Natural Proof Checking: The Naproche Project

Peter Koepke, University of Bonn, Germany

Mathematical Institute

Logic Colloquium 2011

Barcelona, July 12, 2011

Mathematical logic models mathematics

	Mathematics	$\leftrightarrow$	Mathematical logic	
Language -	basic notions		symbols	- Syntax
	statements		formulas	
	proofs		formal derivations ($\vdash$)	
Ontology -	structures		sets, relations, etc	- Semantics
	truth		satisfaction relation	
	implication		logical implication ($\models$)	

Mathematical logic models mathematics

- excellent agreement between ontology and semantics: a *group* is a *set* such that ...
- complete agreement between syntax and semantics: Gödel's completeness theorem: $\vdash = \models$
- hence: every proof can be replaced by a formal derivation
- formal mathematics: to actually produce formal derivations from informal proofs

On the feasability (complexity) of formal mathematics

N. Bourbaki:

[...] 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, [...]

J. McCarthy:

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.

Informal proofs ...

- use informal mathematical language, combining natural language and formulas
- can be studied linguistically as texts in a specialised language
- are directed at human experts, omitting background theories and many details
- can be studied within argumentation theory

Bridging the gap

informal language

$\cup$

controlled natural language of mathematics

$\downarrow$ *computational linguistics*

formal language

$\downarrow$

TPTP first order language

Bridging the gap

informal argument

↓

sequence of TPTP statements

↓ *automatic theorem provers*
+ *background theory*

formal derivation for each statement?

↓

acceptance / nonacceptance

The Naproche project: Natural language proof checking

- studies the syntax and semantics of the language of mathematical proofs, emphasizing natural language and natural argumentation, relating them to formal mathematics
- models natural language proofs using computer-supported methods of formal linguistics (natural language processing, NLP) and formal logic (automatic theorem provers, ATPs)
- with Bernhard Schröder, Marcos Cramer, (Daniel Kuehlwein, Merlin Carl, Jip Veldman)

The Naproche system

- 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.

- Naproche language: controlled natural language for (parts of) mathematics

- background ontology (sets, functions, types, ...)

- bridging gaps in proofs by using ATPs

Linguistic analysis

- formal grammars, e.g., phrase structure grammar
- standard techniques of computational linguistics like tokenizing, parsing
- parsing mathematical notation like $\sum_{n=1}^{\infty} \frac{1}{n^2}$ in combination with natural language parsing
- discourse representations
- proof representation structures

Layers of the Naproche system

↓ Standard editor or web editor

TeX-style input text

↕ Natural language processing (NLP)

Proof representation structure (PRS)

↕ First-order translation

First-order logic format (TPTP)

↕ Proof checker or automatic theorem prover (ATP)

“Accepted”/“Not accepted”, with error messages

E. Landau, *Grundlagen der Analysis*, 1930

Theorem 30 (Distributive Law):

$$x(y+z) = xy + xz.$$

Preliminary Remark: The formula

$$(y+z)x = yx + zx$$

which results from Theorem 30 and Theorem 29, and similar analogues later on, need not be specifically formulated as theorems, nor even be set down.

Proof: Fix x and y, and let $\mathfrak{M}$ be the set of all z for which the assertion holds true.

I) $x(y+1) = xy' = xy + x = xy + x \cdot 1$;

1 belongs to $\mathfrak{M}$.

II) If z belongs to $\mathfrak{M}$, then

$$x(y+z) = xy + xz,$$

hence

$$x(y+z') = x((y+z)') = x(y+z) + x = (xy+xz) + x = xy + (xz+x) = xy + xz',$$

so that z' belongs to $\mathfrak{M}$.

Therefore, the assertion always holds.

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.

Chapter 1 from Landau in Naproche

by Merlin Carl, Marcos Cramer, Daniel Kuehlwein

Abstract

This is a reformulation of the first chapter of Landau's *Grundlagen der Analysis* in the Controlled Natural Language of Naproche. Talk about sets is still avoided. One consequence of this is that Axiom 5 (the induction axiom) cannot be formulated; instead we use an induction proof method.

Axiom 3: For every x, $x' \neq 1$.

Axiom 4: If $x' = y'$, then $x = y$.

Theorem 1: If $x \neq y$ then $x' \neq y'$.
Proof:
Assume that $x \neq y$ and $x' = y'$. Then by axiom 4, $x = y$. Qed.

Theorem 2: For all x $x' \neq x$.
Proof:
By axiom 3, $1' \neq 1$. Suppose $x' \neq x$. Then by theorem 1, $(x')' \neq x'$. Thus by induction, for all x $x' \neq x$. Qed.

