achieve ITP of natural, readable texts

- $\mathbb{N}$aproche can handle some textbook-style texts abouts numbers, algebra, set theory, 12 of 100 "Wiedijk theorems", the definition of perfectoid rings, ...

- $\mathbb{N}$aproche shows that formalization languages in Formal Mathematics generally could be made more natural and readable

- On the other hand, Naproche formalizations are challenging, since they require attention to natural language criteria and LaTeX issues on top of standard ITP work

- To overcome long proof-checking times and slow interactivity, and other issues, Adrian De Lon is working on a new implementation of $\mathbb{N}$aproche, called Naproche-ZF (see his lecture on Thursday)

Links

Homepage: `https://naproche.github.io/index.html`

Download: `https://isabelle.in.tum.de/`

Naproche on the Web: `https://naproche.github.io/try/#/`

Documentation:

Andrei Paskevich: *The syntax and semantics of the ForTheL language*, `http://nevidal.org/download/forthel.pdf`

De Lon, A., Koepke, P., Lorenzen, A., Marti, A., Schütz, M., Wenzel, M. (2021). *The Isabelle/Naproche Natural Language Proof Assistant.* In: CADE 2021. Lecture Notes in Computer Science(), vol 12699. Springer

$\mathbb{N}$*aproche Tutorial*, within Isabelle at `$ISABELLE_NAPROCHE/examples/TUTORIAL.ftl.pdf`

Thank you!