MathLang

Fairouz Kamareddine
Heriot-Watt University, Edinburgh

Thursday 5 February 2015
What do we want? Open borders for productive collaboration or that we each stick to our borders without including and benefiting from other input?

- Do we believe in the chosen framework? Should all the world believe in the same framework? Does one framework fit all? Can such a framework exist?

- Think of Capitalism, Communism, dictatorship, nationalism, etc... Which one worked in history?

- But then, if we are committed to pluralism, are we in danger of being wiped out because being inclusive may well lead to contradictions?

- Oscar Wilde: I used to think I was indecisive, but now I’m not sure anymore.
Things are not as somber: There is no perfect framework, but some can be invaluable

- De Bruijn used to proudly announce: *I did it my way.*

- I quote Dirk van Dalen: *The Germans have their 3 B’s, but we Dutch too have our 3 B’s: Beth, Brouwer and de Bruijn.*
There is a fourth B:
They look good together
Is our world going through identity crisis?

Logic was dormant until the 17th century when Leibniz wanted to prove things like the existence of god in a mechanical manner.

But the biggest kick off was in the 19th century, when the need for a more precise style in mathematics arose, because controversial results had appeared in analysis.

- 1821: Many controversies in analysis were solved by Cauchy. E.g., he gave a precise definition of convergence in his Cours d’Analyse [Cauchy, 1821].

- 1872: Due to the more exact definition of real numbers given by Dedekind [Dedekind, 1872], the rules for reasoning with real numbers became even more precise.

- 1895-1897: Cantor began formalizing set theory [Cantor, 1895, 1897] and made contributions to number theory.
Formal systems in the 19th century symbols (not natural language) define logical concepts

• 1889: Peano formalized arithmetic [Peano, 1889], but did not treat logic or quantification.

• 1879: Frege was not satisfied with the use of natural language in mathematics:

  “. . . I found the inadequacy of language to be an obstacle; no matter how unwieldy the expressions I was ready to accept, I was less and less able, as the relations became more and more complex, to attain the precision that my purpose required.”

  (Begriffsschrift, Preface)

Frege therefore presented Begriffsschrift [Frege, 1879], the first formalisation of logic giving logical concepts via symbols rather than natural language.
The rest is history

- Paradoxes, type theory, set theory, category theory, formalisation of mathematics (Principia, Hilbert and Ackermann, etc., etc.).

- Much is written on this subject. For the type theoretical view, see my book with Laan and Nederpelt.

- The foundational crisis was concurrent with the building of the physical computer.

- Then, the earlier work of Frege, Russell and Whitehead, Hilbert, etc., on the formalisation of mathematics, were now being complemented/replaced in the 1960s by the computerisation of mathematics.

- De Bruijn’s Automath and Trybulec’s Mizar were conceived around 1967.
And now, more than 100 years since Frege/Principia, and almost 50 years since the start of Automath

- We learned so much. But neither Principia nor Automath are living systems. Instead, there are many other systems out there.

- Can we find ways to combine the best of all these systems? Can we find ways to get more people to be involved in the computerisation/formalisation of mathematics without feeling discouraged by the complexity of one system?

- If we put all in one framework, this framework becomes exclusive.

- We want an open border framework which allows different beliefs.
From chapter 1, § 2 of E. Landau’s *Foundations of Analysis* (Landau 1930, 1951).

**Theorem 6. [Commutative Law of Addition]**

\[ x + y = y + x. \]

**Proof** Fix \( y \), and let \( \mathbb{M} \) be the set of all \( x \) for which the assertion holds.

1) We have

\[ y + 1 = y', \]

and furthermore, by the construction in the proof of Theorem 4,

\[ 1 + y = y', \]

so that

\[ 1 + y = y + 1 \]

and \( 1 \) belongs to \( \mathbb{M} \).

2) If \( x \) belongs to \( \mathbb{M} \), then

\[ x + y = y + x, \]

Therefore

\[ (x + y)' = (y + x)' = y + x'. \]

By the construction in the proof of Theorem 4, we have

\[ x' + y = (x + y)', \]

hence

\[ x' + y = y + x', \]

so that \( x' \) belongs to \( \mathbb{M} \). The assertion therefore holds for all \( x \). \( \square \)
The problem with formal logic

• No logical language is an alternative to CML
  – A logical language does not have mathematico-linguistic categories, is not universal to all mathematicians, and is not a good communication medium.
  – Logical languages make fixed choices (first versus higher order, predicative versus impredicative, constructive versus classical, types or sets, etc.). But different parts of mathematics need different choices and there is no universal agreement as to which is the best formalism.
  – A logician reformulates in logic their formalization of a mathematical-text as a formal, complete text which is structured considerably unlike the original, and is of little use to the ordinary mathematician.
  – Mathematicians do not want to use formal logic and have for centuries done mathematics without it.

• So, mathematicians kept to CML.

• But CML is difficult to computerise and formalise (either in a logical framework or computer system).
What are the options for computerization?

Computers can handle mathematical text at various levels:

- Images of pages may be stored. While useful, this is not a good representation of *language* or *knowledge*.
- Typesetting systems like LaTeX, TeXmacs, can be used.
- Document representations like OpenMath, OMDoc, MathML, can be used.
- Formal logics used by theorem provers (Coq, Isabelle, HOL, Mizar, Isar, etc.) can be used.

