Computing Presuppositions and Implicatures in Mathematical Discourse

Claus Zinn
Division of Informatics, University of Edinburgh
email:zinn@cogsci.ed.ac.uk

Abstract

In any well-written mathematical discourse a certain amount of mathematical and meta-mathematical knowledge is presupposed and implied. We give an account on presuppositions and implicatures in mathematical discourse and describe an architecture that allows to effectively interpret them. Our approach heavily relies on proof methods that capture common patterns of argumentation in mathematical discourse. This pragmatic information provides a high-level strategic discourse understanding and allows to compute the presupposed and implied information.

1 Introduction

Mathematical discourse is an exciting text-genre. It comes with a precise language, much structure and a well-organised domain of discourse. The goal of mathematical discourse, or *informal proof*, is to establish truth and over the centuries well-developed patterns evolved that define the uses of argument in mathematics. Although the mathematical language is widely spoken, it never got much attention from the linguist community. But mathematical discourse offers an ideal testbed to test existing and to develop new linguistic theories!

In any well-written mathematical discourse, at any point of the argument, a certain amount of mathematical and meta-mathematical knowledge is presupposed and implied. The proof author, for ease of argument, takes much for granted. The proof reader is not only assumed to verify what is asserted explicitly (and its verification implies some higher degree of understanding), in order to do so, he must identify its presupposed and implied content. The communication of mathematical insight can be highly effective if author and reader know each other and work within the same subfield of mathematics.[1] Proofs that occur in elementary textbooks have to appeal to a wide audience of potential readers most of which are not experts in the subject being introduced. Here, the common ground

[1] As the mathematician and Fields Medal recipient Thurston says in [Thu94]:

> *"Within a subfield, people develop a body of common knowledge and known techniques. By informal contact, people learn to understand and copy each other's way of thinking, so that ideas can be explained clearly and easily. Mathematical knowledge can be transmitted amazingly fast within a subfield. When a significant theorem is proved, it often (but not always) happens that the solution can be communicated in a matter of minutes from one person to another within a subfield. The same proof would be communicated and generally understood in an hour talk to members of the subfield. It would be subject of a 15- or 20-page paper, which could be read and understood in a few hours or perhaps days by members of the subfield."*

between author and reader will be much smaller. In both target groups (specialised audience of experts or anonymous students), however, to *understand* the proof, a number of proof readers will struggle for hours or even days to untangle the details of a mathematical argument they consider complex or perhaps unintuitive. They find it difficult to keep track of the proof obligations, to identify the critical parts of the argument, or to reconstruct details that were not given explicitly.

Gricean Principles in Mathematical Discourse. The proof author will and must apply the *Gricean conversational principles* making his argument "such as is required, at the stage at which it occurs, by the accepted purpose or direction of the talk exchange in which you are engaged" [Gri68]. Now, how do the Gricean maxims of quantity, quality, relation and manner instantiate for mathematical discourse?

Quality. Of course, a defining cornerstone of mathematics is to respect the Gricean maxims of quality. Mathematicians do not assert the truth of some statement if they lack adequate evidence for it. In the very contrast to non-scientific domains of discourse, the truth of each assertion must be either established by proof, postulated by declaring it as axiom or conditionally assumed for the sake of argument. Proof is established by presenting a highly ordered and structured argument. Moreover, the uses of arguments in mathematical discourse is conventionalised by proof methods imposing such structure.

Manner. To comply with being perspicuous, mathematicians defined and still define their own special purpose language. Over the centuries special syntactic constructions and notations evolved that facilitate concise and precise expression of mathematical statements. Each new concept or notation should be explicitly defined before its first use. Equally, any departure from existing definitions and notations should be made explicit. In mathematical discourse, due to its language, obscurity and ambiguity of expression can easily be avoided.[2]

Another aspect of the maxim of manner is the already mentioned conventionalised uses of arguments in mathematical discourse which relates to the Gricean imperative "be orderly!". In order to present a proof of a statement of the form $A \wedge B$ one first presents a proof for A, and then presents a proof for B. When one presents a proof per induction, one often starts by presenting the base cases, and then with presenting (the often more interesting and complex) step case.

Quantity and Relation. While the Gricean categories quality and manner seem to be easy definable for mathematical discourse, the maxims quantity ("make your contribution as informative as is required, but do not be over-informative!") and relation ("be relevant!") are not. For mathematical discourse, one might be inclined to distinguish relevancy and informativeness in the following way. Be relevant means: use only axioms and theorems and make only assumptions that are needed for the argument; being informative means: tell the proof reader only about the "interesting" axioms, theorems and assumptions used in the argument. Relevancy can be computed in an objective manner — any axiom, theorem or assumption used to logically derive the truth of some statement is relevant — informativeness depends on the proof reader and must be computed in function of his knowledge and interests.

[2] In [Lam93], Lamport gives the following example illustrating how the use of new syntactic constructions allowed for more brief and precise statements. The sentence

> *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.*

reflects the style of seventeenth century mathematicians. The sentence

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

is the modern version. Variables are given names, and formulas are written in a more structured fashion.

Clearly, to respect all of the Gricean principles at once is hard — how to be relevant and not over-informative? The proof author thus faces the difficult task to balance the perspicuity of the argument (make it clearly or easily understood, evident, transparent) with its lengthiness. Providing a copieuseness of detail can quite easily result in tedious and tiresome reading yielding to the "do not see the forest because of all these trees" phenomena. — Obviously, mathematical discourse must not violate the maxim of quantity. A well presented mathematical discourse permits the reader to study the argument at various levels of detail.[3]

In general, however, giving one typical informal textbook proof to N proof readers might result into N or more different interpretations of it. Some readers might have problems in understanding a mathematical discourse, because (i) the presented argument is not as informative as it should be (the argument contains a gap that the reader cannot fill in); (ii) the presented argument does not contain relevant information (the argument contains a gap the author should have provided); (iii) the presented argument contains mathematically and logically incorrect conclusions; (iv) the reader is not familiar with the notations and concepts being used in the argument; (v) the reader is not familiar with the proof methods being employed or fails to recognise them.

