Mathematical Domain Reasoning Tasks in Natural Language Tutorial Dialog on Proofs*

Christoph Benzmüller and **Quoc Bao Vo**
Saarland University, Saarbrücken, Germany
chris|bao@ags.uni-sb.de
http://ags.uni-sb.de/˜chris|bao

Abstract

We study challenges that are imposed to mathematical domain reasoning in the context of natural language tutorial dialog on mathematical proofs. The focus is on proof step evaluation:
(i) How can mathematical domain reasoning support the resolution of ambiguities and underspecified parts in proof steps uttered by a student?
(ii) How can mathematical domain reasoning support the evaluation of a proof step with respect to the criteria *soundness*, *granularity*, and *relevance*?

Introduction

The final goal of the DIALOG project[1] is a natural tutorial dialog on mathematical proofs between a student and an assistance system for mathematics. Natural language (NL) tutorial dialog on mathematical proofs is a multi-disciplinary scientific challenge situated between (i) advanced NL processing, (ii) flexible tutorial dialog, and (iii) dynamic, abstract level mathematical domain reasoning (MDR[2]). There is still relatively few data available that can guide research in this area. We, therefore, approached the project by using a methodology with a strong initial emphasis on empirical investigations and a top-down modeling of the over-all architecture followed by refinements of the architecture, down to implementation.

First a relevant corpus has been collected and analyzed in the DIALOG project. The phenomena that have been identified through corpus analysis demonstrate, for instance, the need for deep semantical analysis, the importance of a tight integration of NL processing and MDR, and the relevancy of dynamic, abstract-level proof development techniques supporting human-oriented MDR. In particular, the explicit abstract-level representation of proof steps (logically sound or unsound) as uttered by the students is a crucial prerequisite for their subsequent analysis by MDR means in a tutorial dialog setting. Additionally, from a logical point of view, proof steps are highly underspecified (e.g. logically relevant references are left implicit) causing an additional challenge for bridging the gap between NL analysis and MDR.

In this paper we focus on the challenges imposed to MDR:

(i) How can MDR support the resolution of ambiguities and underspecified parts in proof steps uttered by a student?

(ii) How can MDR support the evaluation of a student proof step with respect to the criteria *soundness*, *granularity*, and *relevance*?

In the next section we present an example dialog from our DIALOG corpus and point to some revealed phenomena. We then discuss the MDR challenges from a general viewpoint. Subsequently we present our first concrete approach to solve these challenges. Finally, we discuss some related work and conclude the paper.

Phenomena and Challenges

A Wizard-of-Oz experiment (Dahlbäck, Jönsson, & Ahrenberg 1993) has been performed in the DIALOG project in order to obtain a corpus of tutorial dialogs on mathematical proofs. Twenty four subjects with varying background in humanities and sciences participated in this experiment. Their prior mathematical knowledge ranged from little to fair. The experiment employed typed user and tutor (wizard) input as opposed to spoken language. This experiment and the corpus obtained is discussed in more details in (Wolska *et al.* 2004). The complete corpus comprises 66 recorded dialogs containing on average 12 turns and is available from the DIALOG web-page[3]. It contains 1115 sentences in total, of which 393 are student sentences. An example dialog is shown in Fig. 1.

Investigation of the corpus resulted in an overwhelming list of key phenomena raising interesting and novel research challenges (Benzmüller *et al.* 2003). This was not expected, in particular, because of the simplicity of the mathematical domain (naive set theory) chosen for this experiment. Many

*We thank Manfred Pinkal, Jörg Siekmann, Ivana Kruijff-Korbayová, Armin Fiedler, Magdalena Wolska, Helmut Horacek, Serge Autexier, Dimitra Tsovaltzis, Marvin Schiller, and Mark Buckley.

[1] The DIALOG project is a collaboration between the Computer Science and Computational Linguistics departments of Saarland University as part of the Collaborative Research Center on *Resource-Adaptive Cognitive Processes*, SFB 378 (`http://www.coli.uni-saarland.de/projects/sfb378/`).

[2] We use 'MDR' in the remainder as an abbreviation for both 'mathematical domain reasoning' and 'mathematical domain reasoner'; the precise meaning will be clear in each context.

