Naturalness in Formal Mathematics

Peter Koepke, Mathematical Institute, University of Bonn, Germany

Mal'cev Meeting

Novosibirsk, 12 November 2013

A formal proof of the Feit-Thompson Theorem

G. Gonthier et al, September 2012, using the Coq proof assistant, proved

```
The Odd Order theorem

Theorem stripped_Odd_Order T mul one inv
(G : T → Type) (n : natural) :
group_axioms T mul one inv →
group T mul one inv G →
finite_of_order T G n →
odd n → solvable_group T mul one inv G.
```

A formal proof of the Feit-Thompson Theorem

```
Inductive solvable_group T mul one inv (G : T
→ Type) :=
| TrivialGroupSolvable
    (G_trivial : ∀ x, equivalent (G x) (equal
T x one))
| AbelianExtensionSolvable (H : T → Type)
  (H_solvable : solvable_group T mul one inv
H)
(G_on_H_abelian : abelian_factor T mul one inv
G H).
```

A formal proof of the Feit-Thompson Theorem

- H. Bender and G. Glauberman. *Local analysis of the Odd Order Theorem*

- T. Peterfalvi. *Character theory for the Odd Order Theorem*

- preliminaries from group theory, algebra, linear algebra, ...

- sets, finite sequences, ...

- logical preliminaries

The Gödel completeness theorem

Every logically true mathematical statement has a formal derivation.

Principia Mathematica

***54·43.** $\vdash :. \alpha, \beta \in 1 . \supset : \alpha \cap \beta = \Lambda . \equiv . \alpha \cup \beta \in 2$

Dem.

$\vdash . *54{\cdot}26 . \supset \vdash :. \alpha = \iota`x . \beta = \iota`y . \supset : \alpha \cup \beta \in 2 . \equiv . x \neq y .$

[*51·231] $\equiv . \iota`x \cap \iota`y = \Lambda .$

[*13·12] $\equiv . \alpha \cap \beta = \Lambda$ (1)

$\vdash . (1) . *11{\cdot}11{\cdot}35 . \supset$

$\vdash :. (\exists x, y) . \alpha = \iota`x . \beta = \iota`y . \supset : \alpha \cup \beta \in 2 . \equiv . \alpha \cap \beta = \Lambda$ (2)

$\vdash . (2) . *11{\cdot}54 . *52{\cdot}1 . \supset \vdash .$ Prop

From this proposition it will follow, when arithmetical addition has been defined, that $1 + 1 = 2$.

The Gödel completeness theorem

Every logically true mathematical statement has a formal derivation.

Every true mathematical statement has a formal derivation within some (foundational) axiom system.

The Gödel completeness theorem

Every logically true mathematical statement has a formal derivation.

Every true mathematical statement has a formal derivation within some (foundational) axiom system.

Every mathematical proof can be replaced by a formal derivation.

The Gödel completeness theorem

Every logically true mathematical statement has a formal derivation.

Every true mathematical statement has a formal derivation within some (foundational) axiom system.

Every mathematical proof can be replaced by a formal derivation.

Mathematics can be in principle be carried out completely formal (*Formal mathematics*).

Proof styles and granularities

- Feit-Thompson, journal article: 250p.

- Bender-Glauberman, T. Peterfalvi, monographs: 360p.

- Coq formalization of these books: 47000 lines (of code)

- (formalization of mathematics in Coq so far: 170000 lines)

- Gödel completeness: complete derivation in some formal system: ???? lines

Reading / understanding / checking proofs

- Mathematical input language

- (Internal representation)

- Successively reducing statements in the proof to previous statements

- by insight and reasoning

- and/or by proof search in a formal system / automatic theorem proving / ATP

Formal proofs - derivations

N. Bourbaki:

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, [...]

Formal proofs - derivations

N. Bourbaki:

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, [...]

Computer-supported formal proofs

J. McCarthy, 1962:

Checking mathematical proofs is potentially one of the most interesting and useful applications of automatic computers. ... 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.

Computer-supported formal proofs

J. McCarthy, 1962:

Checking mathematical proofs is potentially one of the most interesting and useful applications of automatic computers. ... 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.

Formal mathematics / automatic proof checking

- Mathematical input language

- Internal representation

- Successively reducing statements in the proof to previous statements

- by insight and reasoning

- and/or by proof search in a formal system / automatic theorem proving / ATP

Examples of formal mathematics systems

- Automath, de Bruijn, ~1967
- Mizar, Trybulec, ~1973
- **Isabelle/Isar**, Paulson, Nipkow, Wenzel, ~2002
- **Coq**
- **HOL Light**, Harrison
- many other systems

Why does formal mathematics not catch on?

Freek Wiedijk, 2007:

> The other reason that there has not been much progress on the vision from the QED manifesto is that currently formalized mathematics does not resemble real mathematics at all. Formal proofs look like computer program source code.

Why does formal mathematics not catch on?

