# AutoFormalization: Past, Present, Surprises

x

Josef Urban

Czech Technical University in Prague

July 10, 2024, Bonn
HIM Trimester “Prospects of Formal Math”

European Research Council
Established by the European Commission

# Brief History of Related Ideas (incomplete?)

- 1929/34 Jaskowski: “On the rules of supposition” (“natural proofs”)
- 1962 McCarthy: “Checking mathematical proofs is potentially one of the most interesting and useful applications of automatic computers”
- 1963: P. Abrahams: “Machine verification of mathematical proofs”
- “checking a textbook proof would require much more” (PhD at MIT)
- 60’s/70s: SAD and Mizar - “human-friendly” (controlled natural?) formal languages
- 1990 D. Simon: “Checking Natural Language Proofs” (Phd, Texas)
- 1993 L. Lamport. “How to write proofs” (structured proofs)
- 1990s-... A. Ranta: Grammatical Framework (GF)
- 2002 M. Wenzel: Isabelle, Isar (Phd - “structured” proof docs)
- 2003 C. Zinn: “Understanding informal mathematical discourse.” (Phd, discourse representation theory, ATPs, manual parsing)
- 2000-8 A. Paskevitch: ForTheL/SAD
- 2005-... P. Koepke et al: Naproche
- 2008-2013 M. Ganesalingam: The Language of Mathematics
- ... more

# How it started for me

Date: Wed, 5 May 2004 18:45:37 +0200 (CEST)
From: Josef Urban <urban@ktilinux.ms.mff.cuni.cz>
To: <keimel@mathematik.tu-darmstadt.de>
cc: <trybulec@math.uwb.edu.pl>
Subject: Compendium of Continuous Lattices

Dear prof. Keimel,
[...]
recently I have linked the Mizar system with the modern ATPs.

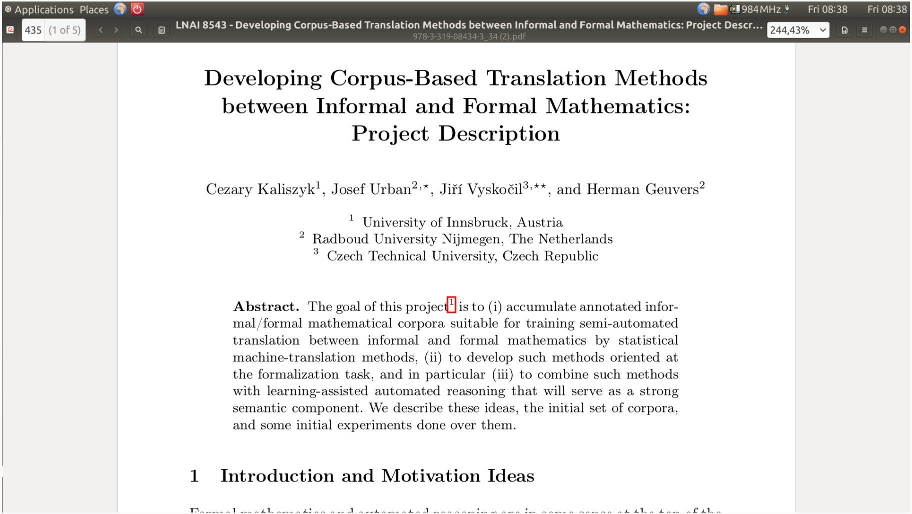
I am also interested in translating mathematical texts written in
natural language (i.e. TeX or Latex) to the formalized Mizar language.

I hope that the link with theorem provers could be used
as an additional semantic filter for the natural language parsers.

I am writing to you, because you are one of the authors of the
"Compendium of Continuous Lattices", which has been from a large
part formalized in Mizar, and therefore it could be very suitable
for such experiment with automated translating.

[...later email...]
I should note that this is considered to be quite a hard task, and it is

# 2014 Learning-Assisted Autoformalization Declaration




## Developing Corpus-Based Translation Methods between Informal and Formal Mathematics: Project Description

Cezary Kaliszyk[1], Josef Urban[2,⋆], Jiří Vyskočil[3,⋆⋆], and Herman Geuvers[2]

[1] University of Innsbruck, Austria
[2] Radboud University Nijmegen, The Netherlands
[3] Czech Technical University, Czech Republic

**Abstract.** The goal of this project[1] is to (i) accumulate annotated informal/formal mathematical corpora suitable for training semi-automated translation between informal and formal mathematics by statistical machine-translation methods, (ii) to develop such methods oriented at the formalization task, and in particular (iii) to combine such methods with learning-assisted automated reasoning that will serve as a strong semantic component. We describe these ideas, the initial set of corpora, and some initial experiments done over them.

