Premise Selection in the Naproche System

M. Cramer, P. Koepke, D. Kühlwein, and B. Schröder

Mathematical Institute, University of Bonn
German Linguistics, University of Duisburg-Essen
{cramer,koepke,kuehlwei}@math.uni-bonn.de
bernhard.schroeder@uni-due.de
http://www.naproche.net

Abstract. Automated theorem provers (ATPs) struggle to solve problems with large sets of possibly superfluous axiom. Several algorithms have been developed to reduce the number of axioms, optimally only selecting the necessary axioms. However, most of these algorithms consider only single problems. In this paper, we describe an axiom selection method for series of related problems that is based on logical and textual proximity and tries to mimic a human way of understanding mathematical texts. We present first results that indicate that this approach is indeed useful.

Key words: formal mathematics, automated theorem proving, axiom selection

1 Introduction

Reducing the search space of ATP problems is a long standing problem in the ATP community. In 1987 Larry Wos called a solution to the problem of definition expansion and contraction "one of the more significant advances in the field of automated reasoning" [22]. In recent years, several algorithms have been developed to tackle this problem. (e.g. SRASS [17], SInE [8], Gazing [1], MaLARea [18], and the work by Meng and Paulson [13]). In this paper, we describe an axiom selection method for series of related problems that is based on logical and textual proximity.

The Naproche project (NAtural language PROof CHEcking) studies the semi-formal language of mathematics as used in mathematical journals and textbooks from the perspectives of linguistics, logic and mathematics. As part of the Naproche project, we develop the Naproche system [5], a program that can automatically check texts written in the Naproche controlled natural language (CNL) for logical correctness. We test our system by reformulating parts of mathematical textbooks and the basics of mathematical theories in the Naproche CNL and checking the resulting texts.

The checking process is similar to how a human reader would verify the correctness of a text. Each statement in the text that is not an axiom[1], a definition

[1] Here, axiom is used in the mathematical sense, e.g. the axiom of choice.

or an assumption must follow from the information given so far. Using logical terms, we can say that the statement has to follow from its premises. In the Naproche system, such statements create proof obligations. A proof obligations is an ATP problem with the statement as conjecture and the premises as axioms.

The longer a text is, the more premises are available, which makes proof obligations harder to discharge. Thus, we need to find a way to reduce the number of premises, i.e. reduce the search space of the ATP. For single ATP problems, successful algorithms (e.g. SInE, SRASS) exist. There are also programs that were developed for larger theories, e.g. Gazing and MaLARea. However, to our knowledge there is no system that is based on a 'human' understanding of a proof. The texts we are dealing with read like normal mathematical proofs in natural language. We developed a premise selection algorithm that tries to use some of the information implicit in the human structuring of the proof text.

We first give a quick overview of the Naproche system. Section 2 explains our premise selection algorithm. First results are presented in section 3.

2 The Naproche System

The Naproche system [5] checks texts that are written in a controlled natural language for mathematics for correctness. We call this controlled natural language the Naproche CNL. Texts written in the Naproche CNL read like normal mathematical texts. The Naproche CNL is described in a separate paper (see [3]). A quick overview can be found online[2].

The input text is first translated into a linguistic representation called Proof Representation Structure (PRS, see [3], [4]). From such a PRS the program determines which statements have to be checked and creates the corresponding proof obligations [10]. For the actual proving we use the TPTP infrastructure [16]. The proof obligations are translated into ATP problems in the TPTP format and then sent to an ATP.

There are two main long term goals of the Naproche Project: Firstly, to provide a more natural system for formalising mathematics, and secondly to function as a tool that can help undergraduate students to learn how to write formally correct proofs and thus get used to the semi-formal language of mathematics.

2.1 An Example Text

We present a short example texts taken from the Naproche translation of Euclid's Elements [7]. Note that Naproche uses LaTeX-sourcecode as input. The example shows the compiled version.

Example: Let a, b and c be distinct points. By Theorem 1 there is a point d, such that $\overline{da} = \overline{db} = \overline{ab}$. Let M be the line such that b and d are on M. Let α be the circle such that b is the center of α, and c is on α.