The easiness to create ambiguous statements (even in the special purpose language of mathematics), and to leave much of a mathematical argument unsaid lead Frege to his *Begriffsschrift* [Fre67] and Hilbert to his *logischen Grundlagen der Mathematik* [Hil22]: mathematics has to be expressed in a well-defined formal language using a well-defined set of formal reasoning schemes. In *formal mathematical discourse*, there is, by definition, no room for ambiguity. But formal mathematical discourse violates the Gricean maxim of quantity. It is tediously long, tiresome to read, and it is hard to extract its interesting parts. It also violates the Gricean maxim of manner: its notation is obscure and does not match the expressiveness of its informal counterpart[4]; its reasoning patterns do not match the kind of reasoning mathematicians perform when presenting informal proofs. These are reasons for the fact that computer-assisted proving has not been widely adopted in the standard mathematician's working environment.

Goal. The underlying idea of this research work is to tackle the problem of text understanding for the text genre of mathematical discourse. Taken its specific characteristics, can we built a system that understands mathematical discourse?

Our definition of "understanding mathematical discourse" is to be able to extract and determine its asserted, presupposed and implied content. The text understanding system must therefore be able to translate an informal mathematical discourse into a formal mathematical discourse, the latter representing the meaning of the former.

Overview. The rest of the paper is organised as follows. In section 2 we give an example proof and use it to highlight and concretise the points we have just made. In section 3 we describe the architecture of our system that is designed to understand informal mathematical discourse. This includes a discussion of its semantics/pragmatics interface, the use of *proof methods* and so-called *proof representation structures*. Text understanding requires much reasoning, and this is particularly true for the chosen text genre. Section 3 concludes by illustrating the amount of inference that has to be invested to achieve text comprehension in mathematics. Section 4 relates our approach to the work of others. Section 5 concludes.

[3] *Vide* our brief discussion of Lamport-style proof presentation in Section 4.

[4] Generally, this is the case — but much could be done to make formal languages more expressive and less obscure without compromising their qualities.

2 Examples

Before analysing mathematical discourse for its implicit content, have a look at the following example (taken from [AL98], adapted from [vdS92]):

(1) a. If baldness is hereditary, then Jack's son is bald.

b. If Jack has a son, then Jack's son is bald.

The presupposition that Jack has a son (triggered by *Jack's son*) is entailed by (1a), but not by (1b). That Jack is bald is clearly an implicature of (1a) and, to a lesser extent, could also been inferred from (1b). In (1b), the fact that Jack's son is bald could have several reasons, for example, he might prefer this hair style, or he might undergo a chemical therapy to treat his cancer disease, or he might be naturally bald. Uttering *Jack's son is bald* as in (1b) strongly suggests that Jack's son is bald because he is the son of Jack. And this opens the possibility to associate baldness with either heredity (as in (1a)) or the weird character of Jack (being a fan of this hair style) and his parental influence on his son.

If we interpret sentence (1a) in a context that contains the fact *Jack has three children*, we intuitively conclude that Jack has two daughters and a son. Uttering (1b) in the same context does not yield the same conclusion. Here, we might conclude that whoever utters (1b) does not know if Jack has a son (or two sons or three sons) or not. Note that uttering *Yes, but to which extent has Jack's baldness affected his career or his personal life?* presupposes that Jack is bald. Uttering this sentence in the context being established by (1a) *Jack's baldness* does not assert new information. Uttering it after (1b), however, makes it more likely that Jack's baldness is responsible for the baldness of his son. This example illustrates the strong interaction between asserted content, presuppositions and implicatures. Depending on the context, *Jack's son* can both give rise to presuppositions and implicatures.

The next discourse is taken from a standard textbook on elementary number theory [LeV65]; (2a) is the theorem to be proven, (2b-2f) its proof.

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

b. The theorem is true for $a = 2$.

c. Assume it to be true for $2, 3, 4, \ldots, a-1$.

d. If a is prime, we are through.

e. Otherwise a has a divisor different from 1 and a, and we have $a = bc$, with $1 < b < a$, $1 < c < a$.

f. The induction hypothesis then implies that $b = \prod_{i=1}^{s} p'_i$, $c = \prod_{i=1}^{t} p''_i$, with p'_i, p''_i primes and hence $a = p'_1 p'_2 \ldots p'_s p''_1 \ldots p''_t$.

Obvious presupposition triggers are *otherwise* (presupposing that a different case has been already been treated) and *the induction hypothesis* (presupposing that there is such an assumption). To handle this kind of presupposition a high-level proof understanding is required. For the given example, we have to recognise that the argument is a proof per Nœtherian induction (aka *course of values induction*) where (2c) is the induction hypothesis; and that a is a product of primes is the remaining proof obligation. Within the step case (2d-2f) a proof per cases is performed.

Cue phrases like *then implies*, *and we have*, and *and hence* all presuppose a logical relation between the clause they initiate and prior discourse. In a sense, the clue word *therefore* in clauses like *therefore A* anaphorically refers to a set of premises Γ such that A is a logical consequence of it. However, Γ will rarely completely consist of material that has been made linguistically explicit by the proof author. To establish logical consequence, the missing elements of Γ must be constructed by the proof reader.

The discourse also contains a number of more or less obvious implicatures. The *we are through* in (2d) marks the end of the first case: a is prime, therefore a is a product of primes, and this fulfils the remaining proof obligation in this case.

Sentence (2e) initiates the second case. The *Otherwise* ... in (2e) is elliptic and has to be reconstructed as *If a is not prime, then* .... The reconstruction *a is not prime* implies that *a has a divisor different from* 1 *and* a, a fact that the proof author explicitly asserts. The second part of (2e), $a = bc$ *with* $1 < b < a$, $1 < c < a$, is implied by a having divisors between 1 and a.

