

Understanding Mathematical Discourse

Claus Zinn

Lehrstuhl für Künstliche Intelligenz (IMMD VIII)
Universität Erlangen-Nürnberg

Abstract

Discourse Understanding is hard. This seems to be especially true for mathematical discourse, that is proofs. Restricting discourse to mathematical discourse allow us, however, to study the subject matter in its purest form. This domain of discourse is rich and well-defined, highly structured, offers a well-defined set of discourse relations and forces/allows us to apply mathematical reasoning.

We give a brief discussion on selected linguistic phenomena of mathematical discourse, and an analysis from the mathematician's point of view. Requirements for a theory of discourse representation are given, followed by a discussion of proofs plans that provide necessary context and structure. A large part of semantics construction is defined in terms of proof plan recognition and instantiation by matching and attaching.

1 Introduction

Human-based proof verification. Understanding a mathematical discourse means to being able to verify the correctness of the given mathematical argument. Verifying a proof is by no means trivial and often time consuming. As DAVIS shows, only some mathematicians would volunteer to check a fifty-page proof [7]. Often, only a small number of mathematicians of some sub-discipline have the necessary qualification to check a proof of their domain completely for correctness. The verification process itself is error-prone. It is therefore inevitable that even many published proofs are either incomplete and/or error-prone.

Verifying a proof is also of social nature [9]:

“The acceptance of a theorem by practising mathematicians is a social process which is more a function of understanding and significance than of rigorous proof.”

HANNA gives some criteria why mathematicians accept a proof [9, p. 70]:

- 1. They understand the theorem, the concepts embodied in it, its logical antecedents, and its implications. There is nothing to suggest it is not true;*
- 2. The theorem is significant enough to have implications in one or more branches of mathematics (and is thus important and useful enough to warrant detailed study and analysis);*
- 3. The theorem is consistent with the body of accepted mathematical results;*
- 4. The author has an unimpeachable reputation as an expert in the subject matter of the theorem.*

5. *There is a convincing mathematical argument for it (rigorous or otherwise), of a type they have encountered before.*

If there is a rank order of criteria for admissibility, then these five criteria all rank higher than rigorous proof.

HANNA argues further that the “*mathematician is much more interested in the message embodied in the proof than its formal codification and syntax. The mechanics of proof are seen as a necessary but ultimately less significant aspects of mathematics. Certainly being able to follow the steps of a proof is not the same as understanding it*”. HANNA then cites Bourbaki¹:

“A proof is not really ‘understood’ as long as he (the mathematician) has only verified the correctness of the deductions involved step by step, without trying to understand clearly the ideas which led to the construction of this chain of deductions in preference to all others”

In 1976, ULAM estimated the number of published theorems and their proofs by 200.000 [19]. According to Hanna [9], only a small part of these theorems get recognized by the mathematical community and accepted as true statements. Will we have, in which distant future, computer programs which do the proof reading, or which assist mathematicians in doing so?

Machine-based proof verification. Imagine a machine that is capable to understand mathematical discourse. You can give a natural language proof for some theorem to the machine, it then analyses the input and communicates its knowledge about what it has read, analysed and understood. Of course, building a machine that understands discourse is one of the main goals of Natural Language Processing. Restricting discourse to mathematical discourse allows us, however, to study the subject matter in its purest form. Fig. 1 depicts the main characteristics of mathematical discourse.

-
- + well-defined and rich domain of discourse
 - + highly structured mathematical discourse
 - + well-defined and small set of discourse relations
 - + mathematical reasoning
-

Figure 1: Mathematical discourse

Building an ontology for a mathematical domain, say elementary number theory, is much easier than for a more natural domain of discourse, say, for the reservation/booking domain. This is because mathematicians are used to define concepts before using them — respecting certain standards on *how to define*. A mathematical discourse is a highly structured form of discourse. Mathematicians agree on a variety of different methods for *how to prove* theorems. Often, the form and content of the theorem clearly indicate applicable proof methods. From the linguistic point of view, these proof methods define discourse plans which serve as good estimates on what to expect next, and therefore, to a large extent, facilitate the task of discourse understanding. Equally, discourse markers help to identify the logical relations that hold between parts of the proof. To being able to follow the reasoning line of the proof author, it is necessary to model mathematical reasoning itself — a task, however, which we consider much easier than modeling common sense reasoning of everyday discourses.

But what does it mean to understand a mathematical argument? In contrast to Hanna’s criteria, we propose two answers:

Operational: the proof understander is able

- to answer questions about the proof accurately;
- to extract the main proof idea;
- to detect gaps and flaws in the argumentation line;
- to give a more rigorous proof.

Formal: the proof understander is able to generate a formal proof by using the informal mathematical argument.

Note that it might well be the case that the “formal semantics” is a prerequisite for implementing the “operational semantics”.

Formal proofs vs. informal proofs. It is interesting to compare informal proofs taken from textbook proofs with formal proofs, generated automatically/interactively by/with a proof system. This gives us first ideas which difficulties one has to face.

