Foundational libraries in Naproche

Marcel Schütz

CICM 2023, Cambridge, UK, 4-8 September 2023

EuroProofNet Joint WG4-WG5 meeting, Cambridge, UK, 6-8 September 2023

September 7, 2023

The Naproche System

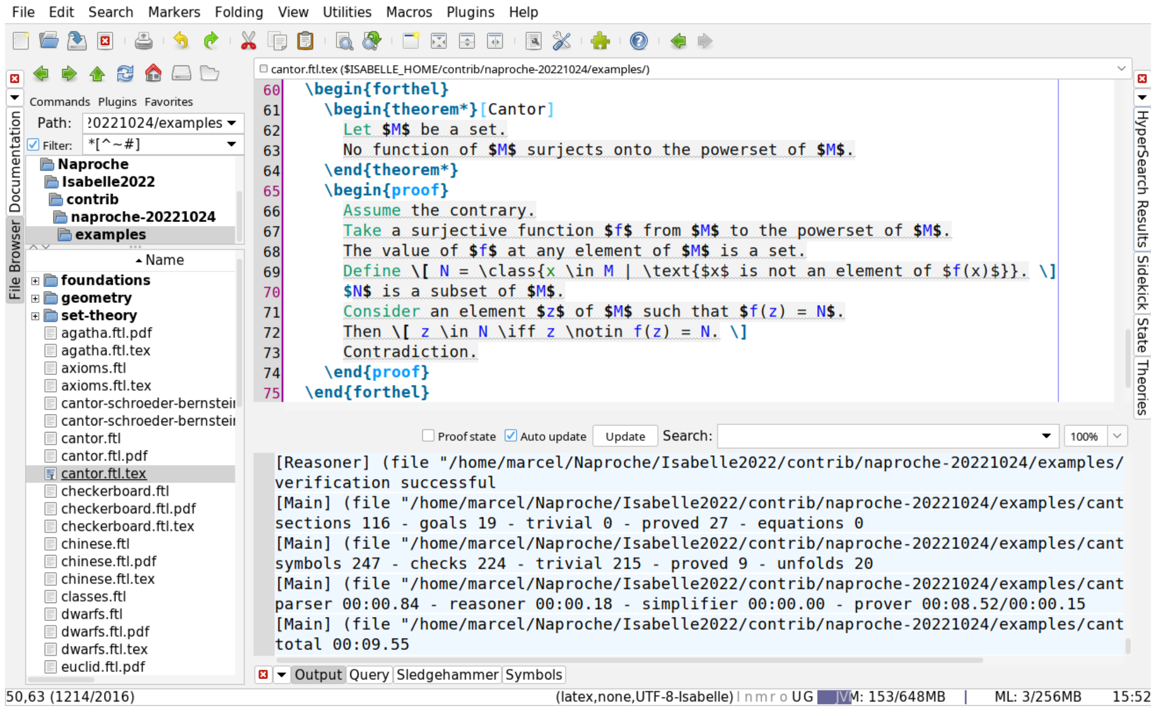
Naproche/ForTheL

Naproche = **Nat**ural **pro**of **che**cking

- Proof assistant
- Component of Isabelle

ForTheL = **For**mula **The**ory **L**anguage

- Naproche's input language
- Controlled natural language
- $\LaTeX$-compatible



Cantor's Theorem in Isabelle/jEdit

The Naproche System

Verification & $\LaTeX$ Integration

```
\begin{forthel}
  \begin{theorem*}[Cantor]
    Let $M$ be a set.
    No function of $M$ surjects onto the powerset of $M$.
  \end{theorem*}
  \begin{proof}
    Assume the contrary.
    Take a surjective function $f$ from $M$ to the powerset of $M$.
    The value of $f$ at any element of $M$ is a set.
    Define \[ N = \class{x \in M | \text{$x$ is not an element of $f(x)$}}. \]
    $N$ is a subset of $M$.
    Consider an element $z$ of $M$ such that $f(z) = N$.
    Then \[ z \in N \iff z \notin f(z) = N. \]
    Contradiction.
  \end{proof}
\end{forthel}
```

Naproche

FOL (TPTP)

E, Vampire

`verification successful`

or

`verification failed`

pdflatex

Theorem (Cantor). Let M be a set. No function of M surjects onto the powerset of M.