[3] `http://www.ags.uni-sb.de/˜chris/dialog/`

T1: Bitte zeigen Sie: $K((A \cup B) \cap (C \cup D)) = (K(A) \cap K(B)) \cup (K(C) \cap K(D))$! *[Engl.: Please show: $K((A \cup B) \cap (C \cup D)) = (K(A) \cap K(B)) \cup (K(C) \cap K(D))$!]*

S1: nach deMorgan-Regel-2 ist $K((A \cup B) \cap (C \cup D)) = (K(A \cup B) \cup K(C \cup D))$. *[Engl.: by deMorgan-Rule-2 $K((A \cup B) \cap (C \cup D)) = (K(A \cup B) \cup K(C \cup D))$ holds.]*

T2: Das ist richtig! *[Engl.: This is correct!]*

S2: $K(A \cup B)$ ist laut deMorgan-1 $K(A) \cap K(B)$ *[Engl.: $K(A \cup B)$ is $K(A) \cap K(B)$ according to deMorgan-1]*

T3: Das stimmt auch. *[Engl.: That is also right.]*

S3: und $K(C \cup D)$ ist ebenfalls laut deMorgan-1 $K(C) \cap K(D)$ *[Engl.: and $K(C \cup D)$ is also $K(C) \cap K(D)$ according to deMorgan-1]*

T4: Auch das stimmt. *[Engl.: That also is right.]*

S4: also folgt letztendlich: $K((A \cup B) \cap (C \cup D)) = (K(A) \cap K(B)) \cup (K(C) \cap K(D))$. *[Engl.: hence follows finally: $K((A \cup B) \cap (C \cup D)) = (K(A) \cap K(B)) \cup (K(C) \cap K(D))$.]*

T5: Das stimmt … *[Engl.: This is correct …]*

Figure 1: An example dialog. **T** and **S** mark tutor (i.e. wizard) and student turns respectively. 'K' refers to the 'set complement' relation. German has been the language of choice.

of the identified phenomena are relevant not only for the tutorial NL dialog context but have a much wider impact for NL interactions in human-oriented theorem proving. This paper focuses on phenomena that are relevant for MDR:

Notion of Proof. For analyzing the notion of human-oriented mathematical proofs, primarily shaped-up textbook proofs have been investigated in the deduction systems community (Zinn 2004). The DIALOG corpus provides an important alternative view on it, since textbook proofs neither reveal the actual dynamics of proof construction nor do they show the weaknesses and inaccuracies of the student's utterances, i.e., the student's proof step directives. The corpus also illustrates the style and logical granularity of human-constructed proofs. The style is mainly declarative, for example, the students declaratively described the conclusions and some (or none) of the premises of their inferences. This is in contrast to the procedural style employed in many proof assistants where proof steps are invoked by calling rules, tactics, or methods, i.e., some proof refinement procedures.

The hypothesis that assertion level reasoning (Huang 1994) plays an essential role in this context has been confirmed. The phenomenon that assertion level reasoning may by highly underspecified in human-constructed proofs, however, is a novel finding (Autexier *et al.* 2003).

Underspecification is a well known phenomenon in linguistic analysis. The corpus reveals that underspecification also occurs in the content and precision of mathematical utterances (proof step specification) and thus carries over to MDR. Interestingly underspecification also occurs in shaped-up textbook proofs but has only very recently been addressed (Zinn 2004). To illustrate the underspecification aspect we use example utterance **S4** in Fig. 1: Utterance **S4** is logically strongly underspecified. Here, it is neither mentioned from what assertion(s) in the discourse this statement exactly follows nor how these assertions are used. However, such detailed information is typically required in proof assistants to execute the student's proof step directive, i.e., to 'understand' and 'logically follow' the student's argumentation.

Proof Step Evaluation (PSE) is an interesting novel application for theorem proving systems. A (next) proof step uttered by a student within a tutorial context has to be analyzed with respect to the following criteria:

Soundness: Can the proof step be reconstructed by a formal inference system and logically and tutorially verified?

Granularity: Is the 'argumentative complexity' or 'size' of the proof step logically and tutorially acceptable?

Relevance: Is the proof step logically and tutorially useful for achieving the goal?

Resolution of underspecification and PSE motivate a specific module supporting these tasks in tutorial NL dialog on proofs; in the remainder we call such a module *proof manager (PM)*.

MDR Challenges from a General Viewpoint

Ambiguity and Underspecification Resolution The corpus reveals that ambiguities may arise at different phases of processing between the linguistic analysis and MDR. Consider, for instance, the following student utterance:

S: A enthaelt B [*Engl.*: A contains B]

In this utterance 'enthaelt' ('contains') is ambiguous as it may refer to the set relations 'element-of' and 'subset-of'. The ambiguity arises during linguistic analysis. It can be resolved, for instance, by type-checking provided that type information on A and B is available: if both symbols are of the same 'set type' then 'enthaelt' means 'subset-of'. However, type checking cannot differentiate between '$\subset$' and '$\subseteq$' as potential readings. The phenomenon is even better illustrated by the following two utterances in which important bracketing information is missing ('K' refers to the 'set complement' operation and 'P' to the 'Power set' operation):

S': $P((A \cup C) \cap (B \cup C)) = PC \cup (A \cap B)$

S": $K((A \cup C) \cap (B \cup C)) = KC \cup (A \cap B)$

In **S'** type information (if available) can be employed to rule out the reading $P(C) \cup (A \cap B)$ for the term to the right. However, type information is not sufficient to differentiate between the readings $K(C) \cup (A \cap B)$ and $K(C \cup (A \cap B))$ in **S"**. Here only MDR can detect that the first reading leads to a logically wrong statement and the second reading to a correct one. As we cannot assume that the domain model statically represents all correct mathematical statements this calls for dynamic MDR support in the resolution of ambiguities that, as given here, may arise during linguistic analysis. Now consider the following slight modification (wrt. reference to deMorgan rule) of utterances **T1** and **S1** from Fig. 1.