### 1 Introduction and Motivation Ideas

- CICM, July 10, 2014, exactly 10 years ago (I planned none of these dates ;-)
- `cicm-conference.org/2014/cicm.php?event=&menu=schedule-thursday`

# 3-4 Corners of NLP/Translation Methods in Formalization

- Informal to formal: Translating natural language to formal proofs
- Controlled natural language: A middle ground between informal and formal
- Formal to formal: Translating between different formal systems
- Alignments: Fully manual (annotations), Gauthier & Kaliszyk - much more semantic?, Just neural?
- Hybrid approaches: LMs for rephrasing, formal tools for verification, PCFGs

# Personal Surprises in Autoformalization

- Thibault's automated alignments working across HOL4/Light/Isabelle
- Effectiveness of PCFG settings in conjecturing provable statements
- Impact of different neural architectures and settings (e.g., attention)
- Success of back-translation methods on unsupervised corpora (large scale collaborative project a la blueprint?)
- `https://github.com/JUrban/extract-defs`
- Jesse Han's work on fine-tuning through few-shot prompting
- Potential of smaller/smarter architectures (e.g., GNNs) for terminology invention
- Limits vs potential of today's large LMs (see Wenda/Moa's talks?)
- GPT-2's perfect Mizar grammar/proof mastery vs non-mastery of harder tasks

# Our Autoformalization Attemps

- Goal: Learn understanding of informal math formulas and reasoning
- Experiments with the CYK chart parser linked to semantic methods
- demo: `http://grid01.ciirc.cvut.cz/~mptp/demo.ogv`
- Experiments with neural methods
- Combined with semantic methods: Type checking, theorem proving
- Feedback loops between the learning and the semantic methods
- Math is a much nicer area than unrestricted NLP:
- We (believe we) can express informal math formally, prove things, etc.
- If we achieve grounding math, we might ground scientific texts, law, etc.
- Early Corpora: Flyspeck, Mizar, Proofwiki
- Today: anything is a fair game (Isabelle, Lean, Coq, Metamtah, Stacks, Arxiv)

# AITP Challenges/Bets from 2014

- 3 AITP bets for 10k EUR from my 2014 talk at Institut Henri Poincare (`tinyurl.com/yb55b3jv`)
  - In 20 years, 80% of Mizar and Flyspeck toplevel theorems will be provable automatically (same hardware, same libraries as in 2014 - about 40% then)
  - In 10 years: 60% (DONE already in 2021 - 3 years ahead of schedule)
  - In 25 years, 50% of the toplevel statements in LaTeX-written Msc-level math curriculum textbooks will be parsed automatically and with correct formal semantics (this may be faster than I expected)

# PCFG

- PCFG demo: `http://grid01.ciirc.cvut.cz/~mptp/demo.ogv`
  - Selection of 1k-5k nearest neighbors (RAG?)
  - Building PCFG parser from selected examples
  - Alternative to pure reliance on LLMs
- PCFG as an alternative to black-box models?
- Future: Learning non-trivial tree transformations:
  - Probabilistic methods
  - Evolutionary algorithms
  - Neural approaches (inspired by our OEIS work)
- Bridging gap between Grammatical Framework (GF) and black-box models
- Potential for elaboration in transformation process
- Exploring balance between interpretability and performance

# Statistical/Semantic Parsing of Informalized HOL

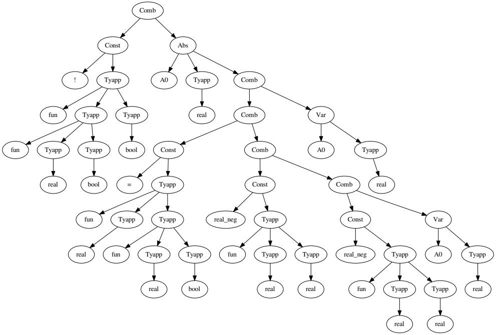
- Training and testing examples exported form Flyspeck formulas
  - Along with their informalized versions
- Grammar parse trees
  - Annotate each (nonterminal) symbol with its HOL type
  - Also “semantic (formal)” nonterminals annotate overloaded terminals
  - guiding analogy: word-sense disambiguation using CYK is common
- Terminals exactly compose the textual form, for example:
- REAL_NEGNEG: $\forall x. - - x = x$