In (2f), the proof author claims that the induction hypothesis implies that both b and c can be represented as a product of primes. It must be inferred from the proof reader that representing a in terms of b and c fulfils the remaining proof obligation, namely that in $a = p'_1 p'_2 \ldots p'_s p''_1 \ldots p''_t$, a has actually been represented as a product of one or more primes.[5]

This short analysis already suggests that, in order to make sense of the asserted content, reasoning is necessary to infer the presupposed and the implied content. A text understander must be able to identify how a given sentence relates to a given proof context, and to determine this relation it will be necessary to add content that the proof author did not supply. In the next section we describe a framework for mechanically computing this information.

3 Architecture of VIP

Fig. 1 depicts the architecture of VIP, our system for verifying informal proofs. VIP

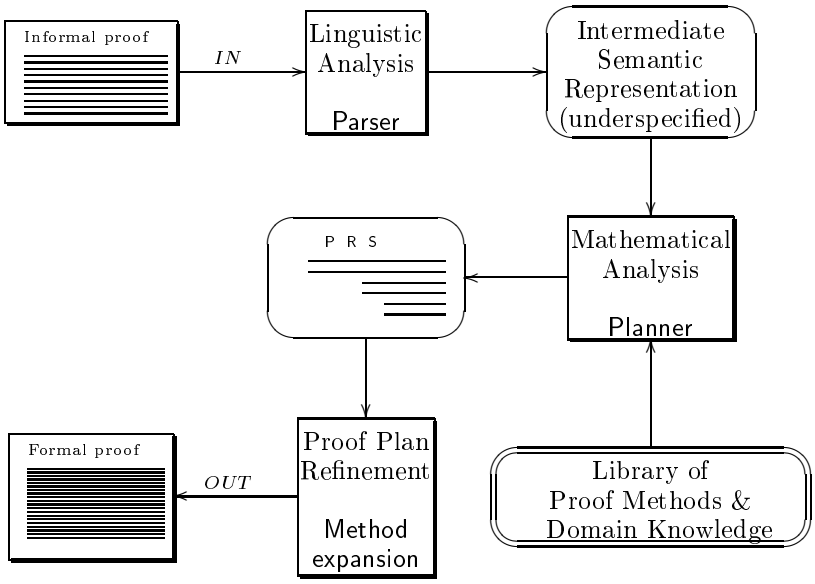

Figure 1: Architecture of VIP

gets as input an informal proof. For each sentence of this proof, the parser performs a linguistic analysis which results into a set of intermediate semantic representations. These representations are under-specified (extended discourse representation) structures. That is, they contain referential expressions that have to be resolved into subsequent processing

[5]Note that considerable effort must be invested to interpret the notation being used in this sentence.

— and most of the expressions that occur in this intermediate semantic representation are of anaphoric character.

For each sentence representation, the planner module tries to incorporate it into a proof representation structure capturing the current proof context. In general, the planner might be able to incorporate more than one intermediate representation into the current proof context. Also, for one given intermediate semantics representation, since there may be more than one possible attachment site, more than one possible proof continuation may arise. In both cases, the planner module maintains a set of proof continuations each of which is represented by a PRS, some of which may be ruled out if they do not allow to be continued in subsequent processing.

The proof planner is supported by a library of proof methods. If the planner detects a new proof obligation, it can search the library for applicable proof methods to handle it. Doing so, the planner can predict potential future proof continuations. Also, the proof planner needs access to a library of domain knowledge.

If the textbook proof has been processed in this manner, the PRS will contain a complete proof plan. A proof refinement module will take this proof plan as input, expand all proof methods that occur in this proof plan to inference-level steps, such that a formal proof will result. A set of formal proofs is returned. We view each of these formal proofs as one possible semantics of the given informal textbook proof.

In the remainder of this section I describe VIP's semantics/pragmatics interface. We start with a description of proof representation structures. Then I sketch how the proof planner combines semantically under-specified representations from the syntax/semantics interface with pragmatic knowledge (mathematical and meta-mathematical knowledge) to incrementally constructs a representation of mathematical discourse.

3.1 Linguistic Analysis — Parser

The parser module of VIP gets a tokenized sentence[6] and returns its underspecified semantic representation. These intermediate structures are extended discourse representation structures because they provide multiple kinds of discourse entities. The introduction of different types of discourse referents has been necessary to properly distinguish between constants, variables (quantified and free variables), terms, and assertions.

The lexical entries for constants (say 3), variables (say a), and terms (say f) are:

$$\lambda R \boxed{\begin{array}{|c|}\hline \text{C:3} \\ \hline \\ \hline \end{array}} + \text{R(3)} \qquad \lambda R \boxed{\begin{array}{|c|}\hline \text{Q:a} \\ \hline \\ \hline \end{array}} + \text{R(a)} \qquad \lambda x_1 \ldots \lambda x_n \lambda R \boxed{\begin{array}{|c|}\hline i : \iota x(x = f(x_1, \ldots x_n)) \\ \hline \\ \hline \end{array}} + \text{R(i).}$$

For variables, Q is either $\forall$, $\exists$, F(ree), or U(nknown); if the name of the variable is unknown, then this is indicated by U(nknown).

Extended discourse representation structures can be under-specified for two reasons: (i) the sentence contains variables with unknown quantifications; (ii) the sentence contains inter-sentential anaphoric expressions. In both cases, to fully determine the semantic content, prior discourse and pragmatic knowledge has to be taken into account.

Quantification of Variables. Mathematically, the occurrence of a in (2a) is universally quantified while all occurrences of a in (2c-2f) are free. Linguistically, the quantification of a in (2a) has been made explicit by using the quantifier *every*. Note that the if-then construction in (2d) does neither explicitly nor implicitly quantify a.[7]

For example, the parser returns the fully-specified DRS for (2a): $[[\forall : a, C : 1 | a > 1] \rightarrow [\text{—} | prod\_primes(a)]]$. For a slightly different formulation of (2a), *If* $a > 1$, *then* a

[6] The LaTeX input has been tokenized by hand. Sentence boundaries have been marked.

[7] This is a difference to standard DRT [KR93] where discourse entities introduced in the premise of a conditional get a universal reading.

can be represented as a product of one or more primes., we obtain $[[U : a, C : 1 | a > 1] \rightarrow [\text{—} | prod\_primes(a)]]$. The status of a has to be determined from the context.