We are gradually developing a system named Mathlang which we hope will eventually allow building a bridge between the latter 3 levels.

This talk aims at discussing the motivations rather than the details.
Full formalization difficulties: choices

A CML-text is structured differently from a fully formalized text proving the same facts. Making the latter involves extensive knowledge and many choices:

• The choice of the underlying logical system.

• The choice of how concepts are implemented (equational reasoning, equivalences and classes, partial functions, induction, etc.).

• The choice of the formal foundation: a type theory (dependent?), a set theory (ZF? FM?), a category theory? etc.

• The choice of the proof checker: Automath, Isabelle, Coq, PVS, Mizar, HOL, ...

An issue is that one must in general commit to one set of choices.
Full formalization difficulties: informality

Any informal reasoning in a CML-text will cause various problems when fully formalizing it:

- A single (big) step may need to expand into a (series of) syntactic proof expressions. *Very long expressions can replace a clear CML-text.*

- The entire CML-text may need *reformulation* in a fully *complete* syntactic formalism where every detail is spelled out. New details may need to be woven throughout the entire text. The text may need to be *turned inside out.*

- Reasoning may be obscured by *proof tactics,* whose meaning is often *ad hoc* and implementation-dependent.

Regardless, ordinary mathematicians do not find the new text useful.
From Module Arith.Plus of Coq standard library (http://coq.inria.fr/).

Lemma plus_sym: (n,m:nat)(n+m)=(m+n).

Proof.

Intros n m ; Elim n ; Simpl rew ; Auto with arith.

Intros y H ; Elim (plus_n_-_Sm m y) ; Simpl rew ; Auto with arith.

Qed.
Mathlang’s Goal: Open borders between mathematics, logic and computation

• Ordinary mathematicians *avoid* formal mathematical logic.

• Ordinary mathematicians *avoid* proof checking (via a computer).

• Ordinary mathematicians *may use* a computer for computation: there are over 1 million people who use Mathematica (including linguists, engineers, etc.).

• Mathematicians may also use other computer forms like Maple, LaTeX, etc.

• But we are not interested in only *libraries* or *computation* or *text editing*.

• We want *freedom of movement* between mathematics, logic and computation.

• At every stage, we must have *the choice* of the level of formality and the depth of computation.
Aim for Mathlang? (Kamareddine and Wells 2001, 2002)

Can we formalise a mathematical text, avoiding as much as possible the ambiguities of natural language, while still guaranteeing the following four goals?

1. The formalised text looks very much like the original mathematical text (and hence the content of the original mathematical text is respected).

2. The formalised text can be fully manipulated and searched in ways that respect its mathematical structure and meaning.

3. Steps can be made to do computation (via computer algebra systems) and proof checking (via proof checkers) on the formalised text.

4. This formalisation of text is not much harder for the ordinary mathematician than \texttt{\LaTeX}. Full formalization down to a foundation of mathematics is not required, although allowing and supporting this is one goal.

(No theorem prover’s language satisfies these goals.)
<table>
<thead>
<tr>
<th>Mathlang</th>
<th>public documents</th>
<th>✓</th>
</tr>
</thead>
<tbody>
<tr>
<td>computations and proofs</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

- A Mathlang text captures different aspects (grammatical, textual, reasoning, etc., allowing degrees of computer manipulation).
- A **weak type system** checks Mathlang documents at a **grammatical level**.
- A **rewrite system** allows the **textual manipulation** of documents.
- The Mathlang document can be checked for **structural reasoning** (no loops in reasoning, etc.) before the inclusion of any logic.
- The MathLang document can be **completed to a fully formalised text** in a choice of provers (so far: Coq, Mizar and Isabelle) using semi-automated tools.
- As far as possible, the Mathlang text remains **close** to its CML original, allowing confidence that the CML has been captured correctly.
- The formal structure should be suitable for various automated uses.
What is CGa? (Maarek’s PhD thesis)

- CGa is a formal language derived from MV (N.G. de Bruijn 1987) and WTT (Kamareddine and Nederpelt 2004) which aims at expliciting the grammatical role played by the elements of a CML text.

- The structures and common concepts used in CML are captured by CGa with a finite set of grammatical/linguistic/syntactic categories: Term “√2”, set “Q”, noun “number”, adjective “even”, statement “a = b”, declaration “Let a be a number”, definition “An even number is..”, step “a is odd, hence a ≠ 0”, context “Assume a is even”.

- Generally, each syntactic category has a corresponding weak type.
• CGa’s type system derives typing judgments to check whether the reasoning parts of a document are coherently built.

• There is an element 0 in $R$ such that $a + 0 = a$.