Theorem 3: If $x \neq 1$ then there is a u such that $x = u'$.
Proof:
If $1 \neq 1$ then there is a u such that $1 = u'$.
Assume $x' \neq 1$. If $u = x$ then $x' = u'$. So there is a u such that $x' = u'$.
Thus by induction, if $x \neq 1$ then there is a u such that $x = u'$. Qed.

Definition 1:
Define $+$ recursively:
$x + 1 = x'$.
$x + y' = (x + y)'$.

Theorem 5: For all x, y, z, $(x + y) + z = x + (y + z)$.
Proof:
Fix x, y.
$(x + y) + 1 = (x + y)' = x + y' = x + (y + 1)$.
Assume that $(x + y) + z = x + (y + z)$. Then $(x + y) + z' = ((x + y) + z)' = (x + (y + z))' = x + (y + z)' = x + (y + z')$. So $(x + y) + z' = x + (y + z')$.
Thus by induction, for all z, $(x + y) + z = x + (y + z)$. Qed.

Lemma 4a: For all y, $1 + y = y'$.
Proof:
By definition 1, $1 + 1 = 1'$.
Suppose $1 + y = y'$. Then by definition 1, $1 + y' = (1 + y)'$. So $1 + y' = (y')'$.
Thus by induction, for all y $1 + y = y'$. Qed.

Current projects

- formalizing Landau
- major rewrite of the Naproche software for greater modularity and more linguistic variants, including a weak type formalism
- collaboration with A. Paskevich et. al. on System for Automated Deduction (SAD)

Possible applications

- Natural language interfaces to formal mathematics
- Mathematical authoring and checking tools
- Writing texts that are simultaneously acceptable by human readers and formal mathematics systems
- Tutorial applications: teaching how to prove

General issues

- Linguistics: construction and analysis of a mathematical language with a definite first order semantics
- Can the gap between natural proofs and formal derivations be narrowed?
- There are natural(ly looking) proofs that are fully formal with respect to the Naproche system
- Philosophy of mathematics: what is a mathematical proof? Naturalism versus formalism?

Thank You!

The Naproche system:

