

Fine-grained mathematical justifications

Jesse Alama

CENTRIA, New University of Lisbon

January 8, 2011 / LogICCC meets India / Delhi University

The Lisbon node of DiFoS

- DiFos: Dialogical Foundations of Semantics
- Nodes: Lisbon, Tübingen, Amsterdam
- Common theme: dialogue in argumentation and proof
- Lisbon node: focused on interaction in mathematical proof
- Especially focused on interactive theorem proving.
- Today: early results and a program that blends interactive and automated theorem proving on large corpora of formalized mathematical knowledge.



Automated theorem proving

- The task of an automated theorem prover (ATP) is to find a deduction of a given proposition from given assumptions.
 - Or: show that a given formula is unsatisfiable (and possibly produce a proof that it is so);
 - Or: show that a given formula is satisfiable (and possibly produce a description of a model of the formula).



Interactive theorem proving

- We are often interested in proving theorems that are many steps away from our basic assumptions.
- ATPs are generally not well-suited to such tasks, because the search space of deductions is often (extremely!) large and complex.
- *Interactive* theorem provers (ITPs) focus the reasoning task to the construction of *proofs*.
- Proof formalisms:
 - natural deduction (in various styles: J askowski, Fitch, Gentzen, etc.)
 - sequent calculus
- ITPs generally use an ATP in some fashion to check the the most basic steps of arguments expressed in the ITP's proof language.



What's the difference?

- ITPs are generally used to construct *human-readable* proofs.
- The proof formalisms employed by ATPs often produce proofs that take unexpected, counterintuitive paths from the assumptions to the desired conclusion.
 - This is not always the case. (Ed Zalta briefly described a couple days ago a case where a proof discovered by an ATP was analyzed and translated into a human-consumable argument.)
 - However, it often is; analyzing proofs found by ATPs is usually nettlesome.
- One writes a proof with the assistance of an ITP by breaking down the argument of interest, and then querying the ITP to check whether the proof is acceptable.



A zoo of ITPs

Numerous ITPs in various styles are available. Some major ones:

- Isabelle: based on a weak, extensible logic that admits many possible ‘instantiations’ (e.g., classical and intuitionistic logic, higher-order logic vs. set theory, etc.), tactic or natural deduction proof style
- HOL, HOL light: higher-order logic based on the so-called LCF (logic for computable functions) style, λ -calculus, intuitionistic logic (via the Curry-Howard correspondence), tactic proof style
- Coq: based on the calculus of inductive constructions, tactic proof style
- Mizar: first-order classical set theory, natural deduction



Focus on mizar

- mizar is one of the oldest ITP that enjoys widespread use.
 - Started in 1973 in Poland.
 - Continues to be developed primarily there.
- Its foundations—classical first-order set theory—are the most attractive from the perspective of traditional foundations of mathematics.
 - mizar is actually based on Tarski-Grothendieck set theory, which is stronger than **ZFC**.
- Rich formalism: captures many natural aspects of mathematical practice, such as various kinds of notational conventions and extensions by definitions.
- Moreover, among comparable ITPs, the lion's share of the knowledge formalized in the mizar system is based on pure mathematics.



Items in mizar

■ Propositions (theorems, lemmas)

`X is empty implies X = {}`

`theorem X <> {} implies ex x st x in X`



mizar items (cont'd)

Definitions:

```
definition let X be set;  
  attr X is empty means  
  not ex x being set st x in X;  
end;
```

```
definition let n be Nat;  
  func Seg n -> set equals  
  { k where k is Element of NAT: 1 <= k & k <= n };  
end;
```



mizar items (cont'd)

Schemes

Extensionality { $X, Y() \rightarrow \text{set}, P[\text{set}]$ } : $X() = Y()$
provided for x holds x in $X()$ iff $P[x]$ and
for x holds x in $Y()$ iff $P[x]$

Ind { $P[\text{Nat}]$ } : for k being Element of NAT holds $P[k]$
provided
 $P[0]$ and for k being Element of NAT st $P[k]$ holds $P[k + 1]$



mizar items (cont'd)

Ensuring non-emptiness of types (a requirement in mizar, for better or worse):

```
registration
  cluster {} -> empty;
end;
```

```
registration
  cluster empty set;
end;
```



mizar items (cont'd)

Relations among types:

```
registration
  let E be non empty set;
  cluster non proper -> non empty Subset of E;
  cluster empty -> proper Subset of E;
end;
```



mizar items (cont'd)

Conventions of language:

```
notation
```

```
  let X, Y be set;
```

```
  antonym X meets Y for X misses Y;
```

```
end;
```

```
registration
```

```
  let p;
```

```
  identify len p with dom p;
```

```
end;
```



Formalizing proofs with mizar

(demo)



Task: improving sufficient conditions and discovering necessary conditions

- From a formalized proof of a theorem, we can say, with complete precision, what is *sufficient* for the theorem.
- We can also compute refinements to a theorem, thereby giving a sharper set of sufficient conditions.
- When the set S of sufficient conditions is suitably refined, we can attempt to go backwards and find out whether S is actually *necessary* for the theorem.



