The challenges of the EuroProofNet Working Group 4 on proof libraries

Claudio Sacerdoti Coen

<claudio.sacerdoticoen@unibo.it>

University of Bologna

23/09/2022

Outline

EuroProofNet at a glance

WG4 Libraries of Formal Proofs: objectives and challenges

Indexing and searching in libraries of formulae

Outline

EuroProofNet at a glance

European Research Network on Formal Proofs
COST Action CA20111

- coordinator: Frédéric Blanqui
- 300+ researchers from 40+ countries
- you should freely join if your country is in it
- organizes meetings and schools
- gives grants for Short Term Scientific Missions (STSMs)
- supports women and diversity in science
- promotes formal verification in teaching

EuroProofNet: objectives

1. Capacity Building Objectives
2. Research Coordination Objectives
   - to promote the output of checkable proofs from ATP
   - to make systems interoperable by encoding logics and libraries into Dedukti (LF modulo)
   - to gather proofs in a FAIR database
   - to manage, index, search and exploit the database
   - to apply ML and AI techniques to proofs
   - to improve the use of natural/controlled languages for proofs

Most topics in the range of CICM!
(but restricted to formal libraries)

Most people coming from the TYPES community

Dedukti (LF modulo)

- types are identified up to the symmetric-transitive closure of rewriting rules

  example: $\vdash I : \mathit{True}$ and $2 < 3 \rightsquigarrow \mathit{True}$; therefore $\vdash I : 2 < 3$

- greatly simplifies LF encodings

  example: $\mathit{El}\ (\mathit{arrow}\ A\ A) \rightsquigarrow \mathit{El}\ A\ \rightarrow\ \mathit{El}\ A$
  therefore $\vdash \lambda x : A.x\ :\ \mathit{El}\ (\mathit{arrow}\ A\ A)$

- makes indexing, retrieval and alignment between libraries much harder

  example: indexes should be up to as well
  example: $x + 2$ can be instantiated to $5 - 1$ up to

EuroProofNet: Work Packages

1. WG1 Tools on Proof Systems Interoperability
2. WG2 Automated Theorem Provers
3. WG3 Program Verification
4. WG4 Libraries of Formal Proofs
5. WG5 Machine Learning in Proofs
6. WG6 Type Theory

Outline

EuroProofNet – WG4: Libraries

Objectives:

1. investigate various approaches to efficiently maintain libraries of formal proofs
2. to make a collection of proofs that can be modified, extended, and queried . . .
3. . . . by users who do not have expert knowledge of the entire collection nor of the system that was used to develop the proofs.

EuroProofNet – WG4: Libraries

Tasks:

1. discuss challenges of maintaining and using existing libraries of formal proofs;
2. contribute to creating database of already formalised mathematics;
3. develop the tool for querying libraries of formal proofs with respect to the semantic of search object;
4. that the tool can be efficiently used with Dedukti and within software formalisation efforts.

EuroProofNet – WG4: Libraries

Deliverables:

1. (month 12): Database gathering proofs from Coq, HOL-Light and Matita and their translations.
2. (month 24): Tools for managing the dependencies between proofs, and querying and searching the database.
3. (month 48): Extension of the database and associated tools to other systems like Agda, Minlog, PVS, Lean, Mizar, Atelier B, TLAPS.

EuroProofNet – WG4: Libraries

Challenges:

- Library exporting and dependencies:
  - centralized approach (e.g. AFP) vs decentralized (e.g. opam)
  - how to version libraries and dependencies?
  - what will Dedukti have? how will it manage dependencies?
  - how to trigger automatic translation to/from Dedukti?
  - when to translate between systems?

Engineering challenges

EuroProofNet – WG4: Libraries

Challenges:

- Library reuse:
  - type *t* in system *A* is not translated to type *t* in system *B*
    - how to declare/generate/store alignments?
    - how to transfer between *A.t* in *B* and *B.t*?
  - information how to use things is lost
    - type-classes/instances, automatically inferred arguments, coercions, canonical structures, functors, NOTATIONS, ...
    - how to declare and translate them?