Proof. Assume the contrary. Take a surjective function f from M to the powerset of M. The value of f at any element of M is a set. Define

$$N = \{x \in M \mid x \text{ is not an element of } f(x)\}.$$

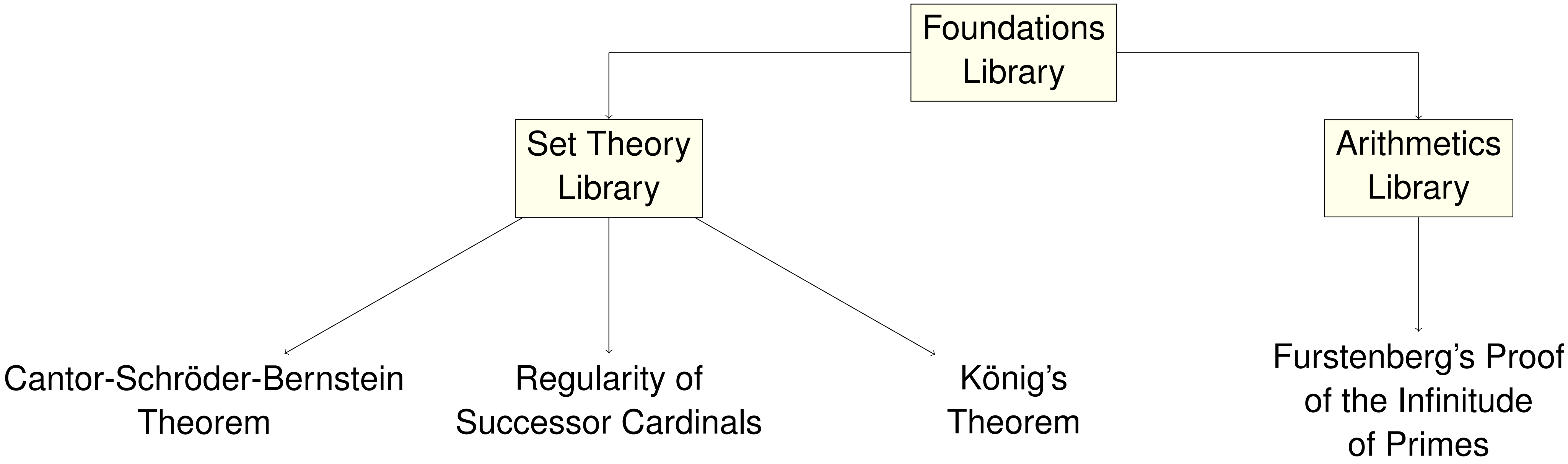
N is a subset of M. Consider an element z of M such that $f(z) = N$. Then

$$z \in N \iff z \notin f(z) = N.$$

Contradiction. □

Libraries in Naproche

Three Case Studies



Libraries in Naproche

Typical examples

ARITHMETIC_03_3235893452210176

Proposition 3.13. Let n, m, k be natural numbers. Then

$$n + (m + k) = (n + m) + k.$$

Proof. Define $\Phi = \{k' \in \mathbb{N} \mid n + (m + k') = (n + m) + k'\}$.

(1) 0 is contained in Φ. Indeed $n + (m + 0) = n + m = (n + m) + 0$.

(2) For all $k' \in \Phi$ we have $k' + 1 \in \Phi$.
Proof. Let $k' \in \Phi$. Then $n + (m + k') = (n + m) + k'$. Hence

$$\begin{aligned} & n + (m + (k' + 1)) \\ &= n + ((m + k') + 1) \\ &= (n + (m + k')) + 1 \\ &= ((n + m) + k') + 1 \\ &= (n + m) + (k' + 1). \end{aligned}$$

Thus $k' + 1 \in \Phi$. Qed.