- (1) A formal proof is written in a formal language, the latter can be easily described in less than one page. An informal proof is written in an informal language, say English, having a large set of stylised syntactic constructions².
- (2) In formal proofs there is no ambiguity. In informal proofs there should be no ambiguity, but there is.
- (3) In a formal proof, for each proof step, it is explicitly given which conclusion is derivable by which set of premises and by which inference rule. In an informal proof, many proof steps are omitted or incomplete (incomplete set of premises, and lack of reference to inference rules used).
- (4) Finding informal or formal proofs is not trivial.
- (5) In a formal proof, the theory in which the proof is stated is explicitly stated. The theory, and nothing else, defines the context. In an informal proof, there is no full and explicit theory that one can refer to. The context is to be completed by the proof reader.
- (6) Formal proofs are structured (e.g., resolution graphs, natural deduction trees, semantics tableaux). Informal proofs are structured, too (e.g., a proof per induction consists of induction base case, induction hypothesis and induction step, the first and the latter contain subproofs themselves).
- (7) In a formal proof, form matters. In an informal proof, meaning matters.
- (8) Machines are good to verify formal proofs, but currently cannot check informal proofs. Humans are good to verify informal proofs, and bad in verifying formal proofs.

In the remainder of this article, I will (i) discuss a selection of linguistic phenomena that occur in informal proofs; (ii) analyse informal proofs mathematically; (iii) define requirements for a theory of discourse representation capturing the specialties of mathematical discourse; (iv) introduce proof plans that introduce necessary context and structure for proof understanding.

2 An analysis of informal proofs

In this section, we give an analysis of mathematical discourse from both the linguistic and mathematical point of view.

2.1 A linguistic analysis of informal proofs

We discuss only problems that are typical for informal mathematical discourse.

2.1.1 Free and bound, anonymous and named variables

Variables and unknowns. In predicate logic, to determine if a variable is free or bound, we only have to analyse the form of the statement, and not at all its meaning. So, in $\forall y : f(x, y)$ the variable x is free, and the variable y is bound. In ordinary mathematics, from the statement alone, there is no way to tell if x is read to be a variable or an unknown. For example, in ordinary mathematics, the statement $x^2 - 1 = (x + 1)(x - 1)$ is true for all values of x , and normally, one would assume x being used as a variable. In the statement $x^2 - 4x + 3 = 0$ one would assume x being used as an unknown, and its value is to be determined. The matter gets complicated by a logical principle which allows unknowns become variables and vice versa.

Look at the following discourse.

- THEOREM 3 (EUCLID'S FIRST THEOREM). *If p is prime, and $p \mid ab$, then $p \mid a$ or $p \mid b$.*
- Suppose that p is prime and $p \mid ab$. If $p \nmid a$ then $(a, p) = 1$, and therefore, by Theorem 24, there are an x and a y for which $xa + yp = 1$ or
- (1)

$$xab + ypb = b.$$

But $p \mid ab$ and $p \mid ypb$, and therefore $p \mid b$.

In this discourse, the p , a , and b that occur in the theorem are variables being universally quantified. But the p , a , b that occur later in the proof are unknowns. We could easily consistently rename all variables in the theorem without affecting the correctness of the proof (albeit the readability might be affected).

Note that the syntactic form of the statement that contains variables helps to determine if a variable occurs free, existentially bound, or universally bound. In 'there is an x and an y ', both x and y are existentially bound. Sometimes, however, the form of the statement is misleading. For the theorem in discourse (1) gets a generic reading that is, all occurring variables get a universal binding. But for the third sentence of this discourse, 'If $p \nmid a$ then $(a, p) = 1$ ', each of 'p', 'a', and 'b' occurs free.

Naming variables. Naming plays an important role in mathematical writing, and is partly determined by convention. In (1) we have several naming actions, for example, when asserting the existence of two natural numbers called x and y for which some equation holds. Expressing the same without the use of terms and formulae would be quite unreadable.

In [11], Lamport gives the following example:

- (2) a. There do not exist four positive integers, the last being greater than two, such that the sum of the first two, each raised to the power of the fourth, equals the third raised to the same power.

- b. There do not exist positive integers x, y, z , and n , with $n > 2$, such that $x^n + y^n = z^n$.

According to Lamport, (2a) reflects the style of seventeenth century mathematicians, and (2b) is the modern version. Variables are given names, and formulas are written in a more structured fashion. But names should only be introduced when necessary. As Krantz points out,

- (3) a. Every nonnegative real number has a square root.
 b. $\forall x \exists y : x \geq 0 \Rightarrow y^2 = x$

(3a) is to be preferred to (3b). Mathematically, these two assertions are equivalent, but linguistically, they are quite different. Parsing the natural language form will, most probably result into a logical form that contains an unary predicate *square_root* and a binary predicate *has*, both of which cannot be found in (3b).

