ℕaproche - Talking to ATPs

by Peter Koepke

Mathematical Institute and Hausdorff Center for Mathematics, University of Bonn

EuroProofNet School on Natural Formal Mathematics

Bonn, 2 June 2025

Abstract

Interactive Theorem Proving (ITP) can be seen as a process where a human user steers an Automated Theorem Prover (ATP) to certify proof steps sufficient for the theorem under consideration. Steering is achieved by various languages which are connected by logically correct translation mechanisms. In the ℕaproche system, these languages contain the ordinary language of mathematics, the controlled natural language ForTheL, enriched first-order logic, and the ATP input language TPTP.

UNIVERSITÄT BONN

hcm HAUSDORFF CENTER FOR MATHEMATICS

An example ℕaproche text

A ℕaproche Teaser

Peter Koepke

June 2, 2025

Abstract

This is an introduction to the ℕaproche proof system [1] which accepts and checks readable texts written in a (controlled) natural mathematical language, with natural proof structurings.

Simple introduction of natural numbers and prime numbers

An example $\mathbb{N}$aproche text

Contents

Simple introduction of natural numbers and prime numbers

An example $\mathbb{N}$aproche text

Theorem 52 (Euclids Lemma). Let p be a prime number and $p|m * n$. Then $p|m$ or $p|n$.

Simple introduction of natural numbers and prime numbers

Leading up to Euclid's Lemma

An example $\mathbb{N}$aproche text

7 Prime Numbers

[dump on] Let p, d denote natural numbers.

Let n is nontrivial stand for $n \neq 0$ and $n \neq 1$.

Definition 44. p is prime iff p is nontrivial and for every divisor d of p $d = 1$ or $d = p$.

Let a prime number stand for a natural number that is prime.

Lemma 45. 2 is prime.

Lemma 46. Every even prime number is equal to 2.

Lemma 47. 3 is prime.

Lemma 48. Every nontrivial natural number has a prime divisor.

Proof by induction. □

Simple introduction of natural numbers and prime numbers

Leading up to Euclid's Lemma

Detailed analysis of Lemma 48

Lemma 48

7 Prime Numbers

[dump on] Let p, d denote natural numbers.

Let n is nontrivial stand for $n \neq 0$ and $n \neq 1$.

Definition 44. p is prime iff p is nontrivial and for every divisor d of p $d = 1$ or $d = p$.

Let a prime number stand for a natural number that is prime.

Lemma 45. 2 is prime.

Lemma 46. Every even prime number is equal to 2.

Lemma 47. 3 is prime.

Lemma 48. Every nontrivial natural number has a prime divisor.

Proof by induction. □

Mathematical statement in natural language

Considered as fully formal statement by Naproche

Fully formal material on gray background

Other "literate" material on white background

Lemma 48

7 Prime Numbers

[dump on] Let p, d denote natural numbers.

Let n is nontrivial stand for $n \neq 0$ and $n \neq 1$.

Definition 44. p is prime iff p is nontrivial and for every divisor d of p $d = 1$ or $d = p$.

Let a prime number stand for a natural number that is prime.

Lemma 45. 2 is prime.

Lemma 46. Every even prime number is equal to 2.

Lemma 47. 3 is prime.

Lemma 48. Every nontrivial natural number has a prime divisor.

Proof by induction. □

8 Euclid's Lemma

We need that prime numbers are prime elements in the ring of integers, or the halfring of natural numbers. The following argument is taken over almost verbatim from the Wikipedia article on Euclid's Lemma [6].

Definition 49. m and n are coprime iff every common divisor of m and n is equal to 1.

Lemma 50. If m and m are coprime then $m = 1$.

Let a, b denote natural numbers.

Lemma 51. For all nonzero natural numbers n, a, b if $n|a * b$ and n and a are coprime then n divides b.

Proof by induction on $a * b$.

Let n, a, b be nonzero natural numbers such that $n|a * b$ and n and a are coprime. Take a natural number q such that $n * q = a * b$.

Case $n = a$. Then $n = 1$ and $n|b$. qed.

Case $a > n$. Then $q \geq b$.

$$n * (q - b) = (n * q) - (n * b) = (a * b) - (n * b) = (a - n) * b.$$

Mathematical statement in natural language

Considered as fully formal statement by $\mathbb{N}$aproche

Fully formal material on gray background

Other "literate" material on white background

A mathematical view on the text

7 Prime Numbers

[dump on] Let p, d denote natural numbers.

Let n is nontrivial stand for $n \neq 0$ and $n \neq 1$.

Definition 44. p is prime iff p is nontrivial and for every divisor d of p $d = 1$ or $d = p$.

Let a prime number stand for a natural number that is prime.

Lemma 45. 2 is prime.

Lemma 46. Every even prime number is equal to 2.

Lemma 47. 3 is prime.

Lemma 48. Every nontrivial natural number has a prime divisor.

Proof by induction. □

Textbook-like introduction of prime numbers

- *pretyping of variables* p, d

- *definition of* prime

- prime number *as a linguistic alternative*

- *illustrative lemmas 45-47 whose proofs are "left to the reader"*

- *Lemma 48 is an interesting result with the proof hint "by induction"*

The typesetting view

```
\subsection{Prime Numbers}

\begin{forthel}
[dump on]
Let $p,d$ denote natural numbers.

Let $n$ is nontrivial stand for $n \neq 0$ and $n \neq 1$.

\begin{definition}
$p$ is prime iff $p$ is nontrivial and
for every divisor $d$ of $p$ $d = 1$ or $d = p$.
\end{definition}
Let a prime number stand for a natural number that is prime.

\begin{lemma} $2$ is prime.
\end{lemma}

\begin{lemma}
Every even prime number is equal to $2$.
\end{lemma}

\begin{lemma} $3$ is prime.
\end{lemma}

\begin{lemma}
Every nontrivial natural number has a prime divisor.
\end{lemma}
\begin{proof}[by induction]
%Let $n$ be a nontrivial natural number.
%Assume that $n$ is not prime.
%Take a divisor $m$ of $n$ such that $m \neq 1$ and $m \neq n$.
%$m$ is inductively smaller than $n$.
%Every prime divisor of $m$ is a prime divisor of $n$.
\end{proof}
\end{forthel}
```