Figure 1: Example of CGa encoding of CML text
Weak Type Theory

In Weak Type Theory (or WTT) we have the following linguistic categories:

- On the **atomic** level: variables, constants and binders,
- On the **phrase** level: terms $\mathcal{T}$, sets $\mathcal{S}$, nouns $\mathcal{N}$ and adjectives $\mathcal{A}$,
- On the **sentence** level: statements $P$ and definitions $D$,
- On the **discourse** level: contexts $\Gamma$, lines $l$ and books $B$. 
## Categories of syntax of WTT

<table>
<thead>
<tr>
<th>Other category</th>
<th>Abstract syntax</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>expressions</td>
<td>$\mathcal{E} = T</td>
<td>S</td>
</tr>
<tr>
<td>parameters</td>
<td>$\mathcal{P} = T</td>
<td>S</td>
</tr>
<tr>
<td>typings</td>
<td>$\mathcal{T} = S: \text{SET}</td>
<td>S: \text{STAT}</td>
</tr>
<tr>
<td>declarations</td>
<td>$\mathcal{Z} = V^S: \text{SET}</td>
<td>V^P: \text{STAT}</td>
</tr>
<tr>
<td>level</td>
<td>category</td>
<td>abstract syntax</td>
</tr>
<tr>
<td>------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>atomic</td>
<td>variables</td>
<td>$V = V^T</td>
</tr>
<tr>
<td></td>
<td>constants</td>
<td>$C = C^T</td>
</tr>
<tr>
<td></td>
<td>binders</td>
<td>$B = B^T</td>
</tr>
<tr>
<td>phrase</td>
<td>terms</td>
<td>$T = C^T(\vec{P})</td>
</tr>
<tr>
<td></td>
<td>sets</td>
<td>$S = C^S(\vec{P})</td>
</tr>
<tr>
<td></td>
<td>nouns</td>
<td>$N = C^N(\vec{P})</td>
</tr>
<tr>
<td></td>
<td>adjectives</td>
<td>$A = C^A(\vec{P})</td>
</tr>
<tr>
<td>sentence</td>
<td>statements</td>
<td>$P = C^P(\vec{P})</td>
</tr>
<tr>
<td></td>
<td>definitions</td>
<td>$D = D^\varphi</td>
</tr>
<tr>
<td></td>
<td>$D^\varphi = C^T(\vec{V}) := T</td>
<td>C^S(\vec{V}) := S</td>
</tr>
<tr>
<td>discourse</td>
<td>contexts</td>
<td>$\Gamma = \emptyset</td>
</tr>
<tr>
<td></td>
<td>lines</td>
<td>$l = \Gamma \triangleright P</td>
</tr>
<tr>
<td></td>
<td>books</td>
<td>$B = \emptyset</td>
</tr>
</tbody>
</table>
Derivation rules

(1) \( B \) is a weakly well-typed book: \( \vdash B :: \text{book} \).

(2) \( \Gamma \) is a weakly well-typed context relative to book \( B \): \( B \vdash \Gamma :: \text{cont} \).

(3) \( t \) is a weakly well-typed term, etc., relative to book \( B \) and context \( \Gamma \):

\[
B; \Gamma \vdash t :: T, \quad B; \Gamma \vdash s :: S, \quad B; \Gamma \vdash n :: N, \\
B; \Gamma \vdash a :: A, \quad B; \Gamma \vdash p :: P, \quad B; \Gamma \vdash d :: D
\]

\( \text{OK}(B; \Gamma) \). stands for: \( \vdash B :: \text{book}, \ \text{and} \ \ B \vdash \Gamma :: \text{cont} \)
Examples of derivation rules

\begin{itemize}
  \item \text{dvar}(\emptyset) = \emptyset \quad \text{dvar}(\Gamma', x : W) = \text{dvar}(\Gamma'), x \quad \text{dvar}(\Gamma', P) = \text{dvar}(\Gamma')
\end{itemize}

\[
\begin{align*}
\text{OK}(B; \Gamma), \quad x & \in V^{T/S/P}, \quad x \in \text{dvar}(\Gamma) \\
B; \Gamma & \vdash x :: T/S/P \quad \text{(var)}
\end{align*}
\]

\[
\begin{align*}
B; \Gamma & \vdash n :: N, \quad B; \Gamma \vdash a :: A \\
B; \Gamma & \vdash an :: N \quad \text{(adj-noun)}
\end{align*}
\]

\[
\begin{align*}
\vdash \emptyset :: \text{book} \quad \text{(emp-book)}
\end{align*}
\]

\[
\begin{align*}
B; \Gamma & \vdash p :: P \\
\vdash B \circ \Gamma \triangleright p :: \text{book} \quad B; \Gamma & \vdash d :: D \\
\vdash B \circ \Gamma \triangleright d :: \text{book} \quad \text{(book-ext)}
\end{align*}
\]
Properties of WTT

- **Every variable is declared** If $B; \Gamma \vdash \Phi :: \mathbf{W}$ then $FV(\Phi) \subseteq dvar(\Gamma)$.
- **Correct subcontexts** If $B \vdash \Gamma :: \text{cont}$ and $\Gamma' \subseteq \Gamma$ then $B \vdash \Gamma' :: \text{cont}$.
- **Correct subbooks** If $B :: \text{book}$ and $B' \subseteq B$ then $B' :: \text{book}$.
- **Free constants are either declared in book or in contexts** If $B; \Gamma \vdash \Phi :: \mathbf{W}$, then $FC(\Phi) \subseteq \text{prefcons}(B) \cup \text{defcons}(B)$.
- **Types are unique** If $B; \Gamma \vdash A :: W_1$ and $B; \Gamma \vdash A :: W_2$, then $W_1 \equiv W_2$.
- **Weak type checking is decidable** there is a decision procedure for the question $B; \Gamma \vdash \Phi :: \mathbf{W}$.
- **Weak typability is computable** there is a procedure deciding whether an answer exists for $B; \Gamma \vdash \Phi :: ?$ and if so, delivering the answer.
Definition unfolding

• Let $\vdash B :: \text{book}$ and $\Gamma \triangleright c(x_1, \ldots, x_n) := \Phi$ a line in $B$.