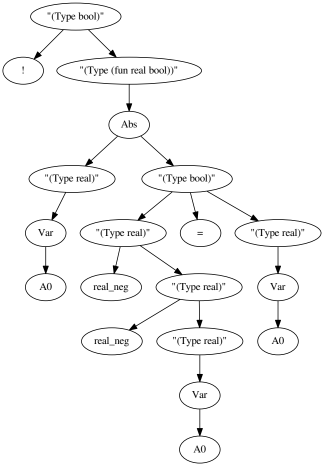
```
(Comb (Const "!" (Tyapp "fun" (Tyapp "fun" (Tyapp "real") (Tyapp "bool"))
(Tyapp "bool"))) (Abs "A0" (Tyapp "real") (Comb (Comb (Const "=" (Tyapp "fun"
(Tyapp "real") (Tyapp "fun" (Tyapp "real") (Tyapp "bool")))) (Comb (Const
"real_neg" (Tyapp "fun" (Tyapp "real") (Tyapp "real"))) (Comb (Const
"real_neg" (Tyapp "fun" (Tyapp "real") (Tyapp "real"))) (Var "A0" (Tyapp
"real"))))) (Var "A0" (Tyapp "real")))))
```

- becomes

```
("(̈Type bool)"̈ ! ("(̈Type (fun real bool))"̈ (Abs ("(̈Type real)"̈
(Var A0)) ("(̈Type bool)"̈ ("(̈Type real)"̈ real_neg ("(̈Type real)"̈
real_neg ("(̈Type real)"̈ (Var A0)))) = ("(̈Type real)"̈ (Var A0))))))
```

# Example grammars



# CYK Learning and Parsing (KUV, ITP 17)

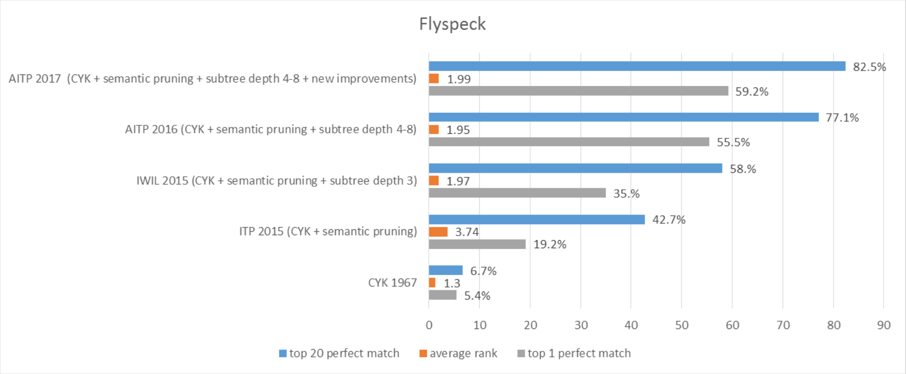
- Induce PCFG (probabilistic context-free grammar) from the trees
  - Grammar rules obtained from the inner nodes of each grammar tree
  - Probabilities are computed from the frequencies
- The PCFG grammar is binarized for efficiency
  - New nonterminals as shortcuts for multiple nonterminals
- CYK: dynamic-programming algorithm for parsing ambiguous sentences
  - input: sentence – a sequence of words and a binarized PCFG
  - output: N most probable parse trees
- Additional semantic pruning
  - Compatible types for free variables in subtrees
- Allow small probability for each symbol to be a variable
- Top parse trees are de-binarized to the original CFG
  - Transformed to HOL parse trees (preterms, Hindley-Milner)
  - typed checked in HOL and then given to an ATP (hammer)