2.1.2 Terms, and means to reference them

As we have seen already in the examples of Lamport and Krantz, it is possible to express the same using natural language, terms and formulae or both. A theory of semantics construction should be able to identify that ‘The greatest common divisor of a and p is 1’, and ‘ $(a,p) = 1$ ’ describe the same. This is different to the formulae ‘ $xa + yp = 1$ ’ and ‘ $xab + ypb = b$ ’ occurring in discourse (1), albeit the fact that term rewriting is semantics preserving.

A first and common kind of reference is connected with the use of terms and formulae in mathematical texts. In the examples (4a–4c), we have expressions that refer to term structures.

- (4) a. We obtain

$$p_1^{a_1} \dots p_i^{a_i - b_i} \dots p_k^{a_k} = p_1^{b_1} \dots p_{i-1}^{b_{i-1}} p_{i+1}^{b_{i+1}} \dots p_k^{b_k}.$$

The left-hand side is divisible by p_i , while the right-hand side is not.

- b. We have $m = 2^{b_1} 3^{b_2} \dots p_j^{b_j}$, with every b either 0 or 1. There are just 2^j possible choices of the exponents and so not more than 2^j different values of m .

- c. We have

$$2^p = (1 + 1)^p = 1 + \binom{p}{1} + \dots + \binom{p}{p} = 2 + \sum_1^{p-1} \binom{p}{l}.$$

Every term on the right, except the first, is divisible by p .

For resolving the referential expressions ‘the left-hand side’ and ‘the right hand side’, apparently, an equation has to be identified, and, respectively, its first or second argument returned.

In the second example, ‘every b ’ quantifies over the set $\{b_1, b_2, \dots, b_j\}$, equally referred to by ‘the exponents’. To handle these referential expressions, the term structure as well as domain knowledge has to be taken into account to form a set of terms that is to be returned as value of this referential expression. Apparently, in example (4c) ‘every term on the right, except the first’ is ambiguous, and requires also advanced grouping mechanism.

2.1.3 Formulae, and means to reference them

In mathematical discourse, we can refer to formulae by proper names. In (1), both ‘Theorem 3’ and ‘Euclid’s first theorem’ refer to the formulae

$$\forall p \forall a \forall b : \text{prime}(p) \wedge \text{div}(p, \text{mult}(a, b)) \rightarrow \text{div}(p, a) \vee \text{div}(p, b).$$

To access a sub-formulae of this formula we could use ‘the premise’, ‘the first part of the premise’ and various other constructions.

Proposition anaphora are common in mathematical texts. Consider the fundamental theorem of arithmetic and one of its proofs in (5).

THEOREM 2-2. Every integer $a > 1$ can be represented as a product of one or more primes.

(5) *Proof:* The theorem is true for $a = 2$. Assume it to be true for $2, 3, 4, \dots, a - 1$. If a is prime, we are through. Otherwise a has a divisor different from 1 and a , and we have $a = bc$, with $1 < b < a$, $1 < c < a$. The induction hypothesis then implies that

$$b = \prod_{i=1}^s p'_i, \quad c = \prod_{i=1}^t p''_i,$$

with p'_i, p''_i primes and hence $a = p'_1 p'_2 \dots p'_s p''_1 \dots p''_t$.

We have underlined three referential expressions, all of which are proposition anaphora. We need to introduce a discourse referent for each of ‘the theorem’, ‘it’, and ‘the induction hypothesis’, all of which are of propositional type. The propositional type is either induced by the verb phrase ‘being true for’ – only propositions can be true; or by ‘implies’ – only propositions can imply other propositions.

Semantics construction is complex. The sentence ‘The theorem is true for $a = 2$ ’ means ‘(It is true that) 2 can be represented as a product of one or more primes’ (resolving ‘the theorem’ to the abstract discourse entity that refers to (the content of) the theorem, and instantiating that universally quantified proposition with 2). Semantics construction for the theorem could yield $\forall a, a > 1, \exists p_1 \dots \exists p_n, a = p_1 \dots p_n \wedge \forall i, 1 < p_i \leq a \wedge \text{prime}(p_i)$.

2.1.4 Discourse markers and discourse relations.

The discourse (5) contains quite a lot of clue words (e.g., *if, otherwise, assume, implies, hence*), and cue phrases (e.g., *is true for, we are through*).

The third proof sentence, starting with *if*, could be read as follows: introduce the assumption *a is prime* and conclude *we are through* (signaling proof termination). However, this sentence has to be embedded into a larger context. In fact, *if* introduces a proof per cases construct, where *a is prime* defines the first case, and where the cue phrase *we are through* terminates this case segment. Obviously, *we are through* does not end the whole proof (there are still some sentences to be processed), so it can only terminate a sub-discourse (this first case). The cue phrase *otherwise* initiates the second case and is a kind of ellipsis which has to be reconstructed to: *a is not prime*. The cue word *and* is not a logical conjunction, but belongs to the cue phrase *and we have* that signals forward reasoning. The last marker of the textbook proof is *hence*, indicating that the subsequent phrase is a logical consequence of the former. Note that the last phrase, $a = p'_1 p'_2 \dots p'_s p''_1 \dots p''_t$, terminates the proof because it is the last sentence; and it says

that we were able to represent for an arbitrary number a its product of primes representation.