Naproche - Natural Language Proof Checking

univer

Bonn Mathematical Logic Group

Menu

home

members

---

web interface

examples

Burali-Forti paradox

Group Theory

Landau

tutorial

---

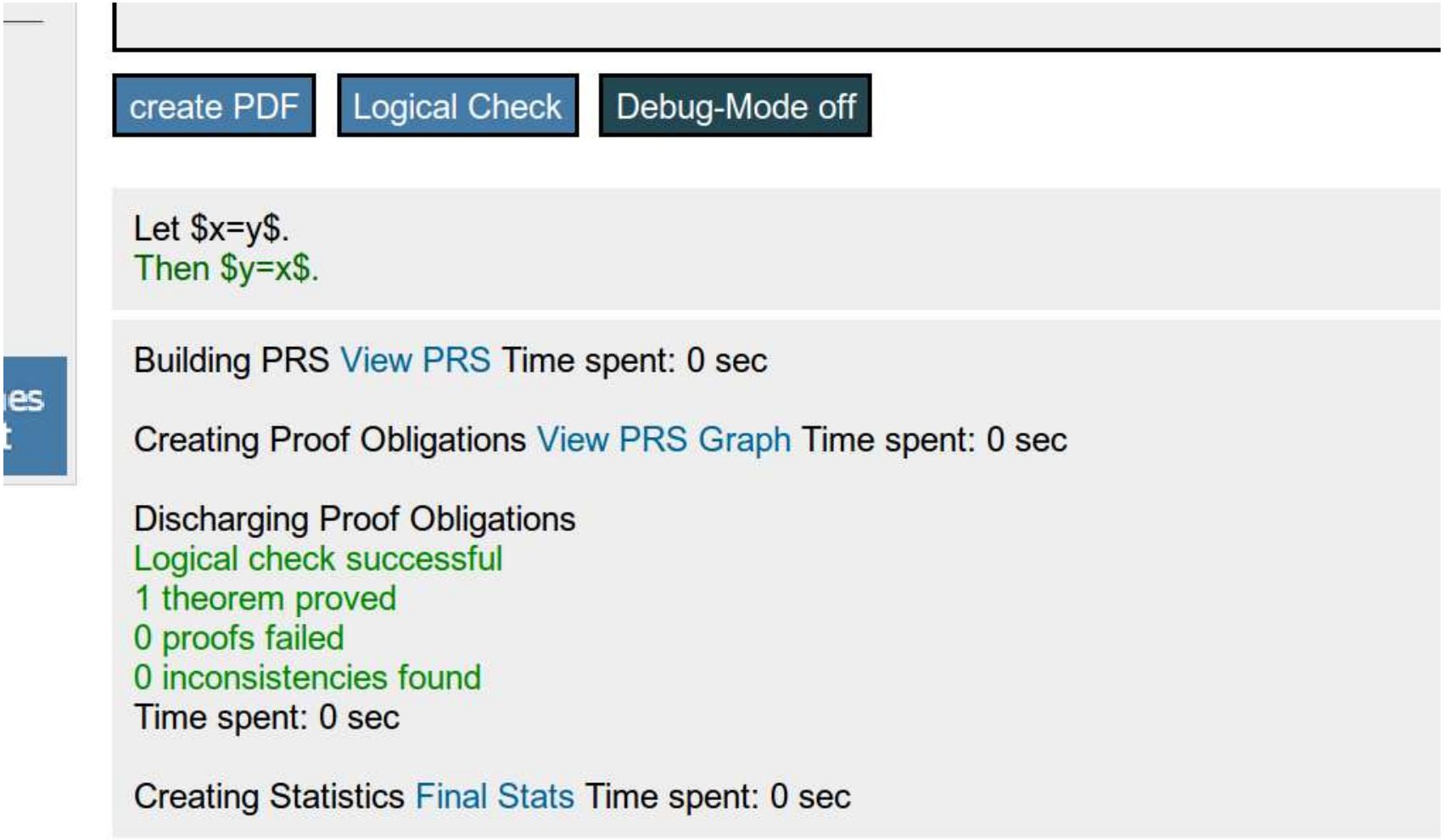
formal mathematics seminar

Enviroments | Arrows | Relations | Symbols

Let $x=y$. Then $y=x$.