Readable output generated from LaTeX by pdfLaTeX

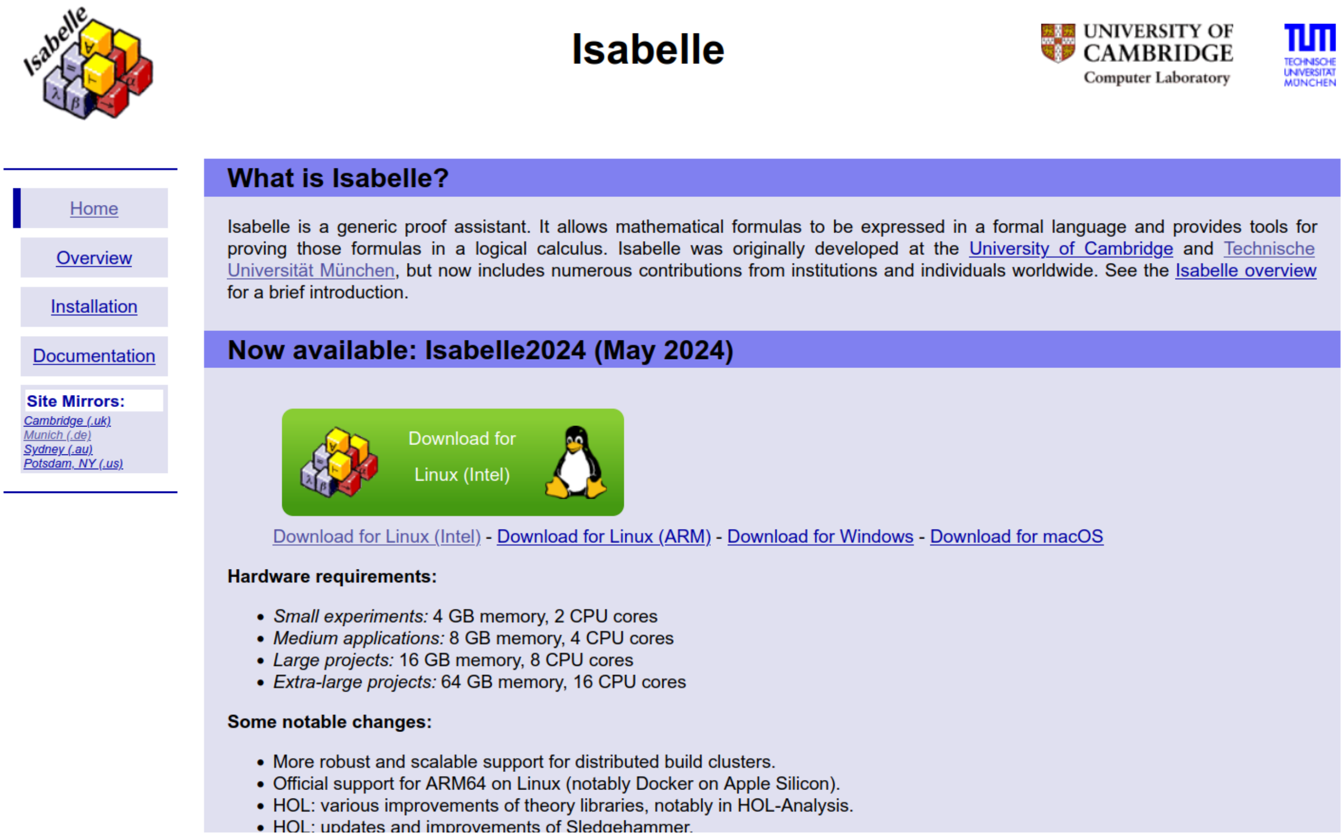
- simple LaTeX

- forthel environments for strictly formal text

- ordinary LaTeX environments for definitions, lemmas and proofs

- LaTeX file (`...ftl.org`) is the input to the Naproche *system*

Working with ℕaproche documents in Isabelle



Isabelle

Home

Overview

Installation

Documentation

Site Mirrors:

Cambridge (.uk)
Munich (.de)
Sydney (.au)
Potsdam, NY (.us)

What is Isabelle?

Isabelle is a generic proof assistant. It allows mathematical formulas to be expressed in a formal language and provides tools for proving those formulas in a logical calculus. Isabelle was originally developed at the University of Cambridge and Technische Universität München, but now includes numerous contributions from institutions and individuals worldwide. See the Isabelle overview for a brief introduction.

Now available: Isabelle2024 (May 2024)

Download for Linux (Intel)

Download for Linux (Intel) - Download for Linux (ARM) - Download for Windows - Download for macOS

Hardware requirements:

- *Small experiments:* 4 GB memory, 2 CPU cores
- *Medium applications:* 8 GB memory, 4 CPU cores
- *Large projects:* 16 GB memory, 8 CPU cores
- *Extra-large projects:* 64 GB memory, 16 CPU cores

Some notable changes:

- More robust and scalable support for distributed build clusters.
- Official support for ARM64 on Linux (notably Docker on Apple Silicon).
- HOL: various improvements of theory libraries, notably in HOL-Analysis.
- HOL: updates and improvements of Sledgehammer.

Working with ℕaproche documents in Isabelle2024

Isabelle

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

TECHNISCHE UNIVERSITÄT MÜNCHEN

- Home
- Overview
- Installation
- Documentation

Site Mirrors:
Cambridge (.uk)
Munich (.de)
Sydney (.au)
Potsdam, NY (.us)

What is Isabelle?

Isabelle is a generic proof assistant. It allows mathematical formulas to be expressed in a formal language and provides tools for proving those formulas in a logical calculus. Isabelle was originally developed at the University of Cambridge and Technische Universität München, but now includes numerous contributions from institutions and individuals worldwide. See the Isabelle overview for a brief introduction.

Now available: Isabelle2024 (May 2024)

Download for Linux (Intel)

Download for Linux (Intel) - Download for Linux (ARM) - Download for Windows - Download for macOS

