Analysis of Mixed Natural and Symbolic Language Input in Mathematical Dialogs

Magdalena Wolska Ivana Kruijff-Korbayová

Fachrichtung Computerlinguistik
Universität des Saarlandes, Postfach 15 11 50
66041 Saarbrücken, Germany
{magda,korbay}@coli.uni-sb.de

Abstract

Discourse in formal domains, such as mathematics, is characterized by a mixture of telegraphic natural language and embedded (semi-)formal symbolic mathematical expressions. We present language phenomena observed in a corpus of dialogs with a simulated tutorial system for proving theorems as evidence for the need for deep syntactic and semantic analysis. We propose an approach to input understanding in this setting. Our goal is a uniform analysis of inputs of different degree of verbalization: ranging from symbolic alone to fully worded mathematical expressions.

1 Introduction

Our goal is to develop a language understanding module for a flexible dialog system tutoring mathematical problem solving, in particular, theorem proving (Benzm¨uller et al., 2003a).[1] As empirical findings in the area of intelligent tutoring show, flexible natural language dialog supports active learning (Moore, 1993). However, little is known about the use of natural language in dialog setting in formal domains, such as mathematics, due to the lack of empirical data. To fill this gap, we collected a corpus of dialogs with a simulated tutorial dialog system for teaching proofs in naive set theory.

An investigation of the corpus reveals various phenomena that present challenges for such input understanding techniques as shallow syntactic analysis combined with keyword spotting, or statistical methods, e.g., Latent Semantic Analysis, which are commonly employed in (tutorial) dialog systems. The prominent characteristics of the language in our corpus include: (i) tight interleaving of natural and symbolic language, (ii) varying degree of natural language verbalization of the formal mathematical content, and (iii) informal and/or imprecise reference to mathematical concepts and relations.

These phenomena motivate the need for deep syntactic and semantic analysis in order to ensure correct mapping of the surface input to the underlying proof representation. An additional methodological desideratum is to provide a uniform treatment of the different degrees of verbalization of the mathematical content. By designing one grammar which allows a uniform treatment of the linguistic content on a par with the mathematical content, one can aim at achieving a consistent analysis void of example-based heuristics. We present such an approach to analysis here.

The paper is organized as follows: In Section 2, we summarize relevant existing approaches to input analysis in (tutorial) dialog systems on the one hand and analysis of mathematical discourse on the other. Their shortcomings with respect to our setting become clear in Section 3 where we show examples of language phenomena from our dialogs. In Section 4, we propose an analysis methodology that allows us to capture any mixture of natural and mathematical language in a uniform way. We show example analyses in Section 5. In Section 6, we conclude and point out future work issues.

2 Related work

Language understanding in dialog systems, be it with text or speech interface, is commonly performed using shallow syntactic analysis combined with keyword spotting. Tutorial systems also successfully employ statistical methods which compare student responses to a model built from preconstructed gold-standard answers (Graesser et al., 2000). This is impossible for our dialogs, due to the presence of symbolic mathematical expressions. Moreover, the shallow techniques also remain oblivious of such aspects of discourse meaning as causal relations, modality, negation, or scope of quantifiers which are of crucial importance in our setting. When precise understanding is needed, tutorial systems either use menu- or template-based input, or

[1]This work is carried out within the DIALOG project: a collaboration between the Computer Science and Computational Linguistics departments of the Saarland University, within the Collaborative Research Center on *Resource-Adaptive Cognitive Processes*, SFB 378 (`www.coli.uni-sb.de/sfb378`).

use closed-questions to elicit short answers of little syntactic variation (Glass, 2001). However, this conflicts with the preference for flexible dialog in active learning (Moore, 1993).

With regard to interpreting mathematical texts, (Zinn, 2003) and (Baur, 1999) present DRT analyses of course-book proofs. However, the language in our dialogs is more informal: natural language and symbolic mathematical expressions are mixed more freely, there is a higher degree and more variety of verbalization, and mathematical objects are not properly introduced. Moreover, both above approaches rely on typesetting and additional information that identifies mathematical symbols, formulae, and proof steps, whereas our input does not contain any such information. Forcing the user to delimit formulae would reduce the flexibility of the system, make the interface harder to use, and might not guarantee a clean separation of the natural language and the non-linguistic content anyway.

3 Linguistic data

In this section, we first briefly describe the corpus collection experiment and then present the common language phenomena found in the corpus.

3.1 Corpus collection

24 subjects with varying educational background and little to fair prior mathematical knowledge participated in a *Wizard-of-Oz* experiment (Benzm¨uller et al., 2003b). In the tutoring session, they were asked to prove 3 theorems[2]:

(i) $K((A \cup B) \cap (C \cup D)) = (K(A) \cap K(B)) \cup (K(C) \cap K(D))$;

(ii) $A \cap B \in P((A \cup C) \cap (B \cup C))$;

(iii) $If\, A \subseteq K(B), then B \subseteq K(A)$.