Freek Wiedijk, 2007:

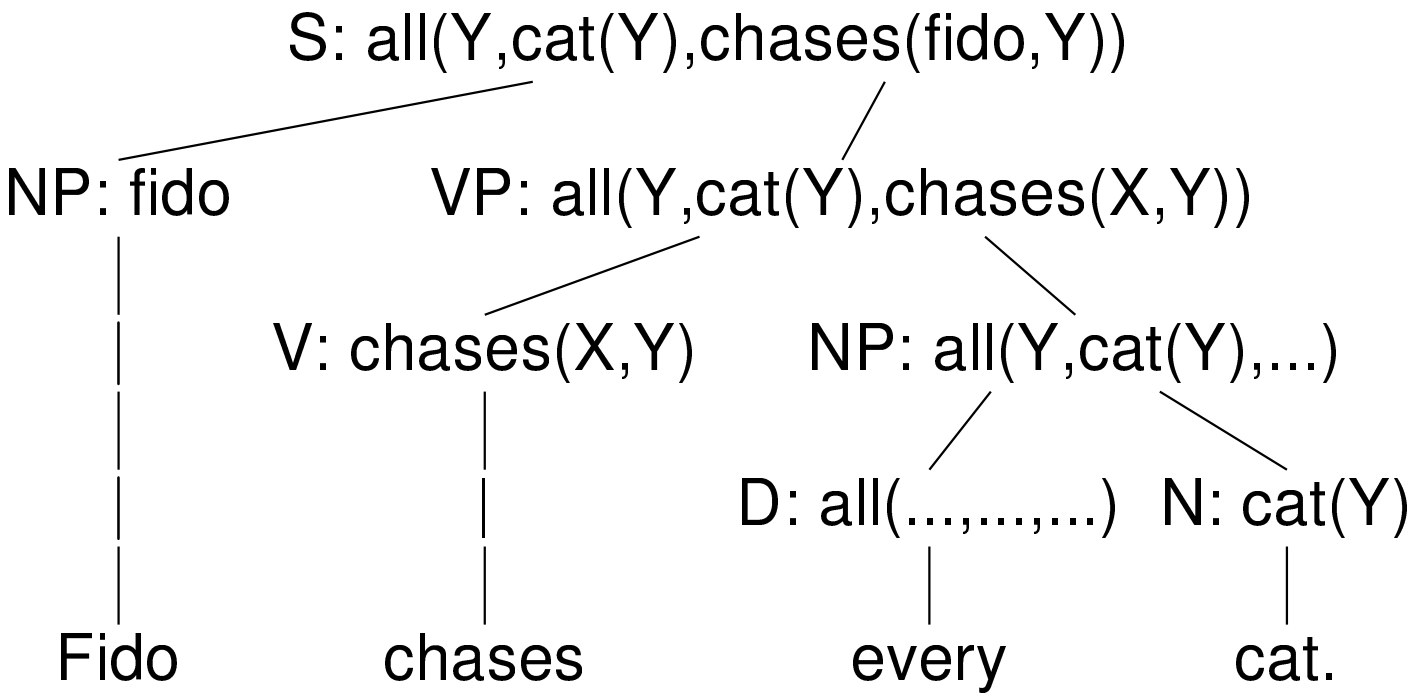
> The other reason that there has not been much progress on the vision from the QED manifesto is that currently formalized mathematics does not resemble real mathematics at all. Formal proofs look like computer program source code.

Mathematical statements

“1 divides every integer.” $\leftarrow\!\!\rightarrow$“Fido chases every cat.”

Linguistic analysis

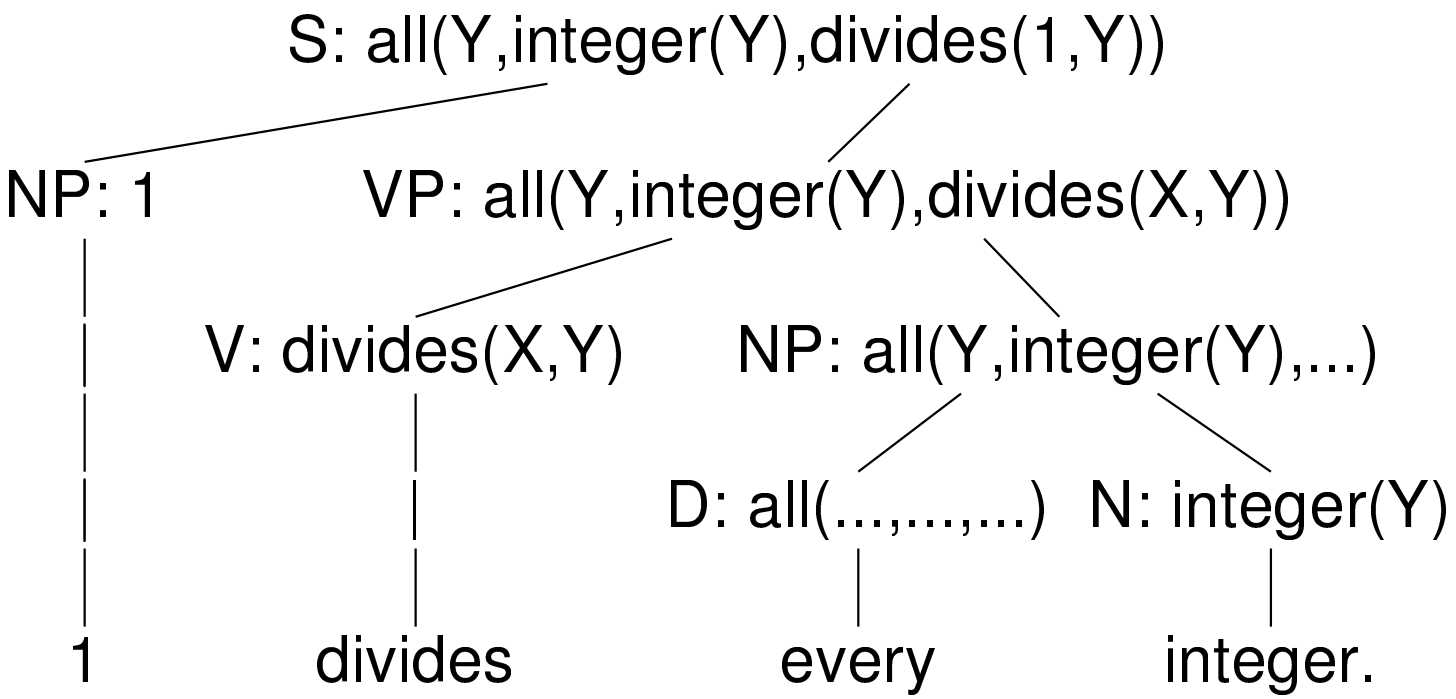
“Fido chases every cat.”