[2] http://www.naproche.net/wiki/doku.php?id=dokumentation:language

2.2 Related Work

There are several projects that are similar to Naproche. We will just name a few:

A. Trybulec's Mizar [12] is arguably the most prominent. It was started in 1973, and by today many non-trivial mathematical theorems have been proved. An active community continues to formulate and prove theorems in Mizar. The results are published regularly in the journal *Formalized Mathematics.*

The Isabelle [14] team is working on Isar [21], a "human-readable structured proof language". The System for Automated Deduction (SAD, [19]) checks texts that are written in its input language, ForThel [20], for correctness.

Claus Zinn did his PhD on *Understanding Informal Mathematical Discourse* [23], but focused on only two examples. The DIALOG group [2] did experiments with mathematical language in a tutoring context. Mohan Ganesalingam [6] studied the language of mathematics in detail, but did not implement his ideas (yet).

What distinguishes Naproche is our focus on deep linguistic analysis of non-annotated natural language. We try to keep our input language as close as possible to the natural language of mathematics.

3 The Premise Selection Algorithm

When verifying the correctness of a proof, mathematicians basically face the same problem as ATPs. They have a given set of premises, i.e. all their mathematical knowledge, and have to derive the conjecture from these premises. Understanding the proof means knowing which premises were used in each step. While the human selection process as a whole is very complicated, there are three parts that can easily be used for automated premise selection.

- Explicit References:
  Explicit references like "*by theorem 4*" are often used in mathematical texts. Such a reference is a clear indication that the referenced object is useful, or even necessary.
- Textual Adjacency:
  While human proofs are not as detailed as formal derivations, they are still done step by step. Usually, the proof steps just before a statement are relevant. The most common example of this are assumptions: *Assume* φ*. Then* ψ. Here, the proof step before *Then* ψ is *Assume* φ, and φ will most likely be needed to prove ψ.
- Logical Relevance:
  Quite often, ideas that were needed in one part of a proof are also needed in another part. I.e. if a definition was needed for the first proof step, it will probably be needed again later in the proof.

In order to capture these ideas we developed Proof Graphs. Each statement of the proof becomes a node in this graph. Two nodes are connected by an (untyped) edge if they are textually or logically close to each other, or if there is an explicit reference from one to the other. We define the distance between two statements as the geodesic distance. i.e. the length of the shortest path from one statement to the other.

Based on Proof Graphs, we can define a premise selection algorithm. Given a proof obligation, the premise selection algorithm determines which of the available premises are given to the ATP. The algorithm was implemented as part of the Naproche system.

Explicit references and textual adjacency are calculated during the linguistic analysis of the text. We say that φ is logically close to ψ if φ was used in the proof on ψ. In the implementation, we use Geoff Sutcliffes program *Proof Summary* which analyses ATP proofs.

The premises selection algorithm proceeds as follows:

– Input: *Conjecture*, *Axioms*, *Distance* and *Time*
1 Determine the distance between the *Conjecture* and the *Axioms.*
2 Select all axioms whose nodes have distance less than *Distance* from the conjecture Node.
3 Create a TPTP problem with the selected axioms and the *Conjecture.*
4 Run an ATP on the problem with time limit *Time.*
5 If the ATP cannot prove the conjecture from the axioms, the starting distance is less than the predefined maximum distance, and the starting time is less than the predefined maximum time, define a new time limit and a new starting distance (e.g. $\mathit{NewTime} = 2 * \mathit{Time}$ and $\mathit{NewDistance} = 2 * \mathit{Distance}$) and try again.
6 If the ATP finds a proof, use the proof given by the ATP to find out which axioms where actually used. Determine the maximum distance of the used axiom to define the new starting distance (e.g. $\mathit{NextDistance} = (4*\mathit{Distance}+\mathit{MaxUsedDistance})/5$) and update the proof graph with this new information.

4 Results

To test the algorithm we checked a Naproche CNL version of the first chapter of Landau's *Grundlagen der Analysis* [11] with and without the premises selection algorithm. This text contains 228 proof obligation with a total number of 7602 premises.