To encourage dialog with the system, the subjects were instructed to enter proof steps, rather than complete proofs at once. Both the subjects and the tutor were free in formulating their turns. Buttons were available in the interface for inserting mathematical symbols, while literals were typed on the keyboard. The dialogs were typed in German.

The collected corpus consists of 66 dialog logfiles, containing on average 12 turns. The total number of sentences is 1115, of which 393 are student sentences. The students' turns consisted on average of 1 sentence, the tutor's of 2. More details on the corpus itself and annotation efforts that guide the development of the system components can be found in (Wolska *et al.*, 2004).

[2] K stands for set complement and P for power set.

3.2 Language phenomena

To indicate the overall complexity of input understanding in our setting, we present an overview of common language phenomena in our dialogs.[3] In the remainder of this paper, we then concentrate on the issue of interleaved natural language and mathematical expressions, and present an approach to processing this type of input.

Interleaved natural language and formulae Mathematical language, often semi-formal, is interleaved with natural language informally verbalizing proof steps. In particular, mathematical expressions (or parts thereof) may lie within the scope of quantifiers or negation expressed in natural language:

A auch $\subseteq$ $K(B)$	[A *also* $\subseteq$ $K(B)$]
A∩B **ist** $\in$ **von** C∪(A∩B)	[... *is* $\in$ *of* ...]
(da ja A∩B=$\emptyset$)	[(*because* $A \cap B = \emptyset$)]
B enthaelt **kein x**∈A	[*B contains no* $x \in A$]

For parsing, this means that the mathematical content has to be identified before it is interpreted within the utterance.

Imprecise or informal naming Domain relations and concepts are described informally using imprecise and/or ambiguous expressions.

A **enthaelt** B	[*A contains B*]
A muss **in** B sein	[*A must be in B*]

where **contain** and **be_in** can express the domain relation of either subset or element;

B **vollstaendig ausserhalb von** A **liegen** muss, also im Komplement von A
[*B has to be entirely outside of A, so in the complement of A*]

dann sind A und B **(vollkommen) verschieden**, haben keine **gemeinsamen Elemente**
[*then A and B are (completely) different, have no common elements*]

where **be_outside_of** and **be_different** are informal descriptions of the empty intersection of sets.

To handle imprecision and informality, we constructed an ontological knowledge base containing domain-specific interpretations of the predicates (Horacek and Wolska, 2004).

Discourse deixis Anaphoric expressions refer deictically to pieces of discourse:

der obere Ausdruck	[*the above term*]
der letzte Satz	[*the last sentence*]
Folgerung aus **dem Obigen**	[*conclusion from the above*]
aus der regel in **der zweiten Zeile**	[*from the rule in the second line*]

[3] As the tutor was also free in wording his turns, we include observations from both student and tutor language behavior. In the presented examples, we reproduce the original spelling.

In our domain, this class of referring expressions also includes references to structural parts of terms and formulae such as "the left side" or "the inner parenthesis" which are incomplete specifications: the former refers to a part of an equation, the latter, metonymic, to an expression enclosed in parenthesis. Moreover, these expressions require discourse referents for the sub-parts of mathematical expressions to be available.

Generic vs. specific reference Generic and specific references can appear within one utterance:

> **Potenzmenge** enthaelt alle Teilmengen, also auch **(A∩B)**
> [*A power set contains all subsets, hence also(A∩B)*]

where "a power set" is a generic reference, whereas "$A \cap B$" is a specific reference to a subset of a specific instance of a power set introduced earlier.

Co-reference[4] Co-reference phenomena specific to informal mathematical discourse involve (parts of) mathematical expressions within text.

> Da, wenn $A_i \subseteq K(B_j)$ sein soll, A_i Element von $K(B_j)$ sein muss. Und wenn $B_k \subseteq K(A_l)$ sein soll, muss es_k auch Element von $K(A_l)$ sein.
> [*Because if it should be that $A_i \subseteq K(B_j)$, A_i must be an element of $K(B_j)$. And if it should be that $B_k \subseteq K(A_l)$, it must be an element of $K(A_l)$ as well.*]

Entities denoted with the same literals may or may not co-refer:

> DeMorgan-Regel-2 besagt: $K(A_i \cap B_j) = K(A_i) \cup K(B_j)$
> In diesem Fall: z.B. $K(A_i)$ = dem Begriff $K(A_k \cup B_l)$
> $K(B_j)$ = dem Begriff $K(C \cup D)$
> [*DeMorgan-Regel-2 means: $K(A_i \cap B_j) = K(A_i) \cup K(B_j)$*
> *In this case: e.g. $K(A_i)$ = the term $K(A_k \cup B_l)$*
> *$K(B_j)$ = the term $K(C \cup D)$*]

Informal descriptions of proof-step actions Sometimes, "actions" involving terms, formulae or parts thereof are verbalized before the appropriate formal operation is performed:

> **Wende** zweimal die DeMorgan-Regel an
> [*I'm* **applying** *DeMorgan rule twice*]
> damit kann ich den oberen Ausdruck wie folgt **schreiben**:...
> [*given this I can* **write** *the upper term as follows:*...]