$\forall Y(\mathrm{cat}(Y) \rightarrow \mathrm{chases}(\mathrm{fido}, Y))$.

Linguistic analysis

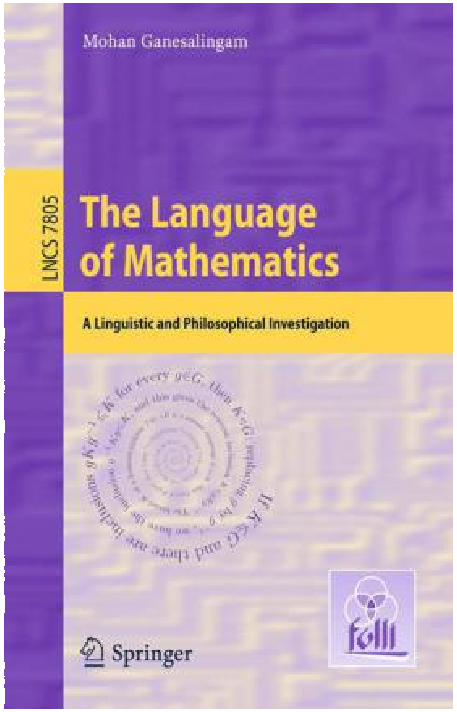
“1 divides every integer.”



$\forall Y\,(\mathrm{integer}(Y) \to 1|Y).$

The Language of Mathematics

- Mohan Ganesalingam: *The Language of Mathematics*,



The Naproche project: Natural language proof checking

- models natural language proofs using computer-supported methods of formal linguistics and formal logic
- joint work with Bernhard Schröder, University of Duisburg-Essen: `www.naproche.net`
- development of a mathematical authoring system with a LaTeX-quality graphical interface
- employing strong ATPs to "bridge logical gaps"

PhD thesis of Marcos Cramer:

E. Landau, *Grundlagen der Analysis*, 1930

Theorem 30: For all $x, y, z, x*(y+z) = (x*y)+(x*z)$.

Proof: Fix x, y. $x*(y+1) = x*y' = x*y + x = (x*y)+(x*1)$.

Now suppose $x*(y+z) = (x*y)+(x*z)$. Then $x*(y+z') = x*((y+z)') = (x*(y+z))+x = ((x*y)+(x*z))+x = (x*y)+((x*z)+x) = (x*y)+(x*z')$.