T1: Please show : $K((A \cup B) \cap (C \cup D)) = (K(A) \cap K(B)) \cup (K(C) \cap K(D))$

S1': by the deMorgan rule we have $K((A \cup B) \cap (C \cup D)) = (K(A \cup B) \cup K(C \cup D))$.

S1' does not lead to an ambiguity during linguistic analysis. It nevertheless leads to an ambiguity in the domain reasoner since the suggested proof step is highly underspecified from

Proof State

(A1) $A \wedge B$.
(A2) $A \Rightarrow C$.
(A3) $C \Rightarrow D$.
(A4) $F \Rightarrow B$.

(G) $D \vee E$.

Some Student Utterances

(a) From the assertions follows D.
(b) B holds.
(c) It is sufficient to show D.
(d) We show E.

Figure 2: PSE example scenario: (A1)-(A4) are assertions that have been introduced in the discourse and that are available to prove the proof goal (G). (a)-(d) are examples for possible proof step directives of the student in this proof situation.

a proof construction viewpoint: **S1'** can be obtained directly from the deMorgan rule $\forall X, Y. K(X \cap Y) = K(X) \cup K(Y)$ (denoted as `deMorgan-2`) by instantiating X with $(A \cup B)$ and Y with $(C \cup D)$. Alternatively it could be inferred from **T1** when applying deMorgan rule $\forall X, Y. K(X \cup Y) = K(X) \cap K(Y)$ (denoted as `deMorgan-1`) from right to left to the subterms of **T1**: $K(A) \cap K(B)$ and $K(C) \cap K(D)$. Differentiating between such alternatives could be crucial in tutoring mathematical proofs.

Proof Step Evaluation: PSE supports the dynamic step-by-step analysis (with criteria soundness, granularity, relevance) of the proof constructed by the student. All three criteria have a pure logical dimension and additionally a tutorial dimension. For instance, a proof step may be formally relevant by pure logical means but it may be considered as not relevant when additional tutorial aspects are taken into account. On the other hand, a student utterance which is sufficiently close to a valid next proof step may be considered tutorially relevant while being logically irrelevant. In this paper we mainly focus on the logical dimension; the hypothesis is that their solution is one important prerequisite for solving the general PSE problem involving also the tutorial dimension. Much further research in this direction is clearly needed. The PSE challenge will now be further illustrated using the artificially simplified example in Fig. 2.

Soundness: Determining whether an uttered proof step is sound requires that the MDR can represent, reconstruct and validate the uttered proof step (including all the justifications used by the student) within the MDR's representation of the proof state. Consider, for instance, utterance (a) in Fig. 2: Verification of the soundness of this utterance boils down to adding D as a new assertion to the proof state and to proving that: **(P1)** $(A \wedge B), (A \Rightarrow C), (C \Rightarrow D), (F \Rightarrow B) \vdash D$. Solving this proof task confirms the logical soundness of utterance (a). If further explicit justifications are provided in the student's utterance (e.g. a proof rule) then we have to take them into consideration and, for example, prove **(P1)** modulo these additional constraints. Soundness is a fairly tractable criterion for which different techniques are readily available (Zinn 2004). PSE with respect to the criteria *granularity* and *relevance*, however, is novel and challenging.