Anaphoric references. For sentence (2b), the parser returns the underspecified DRS: $[[U : a, C : 2, A : \alpha \mid theorem(\alpha), \alpha \doteq ?, true\_for(\alpha, a, 2)]]$. The discourse referent α is of type assertion and must satisfy the conditions $theorem(\alpha)$ and $true\_for(\alpha, a, 2)$. Its antecedent has to be identified by the proof planner.

3.2 Proof Representation Structures

Proof representation structures (PRSs) play a central role in our framework.[8] An example PRS for LeVeque's proof (discourse (2a-2f)) is given in Fig. 2 (please ignore the stars for the moment). PRSs contain discourse referents, discourse conditions and discourse markers.

LET $a \in \mathbb{N}$ be universally quantified	$*^1$
THEOREM $a > 1 \rightarrow prod\_primes(a)$	$*^1$
Proof per Nœtherian induction	$*^2$
LET $a \in \mathbb{N}$ be arbitrary	$*^2$
LET $k \in \mathbb{N}$ be universally quantified	$*^2$
1. *induction_hypothesis*	$*^2$
ASSUME $k < a \rightarrow (k > 1 \rightarrow prod\_primes(k))$	$*^2$
PROVE $a > 1 \rightarrow prod\_primes(a)$	$*^2$
2. *Proof per cases*	
2.1. *first_case*	
2.1.1. ASSUME $prime(a)$	
2.1.2. $prod\_primes(a)$ **by ?** (QED)	
2.2. *second_case*	
2.2.1. ASSUME $\neg prime(a)$	
LET X be ?	
2.2.2. $divisor(X, a)$ **by ?**	
2.2.3. $X \neq 1$ **by ?**	
2.2.4. $X \neq a$ **by ?**	
LET b, c be ?	
2.2.5. $a = bc$ **by ?**	
2.2.6. $1 < b < a$ **by ?**	
2.2.7. $1 < c < a$ **by ?**	
LET $p'_1, p'_2 \ldots, p''_1, p''_2 \ldots$ be ? and prime	
LET s, t, i be ?	
2.2.8. $b = \prod_{i=1}^{s} p'_i$ **by ?**	
2.2.9. $c = \prod_{i=1}^{t} p''_i$ **by ?**	
2.2.10. $a = p'_1 p'_2 \ldots p'_i p''_1 \ldots p''_t$ **by ?** (QED)	

Figure 2: A proof representation structure

Discourse referents are introduced by a LET construct. Conditions are numbered and can be either assumptions or conclusions, the former are prefixed by ASSUME. Each conclusion can be accompanied by a justification list which is a list of numbers, each of which refers to

[8] Albeit the missing box-like appearance, PRSs are an extension of Kamp's discourse representation structures [KR93]. PRSs are also similar in many respects to Lamport-style proof presentations [Lam93].

the condition it prefixes. Instead of boxes, in PRS, accessibility is imposed by a numbering scheme. For example, take the condition labelled 2.2.6. It contains the terms b and a. Their quantification is determined by the closest accessible LET constructs that introduce these names. Note that the free variable a shadows the universally bound variable a. For justifying 2.2.6 the conditions 2.2.1-2.2.5 and 1. are accessible, the conditions 2.1.1 and 2.1.2 are not accessible. Discourse markers are used to provide non-number referents to either assumptions (e.g., *the induction hypothesis*) or sub-structures (e.g., *the second case*).

Note that the PRS of Fig. 2 is still semantically underspecified. Variables marked with *'?'* have scope, but no decision has yet been made about their quantification. Also, no conclusion is justified.

3.3 The Semantics/Pragmatics Interface — Proof Planner

The proof planner module gets an underspecified semantic representation of an informal proof sentence and tries to incorporate it into the PRS. The proof planner consults two libraries: a library of mathematical knowledge and a library of meta-mathematical knowledge. The theory library contains domain knowledge, basically, a set of definitions and theorems. Our experience showed that it is useful to have multiple definitions for the same concept. The proof plan library contains a collection of proof plans/methods (more precisely: proof plan schemas).[9] Proof methods encode general conversational principles in mathematical argumentation; — and clearly, cannot be encoded at the level of lexical entries.

Quite often, the form of the theorem presupposes more than one applicable proof plan. Fig. 3 depicts one of the applicable proof plans for theorems of the form $\forall n \in \mathbb{N} : P(n)$. It introduces two variables (a free and a bound one), an assumption, a discourse marker,

LET $n \in \mathbb{N}$ be universally quantified PROVE $P(n)$
Proof per Nœtherian induction LET $n \in \mathbb{N}$ be arbitrary LET $k \in \mathbb{N}$ be universally quantified *induction hypothesis* ASSUME $k < n \rightarrow P(k)$ PROVE $P(n)$

Figure 3: Plan for Nœtherian induction

and a remaining proof obligation.

For illustrating the PRS construction algorithm, assume an empty PRS, and that the parser returns the following intermediate representation for (2a): $[[\forall : a, C : 1 | a > 1] \rightarrow [— | prod\_primes(a)]]$. Since the PRS is empty, the planner assumes it is reading the semantic representation for a theorem (that is, it labels the assertion with the discourse marker THEOREM), and adds the enriched semantic representation into the PRS. Now, we have a PRS that consists of the top two-lines of the PRS as depicted in Fig. 2. Instead of interpreting the underspecified representation of LeVeque's first proof sentence, the proof planner tries to figure out future proof continuations. Consulting the proof plan library, applicable proof plans are identified, amongst them the proof plan shown in Fig. 3. Instantiating this proof plan for the theorem at hand results in the lines marked '$*^2$' of

[9] However, we cannot hope that any proof plan library is complete. There is a potentially infinite number of different induction schemes, and creative mathematicians often combine existing methods in a novel way. Learning new plans from the content of discourse is a difficult task — also for the human reader!

the PRS in Fig. 2. These starred lines define a number of expectations, and therefore allow us to view remaining informal statements as anaphoric entities that refer to their place in the PRS.