The meaning of the "action verbs" is needed for the interpretation of the intended proof-step.

Metonymy Metonymic expressions are used to refer to structural sub-parts of formulae, resulting in predicate structures acceptable informally, yet incompatible in terms of selection restrictions.

> Dann **gilt fuer die linke Seite**, wenn
> $C \cup (A \cap B) = (A \cup C) \cap (B \cup C)$, der Begriff A ∩ B dann ja schon dadrin und ist somit auch Element davon
> [*Then* **for the left hand side it holds** *that..., the term A ∩ B is already there, and so an element of it*]

[4]To indicate co-referential entities, we inserted the indices which are not present in the dialog logfi les.

where the predicate **hold**, in this domain, normally takes an argument of sort CONST, TERM or FORMULA, rather than LOCATION;

> de morgan **regel** 2 **auf beide komplemente angewendet**
> [*de morgan rule 2 applied to both complements*]

where the predicate **apply** takes two arguments: one of sort RULE and the other of sort TERM or FORMULA, rather than OPERATION ON SETS.

In the next section, we present our approach to a uniform analysis of input that consists of a mixture of natural language and mathematical expressions.

4 Uniform input analysis strategy

The task of input interpretation is two-fold. Firstly, it is to construct a representation of the utterance's linguistic meaning. Secondly, it is to identify and separate within the utterance:
(i) parts which constitute meta-communication with the tutor, e.g.:

> Ich habe die Aufgabenstellung nicht verstanden.
> [*I don't understand what the task is.*]

(ii) parts which convey domain knowledge that should be verified by a domain reasoner; for example, the entire utterance

> $K((A \cup B))$ ist laut deMorgan-1 $K(A) \cap K(B)$
> [*... is, according to deMorgan-1,...*]

can be evaluated; on the other hand, the domain reasoner's knowledge base does not contain appropriate representations to evaluate the correctness of using, e.g., the focusing particle "also", as in:

> Wenn A = B, dann ist A auch $\subseteq K(B)$ und B $\subseteq K(A)$.
> [*If A = B, then A is also $\subseteq K(B)$ and B $\subseteq K(A)$.*]

Our goal is to provide a uniform analysis of inputs of varying degrees of verbalization. This is achieved by the use of one grammar that is capable of analyzing utterances that contain both natural language and mathematical expressions. Syntactic categories corresponding to mathematical expressions are treated in the same way as those of linguistic lexical entries: they are part of the deep analysis, enter into dependency relations and take on semantic roles. The analysis proceeds in 2 stages:

1. After standard pre-processing,[5] mathematical expressions are identified, analyzed, categorized, and substituted with default lexicon entries encoded in the grammar (Section 4.1).

[5]Standard pre-processing includes sentence and word tokenization, (spelling correction and) morphological analysis, part-of-speech tagging.

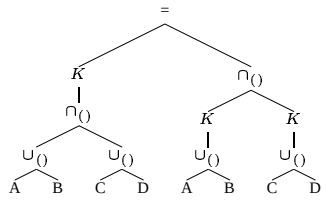

Figure 1: Tree representation of the formula $K((A \cup B) \cap (C \cup D)) = (K(A \cup B) \cap K(C \cup D))$

2. Next, the input is syntactically parsed, and a representation of its linguistic meaning is constructed compositionally along with the parse (Section 4.2).

The obtained linguistic meaning representation is subsequently merged with discourse context and interpreted by consulting a semantic lexicon of the domain and a domain-specific knowledge base (Section 4.3).

If the syntactic parser fails to produce an analysis, a shallow chunk parser and keyword-based rules are used to attempt partial analysis and build a partial representation of the predicate-argument structure.

In the next sections, we present the procedure of constructing the linguistic meaning of syntactically well-formed utterances.

4.1 Parsing mathematical expressions

The task of the mathematical expression parser is to identify mathematical expressions. The identified mathematical expressions are subsequently verified as to syntactic validity and categorized.

Implementation Identification of mathematical expressions within word-tokenized text is performed using simple indicators: single character tokens (with the characters P and K standing for power set and set complement respectively), mathematical symbol unicodes, and new-line characters. The tagger converts the infix notation used in the input into an expression tree from which the following information is available: surface sub-structure (e.g., "left side" of an expression, list of sub-expressions, list of bracketed sub-expressions) and expression type based on the top level operator (e.g., CONST, TERM, FORMULA 0_FORMULA (formula missing left argument), etc.).