Research and engineering challenges

EuroProofNet – WG4: Libraries

Challenges:

- Library indexing and querying:
  - adapt existing tools for indexing and querying up to instantiation/generalization/approximation
  - how to elaborate queries (and results)? (e.g. a query written in Coq)
  - requires alignments as well

Research and engineering challenges

EuroProofNet – WG4: Libraries

Challenges:

- Proof mining:
  - identify proofs in logical fragments (e.g. to allow more translations)
  - bring proofs in a logical fragment
  - devise new/improved translations between logics/systems

Outline

State of the art of retrieval of mathematical knowledge

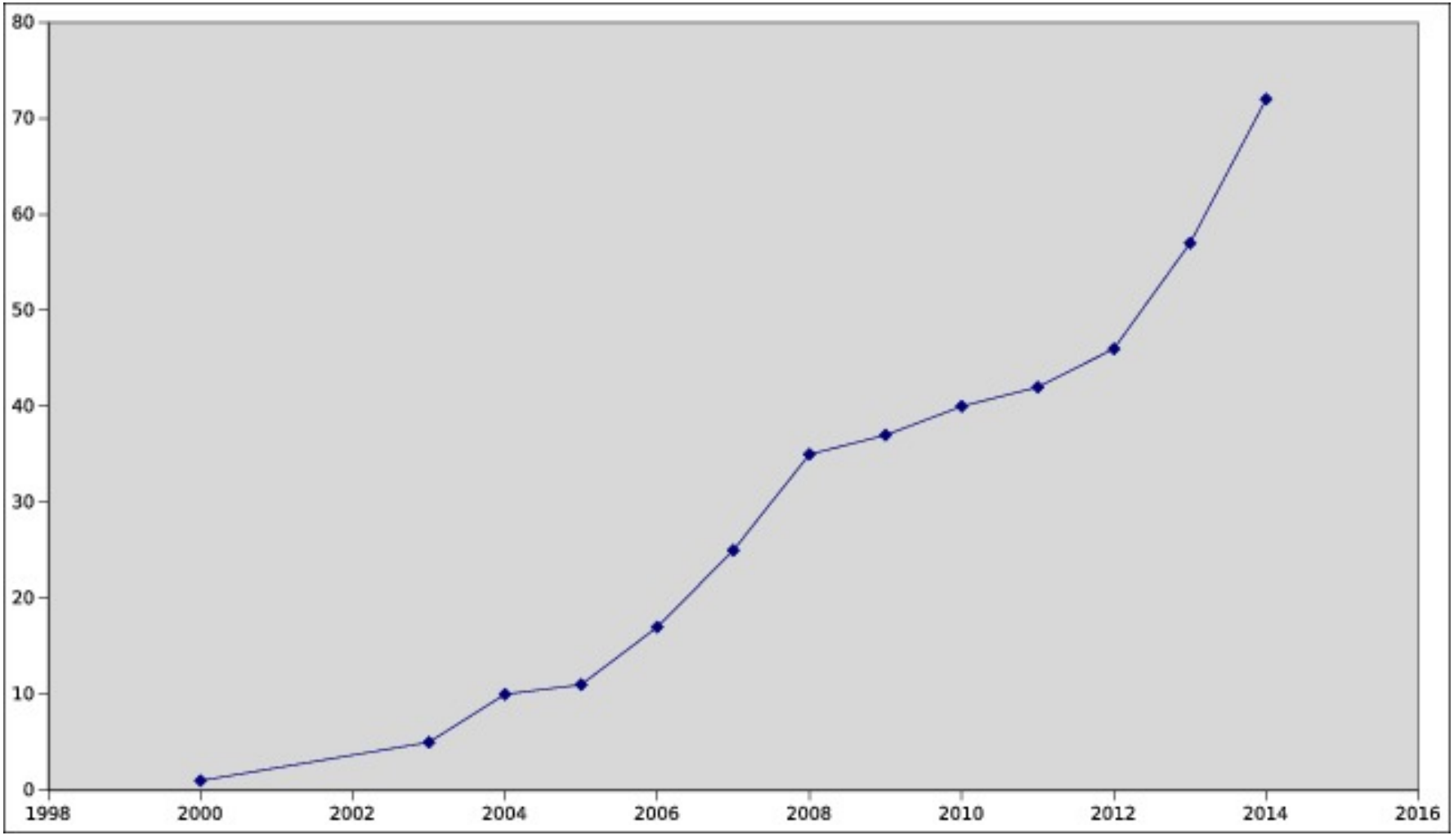
- C. Sacerdoti Coen, F. Guidi,
  A Survey on Retrieval of Mathematical Knowledge,
  Math. Comput. Sci. 10(4): 409-427 (2016)

  Taxonomic study of 72 papers

- NTCIR context on Mathematical Information Retrieval (last one in 2013)

Target both collection of statements and collections of mathematical texts

Progress


80
70
60
50
40
30
20
10
0
1998
2000
2002
2004
2006
2008
2010
2012
2014
2016

Three Taxonomies

Purpose Driven

Why?

Encoding Based

What?

Techniques

How?