Granularity evaluation requires analyzing the 'complexity' or 'size' of proofs instead of asking for the mere existence of proofs. For instance, evaluating utterance (a) above boils down to judging the complexity of the generated proof task **(P1)**. Let us, for example, use Gentzen's natural deduction (ND) calculus as the proof system $\vdash$. As a first and naive logical granularity measure, we may determine the number of $\vdash$-steps in the smallest $\vdash$-proof of the proof task for the proof step utterance in question; this number is taken as the argumentative complexity of the uttered proof step. For example, the smallest ND proof for utterance (a) has '3' proof steps: we need one 'Conjunction-Elimination' step to extract A from $A \wedge B$, one 'Modus Ponens' step to obtain C from A and $A \Rightarrow C$, and another 'Modus Ponens' step to obtain D from C and $C \Rightarrow D$. On the other hand, the smallest ND proof for utterance (b) requires only '1' step: B follows from assertion $A \wedge B$ by 'Conjunction-Elimination'. If we now fix a threshold that tries to capture, in this sense, the 'maximally acceptable size of an argumentation' then we can distinguish between proof steps whose granularity is acceptable and those which are not. This threshold may be treated as a parameter determined by the tutorial setting. However, the ND calculus together with naive proof step counting doesn't always provide a cognitively adequate basis for granularity analysis. The reason is that two intuitively very similar student proof steps (such as **(i)** from $A = B$ and $B = C$ infer $A = C$ and **(ii)** from $A \Leftrightarrow B$ and $B \Leftrightarrow C$ infer $A \Leftrightarrow C$) may actually expand into base-level ND proofs of completely different size. Also related literature has pointed out that standard ND calculus does not adequately reflect human-reasoning (Rips 1994). This problem could become even worse if we chose a machine-oriented calculus such as resolution. Two important and cognitively interesting questions thus concern the appropriate choice of a proof system $\vdash$ and ways to measure the 'argumentative complexity' of a proof step.

Relevance. Relevance asks questions about the usefulness and importance of a proof step with respect to the original proof task. For instance, in utterance (c) the proof goal $D \vee E$ is refined to the new proof goal D using backward reasoning, i.e., the previously open goal $D \vee E$ is closed and justified by a new goal. Answering the logical relevance question in this case requires to check whether a proof can still be generated in the new proof situation. In our case, the task is thus identical to proof task **(P1)**. A backward proof step that is not relevant according to this criterion is (d) since it reduces to the proof task: **(P2)** $(A \wedge B), (A \Rightarrow C), (C \Rightarrow D), (F \Rightarrow B) \vdash E$ for which no proof can be generated. Thus, (d) is a sound refinement step that is not relevant. This simple approach appears plausible but needs to be refined. The challenge is to exclude detours and to take tutorial aspects into account (in a tutorial setting we are often interested in teaching particular styles of proofs, particular proof methods, etc.). This also applies to the more challenging forward reasoning case to identify that, for instance, utterance (b) describes a non-relevant proof step.

Relevance and granularity are interesting, ambitious and important challenges for tutoring of proofs. To address these problems, it's not sufficient to merely establish the existence of proofs but the system has to construct proofs with particular properties. It may be the case that evaluating different criteria requires different 'suitable' theorem provers.

Moreover, the system also needs to closely mirror and reflect reasoning steps as they are typically performed by humans. Generally, the system will need to adapt to the capabilities of individual students and the requirements of varying tutorial settings.

PSE in the Dialog Demonstrator

We have implemented a demonstrator version of a PM which provides dynamic support for resolution of underspecification and PSE based on heuristically guided abstract-level MDR realized on top of the Ωmega-CoRe framework (Autexier 2003). The PM has been integrated into the overall demonstrator of the Dialog project in which it communicates with other components of the system including the linguistic analyzer, the dialog manager, the tutorial manager, and the NL generator. More information on the role of the PM in the Dialog demonstrator system and on its interplay with other modules is given in (Buckley & Benzmüller 2005). Note that we do not address tutoring aspects directly in the PM. Instead the result of the PM's proof step analysis is passed to the *tutorial manager* which then proposes a tutoring move to the dialog manager of the overall system. Tutoring aspects of the Dialog project are discussed in (Fiedler & Tsovaltzi 2003).

The complete system has been applied to several example dialogs from the Dialog corpus and it has been demonstrated in the course of the evaluation of the Dialog project that the system is particularly able to support variations of the dialog presented in Fig.1 (which we will use for illustration purposes). However, our system is currently only applicable to a very restricted subset of example proofs in naive set theory. For these examples the PM's computation costs are acceptable. It remains to be seen whether this is still the case when moving to less elementary mathematical problem domains.

Proof Step Representation and Resolution of Underspecification. The PM needs to "understand" the incoming student proof step and to fit it into the current proof context.

In our implementation, the student proof step is first formatted into a tuple $\langle$ Label, Type, Dir, Formula, Justification-list $\rangle$: Label provides a reference to this proof step. Type indicates whether the student proof step is, for example, an *inference* step, a *variable assignment*, or a *local hypothesis introduction* (these are the options we currently support). Given the proof step type *inference*, Dir indicates the direction of this step as linguistically extracted from the student's utterance. The alternatives are *forward*, *backward*, *sideward*, and *closing*. For instance, when the student asserts that "ϕ follows from ψ and θ" and if we know that ψ and θ are the two premises of the current proof task, then the input analyzer should be able to assign forward inference to Dir. Formula is the asserted formula in this proof step, e.g., the ϕ from above. Justification-list contains all the information the student uses to justify Formula.