For example, the expression $K((A \cup B) \cap (C \cup D)) = (K(A \cup B) \cap K(C \cup D))$ is represented by the formula tree in Fig. 1. The bracket subscripts indicate the operators heading sub-formulae enclosed in parenthesis. Given the expression's top node operator, =, the expression is of type formula, its "left side" is the expression $K((A \cup B) \cap (C \cup D))$, the list of bracketed sub-expressions includes: A∪B, C∪D, $(A \cup B) \cap (C \cup D)$, etc.

Evaluation We have conducted a preliminary evaluation of the mathematical expression parser. Both the student and tutor turns were included to provide more data for the evaluation. Of the 890 mathematical expressions found in the corpus (432 in the student and 458 in the tutor turns), only 9 were incorrectly recognized. The following classes of errors were detected:[6]

1. P((A∪C)∩(B∪C)) =PC ∪(A∩B)
   → P((A∪C)∩(B∪C))=PC ∪(A∩B)
2. a. (A⊆U und B⊆U) b. (da ja A∩B=∅)
   → (A⊆U und B⊆U) → (da ja A∩B=∅)
3. K((A∪B)∩(C∪D)) = K(A ? B) ? K(C ? D)
   → K((A∪B)∩(C∪D)) = K(A ? B) ? K(C ? D)
4. Gleiches gilt mit D (K(C∩D)) ∪ (K(A∩B))
   → Gleiches gilt mit D (K(C∩D)) ∪ (K(A∩B))
   [The same holds with . . .]

The examples in (1) and (2) have to do with parentheses. In (1), the student actually omitted them. The remedy in such cases is to ask the student to correct the input. In (2), on the other hand, no parentheses are missing, but they are ambiguous between mathematical brackets and parenthetical statement markers. The parser mistakenly included one of the parentheses with the mathematical expressions, thereby introducing an error. We could include a list of mathematical operations allowed to be verbalized, in order to include the logical connective in (2a) in the tagged formula. But (2b) shows that this simple solution would not remedy the problem overall, as there is no pattern as to the amount and type of linguistic material accompanying the formulae in parenthesis. We are presently working on ways to identify the two uses of parentheses in a pre-processing step. In (3) the error is caused by a non-standard character, "?", found in the formula. In (4) the student omitted punctuation causing the character "D" to be interpreted as a non-standard literal for naming an operation on sets.

4.2 Deep analysis

The task of the deep parser is to produce a domain-independent linguistic meaning representation of syntactically well-formed sentences and fragments.

By linguistic meaning (LM), we understand the dependency-based deep semantics in the sense of the Prague School notion of sentence meaning as employed in the Functional Generative Description

[6] Incorrect tagging is shown along with the correct result below it, following an arrow.

(FGD) (Sgall et al., 1986; Kruijff, 2001). It represents the literal meaning of the utterance rather than a domain-specific interpretation.[7] In FGD, the central frame unit of a sentence/clause is the head verb which specifies the *tectogrammatical relations* (TRs) of its dependents (*participants*). Further distinction is drawn into *inner participants*, such as Actor, Patient, Addressee, and *free modifications*, such as Location, Means, Direction. Using TRs rather than surface grammatical roles provides a generalized view of the correlations between domain-specific content and its linguistic realization.

We use a simplified set of TRs based on (Hajičová et al., 2000). One reason for simplification is to distinguish which relations are to be understood metaphorically given the domain sub-language. In order to allow for ambiguity in the recognition of TRs, we organize them hierarchically into a taxonomy. The most commonly occurring relations in our context, aside from the inner participant roles of Actor and Patient, are Cause, Condition, and Result-Conclusion (which coincide with the rhetorical relations in the argumentative structure of the proof), for example:

Da [A $\subseteq K(B)$ gilt]$_{<\text{CAUSE}>}$, alle x, die in A sind sind nicht in B
[*As A$\subseteq K(B)$ applies, all x that are in A are not in B*]
Wenn [A $\subseteq K(B)$]$_{<\text{COND}>}$, dann A $\cap$ B=$\emptyset$
[*If A$\subseteq K(B)$, then A$\cap$B=$\emptyset$*]
Da $A \subseteq K(B)$ gilt, [alle x, die in A sind sind nicht in B]$_{<\text{RES}>}$
Wenn A$\subseteq K(B)$, dann [A$\cap$B=$\emptyset$]$_{<\text{RES}>}$

Other commonly found TRs include Norm-Criterion, e.g.