• We write $B \vdash c(P_1, \ldots, P_n) \overset{\delta}{\rightarrow} \Phi[x_i := P_i]$.

• **Church-Rosser** If $B \vdash \Phi \overset{\delta}{\rightarrow} \Phi_1$ and $B \vdash \Phi \overset{\delta}{\rightarrow} \Phi_2$ then there exists $\Phi_3$ such that $B \vdash \Phi_1 \overset{\delta}{\rightarrow} \Phi_3$ and $B \vdash \Phi_2 \overset{\delta}{\rightarrow} \Phi_3$.

• **Strong Normalisation** Let $\vdash B :: \text{book}$. For all subformulas $\Psi$ occurring in $B$, relation $\overset{\delta}{\rightarrow}$ is strongly normalizing (i.e., definition unfolding inside a well-typed book is a well-founded procedure).
Let $M$ be a set, $y$ and $x$ are natural numbers, if $x$ belongs to $M$ then $x + y = y + x$. 
Let $M$ be a set, $y$ and $x$ are natural numbers, if $x$ belongs to $M$ then $x + y \Leftarrow \text{error}$
How complete is the CGa?

- CGa is quite advanced but remains under development according to new translations of mathematical texts. Are the current CGa categories sufficient?

- The metatheory of WTT has been established in (Kamareddine and Nederepelt 2004). That of CGa remains to be established. However, since CGa is quite similar to WTT, its metatheory might be similar to that of WTT.

- The type checker for CGa works well and gives some useful error messages. Error messages should be improved.
Common Mathematical Language (CML)

Tagging

Structuring

Formalising

Text and symbol aspect (TSa)

Core Grammatical aspect (CGa)

Further upcoming aspects

MathLang

Document Rhetorical aspect (DRA)

Further computations e.g. Computer Algebra Systems

Automatic skeleton generation

Fine-structuring

Formal Proof Sketch (e.g. Mizar, Isar, HOL)

Completing-formalising

Complete Proof

Theorem Provers

Mathematical Libraries
What is TSa? Lamar’s PhD thesis

• TSa builds the bridge between a CML text and its grammatical interpretation and adjoins to each CGa expression a string of words and/or symbols which aims to act as its CML representation.

• *TSa plays the role of a user interface*

• *TSa can flexibly represent natural language mathematics.*

• The author wraps the natural language text with boxes representing the grammatical categories (as we saw before).

• *The author can also give interpretations to the parts of the text.*
Interpretations

There is an element 0 in \( \mathbb{R}^R \) such that \( \text{eq } \frac{\mid a \mid}{\frac{\mid a \mid}{0}} = \frac{\mid a \mid}{\frac{\mid a \mid}{0}} \).

\[ \{ 0 : \mathbb{R}; \text{eq } (\text{plus}(a, 0), a); \} \];

There is an element 0 in \( \mathbb{R}^R \) such that \( \text{eq } \frac{\mid a \mid}{\frac{\mid a \mid}{0}} = \frac{\mid a \mid}{\frac{\mid a \mid}{0}} \).

There is an element 0 in \( \mathbb{R}^R \) such that \( \text{eq } \frac{\mid a \mid}{\frac{\mid a \mid}{0}} = \frac{\mid a \mid}{\frac{\mid a \mid}{0}} \).
Rewrite rules enable natural language representation

Take the example \(0 + a0 = a0 = a(0 + 0) = a0 + a0\)
Figure 2: Example for a simple shared souring
reordering/position Sourcing

\[ n \in \mathbb{N} \]

\[ \forall n \in \mathbb{N} : \exists 1 \leq n \leq 2 \]

\[ ann = \text{contains} \]

\[ n \in \mathbb{N} \]

Lethbridge 2015
Figure 3: Example for a position souring
map souring

\[ \text{ann} = \langle \text{map} \rangle \langle \text{list} \rangle \langle a \rangle a \text{ and } \langle b \rangle b \text{ be in } \langle R \rangle \]

This is expanded to

\[ T(\text{ann}) = \langle \langle a \rangle \langle R \rangle \rangle \cap \langle b \rangle \langle R \rangle \]

Lethbridge 2015
How complete is TSa?

• TSa provides useful interface facilities but it is still under development.

• So far, only simple rewrite (sourcing) rules are used and they are not comprehensive. E.g., unable to cope with things like $\underbrace{x = \ldots = x}_n$.

• The TSa theory and metatheory need development.
What is DRa? Retel’s PhD thesis

- DRa Document Rhetorical structure aspect.
- **Structural components of a document** like chapter, section, subsection, etc.
- **Mathematical components of a document** like theorem, corollary, definition, proof, etc.
- **Relations** between above components.
- These enhance readability, and ease the navigation of a document.
- Also, these help to go into more formal versions of the document.
## Relations

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Instances of the <code>StructuralRhetoricalRole</code> class:</strong> preamble, part, chapter, section, paragraph, <em>etc.</em></td>
</tr>
<tr>
<td><strong>Instances of the <code>MathematicalRhetoricalRole</code> class:</strong> lemma, corollary, theorem, conjecture, definition, axiom, claim, proposition, assertion, proof, exercise, example, problem, solution, <em>etc.</em></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Relation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Types of relations:</strong> relatesTo, uses, justifies, subpartOf, inconsistentWith, exemplifies</td>
</tr>
</tbody>
</table>
What does the mathematician do?