In our current approach, all of these fields except from Formula can be left underspecified (i.e. empty). Label can in general be easily generated by referring to Formula or by NL references such as "the previous proof step", "your second proof step", etc. The other fields are usually more ambitious to determine. Before we proceed with describing our solution to underspecification resolution, we elaborate the Justification-list. Justification-list is a list $(J_1, \ldots, J_n)$ of justifications J_i (for $0 \leq i \leq n$). When $n = 0$ then Justification-list is underspecified. Each justification J_i is a tuple $\langle \textsc{Name}, \textsc{Form}, \textsc{Subst} \rangle$: Name refers to an assertion. It can be the label of a previous proof step or of an assertion in a mathematical knowledge base, for example, '*deMorgan-2*'. Form is a formula used to justify the asserted proof step. For instance, instead of referring to deMorgan-2, the student may say: "Since $\overline{A \cap (B \cup C)} = \overline{A} \cup \overline{B \cup C}$, from $\Phi[\overline{A \cap (B \cup C)}]$ we obtain $\Phi[\overline{A} \cup \overline{B \cup C}]$." Subst is an explicitly mentioned instantiation of variables the student has applied in the proof step.

All justifications fields can be left underspecified. The field Subst has been introduced mainly for the purpose of exhaustively capturing the student input in our representation. Given an underspecified justification $\langle \textsc{Name}, \textsc{Form}, \textsc{Subst} \rangle$, Form is generally equivalent to $dereference(\textsc{Name}) + \textsc{Subst}$. Assume, for example, that we already have information on Form $:= \overline{A \cap (B \cup C)} = \overline{A} \cup \overline{B \cup C}$. The PM can determine a possible assertion which has been used (e.g. *deMorgan-2*) together with the substitution the student has applied (here $[A \mapsto X, (B \cup C) \mapsto Y]$). In fact, in most proof step utterances in the Dialog corpus the student justifies her proof step with a reference to the employed assertion Name and by specifying the inferred formula Formula: For instance, a student may say: "By *deMorgan-2*, we have Φ". Unification and heuristically guided theorem proving is employed in the PM to support the analysis and completion of different combinations of given and missing information in justifications. Problematic cases typically arise when the student leaves the justification for her proof step underspecified altogether.

The proof step representation language presented here is the one that has been implemented in the PM. In the meantime this language has been further developed in theory (Autexier *et al.* 2003).

Example 1 The underspecified proof step **S1** in the example dialog (see Fig. 1) is represented in the PM as follows:[4]

```
(input (label 1_1)
  (formula (= (C (N (U a b) (U c d)))
              (U (C (U a b)) (C (U c d)))))
  (type ?)
  (direction ?)
  (justifications
    (just   (reference deMorgan-2)
            (formula ?)
            (substitution ?))))
```

Our PM employs the Ωmega-CoRe calculus (Autexier 2003) as a sound and complete base framework (for classical higher-order reasoning) to support resolution of underspecification and PSE. The internal proof representation of the

[4] `C`, `N`, and `U` stand for `complement`, `intersection`, and `union`, respectively. `?` denotes underspecification.

PM is based on *task structures* which are defined on top of the ΩMEGA-CoRE calculus; for more details on this proof representation framework we refer to (Hübner *et al.* 2004).

In some sense, tasks resemble and generalize sequents in sequent calculi. Proof construction in this "ΩMEGA-CoRE + tasks"-framework employs and generalizes well-known techniques in tableau-based theorem proving (cf. (Hähnle 2001) and the references therein) and the matrix method (Andrews 1981; Bibel 1983). See also (Vo, Benzmüller, & Autexier 2003) for further details.

We present two example strategies employed by the PM to relate the student proof step to the PM's internal representation of the current proof state and to formally reconstruct it in order to determine missing information.

Justify by a unifiable premise: The system looks for subterms of the premises of the present task and for subterms of the available assertions in a knowledge base which are unifiable to the student proof step. Such a justification may require further conditions to be discharged. These conditions are extracted with the help of the ΩMEGA-CoRE framework and they form additional proof obligations which are analyzed by an automated theorem prover.

Justify by equivalence transformation and equality reasoning: This case is a generalization of the above one in the sense that the asserted formula does only follow via equivalence transformation and equality reasoning from the premises and assertions available in the proof state. For this strategy we employ a specifically adapted tableau-based reasoner implemented within the ΩMEGA-CoRE framework.