# Autoformalization based on PCFG and semantics

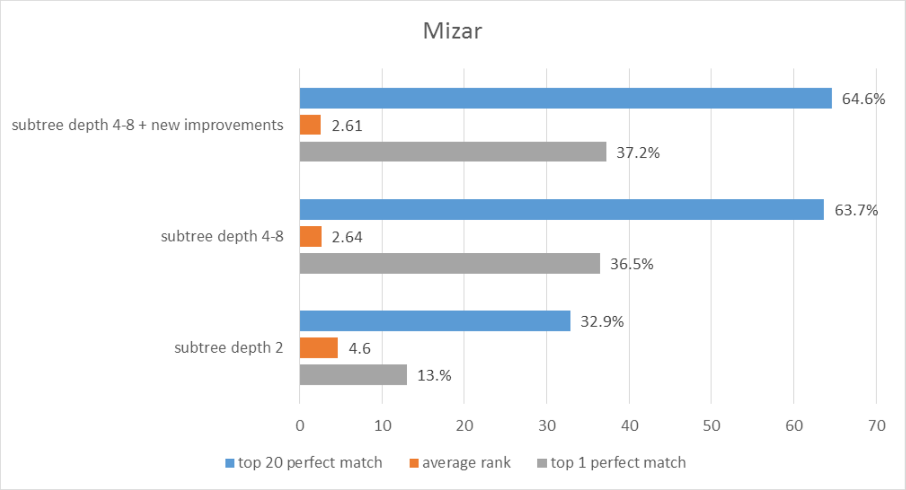
- “`sin ( 0 * x ) = cos pi / 2`”
- produces 16 parses
- of which 11 get type-checked by HOL Light as follows
- with all but three being proved by HOL(y)Hammer
- demo: `http://grid01.ciirc.cvut.cz/~mptp/demo.ogv`

```
sin (&0 * A0) = cos (pi / &2) where A0:real
sin (&0 * A0) = cos pi / &2 where A0:real
sin (&0 * &A0) = cos (pi / &2) where A0:num
sin (&0 * &A0) = cos pi / &2 where A0:num
sin (&(0 * A0)) = cos (pi / &2) where A0:num
sin (&(0 * A0)) = cos pi / &2 where A0:num
csin (Cx (&0 * A0)) = ccos (Cx (pi / &2)) where A0:real
csin (Cx (&0) * A0) = ccos (Cx (pi / &2)) where A0:real^2
Cx (sin (&0 * A0)) = ccos (Cx (pi / &2)) where A0:real
csin (Cx (&0 * A0)) = Cx (cos (pi / &2)) where A0:real
csin (Cx (&0) * A0) = Cx (cos (pi / &2)) where A0:real^2
```

# Flyspeck Progress



# First Mizar Results (100-fold Cross-validation)



# Outline

Neural Parsing

# Neural Autoformalization (Wang et al., 2018,2020)

- generate about 1M Latex - Mizar pairs based on Bancerek's work
- train neural seq-to-seq translation models (Luong – NMT)
- evaluate on about 100k examples
- many architectures tested, some work much better than others
- very important latest invention: *attention* in the seq-to-seq models
- more data very important for neural training – our biggest bottleneck (you can help!)
- Recent addition: unsupervised methods (Lample et all 2018) – no need for aligned data!

# Neural Autoformalization data

| | |
|---|---|
| Rendered LaTeX | If $X \subseteq Y \subseteq Z$, then $X \subseteq Z$. |
| Mizar | `X c= Y & Y c= Z implies X c= Z;` |
| Tokenized Mizar | `X c= Y & Y c= Z implies X c= Z ;` |
| LaTeX | `If $X \subseteq Y \subseteq Z$, then $X \subseteq Z$.` |
| Tokenized LaTeX | `If $ X \subseteq Y \subseteq Z $ , then $ X \subseteq Z $ .` |

# Neural Autoformalization results

| Parameter | Final Test Perplexity | Final Test BLEU | Identical Statements (%) | Identical No-overlap (%) |
|---|---|---|---|---|
| 128 Units | 3.06 | 41.1 | 40121 (38.12%) | 6458 (13.43%) |
| 256 Units | 1.59 | 64.2 | 63433 (60.27%) | 19685 (40.92%) |
| 512 Units | 1.6 | **67.9** | 66361 (63.05%) | 21506 (44.71%) |
| 1024 Units | **1.51** | 61.6 | **69179 (65.73%)** | **22978 (47.77%)** |
| 2048 Units | 2.02 | 60 | 59637 (56.66%) | 16284 (33.85%) |