• The mathematician wraps into boxes and uniquely names chunks of text

• The mathematician assigns to each box the structural and/or mathematical rhetorical roles

• The mathematician indicates the relations between wrapped chunks of texts
Lemma 1. For $m, n \in \mathbb{N}$ one has: $m^2 = 2n^2 \implies m = n = 0$.

Define on $\mathbb{N}$ the predicate:

$$P(m) \iff \exists n. m^2 = 2n^2 \& m > 0.$$ 

Claim. $P(m) \implies \exists m' < m. P(m')$. Indeed suppose $m^2 = 2n^2$ and $m > 0$. It follows that $m^2$ is even, but then $m$ must be even, as odds square to odds. So $m = 2k$ and we have

$$2n^2 = m^2 = 4k^2 \implies n^2 = 2k^2$$

Since $m > 0$, it follows that $m^2 > 0, n^2 > 0$ and $n > 0$. Therefore $P(n)$. Moreover, $m^2 = n^2 + n^2 > n^2$, so $m^2 > n^2$ and hence $m > n$. So we can take $m' = n$.

By the claim $\forall m \in \mathbb{N}. \neg P(m)$, since there are no infinite descending sequences of natural numbers.

Now suppose $m^2 = 2n^2$ with $m \neq 0$. Then $m > 0$ and hence $P(m)$. Contradiction. Therefore $m = 0$. But then also $n = 0$.

Corollary 1. $\sqrt{2} \notin \mathbb{Q}$.

Suppose $\sqrt{2} \in \mathbb{Q}$, i.e. $\sqrt{2} = p/q$ with $p \in \mathbb{Z}, q \in \mathbb{Z} - \{0\}$. Then $\sqrt{2} = m/n$ with $m = \lvert p \rvert, n = \lvert q \rvert \neq 0$. It follows that $m^2 = 2n^2$. But then $n = 0$ by the lemma. Contradiction shows that $\sqrt{2} \notin \mathbb{Q}$. 

Lethbridge 2015

Barendregt 45
Lemma 1.

For \( m, n \in \mathbb{N} \) one has: \( m^2 = 2n^2 \) if and only if \( m = n = 0 \).

Proof.

Define on \( \mathbb{N} \) the predicate:

\[
P(m) \iff \exists n. m^2 = 2n^2 \land m > 0.
\]

Claim. \( P(m) \implies \exists m'. m < m \cdot P(m') \).

Indeed suppose \( m^2 = 2n^2 \) and \( m > 0 \). It follows that \( m^2 \) is even, but then \( m \) must be even, as odds squared have odds. So \( m = 2k \) and we have:

\[2n^2 = m^2 = 4k^2 \implies n^2 = 2k^2 \]

Since \( m > 0 \), it follows that \( m^2 > 0 \), \( n^2 > 0 \) and \( n > 0 \). Therefore \( P(n) \). Moreover, \( m^2 = n^2 + n^2 > n^2 \), so \( m^2 > n^2 \) and hence \( m > n \). So we can take \( m' = n \).

By the claim \( \forall m \in \mathbb{N}. \neg P(m) \), since there are no infinite descending sequences of natural numbers.

Now suppose \( m^2 = 2n^2 \) with \( m \neq 0 \). Then \( m > 0 \) and hence \( P(n) \). Contradiction.

Therefore \( m = 0 \). But then also \( n = 0 \).

\[ \square \]

Corollary 1. \( \sqrt{2} \in \mathbb{Q} \).

Proof. Suppose \( \sqrt{2} \in \mathbb{Q} \), i.e. \( \sqrt{2} = \frac{p}{q} \) with \( p \in \mathbb{Z}, q \in \mathbb{Z} \setminus \{0\} \). Then \( \sqrt{2} = \frac{m}{n} \) with \( m = |p|, n = |q| \neq 0 \). It follows that \( m^2 = 2n^2 \). But then \( n = 0 \) by the lemma. Contradiction shows that \( \sqrt{2} \notin \mathbb{Q} \). \[ \square \]
(A, hasMathematicalRhetoricalRole, lemma)
(E, hasMathematicalRhetoricalRole, definition)
(F, hasMathematicalRhetoricalRole, claim)
(G, hasMathematicalRhetoricalRole, proof)
(B, hasMathematicalRhetoricalRole, proof)
(H, hasOtherMathematicalRhetoricalRole, case)
(I, hasOtherMathematicalRhetoricalRole, case)
(C, hasMathematicalRhetoricalRole, corollary)
(D, hasMathematicalRhetoricalRole, proof)

(B, justifies, A)
(D, justifies, C)
(D, uses, A)
(G, uses, E)
(F, uses, E)
(H, uses, E)
(H, subpartOf, B)
(H, subpartOf, I)
Lemma 1.

For \( m, n \in \mathbb{N} \) one has: \( m^2 = 2n^2 \) \( \iff \) \( n = n = 0 \).

Proof.