Currently *Proof Summary* [16] only supports two ATPs, Metis [9] and EP [15]. Both were used during testing. MaxDistance was set to 20, MaxTime was set to 5 seconds, the start Distance was set to 1, and the start Time set to 1 sec. The other values were defined as follows:

$$\begin{aligned} \text{NewTime} &= 2 * \text{Time} \\ \text{NewDistance} &= 2 * \text{Distance} \\ \text{NextDistance} &= \lceil \tfrac{4*\text{Distance}+\text{MaxUsedDistance}}{5} \rceil \end{aligned}$$

Table 1 shows the results for EP. Without the premises selection algorithm, seven obligations could not be discharged by EP. With the premise selection algorithm enabled, EP was able to discharge all 228 obligations.

		Without PS		With PS	
	Total	Theorem	No Proof	Theorem	No Proof
obligations	228	221	7	228	0
premises	7602	7235	367	3964	0
premises/obligations		32.74	52.43	17.39	N/A
used premises/obligation		2.99	N/A	2.93	N/A
unused premises/obligation		29.75	52.43	5.96	N/A
average distance		8.2	8.15	5.53	N/A
average used distance		3.46	N/A	3.38	N/A
average unused distance		8.68	8.15	5.96	N/A

Table 1. Results for EP 1.0

We also determined the average distance of the used (3.46) and unused premises (8.68). These numbers are a clear indicator that our distance definition is indeed useful.

In Table 2 we see the results for Metis. Without the premises selection algorithm, 44 obligations could not be discharged by Metis. With the premise selection algorithm enabled, 26 obligations could not be discharged.

		Without PS		With PS	
	Total	Theorem	No Proof	Theorem	No Proof
obligations	228	184	44	202	26
premises	7602	5630	1972	2412	1176
premises/obligations		30.6	44.82	11.94	45.23
used premises/obligation		2.16	N/A	2.09	N/A
unused premises/obligation		28.43	44.42	9.85	45.23
average distance		8.98	9.64	5.22	9.49
average used distance		3.49	N/A	3.22	N/A
average unused distance		9.39	9.64	5.64	9.49

Table 2. Results for Metis 2.2

The reason why Metis ends up with a lower number of total premises with premises selection enabled is that the fewer obligations an ATP is able to discharge, the less information we have about logical relevance. This affects the subsequent obligations since fewer formulas are within the search distance. Similar to the EP results, the average distance of used premises is much lower (3.49) than the average distance of unused premises (9.39).

For further testing, we created two problem batches. The first one contained the original 228 problems. For the second batch we took all the modified problems that were created when using EP 1.0 and the premises selection algorithm. We sent the problems to Geoff Sutcliffe and he used his TPTP infrastructure to run seven ATP systems on the problems with a time limit of 300 seconds per obligation. The results can be seen in Table 3.

ATP	Solved(Modified)	Solved(Original)
Bliksem 1.12	225	222
E 1.1	228	228
Geo 2007f	219	210
iProver 0.7	223	222
Metis 2.2	205	193
Prover9 0908	213	221
Vampire 11.0	227	227

Table 3. Results with more ATPs and time limit 300 sec

182 of the modified problems were solved by all the systems. Of the remaining 46, 39 were solved by 6 out of the 7 systems. 169 of the original problems were solved by all the systems. Of the remaining 59, 50 were solved by 6 out of the 7 systems. 6 of the 9 are also in the set of 7 hard ones from the modified versions of the problems.

Of all ATPs tested, only Prover9 performs worse on the modified problem set. We assume that this is due to the fact that we used EP to create the modified problems. We hope that Prover9 would also perform better with premise selection when being used directly in the Naproche system. Unfortunately this cannot be tested at the moment since Prover9 does not provide *Proof Summary* parseable output.

5 Conclusion and Future Work

While the details of the implementation are and should be up for discussion, the results we received suggest that the ideas behind the premise selection algorithm: textual adjacency, references and reusing the same ideas do seem to work and improve the ATP performance.

For further testing, we would like to compare and combine our approach with other axiom selection algorithms. Furthermore, it would be interesting to see how important the different aspects of the proof graph are.

