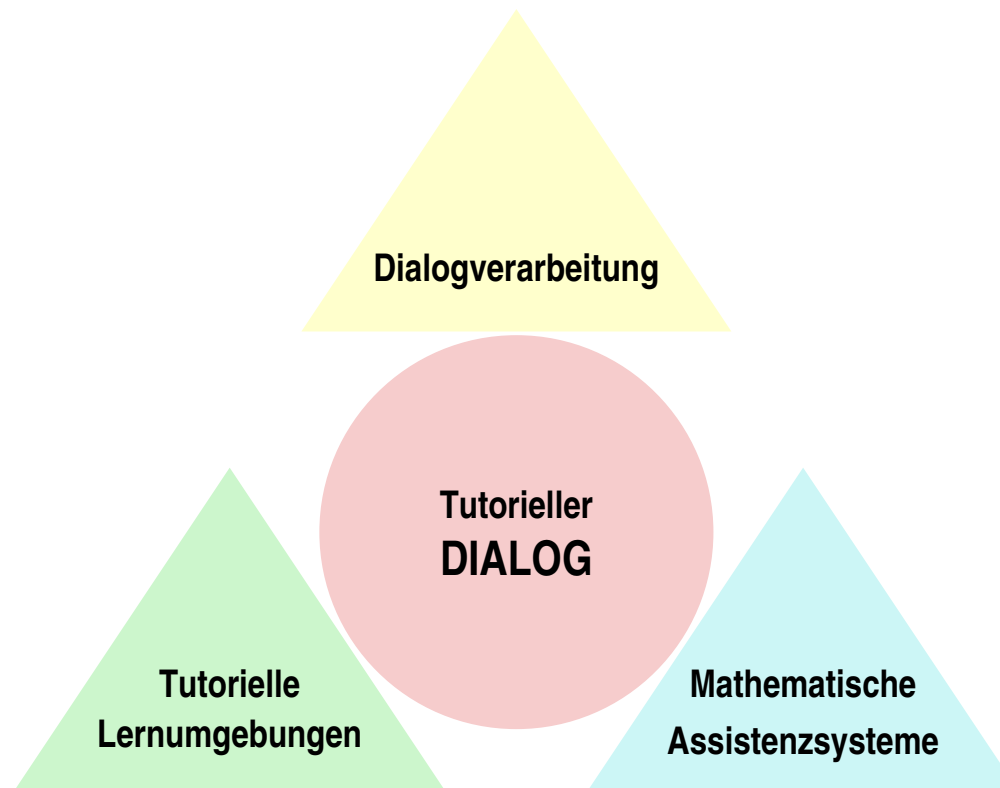


DIALOG: Natural Language-based Interaction with a Mathematics Assistance System

Christoph Benz Müller

Department of Computer Science, Saarland University

4-6 October, Saarbrücken, Germany

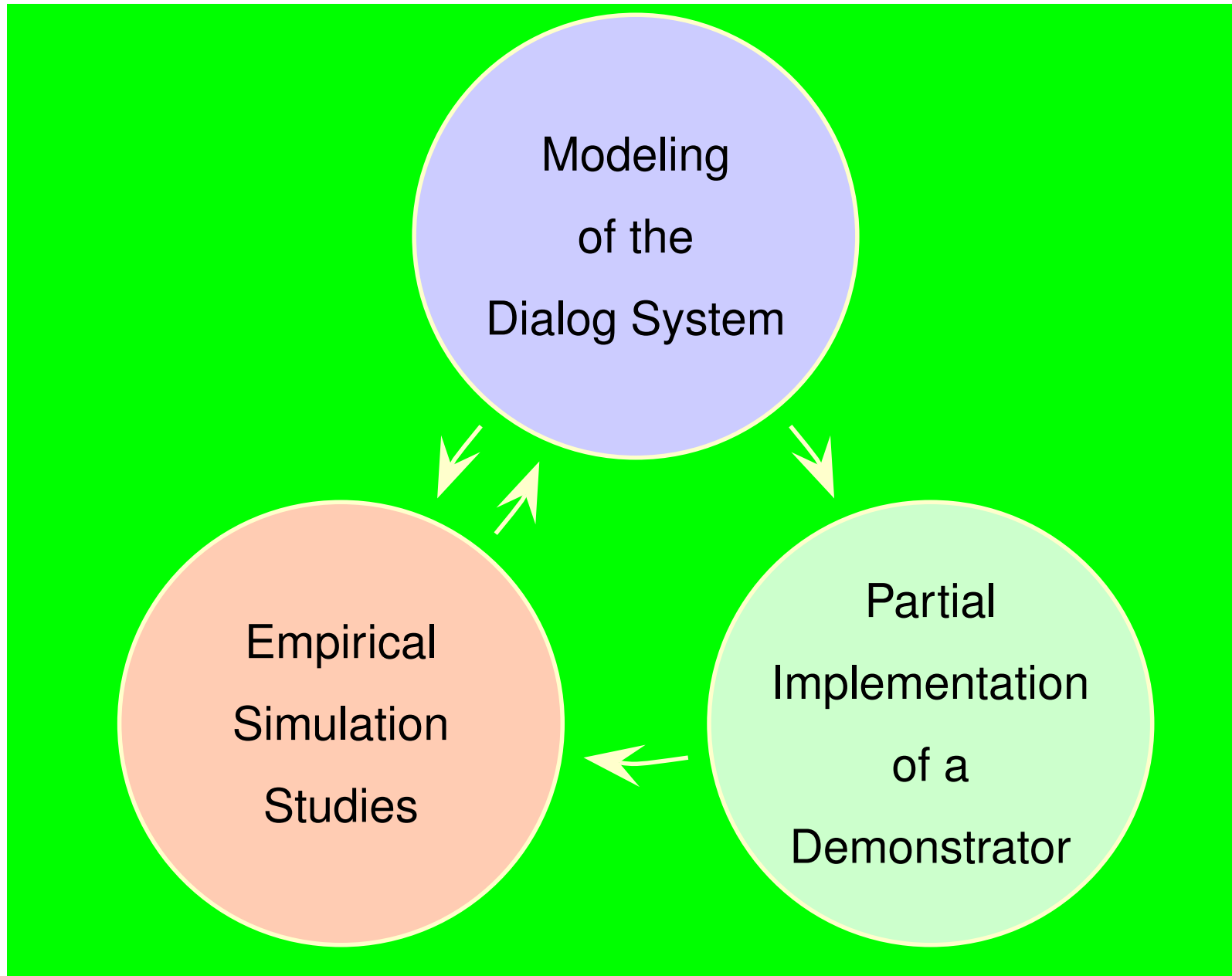


- Joint project (between Coli and CS) as part of the SFB378 on *Resource-adaptive cognitive processes*
- Selected mathematical domain: naive set theory

- **Directly Involved Researchers:** Manfred Pinkal (Coli), Jörg Siekmann (CS), Christoph Benz Müller (CS), Ivana Kruijff-Korbayova (Coli), Magdalena Wolska (Coli), Quoc Bao Vo (CS), Armin Fiedler (CS), Dimitra Tsovaltzi (CS)
- **Collaborators:** Serge Autexier (CS), Malte Gabsdil (Coli), Helmut Horacek (CS), Alexander Koller (Coli), Erica Melis (CS)
- **Hiwis and Students:** Mark Buckley (CS), Hussain Syed Sajjad (CS), Marvin Schiller, Jochen Setz (CS), Michael Wirth (Coli), Sreedhar Ellisetty (CS), Andrea Schuch (Coli), Beata Biehl (Coli), Oliver Culo (Coli)

Method: Progressive Refinement

Dedtreff'04 & LogInf'04

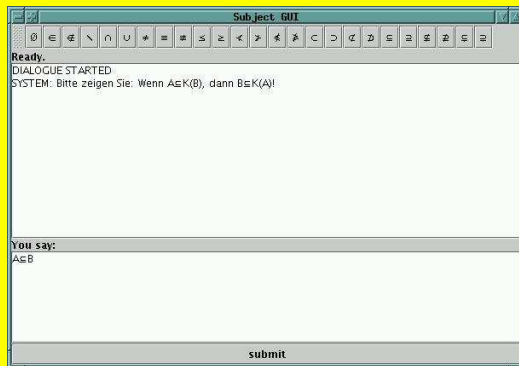
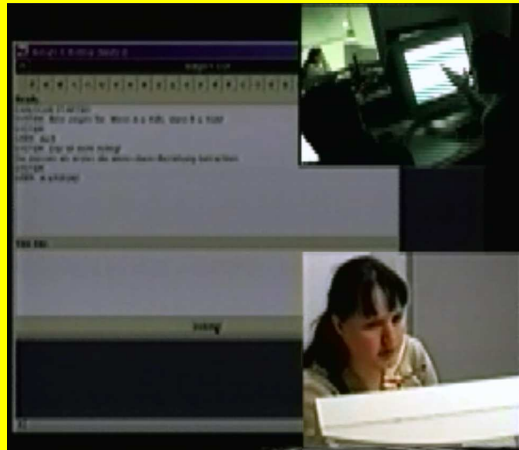