[nach deMorgan-Regel-2]$_{<\text{NORM}>}$ ist $K((A \cup B) \cap \ldots)$=...)
[*according to De Morgan rule 2 it holds that ...*]
$K((A \cup B))$ ist [laut DeMorgan-1]$_{<\text{NORM}>}$ $(K(A) \cap K(B))$
[*... equals, according to De Morgan rule1, ...*]

We group other relations into sets of HasProperty, GeneralRelation (for adjectival and clausal modification), and Other (a catch-all category), for example:

dann muessen alla A und B [in C]$_{<\text{PROP-LOC}>}$ enthalten sein
[*then all A and B have to be contained in C*]
Alle x, [die in B sind]$_{<\text{GENREL}>}$ $\cdots$ [All x that are in B...]
alle elemente [aus A]$_{<\text{PROP-FROM}>}$ sind in $K(B)$ enthalten
[*all elements from A are contained in $K(B)$*]
Aus A$\subseteq$U\B folgt [mit A$\cap$B=$\emptyset$]$_{<\text{OTHER}>}$, B$\subseteq$U\A.
[*From A$\subseteq$U\B follows with A$\cap$B=$\emptyset$, that B$\subseteq$U\A*]

[7] LM is conceptually related to logical form, however, differs in coverage: while it does operate on the level of deep semantic roles, such aspects of meaning as the scope of quantifi ers or interpretation of plurals, synonymy, or ambiguity are not resolved.

where PROP-LOC denotes the HasProperty relation of type Location, GENREL is a general relation as in complementation, and PROP-FROM is a HasProperty relation of type Direction-From or From-Source. More details on the investigation into tectogrammatical relations that build up linguistic meaning of informal mathematical text can be found in (Wolska and Kruijff-Korbayová, 2004a).

Implementation The syntactic analysis is performed using openCCG[8], an open source parser for Multi-Modal Combinatory Categorial Grammar (MMCCG). MMCCG is a lexicalist grammar formalism in which application of combinatory rules is controlled though context-sensitive specification of modes on slashes (Baldridge and Kruijff, 2003). The linguistic meaning, built in parallel with the syntax, is represented using Hybrid Logic Dependency Semantics (HLDS), a hybrid logic representation which allows a compositional, unification-based construction of HLDS terms with CCG (Baldridge and Kruijff, 2002). An HLDS term is a relational structure where dependency relations between heads and dependents are encoded as modal relations. The syntactic categories for a lexical entry FORMULA, corresponding to mathematical expressions of type "formula", are S, NP, and N.

For example, in one of the readings of "B enthaelt $x \in A$", "enthaelt" represents the meaning **contain** taking dependents in the relations Actor and Patient, shown schematically in Fig. 2.

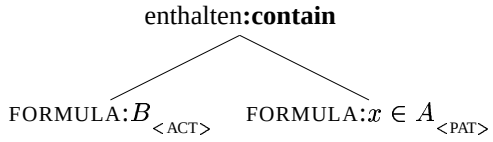

Figure 2: Tectogrammatical representation of the utterance "B enthaelt $x \in A$" [*B contains $x \in A$*].

FORMULA represents the default lexical entry for identified mathematical expressions categorized as "formula" (cf. Section 4.1). The LM is represented by the following HLDS term:

@h1(contain $\wedge$ <ACT>(f1 $\wedge$ FORMULA:B) $\wedge$ <PAT>(f2 $\wedge$ FORMULA: $x \in A$)

where h1 is the state where the proposition **contain** is true, and the nominals f1 and f2 represent dependents of the head **contain**, which stand in the tectogrammatical relations Actor and Patient, respectively.

It is possible to refer to the structural sub-parts of the FORMULA type expressions, as formula subparts are identified by the tagger, and discourse ref-

[8] `http://openccg.sourceforge.net`

erents are created for them and stored with the discourse model.

We represent the discourse model within the same framework of hybrid modal logic. Nominals of the hybrid logic object language are atomic formulae that constitute a pointing device to a particular place in a model where they are true. The satisfaction operator, @, allows to evaluate a formula at the point in the model given by a nominal (e.g. the formula $@_i\phi$ evaluates ϕ at the point i). For discourse modeling, we adopt the hybrid logic formalization of the DRT notions in (Kruijff, 2001; Kruijff and Kruijff-Korbayová, 2001). Within this formalism, nominals are interpreted as discourse referents that are bound to propositions through the satisfaction operator. In the example above, f1 and f2 represent discourse referents for FORMULA:B and FORMULA:$x{\in}A$, respectively. More technical details on the formalism can be found in the aforementioned publications.

4.3 Domain interpretation

The linguistic meaning representations obtained from the parser are interpreted with respect to the domain. We are constructing a domain ontology that reflects the domain reasoner's knowledge base, and is augmented to allow resolution of ambiguities introduced by natural language. For example, the previously mentioned predicate **contain** represents the semantic relation of **Containment** which, in the domain of naive set theory, is ambiguous between the domain relations ELEMENT, SUBSET, and PROPER SUBSET. The specializations of the ambiguous semantic relations are encoded in the ontology, while a semantic lexicon provides interpretations of the predicates. At the domain interpretation stage, the semantic lexicon is consulted to translate the tectogrammatical frames of the predicates into the semantic relations represented in the domain ontology. More details on the lexical-semantic stage of interpretation can be found in (Wolska and Kruijff-Korbayová, 2004b), and more details on the domain ontology are presented in (Horacek and Wolska, 2004).