Unfortunately, the underspecified semantic representation of (2b) cannot be matched to the proof plan.[10] For $a = 2$, we have that $\forall k < a[k > 1 \rightarrow product\_primes(k)]$ is trivially true since there is no k such that $k < a$ and $k > 1$. Therefore, the case $a = 2$ is treated separately.

(2c) can be matched to one of our expectations if we give to the elliptic construct $2, 3, 4, \ldots, a-1$ a universal reading: $\forall n \in \mathbb{N} : n < a$. Note that the a that occurs in this expression refers to the free PRS variable a, and not to the universally quantified PRS variable a that was introduced by processing the theorem.

(2d) contains the discourse marker *if* indicating an assumption. Since we cannot match this assumption with the PRS expectations, the proof planner consults the plan library. Unexpected assumptions are often of the form $A \vee \neg A$ (here, A is *a is prime*), and a proof per cases is likely to take place. The proof planner instantiates a proof per cases plan schema (numbered **2.**) with its two sub-cases **2.1** and **2.2** to the PRS.[11] The so updated PRS allows the proof planner to resolve *if a is prime*. The cue phrase constituent *we are through* marks the end of this first case leaving the proof planner with the obligation to resolve the logical relation between *prod_primes* (2.1.2) and all statements of the proof context that are accessible from 2.1.2. Note that, due to a wrong proof continuation, or to a wrong attachment site selection, we might not be able to identify this logical relation. The occurrence of *otherwise* in (2e) seems to confirm a proof per cases. The remainder of discourse (2e-2f) has to fit into the second case yielding to **2.2.2** to **2.2.10**.

Given the PRS in Fig. 2, the proof planner can easily resolve the definite NP in (2f) to the assumption at line **1.**, labelled with the discourse marker *induction_hypothesis*.

3.4 Inference in Mathematical Discourse - a Closer Look

While the last section demonstrates the basic idea of PRS construction, it does not show all the difficulties that are involved in this task. Note also that most of the information that is contained in the PRS depicted in Fig. 2 is explicitly present in discourse (2a-2f). However, the PRS is still semantically underspecified: most of the variables do not have quantifications yet and all conclusions are still unjustified. Also it is unclear if the PRS in Fig. 2 truly captures the structure of discourse (2a-2f). The proof planner has to validate the proposed PRS and to compute the necessary information to fill in the semantic gaps. In the remainder of this section, we uncover the extent of deduction necessary to do so.

For the following analysis, we were supported by λ-CLAM, a higher-order logic version of the OYSTER/CLAM proof planner [BvHHS90, LD97]. As we will see, this deeper analysis reveals a number of steps that have no equivalent in the textbook proof.[12] Also, the following definitions must have been used:

$$\forall n \in \mathbb{N} : \quad (prime(n) \iff n > 1 \wedge \forall d \in \mathbb{N} : d|n \rightarrow d = 1 \vee d = n)$$

$$\forall a \in \mathbb{N} : \forall b \in \mathbb{N} : \quad (a|b \iff \exists c \in \mathbb{N} : b = ac \wedge a \leq b \wedge c \leq b)$$

$$\forall n \in \mathbb{N} : prod\_primes(n) \iff prime(n) \vee \exists p_1 \in \mathbb{N} : \exists p_2 \in \mathbb{N} : n = p_1 p_2 \wedge p_1 < n \wedge p_2 < n \wedge prod\_primes(p_1) \wedge prod\_primes(p_2)$$

Now, let us start. All boldfaced numbers refer to the PRS of Fig. 2.

[10] However, the planner can resolve the referential expression *the theorem* (the conditions $\alpha \doteq ?$ and $theorem(\alpha)$), and thus can interpret $true\_for(\alpha, a, 2)$ as $prod\_primes(2)$.

[11] This, of course, in only one interpretation among others. Other proof continuations are possible.

[12] For the sake of argument, our description has been a little simplified. A complete formal λ-CLAM–proof is available upon request from the author.

Step 1. The remaining proof obligation, or *goal*, is of the form $A \to B$. To prove statements of this form, the *implication method* can be applied: one assumes A and then uses A to show B:

PROVE $A \to B$
implication method
ASSUME A
PROVE B

This inference is not explicitly part of the textbook proof. It must be derived to facilitate subsequent discourse processing.

Step 2.1.2. To justify $prod\_primes(a)$ involves a couple of inference steps. *Definitional rewriting* on *product_primes* by the definition supplied above, reduces **2.1.2** to

$$prime(a) \vee \exists p_1 \in \mathbb{N} : \exists p_2 \in \mathbb{N} : a = p_1 p_2 \wedge p_1 < a \wedge p_2 < a \wedge product\_primes(p_1) \wedge product\_primes(p_2).$$

Having now a disjunction in the goal, a *proof per elimination* can be performed. There are two possible proof plan schemes:

PROVE $A \to B1 \vee B2$
Proof per elimination-I
ASSUME A
ASSUME $\neg B1$
PROVE $B2$

PROVE $A \to B1 \vee B2$
Proof per elimination-II
ASSUME A
ASSUME $\neg B2$
PROVE $B1$

Applying *proof per elimination-II*, we obtain a new hypothesis (in addition to the induction hypothesis, $a > 1$ and **2.1.1**):

$$\neg(\exists p_1 \in \mathbb{N} : \exists p_2 \in \mathbb{N} : a = p_1 p_2 \wedge p_1 < a \wedge p_2 < a \wedge product\_primes(p_1) \wedge product\_primes(p_2))$$

and a new goal: $prime(n)$. Since the goal is contained in the hypotheses, we are through.