Reconsider the phrase (6a), truncated from discourse (5), being changed into (6b) and (6c).

- (6) a. The induction hypothesis then implies that $b = \prod_{i=1}^s p'_i$.
- b. Obviously $b = \prod_{i=1}^s p'_i$.
- c. It follows that $b = \prod_{i=1}^s p'_i$.

In all these cases, we have a formula, $b = \prod_{i=1}^s p'_i$ that is supposed to be a logical consequence of some other statements. However, the exact nature of the consequence relation remains unclear in the sentences (6b–6c). In (6b), we are told that the equation is obviously true. In (6c), we are told that the equation follows from some former statement, which however, is not given. Only in (6a), we are given a justification for $b = \prod_{i=1}^s p'_i$. But, the induction hypothesis might only be a necessary, but probably not a sufficient condition for this equation. In fact, in all these three given phrases we have to identify the complete set of premises that allows us to conclude $b = \prod_{i=1}^s p'_i$. The task is hard since we cannot expect to find all premises necessary to conclude some statement to be explicitly mentioned in the textbook proof.

It shows that discourse markers help to identify segmentation boundaries, the status of an assertion, and the discourse relation that holds between sentences. To fully determine a discourse relation we need to make explicit or add knowledge that is only implicitly stated or missing.

2.2 Mathematical analysis

We give now a mathematical analysis of discourse (5). The theorem has a form similar to *For every integer $n \geq n_0$, the predicate $P(n)$ holds*. One proof method that can be used to prove a theorem of that form is *mathematical induction*³. It consists of three steps:

1. Verify that P holds for n_0 .
2. Assume $P(n - 1)$ is true.
3. Show that $P(n)$ holds.

However, a variation of mathematical induction, referred to as *generalized induction* is used in the proof under study⁴:

1. Assume $P(i)$ for $n_0 \leq i < n$.
2. Show that $P(n)$ holds.

Knowing the proof method of generalized induction, it is easy to understand the proof. We find the induction hypothesis in the second phrase of the proof. The rest of the proof is the induction step, where $P(n)$ is proved using the fact that $P(i)$ is true for every i between 2 and n . However, it is the first phrase of the informal proof that is hard to understand and misleading. According to the proof method of generalized induction, it cannot be the base case, since the base case is already contained in the induction hypothesis.

Let us write down the proof in more detail. If we break compound phrases into elementary phrases omitting discourse markers (which, nonetheless, are used to interpret the proof), we obtain:

- (7) a. The theorem is true for $a = 2$.
 b. Assume it to be true for $2, 3, 4, \dots, a - 1$.
 c. a is prime.
 d. we are through.
 e. a is not prime.
 f. a has a divisor different from 1 and a .
 g. $a = bc$
 h. $1 < b < a$
 i. $1 < c < a$
 j. $b = \prod_{i=1}^s p'_i$, with p'_i primes.
 k. $c = \prod_{i=1}^t p''_i$, with p''_i primes.
 l. $a = p'_1 p'_2 \dots p'_s p''_1 \dots p''_t$.

The necessity for the first sentence (7a) will be explained in a more formal account. In (7b), we have the induction hypothesis⁵. The propositions (7c) to (7l) constitute the induction step. Statement (7c) initiates a proof per cases: either a is prime or a is not prime. For (7d), the reasoning is: if a is prime then a is a product of one or more primes, and this concludes the proof for this case. The second case is implicitly indicated in the proof with *otherwise*, which we reconstructed in (7e) to *a is not prime*. The statement in (7f) follows from definitional expansion on the concept of prime using (7e) and negation. In (7g–7i), definitional expansion for the concept of divisor, using (7f), is performed. In (7j–7k), the induction hypothesis (7b) is applied two times, for b using (7h) and for c using (7i). In (7l), simple rewriting takes place using (7g) and (7j–7k).

A more detailed account. The statement to be proven, ‘Every integer $a > 1$ can be represented as a product of one or more primes.’ has the logical form

$$(8) \quad \forall n : P(n)$$

The proof is by strong induction. Strong induction can be expression formally by

$$(9) \quad \forall n [(\forall k < n : P(k)) \rightarrow P(n)] \rightarrow \forall n : P(n)$$

If we can prove

$$(10) \quad \forall n [(\forall k < n : P(k)) \rightarrow P(n)]$$

then by (9), (10) and Modus Ponens we get (8).

The proof skeleton of (10) is:

let n be arbitrary

assume $\forall k < n : P(k)$, i.e., $\forall k < n : k > 1 \rightarrow \text{prod_of_primes}(k)$
 conclude $P(n)$, i.e, $n > 1 \rightarrow \text{prod_of_primes}(n)$

Since n has been chosen arbitrarily, the proof holds for all n (\forall -Introduction).

However, note that for $n = 2$, we have that $\forall k < n [k > 1 \rightarrow \text{prod_of_primes}(k)]$ in trivially true since there is no k such that $k < n$ and $k > 1$. Therefore, the proof author make a proof per cases.