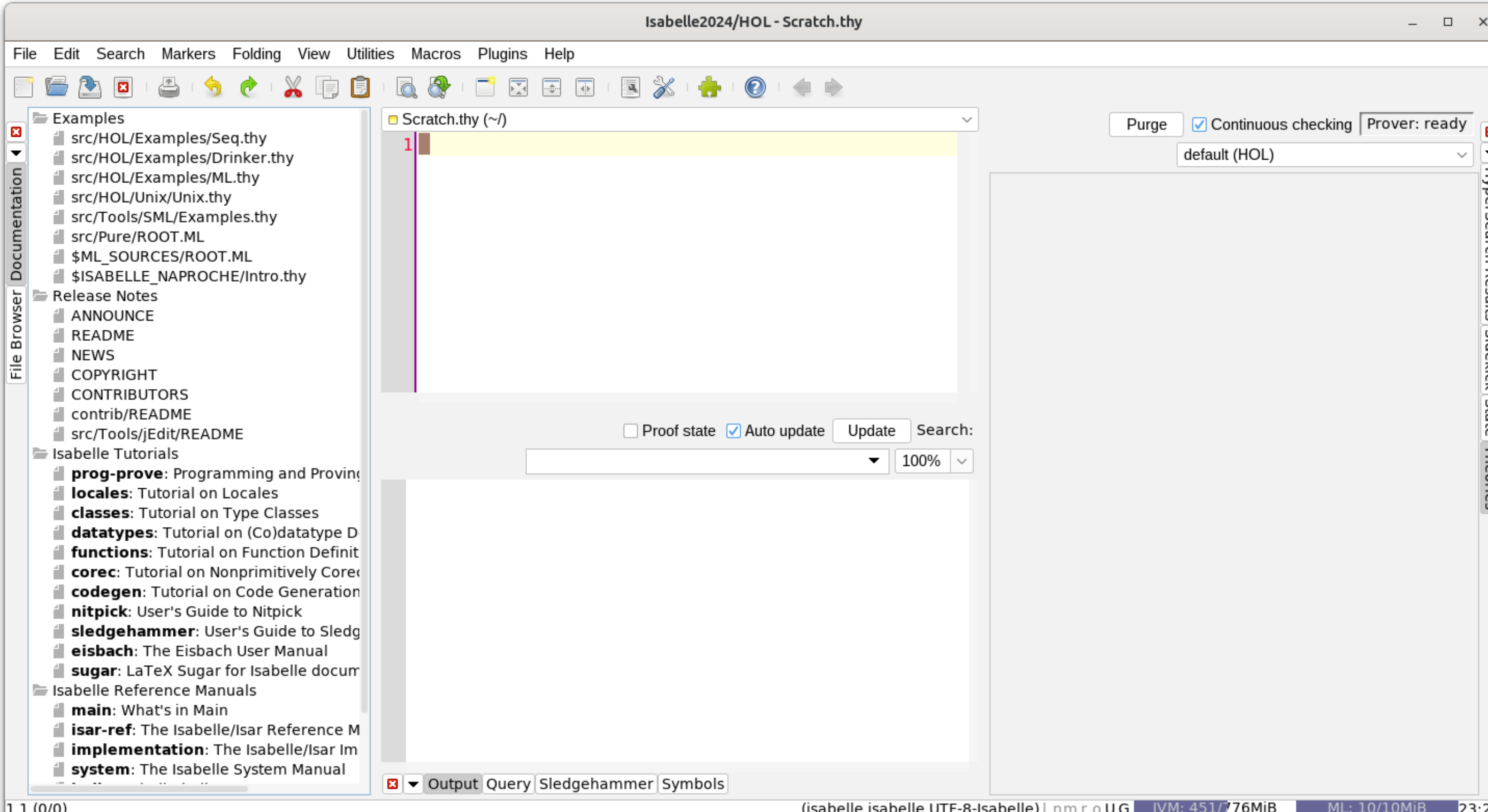
Hardware requirements:

- *Small experiments:* 4 GB memory, 2 CPU cores
- *Medium applications:* 8 GB memory, 4 CPU cores
- *Large projects:* 16 GB memory, 8 CPU cores
- *Extra-large projects:* 64 GB memory, 16 CPU cores

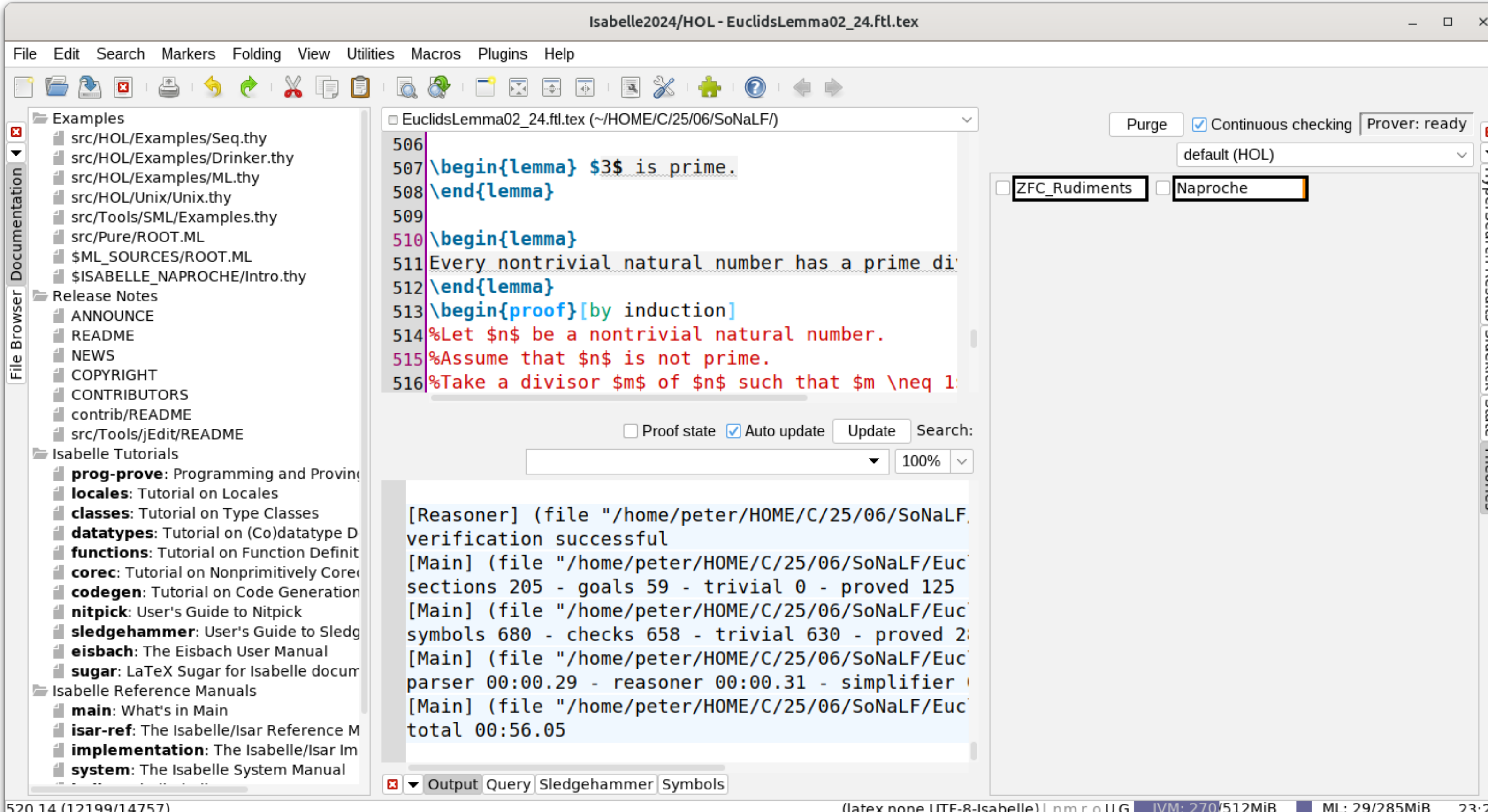
Some notable changes:

- More robust and scalable support for distributed build clusters.
- Official support for ARM64 on Linux (notably Docker on Apple Silicon).
- HOL: various improvements of theory libraries, notably in HOL-Analysis.
- HOL: updates and improvements of Sledgehammer.

Working with ℕaproche documents in Isabelle



Working with $\mathbb{N}$aproche documents in Isabelle



The proof-checking view

7 Prime Numbers

[dump on] Let p, d denote natural numbers.

Let n is nontrivial stand for $n \neq 0$ and $n \neq 1$.

Definition 44. p is prime iff p is nontrivial and for every divisor d of p $d = 1$ or $d = p$.

Let a prime number stand for a natural number that is prime.

Lemma 45. 2 is prime.

Lemma 46. Every even prime number is equal to 2.

Lemma 47. 3 is prime.

Lemma 48. Every nontrivial natural number has a prime divisor.

Proof by induction. □

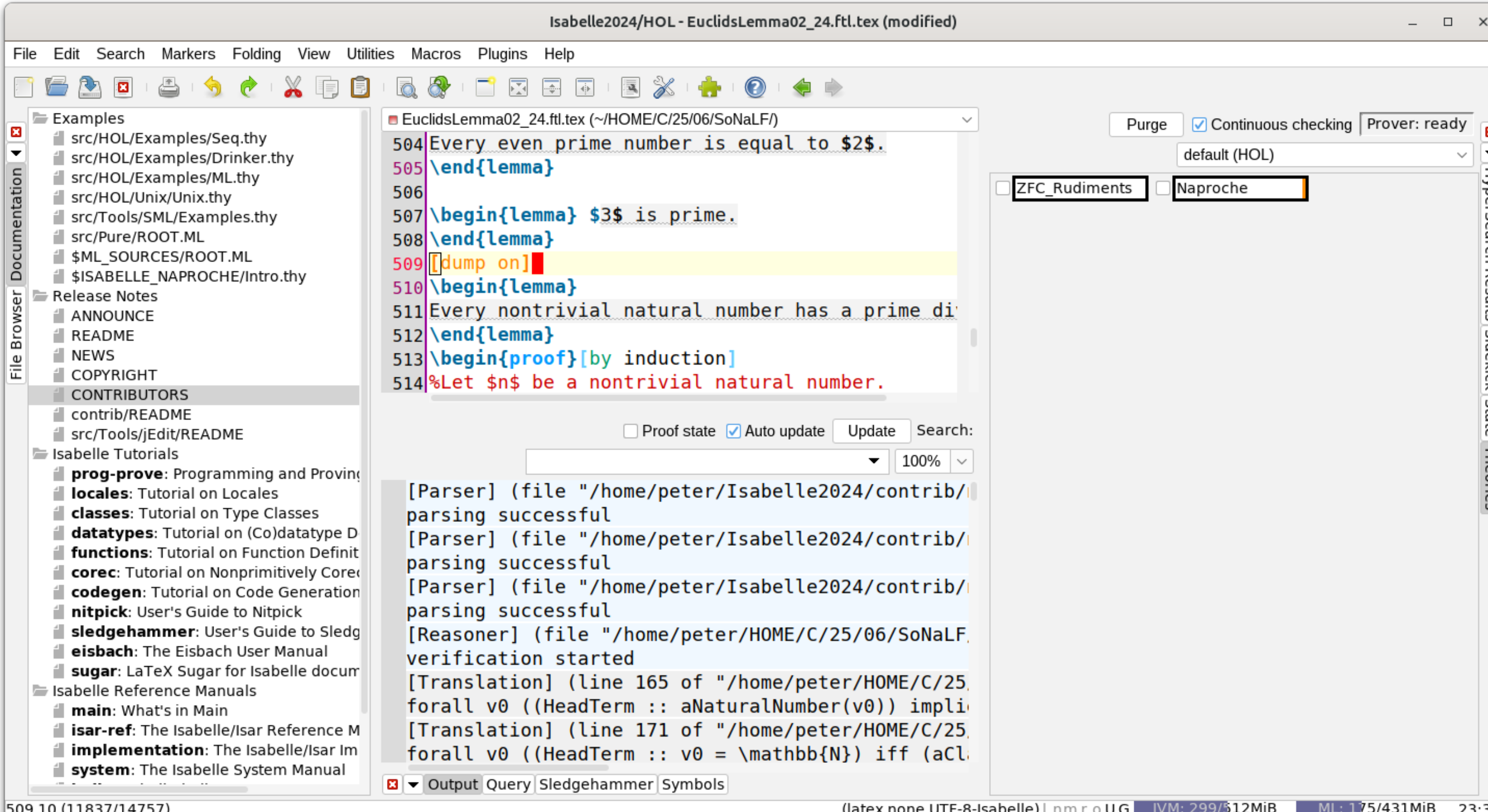
Textbook-like introduction of prime numbers

- Lemma 48 has a short proof "by induction"