Step 2.2.2–2.2.4. Deriving **2.2.2**–**2.2.4** and determining the quantification of X turns out the involve nearly a dozen of inferences. First, we have to perform a sequence of *forward reasoning* steps from the assumption **2.2.1**. This involves definitional rewriting of *prime* followed by consecutive steps that rewrite logical signs:

$$\begin{aligned}
\neg prime(a) &\iff \neg(a > 1 \wedge \forall d \in \mathbb{N} : d|a \to d = 1 \vee d = a) && \text{DeMorgan-1}\\
&\iff \neg(a > 1) \vee \neg\forall d \in \mathbb{N} : d|a \to d = 1 \vee d = a && \text{rewrite-}\forall\\
&\iff \neg(a > 1) \vee \neg\neg\exists d \in \mathbb{N} : \neg(d|a \to d = 1 \vee d = a) && \text{law of double negation}\\
&\iff \neg(a > 1) \vee \exists d \in \mathbb{N} : \neg(d|a \to d = 1 \vee d = a) && \text{rewrite-}\to\\
&\iff \neg(a > 1) \vee \exists d \in \mathbb{N} : \neg(\neg d|a \vee (d = 1 \vee d = a)) && \text{DeMorgan-2}\\
&\iff \neg(a > 1) \vee \exists d \in \mathbb{N} : (\neg\neg d|a \wedge \neg(d = 1 \vee d = a)) && \text{law of double negation}\\
&\iff \neg(a > 1) \vee \exists d \in \mathbb{N} : (d|a \wedge \neg(d = 1 \vee d = a)) && \text{DeMorgan-2}\\
&\iff \neg(a > 1) \vee \exists d \in \mathbb{N} : d|a \wedge \neg(d = 1) \wedge \neg(d = a) && (\text{“}|\text{”} \equiv \text{“divisor of”})
\end{aligned}$$

Second, using the assumption $a > 1$ (which we derived in step **1.**), a form of *Modus Ponens* can be applied (transforming $\Gamma, A, \neg A \vee B \vdash Goal$ to $\Gamma, A, B \vdash Goal$). Forward reasoning from the hypotheses thus yields

$$\exists d \in \mathbb{N} : d|a \wedge \neg(d = 1) \wedge \neg(d = a).$$

Now, we can determine the quantification of X in Fig. 2: it is existentially quantified. The above reasoning also justifies at once each of **2.2.2**–**2.2.4**.

2.2.5–2.2.7. To provide justifications for the steps **2.2.5**–**2.2.7**, we still reason forward from the hypotheses. Performing definitional expansion of $|$, we obtain

$$\exists d \in \mathbb{N} : (\exists c \in \mathbb{N} : a = dc \wedge d \leq a \wedge c \leq a) \wedge \neg(d = 1) \wedge \neg(d = a).$$

This allows us to determine the quantification of both b and c in Fig. 2: both are existentially quantified. Also, the anonymous variables X, introduced in the proof text by *a divisor*, turns out to be named either b or c.

If the library of domain knowledge contains the following lemmata,

$$\begin{aligned} \forall a \in \mathbb{N} : \forall b \in \mathbb{N} : a \leq b \wedge \neg(a = b) &\rightarrow a < b \\ \forall a \in \mathbb{N} : \forall b \in \mathbb{N} : \forall c \in \mathbb{N} : a = bc \wedge \neg(b = 1) &\rightarrow c < a \end{aligned}$$

2.2.6 and **2.2.7** can easily be proven.

2.2.8–2.2.10. We have now established that there exist numbers b and c such that both $b < a$ and $c < a$. For applying the induction hypotheses, some preparatory work is required. First, we apply the *specialisation method* to get rid of their existential quantifiers.

PROVE $\Gamma, \exists x : A(x) \rightarrow B$
Specialisation method
ASSUME $\Gamma, A(y)$
PROVE B

Having now some particular objects b and c with $b < a$ and $c < a$, and using the induction hypothesis we can apply the following proof method

$$\frac{\forall x : p(x) \rightarrow q(x), p(a)}{q(a)}$$

yielding $b > 1 \rightarrow \mathit{product\_primes}(b)$ and $c > 1 \rightarrow \mathit{product\_primes}(c)$.

In order to derive $\mathit{product\_primes}(b)$ and $\mathit{product\_primes}(c)$, we need to show that $b > 1$ and $c > 1$. Assuming the following domain knowledge

$$\forall a \in \mathbb{N} : \forall b \in \mathbb{N} : \forall c \in \mathbb{N} : a = bc \wedge \neg(a = b) \rightarrow c > 1$$

and applying Modus Ponens this can easily been shown. Thus, we successfully justified **2.2.8** and **2.2.9**. The last step of the proof, **2.2.10** is derivable using **2.2.5, 2.2.8** and **2.2.9**.

Conclusion. When communicating proofs, mathematicians omit a bulk of reasoning steps restricting themselves to the "essential". Following the reasoning line of the proof author requires the proof reader to identify the main proof idea determining the structure of the proof. In this particular case, recognising that the proof uses *course of values induction* where the induction step includes a proof per cases enables us to fill in the presupposed and implied information. As we showed, much inference is necessary in order to compute this implicit information.

4 Related Work

Deductive reasoning must play an important role in discourse comprehension. We briefly relate our approach to the work of Hobbs et al., van der Sandt, and Asher and Lascarides. Then, we shortly discuss Lamport's proof presentation style and the technique of proof planning.

Interpretation as Abduction. In [HSAM90], to interpret a natural language sentence its logical form has to be derived. The logical form comes from an assumed syntax/semantics interface, and will, in general, be semantically under-specified. Each logical form of a sentence is interpreted with respect to a domain knowledge base. To establish that the logical form is a logical consequence of the knowledge base, it might be necessary

to add assumptions to the knowledge base. The number of assumptions however should be minimal and [HSAM90] propose a scheme to determine how costly the derivation of the logical form (LF) from a knowledge base is. As a by-product of interpretation, the full content of a LF will be determined, and the LF as well as the additional assumptions that were necessary to deduce it are added to the knowledge base.

Hobbs approach is certainly an interesting approach. However, it has been exposed to a number of well-founded criticism: (i) its complexity is computational intractable; (ii) its solution of weighted abduction, assigning costs to predicates, is ad hoc; (iii) its assumption that inference is monotonic is too strong[13].