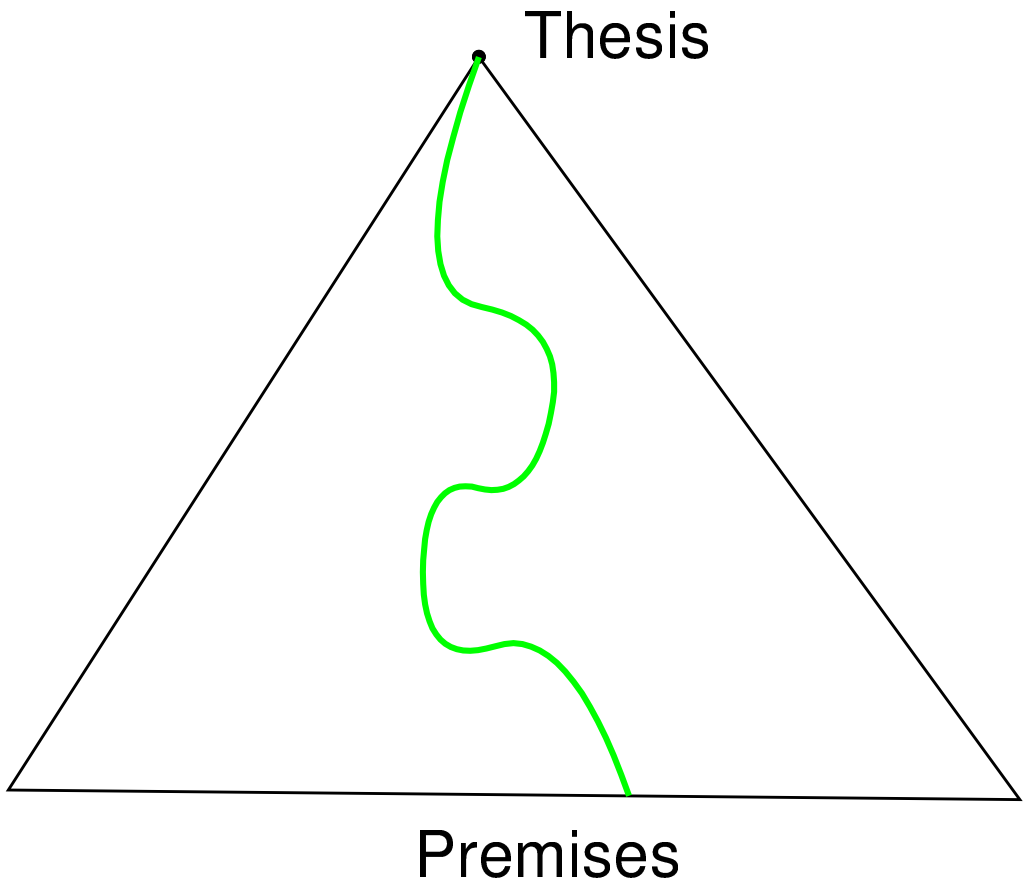
Thus by induction, for all z, $x*(y+z) = (x*y)+(x*z)$. Qed.

Sophisticated linguistic issues:

N. Bourbaki, Algebra:

> In any monoid E the set of invertible elements with the structure induced by that on E is a group. In particular, the set of bijective mappings of a set F onto itself (or set of *permutations* of F) is a group under the law $(f, g) \mapsto f \circ g$, called the *symmetric group of the set* F and denoted by $\mathfrak{S}_F$.

Complexity of Derivation Search in Large Search Space



Derivation Indication Reduces Search Space

- indicating proof steps by explicitly giving intermediate statements (assumptions, claims)

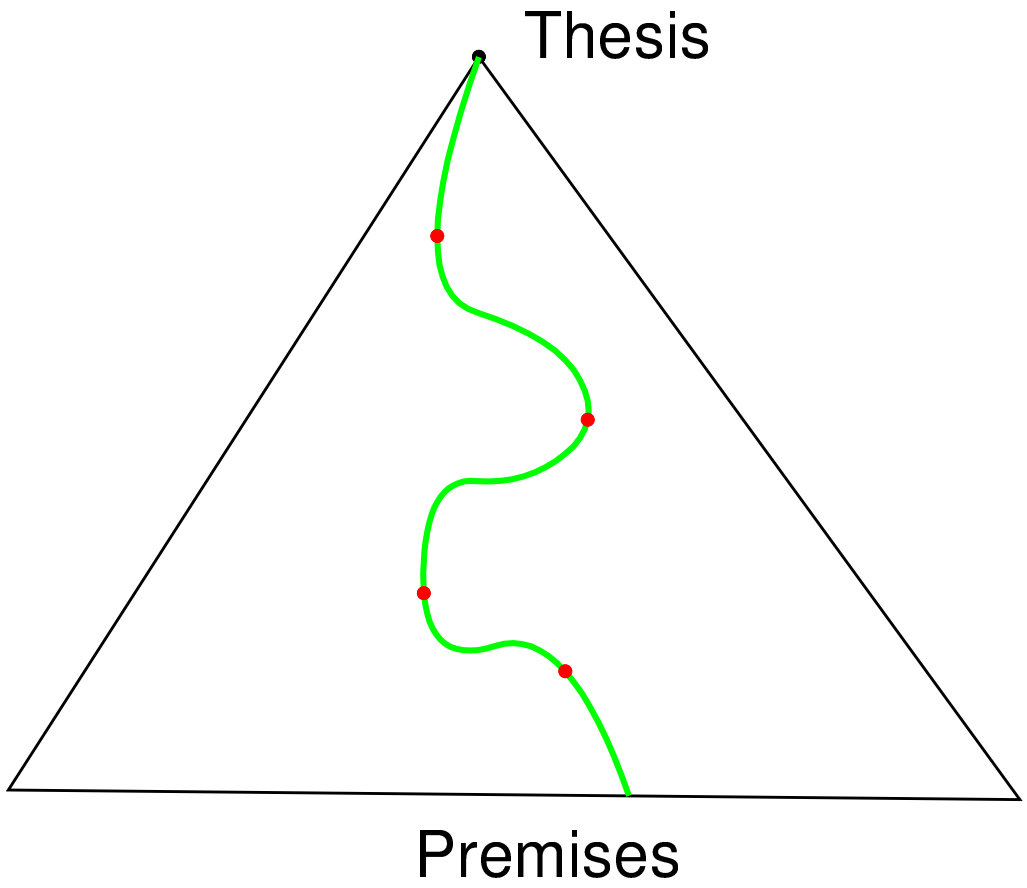
- substitution instances, case distinctions, ...