- Proof is carried out by the E *Automated Theorem Prover (ATP)*

- What is the prover task given to E*?*

Inspecting the interaction of ℕaproche and E in Isabelle: *dump on*



First-order translation and translation into TPTP prover format

```
isPrime(3)
[Reasoner] (line 507 of "/home/peter/HOME/C/25/06/SoNaLF/EuclidsLemma02_24.ftl.tex")
goal:  3 is prime.
[Translation] (line 511 of "/home/peter/HOME/C/25/06/SoNaLF/EuclidsLemma02_24.ftl.tex")
forall v0 ((aNaturalNumber(v0) and (not v0 = 0 and not v0 = 1)) implies ((InductionHypothesis :: forall v1 ((aNaturalNumber(v1) and (not v1 = 0 and not v1 = 1)) implies (iLess(v1,v0) implies exists v2 ((aNaturalNumber(v2) and doDivides(v2,v1)) and isPrime(v2))))) implies exists v1 ((aNaturalNumber(v1) and doDivides(v1,v0)) and isPrime(v1))))
[Reasoner] (line 511 of "/home/peter/HOME/C/25/06/SoNaLF/EuclidsLemma02_24.ftl.tex")
goal:  Every nontrivial natural number has a prime divisor.
[Main] (line 511 of "/home/peter/HOME/C/25/06/SoNaLF/EuclidsLemma02_24.ftl.tex")
fof(m_,hypothesis,( ! [W0] : ( ! [W1] : ((aObject(W0) & aObject(W1)) => aObject(mkPair(W0,W1)))))).
fof(m_,hypothesis,( ! [W0] : ( ! [W1] : ((aMap(W0) & aElementOf(W1,mkDom(W0))) => aObject(mkApp(W0,W1)))))).
fof(m_,hypothesis,( ! [W0] : ( ! [W1] : ((aClass(W0) & aElementOf(W1,W0)) => aObject(W1))))).
fof(m_,hypothesis,( ! [W0] : (aMap(W0) => aClass(mkDom(W0))))).
fof(m_,hypothesis,( ! [W0] : (aFunction(W0) <=> (aMap(W0) & aObject(W0))))).
fof(m_,hypothesis,( ! [W0] : (aSet(W0) <=> (aClass(W0) & aObject(W0))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => aObject(W0)))).
fof(m_,hypothesis,((aClass(sbszmzaztzhzbzblczNrc) & ( ! [W0] : (aElementOf(W0,sbszmzaztzhzbzblczNrc) <=> (aNaturalNumber(W0) & aObject(W0))))) & ( ! [W0] : ((aClass(W0) & ( ! [W1] : (aElementOf(W1,W0) <=> (aNaturalNumber(W1) & aObject(W1))))) => (W0 = sbszmzaztzhzbzblczNrc))))).
fof(maxiomofinfinity,hypothesis,aSet(sbszmzaztzhzbzblczNrc)).
fof(m_,hypothesis,aNaturalNumber(sz0)).
fof(m_,hypothesis,(aNaturalNumber(sz1) & ( ~ (sz1 = sz0)))).
fof(m_,hypothesis,( ! [W0] : ( ! [W1] : ((aNaturalNumber(W0) & aNaturalNumber(W1)) => aNaturalNumber(sdtpldt(W0,W1)))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => ((aNaturalNumber(W0) & ( ~ (W0 = sz0))) => ( ? [W1] : (aNaturalNumber(W1) & (W0 = sdtpldt(W1,sz1)))))))).
```

First-order translation and translation into TPTP prover format

Öffnen • Parser file home peter — Entwurf — Z 4894, Sp 15

6 4 DUS Toronto DUS DUB.txt | notes2.txt | • ideas.txt | Neues Dokument | • eprover 1 sec | • Parser file home pete

511 — 0 von 4

```
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (( ~ (W0 = sz0)) => ( ! [W1] :
sdtbszlzezqdt(W1,sdtasdt(W1,W0)))))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (sdtbszlzezqdt(W0,sz0) => (W0 = sz0))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (sdtbszlzezqdt(W0,sz1) => ((W0 = sz0) | (W0 = sz1)))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (sdtbszlzezqdt(W0,sz2) => (((W0 = sz0) | (W0 = sz1)) | (W0 = sz2)))))).
fof(m_,hypothesis,(((sdtbszlzezqdt(sz0,sz1) & ( ~ (sz0 = sz1))) & (sdtbszlzezqdt(sz1,sz2) & ( ~ (sz1 = sz2)))) & (sdtbszlzezqdt(sz2,sz3) & ( ~ (sz2 = sz3))))).
fof(m_,hypothesis,( ! [W0] : ( ! [W1] : ((aNaturalNumber(W0) & aNaturalNumber(W1)) => (doDivides(W1,W0) <=> ( ? [W2] : (aNaturalNumber(W2) & (W0 = sdtasdt(W1,W2))))))))).
fof(m_,hypothesis,( ! [W0] : ( ! [W1] : ( ! [W2] : (((aNaturalNumber(W0) & aNaturalNumber(W1)) & aNaturalNumber(W2)) => ((doDivides(W0,W1) & doDivides(W1,W2)) => doDivides(W0,W2))))))).
fof(m_,hypothesis,( ! [W0] : ( ! [W1] : ( ! [W2] : (((aNaturalNumber(W0) & aNaturalNumber(W1)) & aNaturalNumber(W2)) => ((doDivides(W0,W1) & doDivides(W0,sdtpldt(W1,W2))) => doDivides(W0,W2))))))).
fof(m_,hypothesis,( ! [W0] : ( ! [W1] : ((aNaturalNumber(W0) & aNaturalNumber(W1)) => ((doDivides(W0,W1) & ( ~ (W1 = sz0))) => sdtbszlzezqdt(W0,W1)))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (isEven(W0) <=> doDivides(sz2,W0))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (isEven(W0) | ( ? [W1] : (aNaturalNumber(W1) & (W0 = sdtpldt(sdtasdt(sz2,W1),sz1)))))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (isPrime(W0) <=> ((( ~ (W0 = sz0)) & ( ~ (W0 = sz1))) & ( ! [W1] : ((aNaturalNumber(W1) & doDivides(W1,W0)) => ((W1 = sz1) | (W1 = W0))))))))).
fof(m_,hypothesis,isPrime(sz2)).
fof(m_,hypothesis,( ! [W0] : (((aNaturalNumber(W0) & isPrime(W0)) & isEven(W0)) => (W0 = sz2)))).
fof(m_,hypothesis,isPrime(sz3)).
fof(m__,conjecture,( ! [W0] : ((aNaturalNumber(W0) & (( ~ (W0 = sz0)) & ( ~ (W0 = sz1)))) => (( ! [W1] : ((aNaturalNumber(W1) & (( ~ (W1 = sz0)) & ( ~ (W1 = sz1)))) => (iLess(W1,W0) => ( ? [W2] : ((aNaturalNumber(W2) & doDivides(W2,W1)) & isPrime(W2)))))) => ( ? [W1] : ((aNaturalNumber(W1) & doDivides(W1,W0)) & isPrime(W1))))))).
```