Case 1 The first case starts with ‘The theorem is true for $a = 2$ ’. Since 2 is prime it is a product of one or more primes.

Case 2 The second case handles $n > 2$. Here, a proof per strong induction is performed as outlined above.

It shows that discourse understanding in the mathematical domain requires a substantial amount of reasoning.

2.3 Requirements for a theory of discourse representation

From the programmer’s point of view, a data structure is needed for maintaining entities being introduced and referred to during discourse, and for conditions that must hold for these entities. This data structure need to be structured because mathematical discourse is structured; and sorted because anaphoric expressions can refer to terms (constants, variables, sets of terms), concepts, propositions, and argumentation structures.

From the representational view, the language for discourse representation must

- be able to distinguish between bound and free variables;
- have different kinds of discourse referents
 - for terms⁶;
 - for propositions (with modality being assumed, postulated, or derived);
 - for argumentation structures (not being discussed here: e.g., ‘the second case’, ‘the first alternative’, ‘repeating the argument’);
- provide a computational component for accessing and grouping terms and sub-formulae;
- have a naming mechanism;
- have means to describe discourse relations that hold between propositional discourse referents;
- have a deductive component, modeling mathematical reasoning, allowing to compute discourse relations between given propositions, or verifying if a given discourse relations really holds;

We believe that Discourse Representation Theory [10, 20] can be adapted to fit our requirements. In the remainder of the paper, however, we will focus on the use of proof plans that are highly valuable for structuring discourse.

3 The use of proof plans

Our approach is characterized by the use of discourse plans (better: *discourse plan schemas*), representing proof methods, and the fact that we view each (elementary and complex) statement as an abstract discourse entity that can be referred to.

The use of discourse plans is twofold. First, to a large extent, they help to structure the proof. And, discourse plans are indispensable if the text under study lacks discourse markers that explicitly signal structure, or when discourse markers give rise to ambiguity. Second, discourse

plans themselves introduce discourse entities: terms, assumptions, goals, and references to sub-proofs which allows us to handle referential expressions like, for example, ‘the theorem’ and ‘the induction hypothesis’, we encountered before. Since discourse plans add context, it is straightforward to represent them in the same representation as the discourse.

Generally, proofs have a hierarchical structure. Since proofs have a recursive structure, proof plans consist of sub-plans, and a major step towards discourse understanding is to identify this structure.

In our approach, semantics construction is a process that can be described as follows. It starts with analysing the theorem, and the identification of applicable proof plans. For the theorem of discourse (1), we can apply a proof plan scheme for universally quantified formulae:

$$(11) \quad \frac{\forall x_1 \dots \forall x_n : P(x_1, \dots, x_n)}{\text{LET } \bar{x}_1 \dots \forall \bar{x}_n \text{ be arbitrary} \\ \text{PROVE } P(\bar{x}_1, \dots, \bar{x}_n)}$$

Above the line, we have the form of a statement to be proven. Below the line, we have the form of an argumentation structure for proving the statement. The formulae after PROVE defines the remaining proof obligation.

Matching it with the logical form of the theorem of discourse (1) results into:

$$(12) \quad \frac{\forall p \forall a \forall b : \text{prime}(p) \wedge \text{div}(p, \text{mult}(a, b)) \rightarrow \text{div}(p, a) \vee \text{div}(p, b)}{\text{LET } \bar{p}, \bar{a}, \bar{b} \text{ be arbitrary} \\ \text{PROVE } \text{prime}(\bar{p}) \wedge \text{div}(\bar{p}, \text{mult}(\bar{a}, \bar{b})) \rightarrow \text{div}(\bar{p}, \bar{a}) \vee \text{div}(\bar{p}, \bar{b})}$$

For the remaining proof obligation we can try either (13a) or (13b), both proofs per elimination.

$$(13) \quad \begin{array}{l} \frac{A \rightarrow B_1 \vee B_2}{\text{a. ASSUME } A \\ \text{ASSUME } \neg B_1 \\ \text{PROVE } B_2} \\ \\ \frac{A \rightarrow B_1 \vee B_2}{\text{b. ASSUME } A \\ \text{ASSUME } \neg B_2 \\ \text{PROVE } B_1} \end{array}$$

As we have demonstrated, theorem analysis defines the context for interpreting the proof. For our example, we get the following proof context:

$$(14) \quad \frac{\forall p \forall a \forall b : \text{prime}(p) \wedge \text{div}(p, \text{mult}(a, b)) \rightarrow \text{div}(p, a) \vee \text{div}(p, b)}{\text{LET } \bar{p}, \bar{a}, \bar{b} \text{ be arbitrary} \\ \text{PROVE } \text{prime}(\bar{p}) \wedge \text{div}(\bar{p}, \text{mult}(\bar{a}, \bar{b})) \rightarrow \text{div}(\bar{p}, \bar{a}) \vee \text{div}(\bar{p}, \bar{b}) \\ \text{ASSUME } \text{prime}(\bar{p}) \wedge \text{div}(\bar{p}, \text{mult}(\bar{a}, \bar{b})) \\ \text{ASSUME } \neg \text{div}(\bar{p}, \bar{a}) \\ \text{PROVE } \text{div}(\bar{p}, \bar{b})}$$

During proof processing other criteria of proof plan applicability will be employed: e.g., referential expressions that refer to abstract discourse entities should be resolvable to abstract discourse entities introduced by a proof plan. This is used to decide between ambiguous proof plan readings.