- references: by statement/lemma/theorem ..., by fact from the literature

- proof/derivation methods: proof by contradiction, contraposition, algebraic manipulation, induction, area-specific methods

- proof similar to some other proof: ‘‘by symmetry‘‘; ‘‘apply the argument ... to the situation ...‘‘

Complexity of Derivation Search



Derivation Indication Reduces Search Space

- indicating proof steps by explicitly giving intermediate statements (assumptions, claims)

- substitution instances, case distinctions, ...

- references: by statement/lemma/theorem ..., by fact from the literature

- proof/derivation methods: proof by contradiction, contraposition, algebraic manipulation, induction, area-specific methods

- proof similar to some other proof: ‘‘by symmetry‘‘; ‘‘apply the argument ... to the situation ...‘‘

Derivation Indication Reduces Search Space

- indicating proof steps by explicitly giving intermediate statements (assumptions, claims)

- substitution instances, case distinctions, ...

- references: by statement/lemma/theorem ..., by fact from the literature

- proof/derivation methods: proof by contradiction, contraposition, algebraic manipulation, induction, area-specific methods

- proof similar to some other proof: ‘‘by symmetry‘‘; ‘‘apply the argument ... to the situation ...‘‘

Derivation Indication Reduces Search Space

- indicating proof steps by explicitly giving intermediate statements (assumptions, claims)

- substitution instances, case distinctions, ...

- references: by statement/lemma/theorem ..., by fact from the literature

- proof/derivation methods: proof by contradiction, contraposition, algebraic manipulation, induction, area-specific methods

- proof similar to some other proof: “by symmetry“; “apply the argument ... to the situation ...“

Derivation Indication Reduces Search Space

- indicating proof steps by explicitly giving intermediate statements (assumptions, claims)

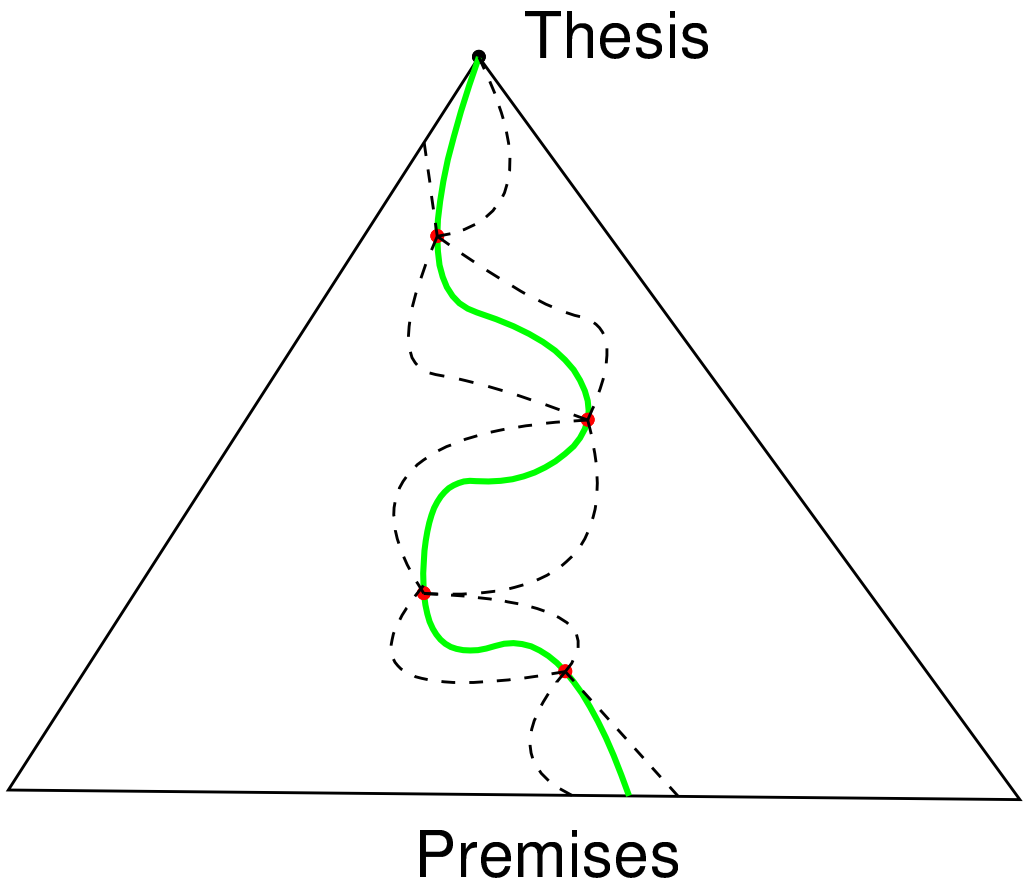
- substitution instances, case distinctions, ...