The main focus for the future will be to create longer and more mathematical texts. Once we do have more material, we will experiment with different modification of the presented Proof Graph.

References

1. Dave Barker-Plummer. Gazing: An Approach to the Problem of Definition and Lemma Use. *J. Autom. Reasoning*, 8(3):311–344, 1992.

2. Christoph Benzmüller, Marvin Schiller, and Jörg Siekmann. Resource-bounded modelling and analysis of human-level interactive proofs. In Matthew Crocker and Jörg Siekmann, editors, *Resource Adaptive Cognitive Processes*, Cognitive Technologies Series. Springer, 2010. In print.
3. M. Cramer, B. Fisseni, P. Koepke, D. Kühlwein, B. Schröder, and J. Veldman. The Naproche Project: Controlled Natural Language Proof Checking of Mathematical Texts. *LNCS*, 5972, 20010.
4. Marcos Cramer. Mathematisch-logische Aspekte von Beweisreprsentationsstrukturen. Master's thesis, University of Bonn, 2009.
5. P. Koepke D. Kühlwein, M. Cramer and B. Schröder. The Naproche System. 2009.
6. Mohan Ganesalingam. *The Language of Mathematics*. PhD thesis, University of Cambridge, 2009.
7. Thomas L. Heath and Euclid. *The Thirteen Books of Euclid's Elements, Books 1 and 2*. Dover Publications, Incorporated, 1956.
8. Krystof Hoder. Automated Reasoning in Large Knowledge Bases. Master's thesis, Charles University, 2008.
9. Joe Hurd. First-Order Proof Tactics in Higher-Order Logic Theorem Provers. In *Design and Application of Strategies/Tactics in Higher Order Logics, number NASA/CP-2003-212448 in NASA Technical Reports*, pages 56–68, 2003.
10. Daniel Kuehlwein. A Calculus for Proof Representation Structures. Master's thesis, University of Bonn, 2008.
11. Edmund Landau. *Grundlagen der Analysis*. Chelsea Publishing Company, 1930.
12. Roman Matuszewski and Piotr Rudnicki. Mizar: the first 30 years. *Mechanized Mathematics and Its Applications*, 4:2005, 2005.
13. Jia Meng and Lawrence C. Paulson. Lightweight relevance filtering for machine-generated resolution problems. *J. Applied Logic*, 7(1):41–57, 2009.
14. Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer, 2002.
15. S. Schulz. E – A Brainiac Theorem Prover. *Journal of AI Communications*, 15(2/3):111–126, 2002.
16. G. Sutcliffe. The TPTP Problem Library and Associated Infrastructure: The FOF and CNF Parts, v3.5.0. *Journal of Automated Reasoning*, 43(4):337–362, 2009.
17. Geoff Sutcliffe and Yury Puzis. SRASS - A Semantic Relevance Axiom Selection System. In *CADE-21: Proceedings of the 21st international conference on Automated Deduction*, pages 295–310, Berlin, Heidelberg, 2007. Springer.
18. Josef Urban, Geoff Sutcliffe, Petr Pudlák, and Jirí Vyskocil. MaLARea SG1- Machine Learner for Automated Reasoning with Semantic Guidance. In *IJCAR*, pages 441–456, 2008.
19. K. Verchinine, A. Lyaletski, and A. Paskevich. *System for Automated Deduction (SAD): a tool for proof verification*, volume 4603 of *LNCS*, pages 398–403. Springer, July 2007.
20. Konstantin Vershinin and Andrey Paskevich. ForTheL - the language of formal theories. *International Journal of Information Theories and Applications*, 7(3):120–126, 2000.
21. Makarius Wenzel. *Isabelle/Isar - a generic framework for human-readable proof documents*, volume 10(23) of *Studies in Logic, Grammar and Rhetoric*. University of Białystok, 2007.
22. L. Wos. The problem of definition expansion and contraction. *J. Autom. Reason.*, 3(4):433–435, 1987.
23. Claus Zinn. *Understanding Informal Mathematical Discourse*. PhD thesis, Friedrich-Alexander-Universitt Erlangen Nürnberg, 2004.