For example, for the predicate **contain**, the lexicon contains the following facts:

contain($ACT_{type:FORMULA}$, $PAT_{type:FORMULA}$)
$\equiv$ (SUBFORMULA$_{PAT}$, embedding$_{ACT}$)
['*a Patient of type* FORMULA *is a* **subformula** *embedded within a* FORMULA *in the Actor relation with respect to the head* **contain**']

contain($ACT_{type:OBJECT}$, $PAT_{type:OBJECT}$)
$\equiv$ CONTAINMENT(container$_{ACT}$, containee$_{PAT}$)
['*the* **Containment** *relation involves a predicate* **contain** *and its Actor and Patient dependents, where the Actor and Patient are the* **container** *and* **containee** *parameters respectively*']

Translation rules that consult the ontology expand the meaning of the predicates to all their alternative domain-specific interpretations preserving argument structure.

As it is in the capacity of neither sentence-level nor discourse-level analysis to evaluate the correctness of the alternative interpretations, this task is delegated to the Proof Manager (PM). The task of the PM is to: (A) communicate directly with the theorem prover;[9] (B) build and maintain a representation of the proof constructed by the student;[10] (C) check type compatibility of proof-relevant entities introduced as new in discourse; (D) check consistency and validity of each of the interpretations constructed by the analysis module, with the proof context; (E) evaluate the proof-relevant part of the utterance with respect to completeness, accuracy, and relevance.

5 Example analysis

In this section, we illustrate the mechanics of the approach on the following examples.

(1) B enthaelt kein $x \in A$ [*B contains no* $x \in A$]
(2) A∩B∈{A∩B}
(3) A enthaelt keinesfalls Elemente, die in B sind.
[*A contains no elements that are also in B*]

Example (1) shows the tight interaction of natural language and mathematical formulae. The intended reading of the scope of negation is over a part of the formula following it, rather than the whole formula. The analysis proceeds as follows.

The formula tagger first identifies the formula <x∈A> and substitutes it with the generic entry FORMULA represented in the lexicon. If there was no prior discourse entity for "B" to verify its type, the type is ambiguous between CONST, TERM, and FORMULA.[11] The sentence is assigned four alternative readings:
(i) "CONST contains no FORMULA",
(ii) "TERM contains no FORMULA",
(iii) "FORMULA contains no FORMULA",
(iv) "CONST contains no CONST 0_FORMULA".

The last reading is obtained by partitioning an entity of type FORMULA in meaningful ways, taking into account possible interaction with preceding modifiers. Here, given the quantifier "no", the expression <x∈A> has been split into its surface parts

[9] We are using a version of ΩMEGA adapted for assertion-level proving (Vo et al., 2003).

[10] The discourse content representation is separated from the proof representation, however, the corresponding entities must be co-indexed in both.

[11] In prior discourse, there may have been an assignment B := ϕ, where ϕ is a formula, in which case, B would be known from discourse context to be of type FORMULA (similarly for term assignment); by CONST we mean a set or element variable such as A, x denoting a set A or an element x respectively.

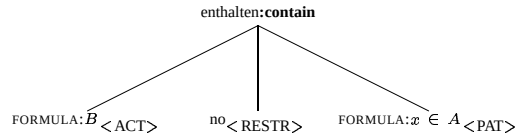

Figure 3: Tectogrammatical representation of the utterance "B enthaelt kein $<x \in A>$" [*B contains no* $x \in A$].

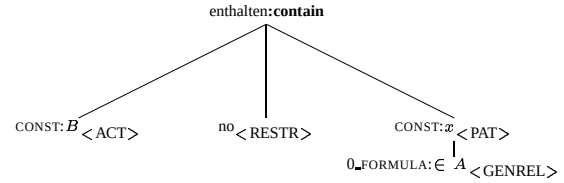

Figure 4: Tectogrammatical representation of the utterance "B enthaelt kein $< [x][\in A] >$" [*B contains no* $< [x][\in A] >$].

as follows: <[x][∈A]>.[12] [x] has been substituted with a generic lexical entry CONST, and [∈A] with a symbolic entry for a formula missing its left argument (cf. Section 4.1).