While we share these views, our main critic is threefold. First, [HSAM90] do not distinguish discourse context (the context that represents prior discourse) from background context (the context that represents domain knowledge). There is only one knowledge base of facts and rules. Discourse update is very simple: After interpreting a sentence its logical form is added to the knowledge base (and all assumptions that were necessary to derive it). However, the non-separation between discourse context and domain knowledge makes it difficult to implement a notion of saliency in discourse interpretation.[14]

Second, [HSAM90]'s knowledge base is a flat structure of facts and rules. It is desirable, however, that discourse interpretation should result into a representation of discourse that preserves its structure — this is particularly true for mathematical discourse. Having discourse structure also massively helps to treat a variety of linguistic phenomena and allows for a much stronger notion of antecedent accessibility.[15]

Third, in [HSAM90]'s approach, there seems to be only one discourse relation, logical consequence. But this does ignore the complexity of discourse and the variety of discourse relations that might hold (background, elaboration, contrast etc.). Determining such discourse relations should be part of the interpretation. Even in mathematical discourse, it does not suffice to compute logical consequence only. A high-level discourse understanding is required that allows to derive discourse structure and discourse goals.

Applying [HSAM90]'s approach to the domain of mathematical discourse in general (and to discourse (2a-2f)) in particular raises the following conceptual problem. For interpreting (2a), [HSAM90] would want to show that the logical form of this sentence can be logically derived from domain knowledge. Finding such a proof, however, is considerably complex. Assuming a knowledge base that contains all the necessary knowledge for deriving this LF and much more, searching it will be too costly. If the knowledge base does not contain all the necessary information (it could still be large), then it is doubtful if abductive inference would assume statements that are mathematically satisfying. In any case, the fact that the discourse (2b-2f) contains indeed a proof for (2a) is completely ignored.

Interpretation within the DRT Framework. In [vdS92], an approach for handling implicit information within Discourse Representation Theory (DRT) [KR93] is presented. Presuppositions are considered as anaphoric entities with semantic content. The cancelling of presuppositions (*vide* (1b)) is realised by binding it to its antecedent. If no suitable antecedent is found, the presupposition is added to the context to provide for such an antecedent (*vide* (1a)). Van der Sandt defines the notion of global, intermediate and local accommodation. He proposes to accommodate a presupposition into the accessible site of the context that produces the widest possible scope without violating consistency

[13] Assumptions are never retracted in order to make a LF derivable.

[14] For resolving the NP *the theorem* in (2b), [HSAM90] would try to abductively derive $\exists x : theorem(x)$. This could only be properly resolved assuming (i) the first statement (2a) has been tagged as theorem; and (ii) it is the most salient theorem.

[15] As we have seen, to properly handle the *if* ... *then* .... *Otherwise* ... structure in (2d-2e), discourse structure is necessary.

and informativeness constraints.

[AL98] argue that van der Sandts proposal to solve the projection problem fails short in not taking enough pragmatic information into account. His accommodation strategy could be more constrained by making use of domain knowledge and by taking rhetorical relations — how does a sentence rhetorically relate to prior discourse — into account. Asher and Lascarides aim to properly handle presuppositions within their SDRT framework (a considerable extension of DRT offering more structure) [Ash93]. They argue that their SDRT discourse update mechanism allows to effectively interface semantics with pragmatics. The discourse update procedure consists of *relating* a sentence to its prior discourse (not just adding it). To compute such rhetorical relations (e.g., background, narration, elaboration), Asher and Lascarides make use of domain knowledge and their DICE inference engine [LA91, AL98].

Our approach. Our point of departure for interpreting mathematical discourse has been DRT [KR93]. However, the particular text genre made it inevitable to extend DRT for mathematical discourse.[16] Some of the extensions went into the direction of Asher and Lascarides' SDRT, for example, the introduction of abstract discourse entities that allow to give a more structured representation of discourse. The result of all our extensions are proof representation structures.

Also, it became clear from the very beginning that DRT's discourse update mechanism, set union of discourse referents and conditions, does not capture the complexity of mathematical discourse. In line with Asher and Lascarides, we argue that a defining part of discourse interpretation must be to determine how a new sentence relates to prior discourse. This necessities to compute a discourse relation between them. For mathematical discourse, before computing logical consequence relations, its structure must be determined. This important task is carried out by the proof planning engine. The application of proof methods enables us to divide the discourse into different segments, each of which contributes in reducing a proof obligation. Logical consequence must only be established between sentences within a segment.

Proof Presentation à la Lamport. In [Lam93], Lamport proposes a method for writing proofs. Writing proofs in his proposed hierarchical structure, according to Lamport, "makes it much harder to prove things that are not true". The Lamport-style proof presentation consists of three parts: (i) a theorem statement, (ii) a proof outline, the proof sketch, and (iii) a highly structured proof of the theorem. Lamport's proof presentation provides a numbering scheme, indentation, and a notion of accessibility. This allows the reader to go through the proof at different levels, where each new proof level provides a new level of detail and formal rigour. Dependencies between mathematical arguments can easily be identified.

However, Lamport does not give any formal definition of his proof presentation style. Looking at his examples, the language he uses in the proof sketch part does not differ from the stylised language used in textbook proofs. The syntactic constructions in the proof part are (i) short (they mostly result into logical forms that are atomic formulae); and (ii) do not contain any complicated referential expressions (e.g., "by the definition of", "the lemma"). In [Lam93], Lamport claims that "the proof style I advocate is a refinement of one, called *natural deduction...*". However, in his proofs there are no explicit references to any of the rules of a natural deduction calculi (although one can easily identify some of them). That is, steps are only justified by other steps without citing the inference rules being involved in the reasoning. For similar criticism, see [RT96].

[16] A selection of linguistic phenomena in mathematical discourse is discussed in [Zin99a, Zin99b].