$\mathbb{N}$aproche's input to E

```
...

fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (isPrime(W0)
<=> ((( ~ (W0 = sz0)) & ( ~ (W0 = sz1))) & ( ! [W1] :
((aNaturalNumber(W1) & doDivides(W1,W0)) => ((W1 = sz1) | (W1 =
W0))))))))).

fof(m_,hypothesis,isPrime(sz2)).

fof(m_,hypothesis,( ! [W0] : (((aNaturalNumber(W0) & isPrime(W0))
& isEven(W0)) => (W0 = sz2)))).

fof(m_,hypothesis,isPrime(sz3)).

fof(m__,conjecture,( ! [W0] : ((aNaturalNumber(W0) & (( ~ (W0 =
sz0)) & ( ~ (W0 = sz1)))) => (( ! [W1] : ((aNaturalNumber(W1) &
(( ~ (W1 = sz0)) & ( ~ (W1 = sz1)))) => (iLess(W1,W0) => ( ? [W2]
: ((aNaturalNumber(W2) & doDivides(W2,W1)) & isPrime(W2))))))
=> ( ? [W1] : ((aNaturalNumber(W1) & doDivides(W1,W0)) &
isPrime(W1))))))).
```

"original"

Definition 1. *p is prime iff p is nontrivial and for every divisor d of p $d = 1$ or $d = p$.*

Lemma 2. *2 is prime.*

Lemma 3. *Every even prime number is equal to 2.*

Lemma 4. *3 is prime.*

Lemma 5. *Every nontrivial n has a prime divisor.*

The “by induction” proof tactic

0.5 Induction

ℕaproche provides an in-built binary relation symbol $\prec$ as a universal inductive relation: if

> (inheritance property) at any point m property P holds at m provided all $\prec$-predecessors of m satisfy P

then

> P holds everywhere.

ℕaproche has a proof tactic ”by induction [on ...]”, which reduces the inductive proof goal ”P holds everywhere” to proving the inheritance property for P.

Initially, there is no specification of $\prec$. The induction proof method for some concrete relation is made available by embedding that relation into $\prec$. Therefore we axiomatically embed the natural order into $\prec$.

Axiom 30. If $m < n$ then $m \prec n$.

Let m is inductively smaller than n stand for $m \prec n$.

Induction in TPTP: iLess as $\prec$

```
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (( ~ (W0 = sz0)) => ( ! [W1] :
sdtbszlzezqdt(W1,sdtasdt(W1,W0)))))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (sdtbszlzezqdt(W0,sz0) => (W0 = sz0))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (sdtbszlzezqdt(W0,sz1) => ((W0 = sz0) | (W0 = sz1)))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (sdtbszlzezqdt(W0,sz2) => (((W0 = sz0) | (W0 = sz1)) | (W0 = sz2)))))).
fof(m_,hypothesis,(((sdtbszlzezqdt(sz0,sz1) & ( ~ (sz0 = sz1))) & (sdtbszlzezqdt(sz1,sz2) & ( ~ (sz1 = sz2)))) &
(sdtbszlzezqdt(sz2,sz3) & ( ~ (sz2 = sz3))))).
fof(m_,hypothesis,( ! [W0] : ( ! [W1] : ((aNaturalNumber(W0) & aNaturalNumber(W1)) => (doDivides(W1,W0) <=> ( ? [W2] :
(aNaturalNumber(W2) & (W0 = sdtasdt(W1,W2))))))))).
fof(m_,hypothesis,( ! [W0] : ( ! [W1] : ( ! [W2] : (((aNaturalNumber(W0) & aNaturalNumber(W1)) & aNaturalNumber(W2)) =>
((doDivides(W0,W1) & doDivides(W1,W2)) => doDivides(W0,W2))))))).
fof(m_,hypothesis,( ! [W0] : ( ! [W1] : ( ! [W2] : (((aNaturalNumber(W0) & aNaturalNumber(W1)) & aNaturalNumber(W2)) =>
((doDivides(W0,W1) & doDivides(W0,sdtpldt(W1,W2))) => doDivides(W0,W2))))))).
fof(m_,hypothesis,( ! [W0] : ( ! [W1] : ((aNaturalNumber(W0) & aNaturalNumber(W1)) => ((doDivides(W0,W1) & ( ~ (W1 = sz0))) =>
sdtbszlzezqdt(W0,W1)))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (isEven(W0) <=> doDivides(sz2,W0))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (isEven(W0) | ( ? [W1] : (aNaturalNumber(W1) & (W0 =
sdtpldt(sdtasdt(sz2,W1),sz1)))))))).
fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (isPrime(W0) <=> ((( ~ (W0 = sz0)) & ( ~ (W0 = sz1))) & ( ! [W1] :
((aNaturalNumber(W1) & doDivides(W1,W0)) => ((W1 = sz1) | (W1 = W0))))))))).
fof(m_,hypothesis,isPrime(sz2)).
fof(m_,hypothesis,( ! [W0] : (((aNaturalNumber(W0) & isPrime(W0)) & isEven(W0)) => (W0 = sz2)))).
fof(m_,hypothesis,isPrime(sz3)).
fof(m__,conjecture,( ! [W0] : ((aNaturalNumber(W0) & (( ~ (W0 = sz0)) & ( ~ (W0 = sz1)))) => (( ! [W1] : ((aNaturalNumber(W1)
& (( ~ (W1 = sz0)) & ( ~ (W1 = sz1)))) => (iLess(W1,W0) => ( ? [W2] : ((aNaturalNumber(W2) & doDivides(W2,W1)) &
isPrime(W2)))))) => ( ? [W1] : ((aNaturalNumber(W1) & doDivides(W1,W0)) & isPrime(W1))))))).
```

Naproche's input to E

```
...

fof(m_,hypothesis,( ! [W0] : (aNaturalNumber(W0) => (isPrime(W0)
<=> ((( ~ (W0 = sz0)) & ( ~ (W0 = sz1))) & ( ! [W1] :
((aNaturalNumber(W1) & doDivides(W1,W0)) => ((W1 = sz1) | (W1 =
W0))))))))).

fof(m_,hypothesis,isPrime(sz2)).

fof(m_,hypothesis,( ! [W0] : (((aNaturalNumber(W0) & isPrime(W0))
& isEven(W0)) => (W0 = sz2)))).

fof(m_,hypothesis,isPrime(sz3)).

fof(m__,conjecture,( ! [W0] : ((aNaturalNumber(W0) & (( ~ (W0 =
sz0)) & ( ~ (W0 = sz1)))) => (( ! [W1] : ((aNaturalNumber(W1) &
(( ~ (W1 = sz0)) & ( ~ (W1 = sz1)))) => (iLess(W1,W0) => ( ? [W2]
: ((aNaturalNumber(W2) & doDivides(W2,W1)) & isPrime(W2))))))
=> ( ? [W1] : ((aNaturalNumber(W1) & doDivides(W1,W0)) &
isPrime(W1))))))).
```