Thus every natural number is an element of Φ. Therefore $n + (m + k) = (n + m) + k$. $\square$

Arithmetics: Proof by induction

FOUNDATIONS_10_1897613305577472

Axiom 10.29 (Choice). Let X be a system of nonempty sets. Then there exists a map f such that $\mathrm{dom}(f) = X$ and $f(x) \in x$ for any $x \in X$.

Foundations: Axiom of choice

SET_THEORY_02_229593678086144

Definition 2.1. An ordinal number is a transitive set α such that every element of α is a transitive set.

Let an ordinal stand for an ordinal number.

SET_THEORY_02_5852994258075648

Definition 2.2. Ord is the class of all ordinals.

Set theory: Definition of ordinal numbers

The LaTeX Workflow

Internal Structuring

Libraries are structured as books with a chapter-structure

→ Chapters can be referenced by their file names:

`set-theory/sections/02_ordinals.ftl.tex`

→ Chapters depend on each other:

[readtex `foundations/sections/11_binary-relations.ftl.tex`]

→ Definitions, theorems etc. can be referenced by unique IDs:

`SET_THEORY_02_229593678086144`

Chapter 2

Ordinal numbers

File: `set-theory/sections/02_ordinals.ftl.tex`

[readtex `foundations/sections/11_binary-relations.ftl.tex`]

[readtex `set-theory/sections/01_transitive-classes.ftl.tex`]

SET_THEORY_02_229593678086144

Definition 2.1. An ordinal number is a transitive set α such that every element of α is a transitive set.

Let an ordinal stand for an ordinal number.

SET_THEORY_02_5852994258075648

Definition 2.2. **Ord** is the class of all ordinals.

SET_THEORY_02_2358097091756032

The LaTeX Workflow

Usage in Other Formalizations

Referencing statements from libraries:

→ Using the LaTeX package `xr`:

```
\usepackage{xr}
```

→ Specifying a library to reference to:

```
\externaldocument{set-theory/set-theory}
```

→ Using the referencing command `\cref{...}`:

```
\cref{SET_THEORY_06_8113916590686208}
```

We have $|F[\kappa_i]| \leq |\kappa_i|$ (by proposition 6.10).

```
\usepackage{xr}
\externaldocument{set-theory/set-theory}

...

We have ... (by \cref{SET_THEORY_06_8113916590686208}).

...
```

Referencing a proposition

SET_THEORY_06_8113916590686208

Proposition 6.10. Let x, y be sets and $f : x \to y$ and $a \subseteq x$. Then $|f[a]| \leq |a|$.

The referenced proposition

The Verifying Workflow

Issue I: Scalability

Library	**Checking time**	**Definitions/theorems/axioms**
Foundations	$\sim$ 10 min.	235
Set Theory	$\sim$ 30 min.	100
Arithmetics	$\sim$ 30 min.	176

$\rightarrow$ **Checking time** does not scale well with the size of a formalization in Naproche

The Verifying Workflow

Issue II: Time redundancy

Naproche rechecks each library whenever it is imported to another formalization.

$\rightarrow$ Annoying **time redundancies**

The Verifying Workflow

Issue III: Modularity

We cannot use different theories in one document.

We cannot use theory morphisms.

We cannot work with instances of theories (i.e. mathematical structures).

$\rightarrow$ ForTheL lacks a proper **module system**

Conclusion & Ideas for Future Work

We have: Both formal *and* human-readable libraries that integrate well in the LaTeX workflow

Current Issues:

Scalability

$\rightarrow$ Extending the scope of provers that ForTheL texts can be checked with (e.g. Isabelle, Lean, ...)

Time redundancy

$\rightarrow$ Persistently storing caching results or proof objects

Modularity

$\rightarrow$ Adding new language features to ForTheL

https://github.com/naproche/naproche