Purpose Driven Taxonomy

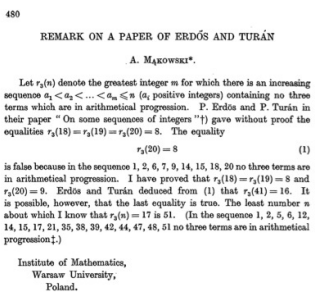
Purpose Driven

Why?

Purpose Driven Taxonomy

Purpose Driven

Why?



480

REMARK ON A PAPER OF ERDŐS AND TURÁN

A. MAKOWSKI*.

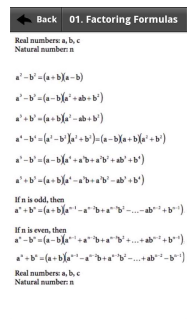
Let $r_3(n)$ denote the greatest integer m for which there is an increasing sequence $a_1 < a_2 < \ldots < a_m \leqslant n$ (a_i positive integers) containing no three terms which are in arithmetical progression. P. Erdős and P. Turán in their paper " On some sequences of integers "† gave without proof the equalities $r_3(18) = r_3(19) = r_3(20) = 8$. The equality

$$r_3(20) = 8 \qquad (1)$$

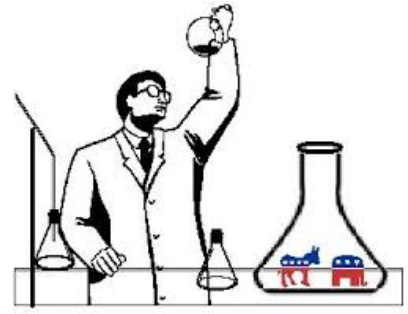
is false because in the sequence 1, 2, 6, 7, 9, 14, 15, 18, 20 no three terms are in arithmetical progression. I have proved that $r_3(18) = r_3(19) = 8$ and $r_3(20) = 9$. Erdős and Turán deduced from (1) that $r_3(41) = 16$. It is possible, however, that the last equality is true. The least number n about which I know that $r_3(n) = 17$ is 51. (In the sequence 1, 2, 5, 6, 12, 14, 15, 17, 21, 35, 38, 39, 42, 44, 47, 48, 51 no three terms are in arithmetical progression‡.)

Institute of Mathematics,
Warsaw University,
Poland.

Document Retrieval



Formula Retrieval



Document Synthesis

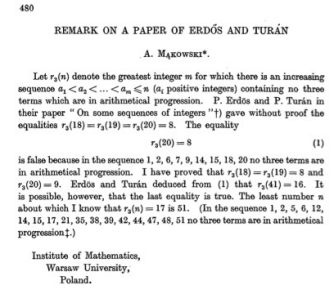
Document Retrieval

Objective: A human wants to recall a set of (fragments) of mathematical documents.

Input: keywords (e.g. for topics), free text, formulae (as examples/to disambiguate).

Output: ranked list of summaries of documents, possibly clustered; results based on similarity and likelyhood of usefulness.

Constraints: balance between precision and recall; only the first results matter; good ranking is fundamental; performance is not.



480

REMARK ON A PAPER OF ERDŐS AND TURÁN

A. Makowski*.

Let $r_3(n)$ denote the greatest integer n for which there is an increasing sequence $a_1 < a_2 < \ldots < a_n \leqslant n$ (a_i positive integers) containing no three terms which are in arithmetical progression. P. Erdős and P. Turán in their paper "On some sequences of integers"† gave without proof the equalities $r_3(18) = r_3(19) = r_3(20) = 8$. The equality

$$r_3(20) = 8 \qquad (1)$$

is false because in the sequence 1, 2, 6, 7, 9, 14, 15, 18, 20 no three terms are in arithmetical progression. I have proved that $r_3(18) = r_3(19) = 8$ and $r_3(20) = 9$. Erdős and Turán deduced from (1) that $r_3(41) = 16$. It is possible, however, that the last equality is true. The least number n about which I know that $r_3(n) = 17$ is 51. (In the sequence 1, 2, 5, 6, 12, 14, 15, 17, 21, 35, 38, 39, 42, 44, 47, 48, 51 no three terms are in arithmetical progression‡.)

Institute of Mathematics,
Warsaw University,
Poland.

Document Retrieval

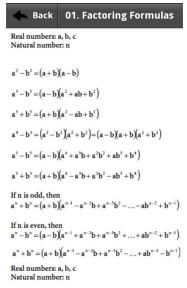
Formula Retrieval

Objective: a program nees to retrieve all formulae in some relation $\mathcal{R}$ with the query $\mathcal{E}$ a (set of) formula(e);

Input: a set of formulae containing metavariables; or a query in some ad-hoc language; rarely additional constraints (keywords, etc.)