For processing the proof of discourse (1), we break its complex sentences into elementary assertions, for each of which an abstract discourse referent is introduced. Discourse markers that connected these elementary assertions (e.g., *and*, *therefore*, *assume*) are maintained (attached to them) for further use. Now, we view each of the elementary discourse constituents as anaphoric referring to its place in the proof plan. The discourse markers, we kept them, define constraints for possible sites of constituent attachment. For example, the cue word *hence* indicates that the constituent it introduces, should be attached to the currently “active” proof plan. For the proof of discourse (1), we find that the first two assumptions ‘ p is prime and $p \mid ab$ ’ (suppose phrase), and ‘ $p \nmid a$ ’ (premise of if-then phrase) met the expectations of proof plan (14), and can be attached. The remainder of the proof is now expected to met the proof obligation ‘ $p \mid b$ ’ (which indeed is the case).

For example, for attaching $(a, p) = 1$, the beginning of the remainder of the proof, we have only one possible attachment point. It is the beginning of the sub-discourse for proving $div(\bar{p}, \bar{b})$.

We get

$$\begin{array}{l} \forall p \forall a \forall b : prime(p) \wedge div(p, mult(a, b)) \rightarrow div(p, a) \vee div(p, b) \\ \hline \text{LET } \bar{p}, \bar{a}, \bar{b} \text{ be arbitrary} \\ \text{PROVE } prime(\bar{p}) \wedge div(\bar{p}, mult(\bar{a}, \bar{b})) \rightarrow div(\bar{p}, \bar{a}) \vee div(\bar{p}, \bar{b}) \\ (15) \quad \text{ASSUME } prime(\bar{p}) \wedge div(\bar{p}, mult(\bar{a}, \bar{b})) \\ \quad \text{ASSUME } \neg div(\bar{p}, \bar{a}) \\ \quad \text{PROVE } div(\bar{p}, \bar{b}) \\ \quad (\bar{a}, \bar{p}) = 1 \text{ by some justification } \Gamma. \end{array}$$

The discourse marker ‘then’ indicates some forward/backward reasoning without introducing structure. We view ‘then’ as defining a binary discourse relation, $then((\bar{a}, \bar{p}) = 1, \Gamma)$, where Γ can be viewed as an anaphoric entity that refers to all necessary premises which are “logically” necessary for drawing the conclusion $(\bar{a}, \bar{p}) = 1$. For resolving this referential expressions, we have two accessible assumptions (and the ‘a’, ‘p’ of ‘ $(a, p) = 1$ ’ resolve to the arbitrarily chosen \bar{p} and \bar{a} introduced by the proof plan).

Discourse markers like *either* or *otherwise* indicate that a new subproof has to be opened and attached. Note that elementary assertions themselves, e.g., the constituent *we are through*, can mark the end of a sub-proof so that consecutive phrases can no longer been attached to this part of the proof which become marked as “closed”.

Fig. 2 depicts a proof representation structure for discourse (5). It begs some similarities with the style of proof writing advocated by Lamport [11]. This structure offers a convenient way to refer to a formula. This propositional discourse referents are just the proof line numbers. Also, the representation is structured indicated by indentation and the proof line numbering. It offers an intuitive notion of accessibility for both names (introduced by let) and formulae (either assumed or derived). For example, in proof step **1.2.1.2.2** we can use assumption **1.2.1.2.1** but not assumption **1.2.1.1.1**. In proof step **1.1.1**, we cannot refer to the name ‘b’, since it is only introduced later in sub-discourse **1.2.1.2**.

Note that the proof representation structure is still incomplete. That is, for some names introduced we have to decide if they are either bound or free (e.g., the ‘b’ and ‘c’ being intro-

LET a be universally quantified
 LET k be universally quantified st $k < a$
 PROVE $(k > 1 \rightarrow \text{prod_primes}(k)) \rightarrow (a > 1 \rightarrow \text{prod_primes}(a))$

LET \bar{a} be arbitrary

1. induction_hypothesis

ASSUME $k > 1 \rightarrow \text{prod_primes}(k)$

1.1. first_case

1.1.1. ASSUME $\bar{a} = 2$

1.1.2. QED

1.2. second_case

1.2.1. ASSUME $\bar{a} \neq 2$

1.2.1.1. first_case

1.2.1.1.1. ASSUME $\text{prime}(\bar{a})$

1.2.1.1.2. QED

1.2.1.2. second_case

1.2.1.2.1. ASSUME $\neg \text{prime}(\bar{a})$

LET X be ?