Example 1 (contd.) Our simple example illustrates the above strategies:

1. The asserted formula in the student proof step is unifiable at top-level with the `deMorgan-2` rule. Thus, we recompute a *forward* proof step:

   $$\overline{(A \cup B) \cap (C \cup D)} = (\overline{A \cup B}) \cup (\overline{C \cup D})$$

   is obtained by `deMorgan-2` using the *substitution*:

   $$[X \mapsto (A \cup B); Y \mapsto (C \cup D)]$$

2. On the other hand, our system is able to identify the discrepancies between the asserted formula and the goal formula of the current proof task. Identifying a possible *backward* reasoning step the system thus carries out the following transformation:

   $$\overline{(A \cup B) \cap (C \cup D)} = (\overline{A} \cap \overline{B}) \cup (\overline{C} \cap \overline{D})$$

   is reduced to the new goal formula

   $$\overline{(A \cup B) \cap (C \cup D)} = (\overline{A \cup B}) \cup (\overline{C \cup D})$$

   by rewriting the subterms: $(\overline{A} \cap \overline{B})$ and $(\overline{C} \cap \overline{D})$ with the subterms $(\overline{A \cup B})$ and $(\overline{C \cup D})$, respectively, using the rule `deMorgan-1`.

For the initially underspecified input proof step representation we have thus computed two possible fully specified logical interpretations.

Proof Step Evaluation The PM is now facing the problem of evaluating both identified proof step interpretations along the PSE criteria. Note that soundness has already been partly addressed during the above phase, since we were able to reconstruct the underspecified proof step in at least one way in the current proof state.

Employing heuristically guided theorem proving techniques, our PM finally identifies the following ratings and annotations for our two proof step interpretations:[5]

1.
```
(evaluation
  (reference ...)
  (formula (= (C (N (U a b) (U c d)))
              (U (C (U a b)) (C (U c d)))))
  (substitution ((x (U a b) y (U c d))))
  (direction FORWARD)
  (justification DeMorgan-2)
  (soundness 1)
  (relevance 0.9)
  (granularity 1))
```
2.
```
(evaluation
  (reference ...)
  (formula (= (C (N (U a b) (U c d)))
              (U (C (U a b)) (C (U c d)))))
  (substitution ...)
  (direction BACKWARD)
  (justification (((C (U c d)) . (N (C c) (C d)))
                  ((C (U a b)) . (N (C a) (C b)))))
  (soundness 1)
  (relevance 0.9)
  (granularity 0.5))
```

The overall system then determines a preference for interpretation (1.) since it shares the justification used by the student, *viz.* the rule `deMorgan-2`. Furthermore, the former inference is considered to be granularly more appropriate than the latter. This is because the former employs only one application of the rule `deMorgan-2` while the latter applies the rule `deMorgan-1` twice. As discussed in the previous section, this is generally an over-simplified way to determine the relative granularity of a proof step. A more precise, separate soundness investigation in the PSE phase would also rule out interpretation (2.), provided that the students explicit reference to deMorgan-2 is taken into account.

Further Proof Management Tasks It is important that the system and the student share a mutual understanding about the situation they are confronting. And we have already motivated that the system should be capable of adequately representing the context and the situation in which the student is currently operating and reasoning about. Generally, we consider different classes of situations. Two examples are:

Problem-solving situations: In these situations, alternative problem solving strategies are considered to tackle the problem, e.g. looking for similar problems whose solutions

[5] The ellipses indicate that the field refers to some internal representation which is left out to save space. Note also that the *relevance* rating for both interpretations is 0.9 to allow a margin for error unless the proof step is found to be used in *every* possible proofs in which case the *relevance* rating will be 1.

are known, finding a lemma whose application could bridge the gap between the premises and the goal, searching for applicable proving methods such as *proof by induction*, *diagonalization proof*, etc.

Proof situations: Once a student proof step has been identified as related to an available proof situation in the maintained proof history, a new *current proof situation* is computed and updated into the proof history. The current proof situation consists of the "relevant" proof fragments which have been identified up to this point.

The tasks of reconstructing theorem prover-oriented proof fragments from the student proof steps, organizing the relevant proof fragments into (partial) proofs, keeping track of the proof history and other relevant information for future backtracking, etc. are all handled by the PM. It's also important to note that while the problems of resolving underspecification and PSE have been discussed separately, they are solved in combination since they are mutually dependent.

In general, judging the student's utterances in a mathematics tutoring scenario is a very complex task addressing many AI problems including NL understanding, plan recognition, ambiguity resolution, step-wise proof construction, management of proofs, etc. In our first implementation of the PM, we clearly had to make several simplifications which can later be generalized if future experiments indicate the need for this. We give some examples:

Granularity and the Direction of Inference: If the direction of an inference is not made explicit by the student, the PM tries to determine it by considering the granularity of the proof justifying a forward reasoning step and the granularity of the proof justifying a backward directed goal reduction step; cf. our example from before. If the former is considered to be more difficult than the latter, the system conjectures that this proof step is a forward proof step; otherwise, it is considered to be a backward proof step.

Student Modeling: The granularity of a proof step is relative to the student's knowledge and expertise in the domain under consideration. In the present implementation, the student model and other relevant information have not been taken into account when appraising the student proof step.