Output: an unordered (less frequently, ordered) set of identifiers of matching formulae

Constraints: recall must be maximized; speed is critical; to speed up, use a decidable $\mathcal{R}' \supseteq \mathcal{R}$



Formula Retrieval

Encoding Based Taxonomy

Encoding Based

What?

Encoding Based Taxonomy

Encoding Based

What?

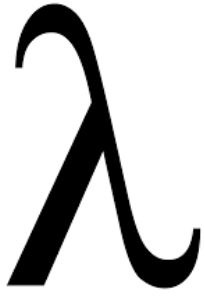
$$\int_0^a x^k dx = \frac{a^{k+1}}{k+1}$$

Presentation

```
XML`MathML`ExpressionToMathML[x^2]

<math xmlns='http://www.w3.org/1998/Math/MathML'>
 <semantics>
  <msup>
   <mi>x</mi>
   <mn>2</mn>
  </msup>
  <annotation-xml encoding='MathML-Content'>
   <apply>
    <power/>
    <ci>x</ci>
    <cn type='integer'>2</cn>
   </apply>
  </annotation-xml>
 </semantics>
</math>
```

Content



Semantics

Purpose Dominates Encoding

Formula retrieval

- always formulated on content or semantics
- on semantics: e.g. what lemmas can be applied to progress in the proof?
- on content: e.g. reuse of lemmas across different systems

Purpose Dominates Encoding

Document retrieval

- formulation is (mostly) agnostic of the encoding
- but queries are likely to be in presentation
- thus queries need to be elaborated first

Taxonomy of Techniques

Taxonomy of Techniques

How?

Taxonomy of Techniques

Taxonomy of Techniques

How?

=

×

n

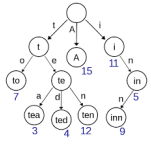
1 Main Technique

n Modular Techniques

Main Techniques

Main Techniques

Reduction to Full Text Search



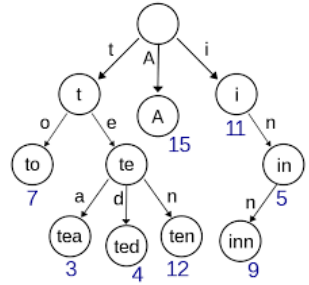
Structure-Based Indexing via Tries/Substituion Trees

Reduction to SQL or ad-hoc

Reduction to XQuery

Structure-Based Indexing via Tries/Substitution Trees

- Stores the library in a huge trie $\Rightarrow$ fast (until we will run out of RAM...)
- Shines on formula retrieval
- precision maximized, poor recall
- $\mathcal{R}$ restricted to instantiation/generalization only
- requires combination with modular techniques to enlarge the class of $\mathcal{R}$



Structure-Based Indexing via Tries/Substitution Trees

Reduction to SQL or ad-hoc

- Used for formula retrieval and document synthesis
- Implemented by theorem provers
- Classifies formulae extracting features (e.g. set of constants, predicate in conclusion position, number of hypotheses, etc.)
- The structure of formulae up-to can be captured in SQL $\mathcal{R}' \supseteq \mathcal{R}$ minimizing the number of SQL queries issued
- Good balance between precision and recall

Reduction to SQL or ad-hoc

Modular Techniques

Modular Techniques

Segmentation

Normalization

Approximation

Enrichment

Query Reduction

Normalization

- Improves recall, precision not harmed
- Normalization induces an equivalence relation $\equiv$
- Queries up-to-$\equiv$ iff $\equiv\mathcal{R}\equiv \subseteq \mathcal{R}$
- For document retrieval: $\equiv$ compatible with similarity and ranking
  Otherwise: major loss of precision
- names to De Brujin indexes; associative/commutative; derived notions (e.g. $\geq$ vs $\leq$); logical equivalence/type isomorphism (e.g. prenex normal forms).

Normalization

Approximation

- Improves recall, decreases precision
- Confused with normalization:
  Lossy transformation of the library
- Replace: formulae with types;
  variable names with placeholder;
  numerical constants with
  placeholder.
- More efficient than query reduction
  (indexing time transformation)

Approximation

Enrichment

- Improves precision and recall
- Applied to both document and formula retrieval
- Augments the information stored/required in the library/query
- Infers new knowledge
- Heuristic generation of parallel markup; automatic/interactive disambiguation of formulae (from presentation to content/semantics); inference of metadata from context analysis, co-occurence analysis, usage analysis (latent semantics)

Enrichment

Query Reduction

- Trades precision for recall
- Selectively drops or weakens some constraints in the query
- Results of weakened queries ranked after results of original one
- Constant identified with co-occurring ones; too frequently occurring item dropped; match formulae only looking at top-level structure.

Query Reduction