"original"

Definition 6. *p is prime iff p is nontrivial and for every divisor d of p $d = 1$ or $d = p$.*

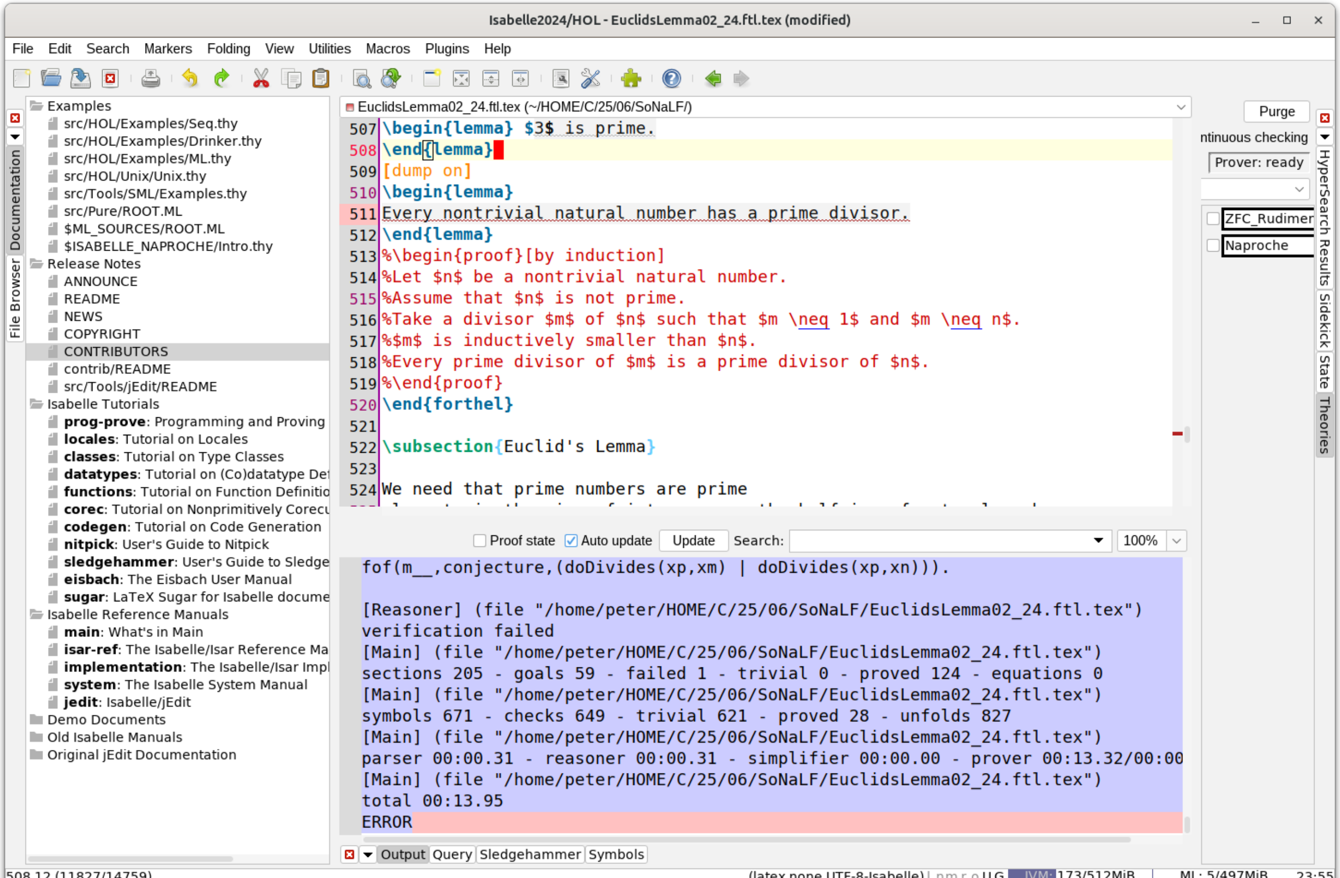
Lemma 7. *2 is prime.*

Lemma 8. *Every even prime number is equal to 2.*

Lemma 9. *3 is prime.*

Lemma 10. *Every nontrivial n has a prime divisor.*

E fails without "by induction"; some statistics



The linguistic view: analyzing mathematical language

7 Prime Numbers

[dump on] Let p, d denote natural numbers.

Let n is nontrivial stand for $n \neq 0$ and $n \neq 1$.

Definition 44. p is prime iff p is nontrivial and for every divisor d of p $d = 1$ or $d = p$.

Let a prime number stand for a natural number that is prime.

Lemma 45. 2 is prime.

Lemma 46. Every even prime number is equal to 2.

Lemma 47. 3 is prime.

Lemma 48. Every nontrivial natural number has a prime divisor.

Proof by induction. □

- *simple (argumentative) sentences with symbolic material*
- *LaTeX conventions*
- *grammatical analysis*
- *parsing*
- *softly typed language*
- *translating into first-order logic*

Phrase structure grammar

every nontrivial natural number has a prime divisor

statement $\longrightarrow$ *subject predicate*

subject $\longrightarrow$ `every nontrivial natural number`

predicate $\longrightarrow$ `has a prime divisor`

The syntax and semantics of the ForTheL language*

Andrei Paskevich
Université Paris XII — Val de Marne, Créteil, France
Kiev National Taras Shevchenko University, Kiev, Ukraine

December 2007

Contents

A simplified fragment of the ForTheL phrase structure grammar

`every nontrivial natural number has a prime divisor`

$simpleStatement \longrightarrow terms\ doesPredicate\ \{\texttt{and}\ doesPredicate\}$

$terms \longrightarrow term\ \{(\texttt{,} \mid \texttt{and})\ term\}$

$term \longrightarrow quantifiedNotion \mid definiteTerm$

$quantifiedNotion \longrightarrow (\texttt{every} \mid \texttt{each} \mid \texttt{all} \mid \texttt{any})\ notion$