WOZ-Experiment → Own Corpus Dedtreff'04 & LogInf'04

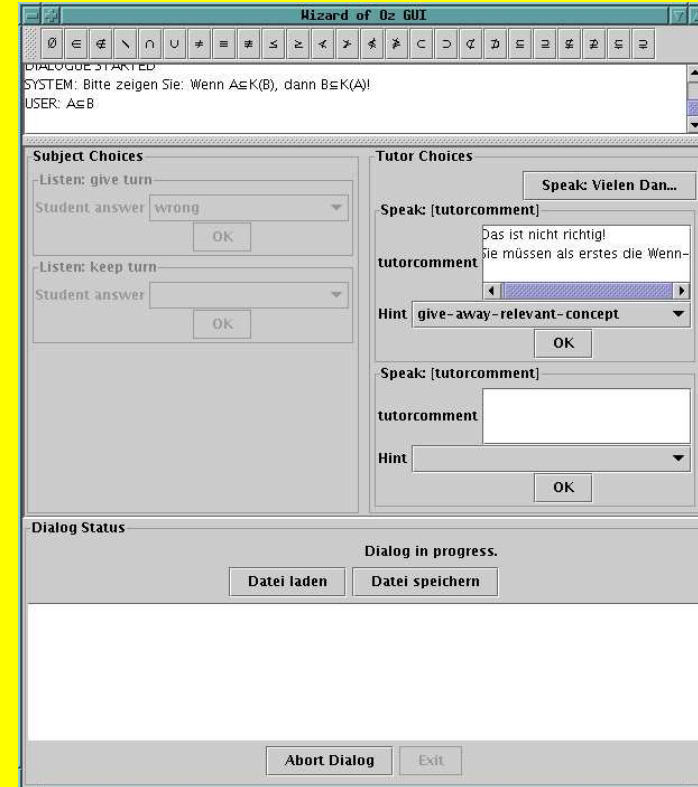
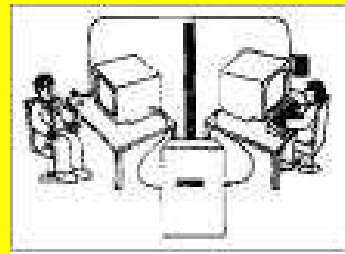
- collect data
(for analysis of phenomena)
- test coarse grained model of system
- test hinting algorithm
- 24 Subjects:
 - ▶ university students
 - ▶ varying background
 - ▶ varying math knowledge
- Wizard:
 - ▶ mathematician with tutoring experience
 - ▶ assisted by developers of hinting algorithm
- Experimenter

WOZ-Experiment → Own Corpus

Dedtreff'04 & LogInf'04



The subject window



The wizard window

1. Preparation and pre-test:
 - fill in background questionnaire
 - study written lesson material (basic concepts and a collection of lemmata)
 - (try to) prove (on paper) the theorem $K(A) \in P(K(A \cap B))$
2. Tutoring session: evaluate a tutoring system with NL dialog capabilities (enter partial steps of a proof and having an interaction with the system)
3. Post-test and evaluation questionnaire:
 - (try to) prove (on paper) another theorem
 - fill in a questionnaire addressing various aspects of the system and its usability

Example: didactic, vp16, dryrun

T1: Bitte zeigen Sie : $K((A \cup B) \cap (C \cup D)) = (K(A) \cap K(B)) \cup (K(C) \cap K(D))$
!</s>

S1: (correct) nach deMorgan-Regel-2 ist $K((A \cup B) \cap (C \cup D)) = (K(A \cup B) \cup K(C \cup D))$ </s>

T2: Das ist richtig !</s>

S2: (correct) $K(A \cup B)$ ist laut DeMorgan-1 $K(A) \cap K(B)$ </s>

T3: Das stimmt auch .</s>

S3: (correct) und $K(C \cup D)$ ist ebenfalls laut DeMorgan-1 $K(C) \cap K(D)$ </s>