Define on \( \mathbb{N} \) the predicate:

\[ P(m) \text{ uses } \exists n. m^2 = 2n^2 \& m > 0. \]

Claim. \[ P(m) \implies m < m.P(m'). \]

Indeed suppose \( m^2 = 2n^2 \) and \( m > 0 \). It follows that \( m^2 \) is even, but then \( m \) must be even, and \( n \) must be odd. So \( m = 2k \) and we have:

\[ 2n^2 = m^2 = 4k^2 \implies n^2 = 2k^2 \]

Since \( m > 0 \), it follows that \( m^2 > 0, n^2 > 0 \) and \( n > 0 \). Therefore \( P(n) \). Moreover, \( m^2 = n^2 + n^2 > n^2 \), so \( m^2 > n^2 \) and hence \( m > n \). So we can take \( m' = n \).

By the claim \( \forall m \in \mathbb{N}. \neg P(m) \), since there are descending sequences of natural numbers.

Now suppose \( m^2 = 2n^2 \) with \( m \neq 0 \). Then \( m > 0 \) and hence \( P(m) \). Contradiction.
The automatically generated dependency Graph

Diagram:

- **A**
- **E**
- **F**
- **G**
- **B**
- **H**
- **I**
- **C**
- **D**

Dependency relationships:

- **A** uses **E**
- **E** justifies **G**
- **G** uses **B**
- **B** subpartOf **H**
- **H** subpartOf **I**
- **I** uses **C**
- **C** justifies **D**
An alternative view of the DRa (Zengler’s thesis)
The Graph of Textual Order: GoTO

Zengler’s thesis

- To be able to examine the proper structure of a DRa tree we introduce the concept of textual order between two nodes in the tree.

- Using textual orders, we can transform the dependency graph into a GoTO by transforming each edge of the DG.

- So far there are two reasons why the GoTO is produced:
  1. Automatic Checking of the GoTO can reveal errors in the document (e.g. loops in the structure of the document).
  2. The GoTO is used to automatically produce a proof skeleton for a prover (we use a variety: Isabelle, Mizar, Coq).

- We automatically transform a DG into GoTO and automatically check the GoTO for errors in the document:
1. Loops in the GoTO (error)
2. Proof of an unproved node (error)
3. More than one proof for a proved node (warning)
4. Missing proof for a proved node (warning)
Graph of Textual Order for the DRa tree example
How complete is DRa?

• The dependency graph can be used to check whether the logical reasoning of the text is coherent and consistent (e.g., no loops in the reasoning).

• However, both the DRa language and its implementation need more experience driven tests on natural language texts.

• Also, the DRa aspect still needs a number of implementation improvements (the automation of the analysis of the text based on its DRa features).

• Extend TSa to also cover DRa (in addition to CGa).

• Extend DRa depending on further experience driven translations.

• Establish the soundness and completeness of DRa for mathematical texts.
Different provers have

- different syntax

- different requirements to the structure of the text
e.g.
- no nested theorems/lemmas
- only backward references
- ...

- Aim: Skeleton should be as close as possible to the mathematician’s text but with re-arrangements when necessary

*Example of nested theorems/lemmas (Moller, 03, Chapter III,2)*

*The automatic generation of a proof skeleton*
The DG for the example
Straight-forward translation of the first part

Definition : \(<\text{def1}>\).

Definition : \(<\text{def2}>\).

Theorem th1: \(<\text{th1}>\).
Proof.
\(<\text{pr1}>\)
Qed.
Solution: Re-ordering
Definition 1

Definition 2

Theorem 1

Proof of Theorem 1

Theorem 2

Lemma 1

Proof of Lemma 1

Proof of Theorem 2

Definition: \(<\text{def1}>\).

Definition: \(<\text{def2}>\).

Theorem \(\text{th1} \): \(<\text{th1}>\).
Proof.
\(<\text{pr_th1}>\)
Qed.

Lemma \(\text{lem1} \): \(<\text{lem1}>\).
Proof.
\(<\text{pr_lem1}>\)
Qed.

Theorem \(\text{th2} \): \(<\text{th2}>\).
Proof.
\(<\text{pr_th2}>\)
Qed.

Finishing the skeleton
Skeleton for Mizar
definition def1:
  <def1>
end;

definition def2:
  <def2>
end;

theorem th1:
  <th1>
    proof
    <pr_th1>
end;

lem1:
  <lem1>
    proof
    <pr_lem1>
end;

theorem th2:
  <th2>
    proof
    <pr_th2>
end;
DRa annotation into Mizar skeleton for Barendregt's example (Retel's PhD thesis)
The generic algorithm for generating the proof skeleton (SGa, Zengler’s thesis)

A vertex is ready to be processed iff:

• it has no incoming \( \prec \) edges (in the GoTO) of unprocessed (white) vertices

• all its children are ready to be processed

• if the vertex is a proved vertex: its proof is ready to be processed

Consider the DG and GoTO of a (typical and not well structured) mathematical text:
The final order of the vertices is:

Lemma 2
Proof 2
  Definition 2
  Claim 2
  Proof C2
Lemma 1
Proof 1
  Definition 1
  Claim 1
  Proof C1
Figure 6: A flattened graph of the GoTO of figure 5 without nested definitions
Figure 7: A flattened graph of the GoTO of figure 5 without nested claims
## The Mizar and Coq rules for the dictionary