The Nature of Mathematical Discourse and Proof Planning. Mathematical discourse is different from other text genres. It has several advantages. First of all, mathematical discourse is a highly structured form of discourse. This is because (i) the domain of discourse is highly structured (a beautifully spinned web of mathematical truths); and (ii) proof methods/plans, encoding conversational principles, govern the uses of arguments in mathematical discourse. Given discourse structure, there is only one rhetorical discourse relation, logical consequence. Its computation involves mathematical reasoning that is monotonic and goal-oriented, thus constraining the search space of possible inferences.

Proof plans are an enabling technology for generating high-level proofs. The technique of proof planning has been invented by [Bun88]. This requires the analysis of families of related proofs, the identification of common patterns in them (called proof plans) and the use of these proof plans to guide future proofs from the same family. Proof planning has been implemented in the OYSTER/CLAM/XBARNACLE system [BvHHS90, LD97]. CLAM builds a proof plan customised to the current conjecture and then uses this plan to instruct the theorem prover, OYSTER, how to prove the theorem. All the search is conducted during proof planning; the execution of the proof plan requires no search.

5 Conclusion

An essential part of text understanding is to make explicit the implicit parts of discourse, its presuppositions and implications. For mathematical discourse, we showed that it is even impossible to understand the explicitly asserted context without being able to compute the non-asserted implicit parts of discourse. A reader, human or machine, must be able to "read between the lines".

In this paper, we described our system for understanding informal mathematical discourse. For representing mathematical discourse, we introduced proof representation structures as the central data structure. For constructing PRSs, we proposed a discourse update algorithm that is powered by pragmatics. A proof planner incorporates an underspecified semantic representation into the proof context by making use of mathematical and meta-mathematical knowledge. We argued that proof plans (capturing common patterns of reasoning in mathematical proofs) enable us to gain a high level discourse understanding allowing us to follow the proof author's argumentation line. A high-level discourse understanding is a prerequisite for computing the parts "between the lines". As we demonstrated, much inference is required to compute this implicit information.

This paper contains (more or less explicit) the idea to view *discourse understanding* in terms of *discourse formalisation*; in the sense of rendering exact and explicit vague English statements uttered in some context. This is a sensible approach for mathematical discourse because, in principle, mathematics is reducible to axiomatic set theory: any mathematical proof can be translated into a formal proof within this formal system. It is worth asking if *discourse interpretation as discourse formalisation* carries over to other text genres.

References

[AL98] N. Asher and A. Lascarides. The semantics and pragmatics of presupposition. *Journal of Semantics*, 15(3):239–300, 1998.

[Ash93] N. Asher. *Reference to abstract objects in discourse*. Kluwer Academic Publishers, 1993.

[Bun88] A. Bundy. The use of explicit proof plans to guide inductive proofs. In R. Lusk and R. Overbeek, editors, *Ninth Conference on Automated Deduction*, pages 111 – 120. Springer Verlag, 1988.

[BvHHS90] A. Bundy, F. van Harmelen, C. Horn, and A. Smaill. The oyster-clam system. In M. E. Stickel, editor, *Proceedings of the 10th International Conference on Automated Deduction*. Springer-Verlag, 1990.

[Dav91] S. Davis, editor. *Pragmatics: A Reader*. Oxford: OUP, 1991.

[Fre67] G. Frege. *Kleine Schriften*. Darmstadt Wiss. Buchges., 1967. Hrsg. von I. Angelelli.

[Gaz79] G. Gazdar. *Pragmatics: Implicature, Presupposition, and Logical Form*. Academic Press, 1979.

[Gri68] H.P. Grice. Logic and conversation. pages 305–315. 1968. Reprinted in [Dav91].

[Hil22] D. Hilbert. Die logischen Grundlagen der Mathematik. *Mathematische Annalen*, 88:151–165, 1922.

[HSAM90] J.R. Hobbs, M.E. Stickel, D.E. Appelt, and P. Martin. Interpretation as abduction. Technical Report Research report, number 499, SRI, 1990.

[KR93] H. Kamp and U. Reyle. *From Discourse to Logic*. Kluwer Academic Publishers, 1993.

[LA91] A. Lascarides and N. Asher. Discourse relations and defeasible knowledge. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 55–63. ACL, 1991.

[Lam93] L. Lamport. How to write proofs. In *Global Analysis in Modern Mathematics*, pages 311–321. Publish or Perish, Houston, Texas, U.S.A., February 1993. A symposium in honor of Richard Palais' sixtieth birthday.

[LD97] H. Lowe and D. Duncan. XBarnacle: Making theorem provers more accessible. In *CADE-14*, 1997.

[LeV65] W. J. LeVeque. *Elementary theory of numbers*. Series in introductory mathematics. Addison-Wesley, 1965.

[Lev83] S. C. Levinson. *Pragmatics*. Cambridge University Press, 1983.

[RT96] P. Rudnicki and A. Trybulec. A Note on "How to write a proof". Technical Report TR96-08, Computing Science Department, University of Alberta, Edmonton, Alberta, Canada, 1996.

[Sol90] D. Solow. *How to read and do proofs*. John Wiley & Sons, 1990.

[Sta77] R. C. Stalnaker. Pragmatic Presuppositions. In A. Roger, B. Wall, and J.P. Murphy, editors, *Proceedings of the Texas Conference on Performatives, Presuppositions and Implicature*, pages 135–147. Washington: Center for Applied Linguistics, 1977. Reprinted in [Dav91].

[Thu94] W. P. Thurston. On proof and progress in mathematics. *Bulletin (New Series) of the American Mathematical Society*, 30(2):161–177, April 1994.

[vdS92] R.A. van der Sandt. Presupposition projection as anaphora resolution. *Journal of Semantics*, 9:333–377, 1992.

[Zin99a] C. Zinn. Parsing formulae in textbook proofs. In H.C. Bunt and E.G.C. Thijsse, editors, *Proceedings of the Third Int'l Workshop on Computational Semantics (IWCS-3)*, pages 422–424. Tilburg University, 1999.

[Zin99b] C. Zinn. Understanding Mathematical Discourse. In *Proceedings Amstelogue'99, Workshop on the Semantics and Pragmatics of Dialogue*. Amsterdam University, 7-9 May 1999.