Related Work

Empirical findings in the area of intelligent tutoring show that flexible natural language dialog supports active learning (Moore 1993). In the DIALOG project, therefore, the focus has been on the development of solutions allowing flexible dialog. However, little is known about the use of natural language in dialog settings in formal domains, such as mathematics, due to the lack of empirical data.

Input analysis in dialog systems is for most domains commonly performed using shallow syntactic analysis combined with keyword spotting; slot-filling templates, however, are not suitable in our case. Moreover, tight interleaving of natural and symbolic language makes key-phrase spotting difficult because of the variety of possible verbalizations. Statistical methods are employed in tutorial systems to compare student responses with a domain-model built from preconstructed gold-standard answers (Graesser *et al.* 2000). In our context, such a static domain-modeling solution is impossible because of the wide quantitative and qualitative range of acceptable proofs, i.e., generally, our set of gold-standard answers is even infinite.

Related work with regard to interpreting mathematical texts is (Zinn 2004) which analyzes comparably complete, carefully structured textbook proofs, and relies on given text-structure, typesetting and additional information that identifies mathematical symbols, formulae, and proof steps. With respect to our goal of ambiguity and underspecification resolution, (Bos 2003) provides an algorithm for efficient presupposition and anaphora resolution which uses state-of-the-art traditional automated theorem provers for checking consistency and informativeness conditions.

Recent research into dialog modeling has delivered a variety of approaches more or less suitable for the tutorial dialog setting. For instance, scripting is employed in Autotutor (Person *et al.* 2000) and knowledge construction dialogs are implemented in Geometry Tutor (Matsuda & VanLehn 2003). Outside the tutorial domain, the framework of Information State Update (ISU) has been developed in the EU projects TRINDI[6] and SIRIDUS[7] (Traum & Larsson 2003), and applied in various projects targeting flexible dialog. An ISU-based approach with several layers of planning is used in the tutorial dialog system BEETLE (Zinn *et al.* 2003).

Finally, the dialogs in our corpus reveal many challenges for human-oriented theorem proving. Traditional automated theorem provers (e.g. OTTER and Spass) work on a very fine-grained logic level. However, interactive proof assistants (e.g. PVS, Coq, NuPRL, Isabelle) and in particular proof planners (e.g. OMEGA and λClam) support abstract-level reasoning. The motivation for abstract-level reasoning is twofold: (a) to provide more adequate interaction support for the human and (b) to widen the spectrum of mechanizable mathematics. Proof assistants are usually built bottom-up from the selected base-calculus; this often imposes constraints on the abstract-level reasoning mechanisms and the user-interface.

Conclusion

We have identified novel challenges and requirements to MDR in the context of tutorial NL dialogs on mathematical proofs. For instance, we must be able to explicitly represent and reason about ambiguous and underspecified student proof steps in the PM. The represented proof steps may be unsound, of unacceptable granularity or not relevant. The analysis of these criteria is then the task of PSE. Generally, resolution of underspecification and PSE are mutually dependent. Except for pure logical soundness validation of proof steps, none of these requirements can currently be easily supported within state-of-the-art theorem provers. Thus, novel and cognitively interesting challenges are raised to the deduction systems community.

PSE can principally be supported by different approaches — including ones that avoid dynamic theorem proving as

[6] `http://www.ling.gu.se/research/projects/trindi/`

[7] `http://www.ling.gu.se/projekt/siridus/`

presented in this paper. We list some alternative approaches according to increasing difficulty:

1. We could statically choose one or a few ‘golden proofs’ and match the uttered partial proofs against them.
2. We first generate from the initially chosen golden proofs larger sets modulo, for instance, (allowed) re-orderings of proof steps and match against this extended set.
3. We dynamically support PSE with heuristically guided abstract-level MDR.
4. We interpret the problem as challenge to proof theory and try to develop a proper proof theoretic approach to differentiate between ’tutorially good proofs and proof steps’ and ’tutorially less good proofs and proof steps’ in the space of all proofs for a given problem.

The space of all proofs that solve a proof problem is generally infinite which is one reason why a static modeling of finitely many ’golden solutions’ as in approaches (1) and (2) is generally insufficient in our context. Approach (3) is our currently preferred choice and a first, still rather naive, approach to the logical dimension of this challenge has been presented in this paper. Much further research is clearly needed. Approach (4) is the approach we want to additionally investigate in the future; some relevant related work in proof theory to capture a notion of good proofs is presented in (Dershowitz & Kirchner 2003).