<table>
<thead>
<tr>
<th>Role</th>
<th>Mizar rule</th>
<th>Coq rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>axiom</td>
<td>%name : %body ;</td>
<td>Axiom %name : %body .</td>
</tr>
<tr>
<td>definition</td>
<td>definition %name : %nl %body %nl end;</td>
<td>Definition : %body .</td>
</tr>
<tr>
<td>theorem</td>
<td>theorem %name : %nl %body</td>
<td>Theorem %name %body .</td>
</tr>
<tr>
<td>proof</td>
<td>proof %nl %body %nl end;</td>
<td>Proof %name : %body .</td>
</tr>
<tr>
<td>cases</td>
<td>per cases; %nl</td>
<td>%body</td>
</tr>
<tr>
<td>case</td>
<td>suppose %nl %body %nl end;</td>
<td>%body</td>
</tr>
<tr>
<td>existencePart</td>
<td>existence %nl %body</td>
<td>%body</td>
</tr>
<tr>
<td>uniquenessPart</td>
<td>uniqueness %nl %body</td>
<td>%body</td>
</tr>
</tbody>
</table>
Rich skeletons for Coq

<table>
<thead>
<tr>
<th>Rule $N^\alpha$</th>
<th>Annotation $ann$</th>
<th>Coq translation $S_{Coq}(ann)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>coq1)</td>
<td>&lt;#&gt;</td>
<td>Set</td>
</tr>
<tr>
<td>coq2)</td>
<td>&lt;#&gt;</td>
<td>Prop</td>
</tr>
<tr>
<td>coq3)</td>
<td>&lt;id&gt; &lt;N&gt;</td>
<td>id : N</td>
</tr>
<tr>
<td>coq4)</td>
<td>&lt;id&gt; &lt;S&gt;</td>
<td>id : S</td>
</tr>
<tr>
<td>coq5)</td>
<td>&lt;id&gt;</td>
<td>id</td>
</tr>
<tr>
<td>coq6)</td>
<td>&lt;id&gt; $p_1$ ... $p_n$ &lt;N&gt;</td>
<td>$id : S_{Coq}(p_1) \rightarrow ... \rightarrow S_{Coq}(p_n) \rightarrow N$</td>
</tr>
<tr>
<td>coq7)</td>
<td>&lt;id&gt; $p_1$ ... $p_n$ &lt;S&gt;</td>
<td>$id : S_{Coq}(p_1) \rightarrow ... \rightarrow S_{Coq}(p_n) \rightarrow S$</td>
</tr>
</tbody>
</table>
| coq8) | $\text{id : } S_{Coq} \left( \begin{array}{c} p_1 \\ ... \\ p_n \end{array} \right) \rightarrow \cdots \rightarrow S_{Coq} \left( \begin{array}{c} p_n \end{array} \right) \rightarrow \text{Prop} |$
|---|---|
| coq9) | $\text{id : } S_{Coq} \left( \begin{array}{c} p_1 \\ ... \\ p_n \end{array} \right) \rightarrow \cdots \rightarrow S_{Coq} \left( \begin{array}{c} p_n \end{array} \right) \rightarrow \text{Set} |$
| coq10) | $(\text{id } S_{Coq} \left( \begin{array}{c} p_1 \\ ... \\ p_n \end{array} \right) \cdots S_{Coq} \left( \begin{array}{c} p_n \end{array} \right) ) |$
| coq11) | $(\text{id } S_{Coq} \left( \begin{array}{c} p_1 \\ ... \\ p_n \end{array} \right) \cdots S_{Coq} \left( \begin{array}{c} p_n \end{array} \right) ) |$
| coq12) | $(\text{id } S_{Coq} \left( \begin{array}{c} p_1 \\ ... \\ p_n \end{array} \right) \cdots S_{Coq} \left( \begin{array}{c} p_n \end{array} \right) ) |$
| coq13) | $\text{id} |$
| coq14) | $\text{id id}_1 \ldots \text{id}_n := S_{Coq} \left( \begin{array}{c} e \end{array} \right) |$
With these rules almost every axiom, definition and theorem can be translated in a way that it is immediately usable in Coq.
the left hand side of the definition is translated according to rule (coq14)) with subset A B.

The right hand side is translated with the rules coq5), coq10), coq11) and coq12) and the result is

\[\forall x \ (\text{impl} \ (\text{in} \ x \ A) \ (\text{in} \ x \ B))\]

Putting left hand and right hand side together and taking the outer DRa annotation we get the translation

Definition subset A B := forall x (impl (in x A) (in x B))
Figure 8: Theorem 17 of Landau’s “Grundlagen der Analysis”

The automatic translation is:

Theorem th117 x y z : (leq x y /\ leq y z) -> leq x z .
The corresponding translation into Isabelle is:

assumes carriernonempty: "not (set-equal R emptyset)"
An example of a full formalisation in Coq via MathLang

Figure 9: The path for processing the Landau chapter
Theorem 1.6. **Commutative Law of Addition**

\[
\text{eq}\times\text{plus}\times y + x = \text{plus}\times y + \times x.
\]

Figure 10: Simple theorem of the second section of Landau’s first chapter
Figure 11: The annotated theorem 16 of the Landau’s first chapter
Chapter 1
Natural Numbers

1.1 Axioms

We assume the following to be given:

A set (i.e. totality) of objects called natural numbers, possessing the properties called axioms to be listed below.