1.2.1.2.2. $\text{divisor}(X, \bar{a})$

LET b, c be ?

1.2.1.2.3. $\bar{a} = bc$

1.2.1.2.4. $1 < b < \bar{a}$

1.2.1.2.5. $1 < c < \bar{a}$

LET $p'_1, p'_2 \dots, p''_1, p''_2 \dots$ be ? and prime

LET s, t, i be ?

1.2.1.2.6. $b = \prod_{i=1}^s p'_i$

1.2.1.2.7. $c = \prod_{i=1}^t p''_i$

1.2.1.2.8. $\bar{a} = p'_1 p'_2 \dots p'_i p''_1 \dots p''_t$ (QED)

1.2.2. QED

2. QED

Figure 2: A proof representation structure

duced in sub-discourse 1.2.1.2). Also, all justifications for derived statements are missing, so that discourse relations are still not computed.

4 Related work

The idea to formally verify textbook proofs has been propagated by McCarthy: “*Checking mathematical proofs is potentially one of the most interesting and useful applications of automatic computer*” [12]. Abrahams, one of his students, wrote a program that initially was intended to analyse textbook proofs. However, as Abrahams noted in [1], this task “*would require far more intelligence than is possible with the current state of the programming art*”. Therefore, so Abrahams, “*the user must create a rigorous, i.e., completely formalized, proof that he believes represents the intent of the author of the textbook proof, and use the computer to check this rigorous proof*”. Abrahams points further out that “*it is a trivial task to program a computer to check a rigorous proof; however, it is not a trivial task to create such a proof from a textbook proof*”. Abrahams was right. In his implementation and in all later projects (e.g., the Mizar project lead by A. Trybulec [14]), proofs had to be written in a *formal language* using a *restricted set of proof construction commands* in order to verify them. A human user is required to fulfill the formalization task.

Similar attempts to parse texts in the mathematics and physics domain are [4] and [5]. Similar to the argument in [5], our purpose is to study natural language understanding in conjunction with automated reasoning. What formal representation can be obtained from textbook proofs and what is needed to formally verify textbook proofs? The most recent work is Simon’s PhD thesis [16] which, however, fails to seriously address linguistic issues.

Abstract discourse entities are treated by [3]. Asher differs between fact anaphora, event anaphora, concept anaphora and proposition anaphora. Only the latter has been treated in this paper. The DRS construction process we described begs some similarities to the one described in [3]. A major difference is that, due to our domain, we can assume the existence of discourse plans that establish frames in which discourses must take place.

A more general task than analyzing textbook proofs is to process arbitrary argumentative discourse [6]. Cohen considers a one-way communication where there is a speaker who tries to convince a hearer of some particular argument. The argument understanding system plays the role of the hearer and its task is to analyze the structure of the argument being presented by the speaker. Because argumentative discourse is goal oriented, the discourse has a logical structure. Cohen proposes a three-component model for argument analysis: (i) a theory of expected coherent structure; (ii) a theory of linguistic clue interpretation; and (iii) a theory of evidence relationships. For our domain, (i) is the proof plan; for (ii) we gave linguistic clues; and for (iii) evidence is replaced by establishing a logical relation between propositions of the proof.

An influential theory of discourse structure consisting of the three components *linguistic structure*, *intentional structure* and *attentional state* is developed in [8]. The intentional structure might be matched with the purpose in mathematical discourse to fulfill all proof obligations, whereas the attentional state, in our domain and approach, might coincide with the currently active proof structure being under refinement.

The necessity to perform a multi-level discourse analysis is discussed in [13]. We agree to the fact that a crucial prerequisite to understand the course of the argumentation within a proof is to identify the discourse relations between sentences of that discourse. Deriving the

discourse relations of a given textbook proof means reconstructing the intentional structure (describing how sentences within a discourse segment contribute to a common discourse purpose, namely how to decrease the proof obligations) and the informational structure (describing how sentences within a segment are related to each other by some relation, namely a logical consequence relation) of the proof.

In [2], it is argued that a model of discourse plans and their recognition is needed to explain and understand *helpful behaviour*. The problem of discourse understanding can be viewed as a plan recognition problem. The problem of plan recognition is, according to [15], *to take as input a sequence of actions performed by an actor and to infer the goal pursued by the actor and also to organize the action sequence in terms of the plan structure*. For our domain, the actor is the author of the textbook proof, the input is a sequence of sentences, the textbook reader is the plan recognizer.

5 Conclusion and future work

It is argued that mathematical discourse is an ideal domain for studying discourse. We analysed a textbook proof on elementary number theory from both the linguistics and mathematics perspective. It shows, no surprise, that domain knowledge, linguistic knowledge, discourse knowledge, and reasoning are all necessary to successfully build discourse understanders. We proposed to represent proof plans in the same representation as discourse and sketched our approach that describes semantics construction in terms of proof plan recognition and instantiation by matching and attaching. This has to be worked out. Besides the implementation task, future work includes to view discourse representations as first-class objects that can be manipulated in various ways: (i) restructure the proof, i.e., isolate an embedded lemma and its proof and replace it by a reference to that lemma, and (ii) analyse defective discourses and try to repair them.