create PDF | Logical Check | Debug-Mode off

The Naproche system



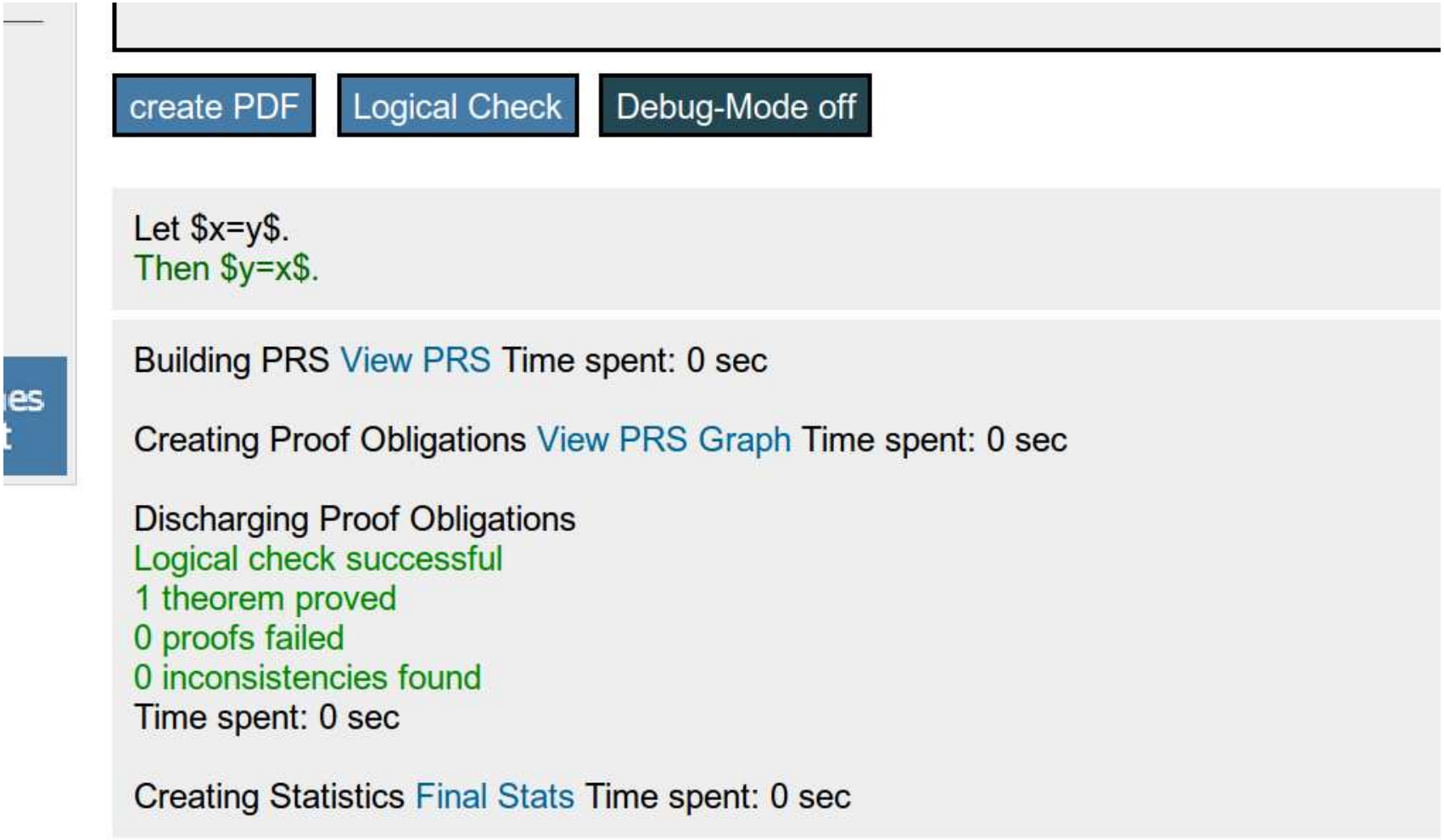
The Naproche system

Proof obligation for $y = x$:

fof('holds(2, 4, 0)', conjecture, vd2 = vd1).

fof('holds(1, 3, 0)', axiom, vd1 = vd2).

The Naproche system



The Naproche system

Axiom 1.
For all x, y, z, $(x*y)*z=x*(y*z)$.

Axiom 2.
For all x, $1*x=x$ and $x*1=x$.

Axiom 3.
For all x, $x*f(x)=1$ and $f(x)*x=1$.

Lemma 1.
If $u*x=x$ then $u=1$.

Proof.
Suppose that $u*x=x$.
Then $(u*x)*f(x)=x*f(x)$. By axiom 1, $u*(x*f(x))=x*f(x)$. So by axiom 3 $u*1=1$.
Then $u=1$ by axiom 2. Qed.

The Naproche system

Lemma 2.
If $x*y=1$ then $y=f(x)$.

Proof.
Assume $x*y=1$.
Then $f(x)*(x*y)=f(x)*1$, i.e. $(f(x)*x)*y=f(x)$. Hence $1*y=f(x)$, i.e. $y=f(x)$.
Qed.

Theorem 1.
$f(x*y)=f(y)*f(x)$.

Proof.
Let $u=(x*y)*(f(y)*f(x))$.
Then $u=x*((y*f(y))*f(x))$ by axiom 1. So $u=x*(1*f(x))=x*f(x)=1$.
Thus $(x*y)*(f(y)*f(x))=1$. Hence $(f(y)*f(x))=f(x*y)$ by lemma 2.
Qed.

The Naproche system

Axiom 1. For all x, y, z, $(x * y) * z = x * (y * z)$.

Axiom 2. For all x, $1 * x = x$ and $x * 1 = x$.

Axiom 3. For all x, $x * f(x) = 1$ and $f(x) * x = 1$.

Lemma 1. If $u * x = x$ then $u = 1$.

Proof. Suppose that $u * x = x$. Then $(u * x) * f(x) = x * f(x)$. By axiom 1, $u * (x * f(x)) = x * f(x)$. So by axiom 3 $u * 1 = 1$. Then $u = 1$ by axiom 2. Qed.

Lemma 2. If $x * y = 1$ then $y = f(x)$.

Proof. Assume $x * y = 1$. Then $f(x) * (x * y) = f(x) * 1$, i.e. $(f(x) * x) * y = f(x)$. Hence $1 * y = f(x)$, i.e. $y = f(x)$. Qed.

Theorem 1. $f(x * y) = f(y) * f(x)$.

Proof. Let $u = (x * y) * (f(y) * f(x))$. Then $u = x * ((y * f(y)) * f(x))$ by axiom 1. So $u = x * (1 * f(x)) = x * f(x) = 1$. Thus $(x * y) * (f(y) * f(x)) = 1$. Hence $(f(y) * f(x)) = f(x * y)$ by lemma 2. Qed.

The Naproche system

Building PRS View PRS Time spent: 4 sec

Creating Proof Obligations View PRS Graph Time spent: 0 sec

Discharging Proof Obligations Logical check successful

17 theorems proved

0 proofs failed

0 inconsistencies found

Time spent: 3 sec

Creating Statistics Final Stats Time spent: 0 sec