- references: by statement/lemma/theorem ..., by fact from the literature

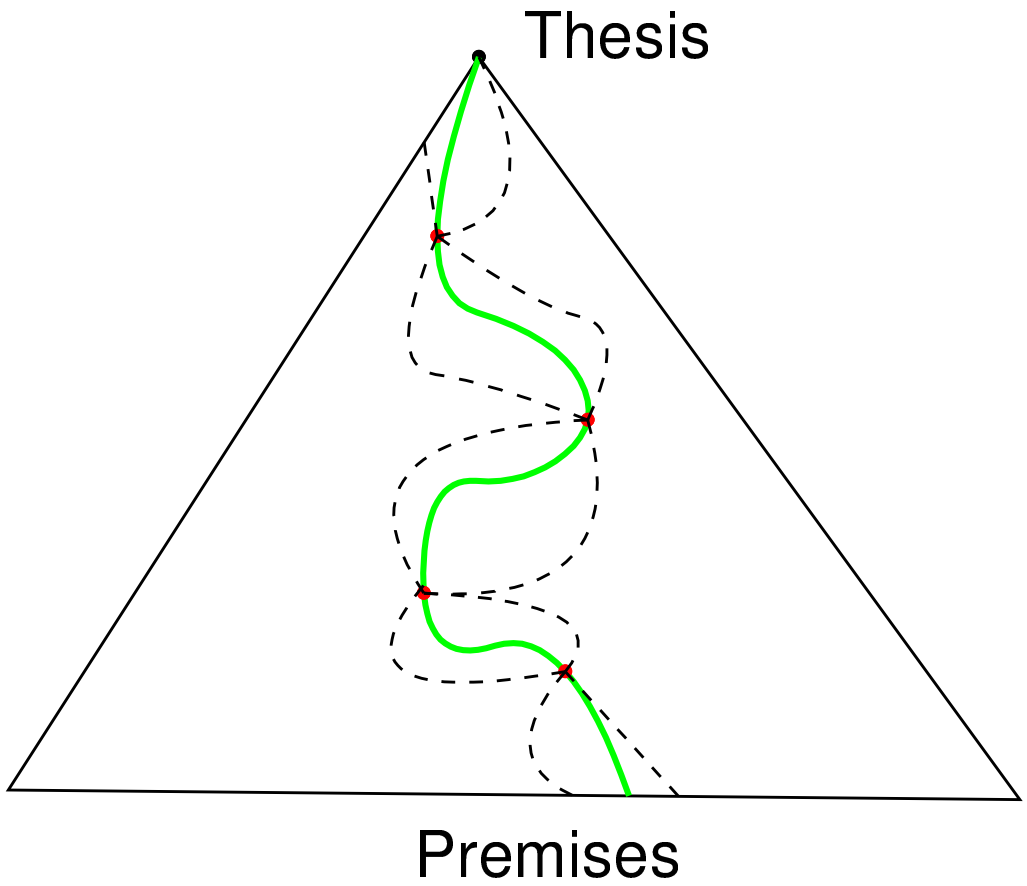
- proof/derivation methods: proof by contradiction, contraposition, algebraic manipulation, induction, area-specific methods

- proof similar to some other proof: ‘‘by symmetry‘‘; ‘‘apply the argument ... to the situation ...‘‘

Complexity of Derivation Search



Derivation Search with ATPs like in Mizar, Naproche, ...



Reasoning

- Mathematicians attempt to use ‘‘natural’’ or ‘‘canonical’’ proof steps

- familiar from undergraduate mathematics: unfolding definitions, dealing with universal quantifiers by considering new variables, ...

- observing mathematical conventions

Reasoning

- Mathematicians attempt to use ‘‘natural’’ or ‘‘canonical’’ proof steps

- familiar from undergraduate mathematics: unfolding definitions, dealing with universal quantifiers by considering new variables, ...

- observing mathematical conventions

Reasoning

- Mathematicians attempt to use ‘‘natural’’ or ‘‘canonical’’ proof steps

- familiar from undergraduate mathematics: unfolding definitions, dealing with universal quantifiers by considering new variables, ...

- observing mathematical conventions

The Ganesalingam-Gowers Project

- http://gowers.wordpress.com/2013/04/14/answers-results-of-polls-and-a-brief-description-of-the-program/

- Mohan Ganesalingam and Timothy Gowers

- automatically generating canonical natural language proofs

- tested for ‘‘naturality‘‘

The Ganesalingam-Gowers Project

- http://gowers.wordpress.com/2013/04/14/answers-results-of-polls-and-a-brief-description-of-the-program/

- Mohan Ganesalingam and Timothy Gowers

- automatically generating canonical natural language proofs

- tested for “naturality“

The Ganesalingam-Gowers Project

- http://gowers.wordpress.com/2013/04/14/answers-results-of-polls-and-a-brief-description-of-the-program/

- Mohan Ganesalingam and Timothy Gowers

- automatically generating canonical natural language proofs

- tested for “naturality“

The Ganesalingam-Gowers Project

- http://gowers.wordpress.com/2013/04/14/answers-results-of-polls-and-a-brief-description-of-the-program/

- Mohan Ganesalingam and Timothy Gowers

- automatically generating canonical natural language proofs

- tested for ‘‘naturality‘‘

The Ganesalingam-Gowers Project

Problem 2. Let X and Y be metric spaces, let $f\colon X \to Y$ be continuous, and let U be an open subset of Y. Then $f^{-1}(U)$ is an open subset of X.