Notes

¹cf. Bourbaki, N., *The architecture of mathematics*, In F. Le Lionnais (ed.), Great currents of mathematical thought (Vol. 1), New York: Dover, 1971.

²There is a booklet of J. Trzeciak that contains hundreds of standard phrases that allow non-native speakers of English to write mathematical arguments in English [18], enriched with terms and formulae.

³See [17], for an introduction to proof methods.

⁴Because in standard mathematical induction, in order to prove $P(n+1)$ we can only refer to $P(n)$. In generalized induction, we can assume that $P(i)$ for every $n_0 < i \leq n$.

⁵Here seems to be a minor flaw. Since it is shown in (7a) that 2 is a product of primes, it is not necessary to assume it in (7b).

⁶To only justify this for constants: in the mathematical domain, a proper name picks out at least one thing, and at most one thing. If we handle constants like proper names in a standard DRT way [10], semantics construction for the sentence ‘2 is prime’ would yield the DRS

x
x = 2
prime(2)

which translates into the logical form $\exists x : x = 2 \wedge \text{prime}(x)$. We believe this representational result unsatisfactory for several reasons. Most of all, we believe the logical form unnatural, and the truth-equivalent $\text{prime}(2)$ is the logical form which is to be preferred, because it is much simpler. In addition, to capture the uniqueness of '2', extra presuppositional information has to be supplied.

References

- [1] P. W. ABRAHAMS, *Machine verification of mathematical proofs*, PhD thesis, MIT, 1963.
- [2] J. ALLEN AND C. PERRAULT, *Analyzing Intention in Utterances*, *Artificial Intelligence*, 15 (1980), pp. 143–178.
- [3] N. ASHER, *Reference to abstract objects in discourse*, Kluwer Academic Publishers, 1993.
- [4] D. G. BOBROW, *Natural language input for a computer problem solving system.*, PhD thesis, MIT, 1964.
- [5] A. BUNDY, L. BYRD, AND G. LUGER, *Solving mechanics problems using meta-level inference*, in 6th. Int'l Joint Conference on Artificial Intelligence, 1979, pp. 1017–1027.
- [6] R. COHEN, *Analyzing the Structure of Argumentative Discourse*, *Computational Linguistics*, 13 (1987), pp. 11–24.
- [7] P. J. DAVIS, *Fidelity in mathematical discourse: Is one and one really two?*, *American Mathematical Monthly*, 79 (1972).
- [8] B. GROSZ AND C. SIDNER, *Attention, Intention, and the Structure of Discourse*, *Computational Linguistics*, 12 (1986), pp. 175–204.
- [9] G. HANNA, *Rigorous proof in mathematics education*, Curriculum Series, The Ontario Institute for Studies in Education, 1983.
- [10] H. KAMP AND U. REYLE, *From Discourse to Logic*, Kluwer Academic Publishers, 1993.
- [11] L. LAMPORT, *How to write proofs*.
<http://www.research.digital.com/SRC/personal/LeslieLampport/proofs/proofs.html>, 1993.
- [12] J. MCCARTHY, *Computer programs for checking mathematical proofs*, in *Recursive Function Theory*, Proceedings of Symposia in Pure Mathematics, vol. 5, American Mathematical Society, 1962.
- [13] J. MOORE AND M. POLLACK, *A Problem for RST: The Need for Multi-Level Discourse Analysis*, *Computational Linguistics*, 18 (1992), pp. 537–544.
- [14] P. RUDNICKI, *An overview of the MIZAR project*, tech. rep., Dept. of Computer Science, University of Alberta, Edmonton, 1992.
- [15] C. SCHMIDT, N. SRIDHARAN, AND J. GOODSON, *The Plan Recognition Problem: An Intersection of Psychology and Artificial Intelligence*, *Artificial Intelligence*, 11 (1978), pp. 45–83.
- [16] D. L. SIMON, *Checking Number Theory Proofs in Natural Language*, PhD thesis, UT Austin, 1990.

- [17] D. SOLOW, *How to read and do proofs*, John Wiley & Sons, 1990.
- [18] J. TRZECIAK, *Writing mathematical papers in english*. Gdańsk Teacher's Press, Institute of Mathematics, Polish Academy of Science, 1993.
- [19] S. M. ULAM, *Adventures of a mathematician*, Scibner, New York, 1976.
- [20] J. VAN EIJCK AND H. KAMP, *Representing Discourse in Context*, vol. Handbook of Logic & Language, ch. 3, pp. 179–237.

Author's Address:

Claus Zinn

Friedrich-Alexander Universität Erlangen-Nürnberg

Lehrstuhl für Künstliche Intelligenz (IMMD VIII)

Am Weichselgarten 9, D - 91058 Erlangen

mail: zinn@immd8.informatik.uni-erlangen.de

www: <http://www8.informatik.uni-erlangen.de/IMMD8/staff/Zinn>