The readings (i) and (ii) are rejected because of sortal incompatibility. The linguistic meanings of readings (iii) and (iv) are presented in Fig. 3 and Fig. 4, respectively. The corresponding HLDS representations are:[13]

— for "FORMULA contains no FORMULA":

s:(@k1(kein∧<RESTR>f2∧<BODY>(e1∧enthalten
∧<ACT>(f1∧FORMULA)∧<PAT>f2))∧@f2(FORMULA))
['*formula B embeds no subformula x∈A*']

— for "CONST contains no CONST 0_FORMULA":

s:(@k1(kein∧<RESTR>x1∧<BODY>(e1∧enthalten
∧<ACT>(c1∧CONST)∧<PAT>x1))∧
@x1(CONST∧<HASPROP>(x2∧0_FORMULA)))
['*B contains no x such that x is an element of A*']

Next, the semantic lexicon is consulted to translate these readings into their domain interpretations. The relevant lexical semantic entries were presented in Section 4.3. Using the linguistic meaning, the semantic lexicon, and the ontology, we obtain four interpretations paraphrased below:

— for "FORMULA contains no FORMULA":

(1.1) 'it is not the case that <PAT>, the formula, x∈A, is a subformula of <ACT>, the formula B';

— for "CONST contains no CONST 0_FORMULA":

[12]There are other ways of constituent partitioning of the formula at the top level operator to separate the operator and its arguments: <[x][∈][A]> and <[x∈][A]>. Each of the partitions obtains its appropriate type corresponding to a lexical entry available in the grammar (e.g., the [x∈] chunk is of type FORMULA_0 for a formula missing its right argument). Not all the readings, however, compose to form a syntactically and semantically valid parse of the given sentence.

[13]Irrelevant parts of the meaning representation are omitted; glosses of the hybrid formulae are provided.

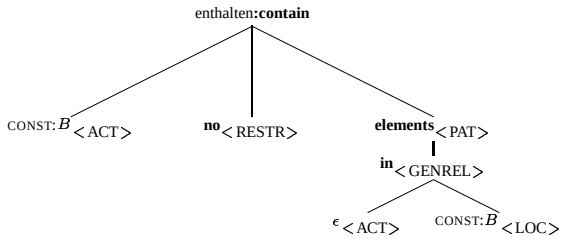

Figure 5: Tectogrammatical representation of the utterance "A enthaelt keinesfalls Elemente, die auch in B sind." [*A contains no elements that are also in B.*].

(1.2a) 'it is not the case that <PAT>, the constant x, ⊆ <ACT>, B, and x∈A',
(1.2b) 'it is not the case that <PAT>, the constant x, ∈ <ACT>, B, and x∈A',
(1.2c) 'it is not the case that <PAT>, the constant x, ⊂ <ACT>, B, and x∈A'.

The interpretation (1.1) is verified in the discourse context with information on structural parts of the discourse entity "B" of type formula, while (1.2a-c) are translated into messages to the PM and passed on for evaluation in the proof context.

Example (2) contains one mathematical formula. Such utterances are the simplest to analyze: The formulae identified by the mathematical expression tagger are passed directly to the PM.

Example (3) shows an utterance with domain-relevant content fully linguistically verbalized. The analysis of fully verbalized utterances proceeds similarly to the first example: the mathematical expressions are substituted with the appropriate generic lexical entries (here, "A" and "B" are substituted with their three possible alternative readings: CONST, TERM, and FORMULA, yielding several readings "CONST contains no elements that are also in CONST", "TERM contains no elements that are also in TERM", etc.). Next, the sentence is analyzed by the grammar. The semantic roles of Actor and Patient associated with the verb "contain" are taken by "A" and "elements" respectively; quantifier "no" is in the relation Restrictor with "A"; the relative clause is in the GeneralRelation with "elements", etc. The linguistic meaning of the utterance in example (3) is shown in Fig. 5. Then, the semantic lexicon and the ontology are consulted to translate the linguistic meaning into its domain-specific interpretations, which are in this case very similar to the ones of example (1).

6 Conclusions and Further Work

Based on experimentally collected tutorial dialogs on mathematical proofs, we argued for the use of deep syntactic and semantic analysis. We presented an approach that uses multimodal CCG with hy-

brid logic dependency semantics, treating natural and symbolic language on a par, thus enabling uniform analysis of inputs with varying degree of formal content verbalization.

A preliminary evaluation of the mathematical expression parser showed a reasonable result. We are incrementally extending the implementation of the deep analysis components, which will be evaluated as part of the next *Wizard-of-Oz* experiment.

One of the issues to be addressed in this context is the treatment of ill-formed input. On the one hand, the system can initiate a correction subdialog in such cases. On the other hand, it is not desirable to go into syntactic details and distract the student from the main tutoring goal. We therefore need to handle some degree of ill-formed input.

Another question is which parts of mathematical expressions should have explicit semantic representation. We feel that this choice should be motivated empirically, by systematic occurrence of natural language references to parts of mathematical expressions (e.g., "the left/right side", "the parenthesis", and "the inner parenthesis") and by the syntactic contexts in which they occur (e.g., the partitioning <[x][∈A]> seems well motivated in "B contains no x∈A"; [x ∈] is a constituent in "x ∈ of complement of B.")