Example

- (example)
- Every path commencing with theorem T ending at axiom A passes through lemma L .
- This shows that $S - \{A\} + L \vdash T$.
- In this case where we know that $S - \{A\} + L$ differs from S (i.e., it is not the case that L proves A from $S - \{A\}$), this shows that we have found a weakening of the sufficient conditions for theorem T .



Dependency relation

Definition: a mizar item a **depends** on a set $S = \{b_1, b_2, \dots, \}$ of mizar items if there exists a mizar text T that contains b_1, b_2, \dots, b_n (in some order), ending with a , such that T is grammatically and logically correct according to the mizar verifier.

Definition: a mizar item a **minimally depends** on a set $S = \{b_1, b_2, \dots, \}$ of mizar items if a depends on S and there is no proper subset S' of S such a depends on S' .



Dependency graph

Nodes: mizar items

Edges: The set of outgoing edges of a node u is the nodes v_1, v_2, \dots, v_n such that the justification of u depends on all of v_1, v_2, \dots, v_n .

The dependency graph of a set of deductions gives information about what is sufficient for the success of theorems.



A challenge

- 2008: my formalization of Euler's polyhedron formula ("for all polyhedra p , we have that $V - E + F = 2$ ", where V , E , and F are, respectively, the numbers of vertices, edges, and faces of a polyhedron)
- The formalization shows that $TG \vdash$ Euler's polyhedron formula, so that **TG** is sufficient for (a formalization of a particular proof of) Euler's polyhedron formula.
- **Challenge:** Tarski-Grothendieck set theory is *much more* than what is needed for Euler's formula. What do we *really* need?



Tarski's universe axiom

- Universe axiom: for every set a there exists a set \mathcal{U} such that
 - $a \in \mathcal{U}$,
 - \mathcal{U} is closed under taking power sets,
 - \mathcal{U} is closed under union,
 - for every subset A of \mathcal{U} for which $|A| < \mathcal{U}$, we have that $A \in \mathcal{U}$
- Consequences of the universe axiom:
 - axiom of infinity
 - axiom of choice
 - power set



'Eliminating' the universe axiom

- By analyzing the dependency graph for Euler's polyhedron formula, one finds that all paths from the theorem to the universe axiom go through either:
 - axiom of infinity
 - axiom of choice
 - power set
- This calculation shows that, from a witness to $\mathbf{TG} \vdash \text{EPF}$, we have a witness for $\mathbf{ZFC} \vdash \text{EPF}$, which is a considerable refinement, in light of the strength of \mathbf{TG} .



From ZFC to ZF

- We can do even better, using the dependency graph of Euler's formula, but using a slightly different method:
- All paths from Euler's formula to the axiom of choice pass through the theorem "Every linearly independent set of vectors in a vector space can be extended to a basis".
- But the proof of Euler's formula uses only finite-dimensional vector spaces, which always have bases (by definition).
- Hence, choice can be eliminated, and we have that **ZF** \vdash EPF.



From ZF to Z

- **Challenge:** Gather all instances of the scheme of replacement (the characteristic axiom that distinguishes **Z** from **ZF**) that occur anywhere in the complete justification of EPF.`
- Show, for each such instance, that the instance is a theorem of **Z**.
 - The proof could be carried out either 'by hand' or with the help of an ATP.



Necessary conditions: reverse mathematics

- Reverse mathematics: discover axioms from theorems (rather than the other way around).
- Example: Bolzano-Weierstrass theorem (every bounded sequence of real numbers has a convergent subsequence).
 - the theorem can be formally proved in \mathbf{ACA}_0 (\mathbf{RCA}_0 + scheme of arithmetical comprehension)
 - show that, working in the much weaker \mathbf{RCA}_0 , that every instance of the scheme of arithmetical comprehension can be proved from the Bolzano-Weierstrass theorem.
 - **Conclusion:** \mathbf{ACA}_0 is equivalent to, or necessary and sufficient for, the Bolzano-Weierstrass theorem.



Necessary conditions: reverse mathematics 'in the small'

- Program: find tractable cases of metamathematical interest where one can *compute*, using an ATP, that certain conditions are *necessary* for certain concrete theorems.
- Applications:
 - Elimination of the axiom of replacement (already sketched)
 - Isolating the need for the axiom of infinity
 - Discovering reversals, à la reverse mathematics (e.g., the necessity for the axiom of choice for certain theorems).



References

- A. Arana, “On formally measuring and eliminating extraneous notions in proofs”, *Philosophia Mathematica* **17** (2009), 189–207.
- The mizar Homepage, <http://mizar.org>.
- A. Quaife, *Automated Development of Fundamental Mathematical Theories*, Kluwer Academic Publishers, 1992.
- S. Simpson, *Subsystems of Second-Order Arithmetic*, 2nd edition, Cambridge University Press, 2010.
- J. Urban, “MPTP—Motivation, implementation, first experiments”, *Journal of Automated Reasoning*, **33**(3–4) (2004), pp. 319–339.