Before formulating the axioms we make some remarks about the symbols = and \neq which be used.

Unless otherwise specified, small italic letters will stand for natural numbers throughout this book.

If \( x \) is given and \( y \) is given, then either \( x \) and \( y \) are the same number; this may be written \( x = y \); or \( x \) and \( y \) are not the same number; this may be written \( x \neq y \).

(\( \neq \) to be read "is not equal to").

Accordingly, the following are true on purely logical grounds:

\[ x = y \quad \text{for every } x, y \]

\[ x \neq y \quad \text{for every } x, y \]

\[ \text{if } x \neq y \quad \text{then } y \neq x \]
Chapter 1 of Landau:

- 5 axioms which we annotate with the mathematical role "axiom", and give them the names "ax11" - "ax15".

- 6 definitions which we annotate with the mathematical role "definition", and give them names "def11" - "def16".

- 36 nodes with the mathematical role "theorem", named "th11" - "th136" and with proofs "pr11" - "pr136".

- Some proofs are partitioned into an existential part and a uniqueness part.

- Other proofs consist of different cases which we annotate as unproved nodes with the mathematical role "case".

Figure 12: The DRa tree of sections 1 and 2 of chapter 1 of Landau’s book
• The relations are annotated in a straightforward manner.

• Each proof justifies its corresponding theorem.

• Axiom 5 ("ax15") is the axiom of induction. So every proof which uses induction, uses also this axiom.

• Definition 1 ("def11") is the definition of addition. Hence every node which uses addition also uses this definition.

• Some theorems use other theorems via texts like: “By Theorem ...”.

• In total we have 36 justifies relations, 154 uses relations, 6 caseOf, 3 existencePartOf and 3 uniquenessPartOf relations.

• The DG and GoTO are automatically generated.

• The GoTO is automatically checked and no errors result. So, we proceed to the next stage: automatically generating the SGa.
Figure 13: The DG of sections 1 and 2 of chapter 1 of Landau's book
The GoTO of section 1 - 4
Definition geq x y := (or (gt x y) (eq x y)).
Definition leq x y := (or (lt x y) (eq x y)).

Theorem th113 x y : (impl (geq x y) (leq y x)).
Proof.
...
Qed.

Theorem th114 x y : (impl (leq x y) (geq y x)).
Proof.
...
Qed.

Theorem th115 x y z : (impl (impl (lt x y) (lt y z)) (lt x z)).
Proof.
...
Qed.
Completing the proofs in Coq

• We defined the natural numbers as an inductive set - just as Landau does in his book.

\[
\text{Inductive nats : Set :=}
| \text{I : nats}
| \text{succ : nats -> nats}
\]

• The encoding of theorem 2 of the first chapter in Coq is

\[
\text{theorem th12 x : neq (succ x) x .}
\]

• Landau proves this theorem with induction. He first shows, that \(1' \neq 1\) and then that with the assumption of \(x' \neq x\) it also holds that \((x')' \neq x'\).

• We do our proof in the Landau style. We introduce the variable \(x\) and eliminate it, which yields two subgoals that we need to prove. These subgoals are exactly the induction basis and the induction step.
Proof.
intro x. elim x.

2 subgoals
x : nats

neq (succ I) I}

forall n : nats, neq (succ n) n -> neq (succ (succ n)) (succ n)

Landau proved the first case with the help of Axiom 3 (for all $x, x' \neq 1$).

apply ax13.

1 subgoal
x : nats

forall n : nats, neq (succ n) n -> neq (succ (succ n)) (succ n)
The next step is to introduce $n$ as natural number and to introduce the induction hypothesis:

```lean
intros n H.
```

1 subgoal

```lean
x : nats
n : nats
H : neq (succ n) n

neq (succ (succ n)) (succ n)
```

We see that this is exactly the second case of Landau’s proof. He proved this case with Theorem 1 - we do the same:

```lean
apply th11.
```

1 subgoal

```lean
x : nats
n : nats
```
H : neq (succ n) n

And of course this is exactly the induction hypotheses which we already have as an assumption and we can finish the proof:

assumption.
Proof completed.

The complete theorem and its proof in Coq finally look like this:

Theorem th12 (x:nats) : neq (succ x) x .
Proof.
intro x. elim x.
apply ax13.
intros n H.
apply th11.
assumption.
Qed.
With the help of the CGa annotations and the automatically generated rich proof skeleton, Zengler (who was not familiar with Coq) completed the Coq proofs of the whole of chapter one in a couple of hours.
Some points to consider

• We do not at all assume/prefer one type/logical theory instead of another.

• The formalisation of a language of mathematics should separate the questions:
  – which type/logical theory is necessary for which part of mathematics
  – which language should mathematics be written in.

• MathLang is independent of any foundation of mathematics.
• Instead of English, one can use Arabic, French, German, etc.

• MathLang aims to support non-fully-formalized mathematics practiced by the ordinary mathematician as well as work toward full formalization.

• MathLang aims to handle mathematics as expressed in natural language as well as symbolic formulas.

• MathLang allows anyone to be involved, whether a mathematician, a computer engineer, a computer scientist, a linguist, a logician, etc.
Bibliography