We also plan to investigate the interaction of modal verbs with the argumentative structure of the proof. For instance, the necessity modality is compatible with asserting a necessary conclusion or a prerequisite condition (e.g., "A und B muessen disjunkt sein." [*A and B must be disjoint.*]). This introduces an ambiguity that needs to be resolved by the domain reasoner.

References

J. M. Baldridge and G.J. M. Kruijff. 2002. Coupling CCG with hybrid logic dependency semantics. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia PA. pp. 319–326.

J. M. Baldridge and G.J. M. Kruijff. 2003. Multi-modal combinatory categorial grammar. In *Proc. of the 10th Annual Meeting of the European Chapter of the Association for Computational Linguistics (EACL'03)*, Budapest, Hungary. pp. 211–218.

J. Baur. 1999. *Syntax und Semantik mathematischer Texte*. Diplomarbeit, Fachrichtung Computerlinguistik, Universit¨at des Saarlandes, Saarbr¨ucken, Germany.

C. Benzm¨uller, A. Fiedler, M. Gabsdil, H. Horacek, I. Kruijff-Korbayov´a, M. Pinkal, J. Siekmann, D. Tsovaltzi, B. Q. Vo, and M. Wolska. 2003a. Tutorial dialogs on mathematical proofs. In *Proc. of IJCAI'03 Workshop on Knowledge Representation and Automated Reasoning for E-Learning Systems*, Acapulco, Mexico.

C. Benzm¨uller, A. Fiedler, M. Gabsdil, H. Horacek, I. Kruijff-Korbayov´a, M. Pinkal, J. Siekmann, D. Tsovaltzi, B. Q. Vo, and M. Wolska. 2003b. A Wizard-of-Oz experiment for tutorial dialogues in mathematics. In *Proc. of the AIED'03 Workshop on Advanced Technologies for Mathematics Education*, Sydney, Australia. pp. 471–481.

M. Glass. 2001. Processing language input in the CIRCSIM-Tutor intelligent tutoring system. In *Proc. of the 10th AIED Conference*, San Antonio, TX. pp. 210–221.

A. Graesser, P. Wiemer-Hastings, K. Wiemer-Hastings, D. Harter, and N. Person. 2000. Using latent semantic analysis to evaluate the contributions of students in autotutor. *Interactive Learning Environments*, 8:2. pp. 129–147.

E. Hajičov´a, J. Panevov´a, and P. Sgall. 2000. A manual for tectogrammatical tagging of the Prague Dependency Treebank. TR-2000-09, Charles University, Prague, Czech Republic.

H. Horacek and M. Wolska. 2004. Interpreting Semi-Formal Utterances in Dialogs about Mathematical Proofs. In *Proc. of the 9th International Conference on Application of Natural Language to Information Systems (NLDB'04)*, Salford, Manchester, Springer. To appear.

G.J.M. Kruijff and I. Kruijff-Korbayov´a. 2001. A hybrid logic formalization of information structure sensitive discourse interpretation. In *Proc. of the 4th International Conference on Text, Speech and Dialogue (TSD'2001)*, Železn´a Ruda, Czech Republic. pp. 31–38.

G.J.M. Kruijff. 2001. *A Categorial-Modal Logical Architecture of Informativity: Dependency Grammar Logic & Information Structure*. Ph.D. Thesis, Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic.

J. Moore. 1993. What makes human explanations effective? In *Proc. of the 15th Annual Conference of the Cognitive Science Society*, Hillsdale, NJ. pp. 131–136.

P. Sgall, E. Hajičov´a, and J. Panevov´a. 1986. *The meaning of the sentence in its semantic and pragmatic aspects*. Reidel Publishing Company, Dordrecht, The Netherlands.

Q.B. Vo, C. Benzm¨uller, and S. Autexier. 2003. Assertion Application in Theorem Proving and Proof Planning. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*. Acapulco, Mexico.

M. Wolska and I. Kruijff-Korbayov´a. 2004a. Building a dependency-based grammar for parsing informal mathematical discourse. In *Proc. of the 7th International Conference on Text, Speech and Dialogue (TSD'04)*, Brno, Czech Republic, Springer. To appear.

M. Wolska and I. Kruijff-Korbayov´a. 2004b. Lexical-Semantic Interpretation of Language Input in Mathematical Dialogs. In *Proc. of the ACL Workshop on Text Meaning and Interpretation*, Barcelona, Spain. To appear.

M. Wolska, B. Q. Vo, D. Tsovaltzi, I. Kruijff-Korbayov´a, E. Karagjosova, H. Horacek, M. Gabsdil, A. Fiedler, C. Benzm¨uller, 2004. An annotated corpus of tutorial dialogs on mathematical theorem proving. In *Proc. of 4th International Conference On Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. pp. 1007–1010.

C. Zinn. 2003. A Computational Framework for Understanding Mathematical Discourse. In *Logic Journal of the IGPL*, 11:4, pp. 457–484, Oxford University Press.