# Neural Fun – Performance after Some Training

| | |
|---|---|
| Rendered LaTeX | Suppose $s_8$ is convergent and $s_7$ is convergent . Then $\lim(s_8+s_7) = \lim s_8 + \lim s_7$ |
| Input LaTeX | `Suppose $ { s _ { 8 } } $ is convergent and $ { s _ { 7 } } $ is convergent . Then $ \mathop { \rm lim } ( { s _ { 8 } } { + } { s _ { 7 } } } ) \mathrel { = } \mathop { \rm lim } { s _ { 8 } } { + } \mathop { \rm lim } { s _ { 7 } } $ .` |
| Correct | `seq1 is convergent & seq2 is convergent implies lim ( seq1 + seq2 ) = ( lim seq1 ) + ( lim seq2 ) ;` |
| Snapshot-1000 | `x in dom f implies ( x * y ) * ( f \| ( x \| ( y \| ( y \| y ) ) ) ) = ( x \| ( y \| ( y \| ( y \| y ) ) ) ) ) ;` |
| Snapshot-2000 | `seq is summable implies seq is summable ;` |
| Snapshot-3000 | `seq is convergent & lim seq = 0c implies seq = seq ;` |
| Snapshot-4000 | `seq is convergent & lim seq = lim seq implies seq1 + seq2 is convergent ;` |
| Snapshot-5000 | `seq1 is convergent & lim seq2 = lim seq2 implies lim_inf seq1 = lim_inf seq2 ;` |
| Snapshot-6000 | `seq is convergent & lim seq = lim seq implies seq1 + seq2 is convergent ;` |
| Snapshot-7000 | `seq is convergent & seq9 is convergent implies lim ( seq + seq9 ) = ( lim seq ) + ( lim seq9 ) ;` |

# Unsupervised NMT Fun on Short Formulas

```
len <* a *> = 1 ;                    len <* a *> = 1 ;
assume i < len q ;                   i < len q ;
len <* q *> = 1 ;                    len <* q *> = 1 ;
s = apply ( v2 , v1 ast t ) ;        s = apply ( v2 , v1 ) . t ;
s . ( i + 1 ) = tt . ( i + 1 )       s . ( i + 1 ) = tau1 . ( i + 1 )
1 + j <= len v2 ;                    1 + j <= len v2 ;
1 + j + 0 <= len v2 + 1 ;            1 + j + 0 <= len v2 + 1 ;
let i be Nat ;                       i is_at_least_length_of p ;
assume v is_applicable_to t ;        not v is applicable ;
let t be type of T ;                 t is_orientedpath_of v1 , v2 , T ;
a ast t in downarrow t ;             a *' in downarrow t ;
t9 in types a ;                      t `2 in types a ;
a ast t <= t ;                       a *' <= t ;
A is_applicable_to t ;               A is applicable ;
Carrier ( f ) c= B                   support ppf n c= B
u in B or u in { v } ;               u in B or u in { v } ;
F . w in w & F . w in I ;            F . w in F & F . w in I ;
GG . y in rng HH ;                   G0 . y in rng ( H1 ./. y ) ;
a * L = Z_ZeroLC ( V ) ;             a * L = ZeroLC ( V ) ;
not u in { v } ;                     u >> v ;
u <> v ;                             u <> v ;
v - w = v1 - w1 ;                    vw = v1 - w1 ;
v + w = v1 + w1 ;                    v + w = v1 + w1 ;
x in A & y in A ;                    assume [ x , y ] in A ;
```

# Imperfect Informalization (CARD_LAR:10)

```
reserve A,B for limit_ordinal infinite Ordinal;
reserve B1,B2,B3,B5,B6,D, C for Ordinal;
reserve X for set;
reserve X for Subset of A;
theorem Th10:
  [#] A is closed unbounded
proof
  thus [#] A is closed
  proof
    let B such that
A1: B in A;
    assume sup ([#] A /\ B)=B;
    thus thesis by A1;
  end;
  sup [#] A = A by ORDINAL2:18;
  hence thesis by Def4;
end;
```

# Imperfect Informalization

```
Let $A$ and $B$ be limit ordinals with $B \in A$.
If the supremum of the intersection of $A$ and $B$
is equal to $B$, then $A$ is a closed set.
\begin{proof}
Assume that $B$ is an element of $A$ such that the supremum of the
intersection of $A$ and $B$ is equal to $B$. By the definition of
limit ordinal, $B$ is a limit point of $A$. Therefore, $A$ is
closed under taking limits and is thus a closed set.  Moreover, by
Theorem 18 from the ordinal arithmetic, the supremum of $A$ is
equal to $A$. Hence, by Definition 4, $A$ is an unbounded set.
Therefore, $A$ is a closed unbounded set.
\end{proof}
```

# Conclusion: Vision of Ubiquitous Formally Checked Reasoning

- Progress towards McCarthy's "objectivity by formal proof" vision
- Direct transcription of mathematical discourse into code?
- What kind of code? Naproche? Lean? Isabelle? Coq?
- Johan's joke: will we speak in code?
- Will we have GF-like explainable translators or only LLMs et al?