Solution 2(a) Let x be an element of $f^{-1}(U)$. Then $f(x) \in U$. Therefore, since U is open, there exists $\eta > 0$ such that $u \in U$ whenever $d(f(x), u) < \eta$. We would like to find $\delta > 0$ s.t. $y \in f^{-1}(U)$ whenever $d(x, y) < \delta$. But $y \in f^{-1}(U)$ if and only if $f(y) \in U$. We know that $f(y) \in U$ whenever $d(f(x), f(y)) < \eta$. Since f is continuous, there exists $\theta > 0$ such that $d(f(x), f(y)) < \eta$ whenever $d(x, y) < \theta$. Therefore, setting $\delta = \theta$, we are done.

The Ganesalingam-Gowers Project: Reasoning

- "Our main priority when writing the program was that the steps it took should be ones that a human would naturally take. ... the program should not do silly things."

- work on the natural mathematical statements

- rewriting definitions

- "peeling off" existential quantifiers

- deleting "used up" hypothesis

- ...

The Ganesalingam-Gowers Project: Reasoning

- "Our main priority when writing the program was that the steps it took should be ones that a human would naturally take. ... the program should not do silly things."

- work on the natural mathematical statements

- rewriting definitions

- "peeling off" existential quantifiers

- deleting "used up" hypothesis

- ...

The Ganesalingam-Gowers Project: Reasoning

- "Our main priority when writing the program was that the steps it took should be ones that a human would naturally take. ... the program should not do silly things."

- work on the natural mathematical statements

- rewriting definitions

- "peeling off" existential quantifiers

- deleting "used up" hypothesis

- ...

The Ganesalingam-Gowers Project: Reasoning

- ‘‘Our main priority when writing the program was that the steps it took should be ones that a human would naturally take. ... the program should not do silly things.‘‘

- work on the natural mathematical statements

- rewriting definitions

- ‘‘peeling off‘‘ existential quantifiers

- deleting ‘‘used up‘‘ hypothesis

- ...

The Ganesalingam-Gowers Project: Reasoning

- ‘‘Our main priority when writing the program was that the steps it took should be ones that a human would naturally take. ... the program should not do silly things.‘‘

- work on the natural mathematical statements

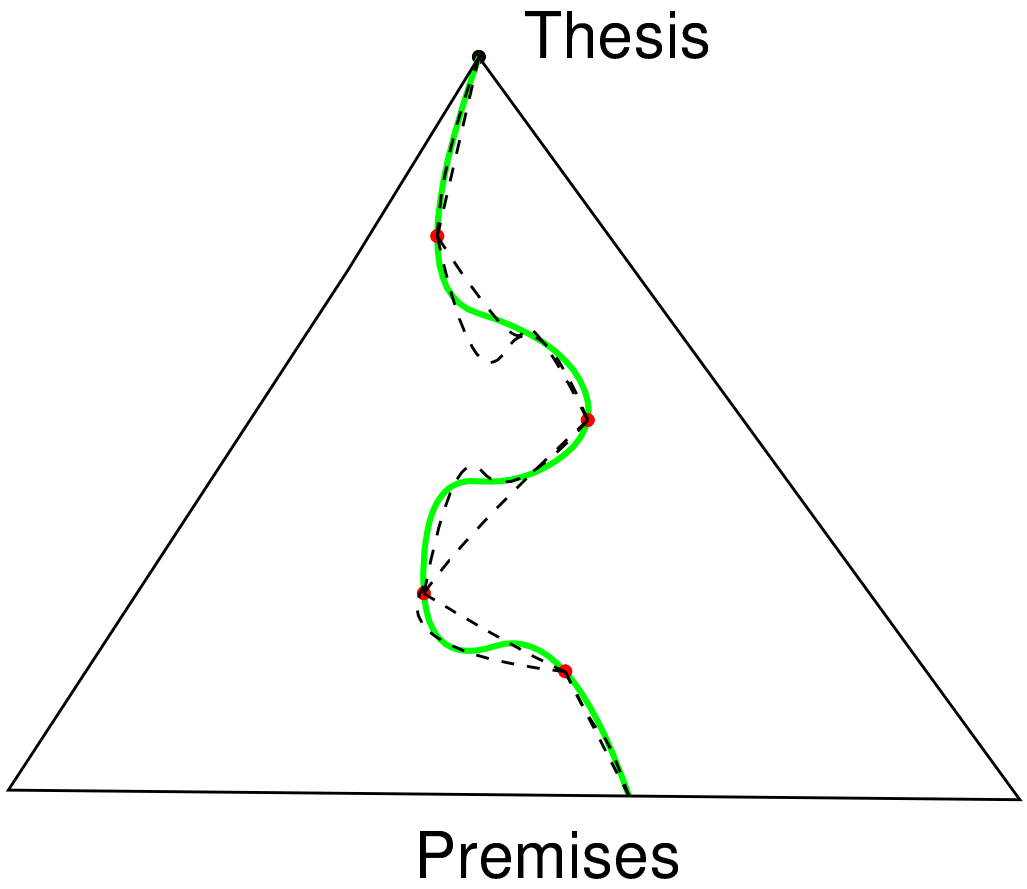
- rewriting definitions

- ‘‘peeling off‘‘ existential quantifiers

- deleting ‘‘used up‘‘ hypothesis

- ...