For (3) we have developed a heuristically guided MDR tool that is capable of representing, constructing and analyzing proofs at the assertion level. In the first place these proofs maybe sound or non-sound. For naive set theory (our mathematical domain of choice so far) this tool has been able to reconstruct and represent student proofs at the same level of argumentative complexity as given in the DIALOG corpus. We conjecture that this is a basic requirement for PSE in tutorial settings. We have also shown how (in the same mathematical domain) our PM resolves ambiguities and underspecification in the student input and how it evaluates the student input along the three major dimensions of *soundness*, *relevance*, and *granularity*. The application of our approach to more challenging mathematical domains and its evaluation therein is future work.

References

Andrews, P. B. 1981. Theorem proving via general matings. *J. of the ACM* 28(2):193–214.

Autexier, S.; Benzmüller, C.; Fiedler, A.; Horacek, H.; and Vo, B. Q. 2003. Assertion-level proof representation with underspecification. *ENTCS* 93:5–23.

Autexier, S. 2003. *Hierarchical Contextual Reasoning*. Ph.D. Dissertation, Saarland University, Germany.

Benzmüller, C.; Fiedler, A.; Gabsdil, M.; Horacek, H.; Kruijff-Korbayová, I.; Pinkal, M.; Siekmann, J.; Tsovaltzi, D.; Vo, B. Q.; and Wolska, M. 2003. Tutorial dialogs on mathematical proofs. In *Proc. of the IJCAI 03 Workshop on Knowledge Representation and Automated Reasoning for E-Learning Systems*.

Bibel, W. 1983. *Automate Theorem Proving*. Friedr. Vieweg.

Bos, J. 2003. Implementing the the binding and accomodation theory for anaphora resolution and presupposition projection. *Computational Linguistics*.

Buckley, M., and Benzmüller, C. 2005. System description: A dialog manager supporting tutorial natural language dialogue on proofs. In *Proc. of the ETAPS Satellite Workshop on User Interfaces for Theorem Provers (UITP)*.

Dahlbäck, N.; Jönsson, A.; and Ahrenberg, L. 1993. Wizard of oz studies – why and how. *Knowledge-Based Systems* 6(4):258–266.

Dershowitz, N., and Kirchner, C. 2003. Abstract saturation-based inference. In *Proc. of LICS 2003*. Ottawa, Ontario: ieee.

Fiedler, A., and Tsovaltzi, D. 2003. Automating hinting in an intelligent tutorial dialog system for mathematics. In *Proc. of the IJCAI 03 Workshop on Knowledge Representation and Automated Reasoning for E-Learning Systems*.

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

Hähnle, R. 2001. Tableaux and related methods. In Robinson, A., and Voronkov, A., eds., *Handbook of Automated Reasoning*, volume I. Elsevier Science. chapter 3, 101–176.

Huang, X. 1994. Reconstructing Proofs at the Assertion Level. In *Proc. of CADE-12*, number 814 in LNAI, 738–752. Springer.

Hübner, M.; Autexier, S.; Benzmüller, C.; and Meier, A. 2004. Interactive theorem proving with tasks. *ENTCS* 103(C):161–181.

Matsuda, N., and VanLehn, K. 2003. Modelling hinting strategies for geometry theorem proving. In *Proc. of the 9th International Conference on User Modeling*.

Moore, J. 1993. What makes human explanations effective? In *Proc. of the 15th Annual Conference of the Cognitive Science Society*.

Person, N. K.; Graesser, A. C.; Harter, D.; Mathews, E.; and the Tutoring Research Group. 2000. Dialog move generation and conversation management in AutoTutor. In *Building Dialog Systems for Tutorial Applications—Papers from the AAAI Fall Symposium*. North Falmouth, MA: AAAI press.

Rips, L. J. 1994. *The psychology of proof*. MIT Press, Cambridge, Mass.

Traum, D. R., and Larsson, S. 2003. The information state approach to dialogue management. In van Kuppevelt, J., and Smith, R., eds., *Current and New Directions in Discourse and Dialogue*. Kluwer. `http://www.ict.usc.edu/~traum/Papers/traumlarsson.pdf`.

Vo, Q. B.; Benzmüller, C.; and Autexier, S. 2003. Assertion application in theorem proving and proof planning. In *Proc. of IJCAI-03*. IJCAI/Morgan Kaufmann.

Wolska, M.; Vo, B. Q.; Tsovaltzi, D.; Kruijff-Korbayová, I.; Karagjosova, E.; Horacek, H.; Gabsdil, M.; Fiedler, A.; and Benzmüller, C. 2004. An annotated corpus of tutorial dialogs on mathematical theorem proving. In *Proc. of LREC*.

Zinn, C.; Moore, J. D.; Core, M. G.; Varges, S.; and Porayska-Pomsta, K. 2003. The be&e tutorial learning environment (beetle). In *Proc. of Diabruck, the 7th Workshop on the Semantics and Pragmatics of Dialogue*.

Zinn, C. 2004. *Understanding Informal Mathematical Discourse*. Ph.D. Dissertation, University of Erlangen-Nuremberg.