$notion \longrightarrow classNoun \mid \ldots$

$classNoun \longrightarrow \{leftAttribute\}\ primClassNoun\ [rightAttribute]$

$primClassNoun \longrightarrow \texttt{natural number}$

$doesPredicate \longrightarrow (\texttt{has} \mid \texttt{have})\ hasPredicate$

$hasPredicate \longrightarrow [\ \texttt{a} \mid \texttt{an} \mid \texttt{the}\ ]\ possessedNoun\ \{\texttt{and}\ [\texttt{a} \mid \texttt{an} \mid \texttt{the}]\ possessedNoun\} \mid \texttt{no}\ possessedNoun$

$possessedNoun \longrightarrow \{leftAttribute\}\ primPossessedNoun$

$primPossessedNoun \longrightarrow (\texttt{divisor} \mid \texttt{divisors})\ [names]$ —— derived from the phrase "`divisor of`"

$leftAttribute \longrightarrow \texttt{nontrivial} \mid \texttt{prime} \mid \ldots$

The ForTheL phrase structure grammar is implemented in $\mathbb{N}$aproche

```
every nontrivial natural number has a prime divisor
```

simpleStatement → *terms doesPredicate* {and *doesPredicate*}

```
simple :: FTL Formula
simple = label "simple statement" $ do
  (q, ts) <- terms
  p  <- conjChain doesPredicate
  q <$> dig p ts
...
```

doesPredicate → (has | have) *hasPredicate*

```
doesPredicate :: FTL Formula
doesPredicate = label "does predicate" $
  ( . . .  <|> hasP <|>  . . .
  where
     . . .
    hasP = has >> hasPredicate
     . . .
```

0.8 Euclid's Lemma

We need that prime numbers are prime elements in the ring of integers, or the halfring of natural numbers. The following argument is taken over almost verbatim from the Wikipedia article on Euclid's Lemma [6].

Definition 49. m and n are coprime iff every common divisor of m and n is equal to 1.

Lemma 50. If m and m are coprime then $m = 1$.

Let a, b denote natural numbers.

Lemma 51. For all nonzero natural numbers n, a, b if $n|a * b$ and n and a are coprime then n divides b.

Proof by induction on $a * b$.

Let n, a, b be nonzero natural numbers such that $n|a*b$ and n and a are coprime. Take a natural number q such that $n * q = a * b$.

Case $n = a$. Then $n = 1$ and $n|b$. qed.

Case $a > n$. Then $q \geq b$.

$$n * (q - b) = (n * q) - (n * b) = (a * b) - (n * b) = (a - n) * b.$$

Thus n divides $(a-n)*b$. n and $a-n$ are coprime. $(a-n)*b < a*b$. $(a - n) * b$ is inductively smaller than $a * b$. Thus n divides b. qed.

Hence $n > a$ and $b \geq q$.

$$(n - a) * q = (n * q) - (a * q) = (a * b) - (a * q) = a * (b - q).$$

0.8 Euclid's Lemma

We need that prime numbers are prime elements in the ring of integers, or the halfring of natural numbers. The following argument is taken over almost verbatim from the Wikipedia article on Euclid's Lemma [6].

Definition 49. m and n are coprime iff every common divisor of m and n is equal to 1.

Lemma 50. If m and m are coprime then $m = 1$.

Let a, b denote natural numbers.

Lemma 51. For all nonzero natural numbers n, a, b if $n|a * b$ and n and a are coprime then n divides b.

Proof by induction on $a * b$.

Let n, a, b be nonzero natural numbers such that $n|a*b$ and n and a are coprime. Take a natural number q such that $n * q = a * b$.

Case $n = a$. Then $n = 1$ and $n|b$. qed.

Case $a > n$. Then $q \geq b$.

$$n * (q - b) = (n * q) - (n * b) = (a * b) - (n * b) = (a - n) * b.$$

Thus n divides $(a-n)*b$. n and $a-n$ are coprime. $(a-n)*b < a*b$. $(a-n)*b$ is inductively smaller than $a*b$. Thus n divides b. qed.

Hence $n > a$ and $b \geq q$.

$$(n - a) * q = (n * q) - (a * q) = (a * b) - (a * q) = a * (b - q).$$

By induction [edit]

The following proof is inspired by Euclid's version of Euclidean algorithm, which proceeds by using only subtractions.

Suppose that $n \mid ab$ and that n and a are coprime (that is, their greatest common divisor is 1). One has to prove that n divides b. Since $n \mid ab$, there exists an integer q such that

$$nq = ab.$$

Without loss of generality, one can suppose that n, q, a, and b are positive, since the divisibility relation is independent of the signs of the involved integers.

To prove the theorem by strong induction, we suppose that it has been proved for all smaller values of ab. There are three cases:

1. If $n = a$, coprimality implies $n = 1$, and n divides b trivially.
2. If $n < a$, then subtracting nb from both sides gives

   $$n(q - b) = (a - n)b.$$

   Thus, n divides $(a - n)\,b$. Since we assumed that n and a are coprime, it follows that $a - n$ and n must be coprime. (If not, their greatest common divisor d would divide their sum a as well as n, contradicting our assumption.)

   The conclusion therefore follows by induction hypothesis, since $0 < (a - n)\,b < ab$.
3. If $n > a$ then subtracting aq from both sides gives

$\mathbb{N}$aproche texts can be readable like textbook mathematics:

Definition 49. m and n are coprime iff every common divisor of m and n is equal to 1.

Lemma 50. If m and m are coprime then $m = 1$.

Let a, b denote natural numbers.

Lemma 51. For all nonzero natural numbers n, a, b if $n|a * b$ and n and a are coprime then n divides b.

Proof by induction on $a * b$.

Let n, a, b be nonzero natural numbers such that $n|a*b$ and n and a are coprime. Take a natural number q such that $n * q = a * b$.

Case $n = a$. Then $n = 1$ and $n|b$. qed.

Case $a > n$. Then $q \geq b$.

$$n * (q - b) = (n * q) - (n * b) = (a * b) - (n * b) = (a - n) * b.$$

Thus n divides $(a-n)*b$. n and $a-n$ are coprime. $(a-n)*b < a*b$. $(a - n) * b$ is inductively smaller than $a * b$. Thus n divides b. qed.

Hence $n > a$ and $b \geq q$.

$$(n - a) * q = (n * q) - (a * q) = (a * b) - (a * q) = a * (b - q).$$

Perspectives

- to increase the coverage of $\mathbb{N}$aproche …
- to build a more efficient $\mathbb{N}$aproche on a set-theoretical basis (Adrian De Lon) …
- to transfer the $\mathbb{N}$aproche approach to other proof systems …
- to use LLMs for language translation and other language processing …

Thank you!

https://naproche.github.io/

https://isabelle.in.tum.de/website-Isabelle2024/