Can One Radically Prune the Local Searches by Reasoning?



Andrei Paskevich' System of Automatic Deduction (SAD)

- started by Victor Glushkov as *Evidence Algorithm*, continued with Alexander Lyaletski and Konstantin Verchinine

- simple phrase structure grammar

- ForTheL: Formula Theory Language, a combination of controlled English with mathematical notation

- has reasoner and ATP

- http://nevidal.org/sad.en.html

Andrei Paskevich' System of Automatic Deduction (SAD)

- started by Victor Glushkov as *Evidence Algorithm*, continued with Alexander Lyaletski and Konstantin Verchinine

- simple phrase structure grammar

- ForTheL: Formula Theory Language, a combination of controlled English with mathematical notation

- has reasoner and ATP

- http://nevidal.org/sad.en.html

Andrei Paskevich' System of Automatic Deduction (SAD)

- started by Victor Glushkov as *Evidence Algorithm*, continued with Alexander Lyaletski and Konstantin Verchinine

- simple phrase structure grammar

- ForTheL: Formula Theory Language, a combination of controlled English with mathematical notation

- has reasoner and ATP

- http://nevidal.org/sad.en.html

Andrei Paskevich' System of Automatic Deduction (SAD)

- started by Victor Glushkov as *Evidence Algorithm*, continued with Alexander Lyaletski and Konstantin Verchinine

- simple phrase structure grammar

- ForTheL: Formula Theory Language, a combination of controlled English with mathematical notation

- has reasoner and ATP

- http://nevidal.org/sad.en.html

Andrei Paskevich' System of Automatic Deduction (SAD)

- started by Victor Glushkov as *Evidence Algorithm*, continued with Alexander Lyaletski and Konstantin Verchinine

- simple phrase structure grammar

- ForTheL: Formula Theory Language, a combination of controlled English with mathematical notation

- has reasoner and ATP

- http://nevidal.org/sad.en.html

Original SAD input

Theorem Main.

For all nonzero natural numbers n,m,p if p * (m * m) = (n * n) then p is compound.

Proof by induction. Let n,m,p be nonzero natural numbers. Assume that p * (m * m) = (n * n). Assume that p is prime. Hence p divides n * n and p divides n. Take q = n / p.
Then m * m = p * (q * q). Indeed p * (m * m) = p * (p * (q * q)).
m < n. Indeed n <= m => n * n <= m * m.

Hence p is compound.

qed.

Combining SAD with natural language

Theorem 1. (Fuerstenberg) *Let* $S = \{\text{Arseq}(0,\, r) \mid r \textit{ is prime}\}$. *Then* S *is infinite.*

Proof. We have $-\bigcup S = \{1, -1\}$, indeed n belongs to $\bigcup S$ iff n has a prime divisor.

Assume that S is finite. Then $\bigcup S$ is closed and $-\bigcup S$ is open. Take p such that $\text{ArSeq}(1, p) \subseteq -\bigcup S$.

Claim. $\text{ArSeq}(1, p)$ has an element that does not belong to $\{1, -1\}$.
Proof. $1 + p$ and $1 - p$ are elements of $\text{ArSeq}(1, p)$. $1 + p \neq 1 \wedge 1 - p \neq 1$. $1 + p \neq -1 \vee 1 - p \neq -1$. *qed*.

Contradiction. $\square$

Combining SAD with natural language and LaTeX-macros

Theorem 2. *The set of prime numbers is infinite.*

Proof. Let A be a finite set of prime numbers. Take a function p and a number r such that p lists A in r steps. ran $p \subseteq \mathbb{N}^+$. $\prod_{i=1}^{r} p_i \in \mathbb{N}^+$. Take $n = \prod_{i=1}^{r} p_i + 1$. n is nontrivial. Take a prime divisor q of n .

Let us show that q is not an element of A. Assume the contrary. Take j such that ($1 \leqslant j \leqslant r$ and $q = p_j$). p_j divides $\prod_{i=1}^{r} p_i$ (by MultProd). Then q divides 1 (by DivMin). Contradiction. qed.

Hence A is not the set of prime numbers. □

General issues

- There are natural(ly looking) proofs that are fully formal with respect to the Naproche system

- this defines a ‘‘fortified formalism’’, using linguistic methods and computer implementations, which allows some natural proofs which are fully formal

- can a ‘‘fortified formalism’’ help to mediate between the ‘‘two streams’’ in the philosophy of mathematics (formalistic / naturalistic)?

Outlook

- Combine best practices from linguistics and various formal mathematics systems to obtain naturalness and power
- will this achieve acceptance by mathematical practioneers?
- Jeremy Avigad: *On a personal note, I am entirely convinced that formal verification of mathematics will eventually become commonplace.*
- What would be the implications of a widespread use of formal mathematics for the methodology, practice, and philosophy of mathematics?

Thank you!