T4: Auch das stimmt .</s>

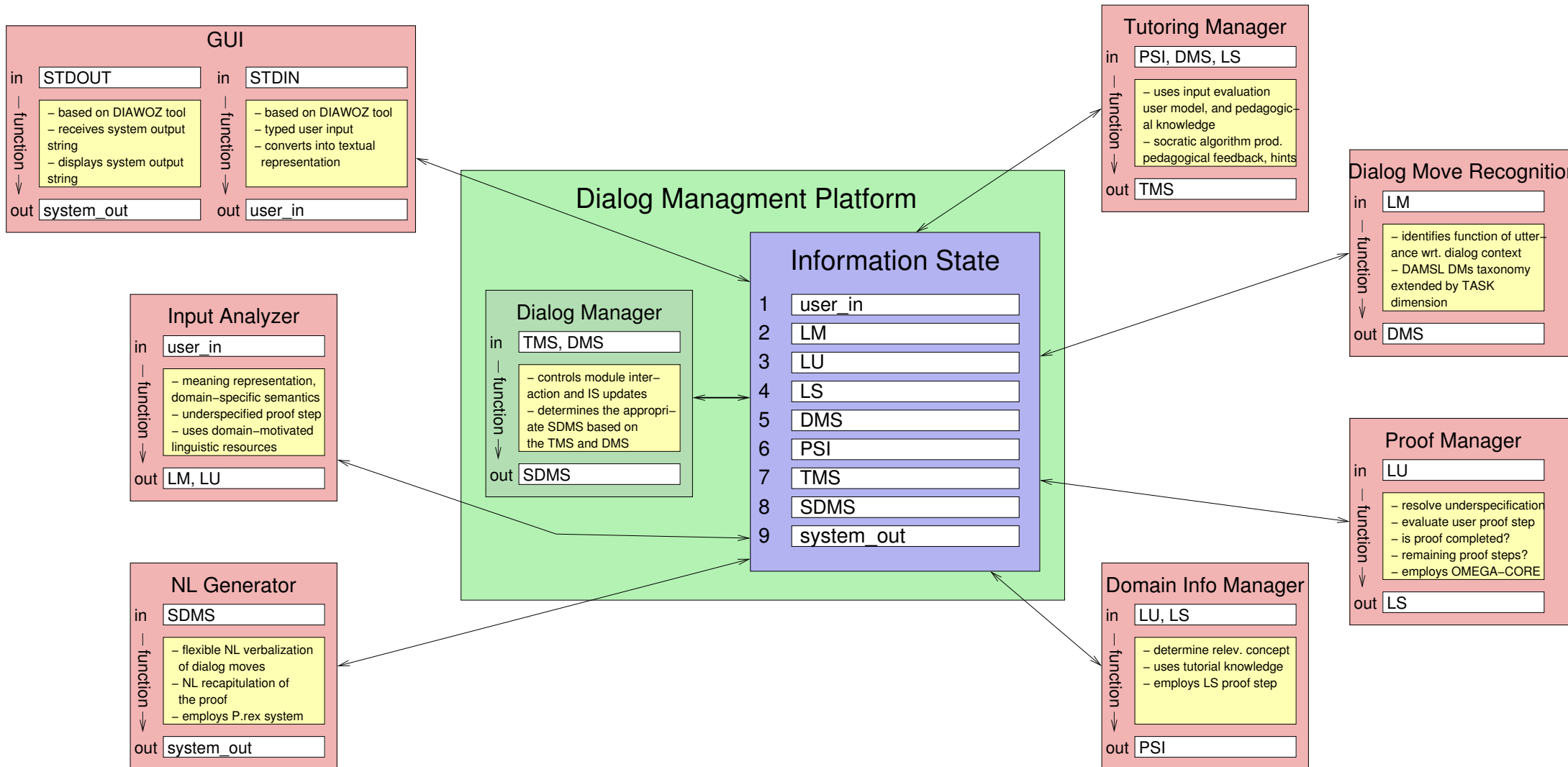
S4: (correct) also folgt letztendlich : $K((A \cup B) \cap (C \cup D)) = (K(A) \cap K(B)) \cup (K(C) \cap K(D))$.</s>

T5: Das stimmt genau .</s> Ich wiederhole noch einmal : Auf die linke Seite der Gleichung kann ich zuerst die zweite und danach die erste de-Morgan-Regel anwenden , so daß sich folgende Argumentationskette ergibt : $K((A \cup B) \cap (C \cup D)) = K(A \cup B) \cup K(C \cup D) = (K(A) \cap K(B)) \cup (K(C) \cap K(D))$.</s>

- opened and entered new field of NL based mathematical tutoring dialogs
- foremost aim: obtain a general view of the **interplay between** advanced NL processing in a flexible tutoring dialog, and dynamic, abstract-level mathematical domain reasoning.
- moved from collecting empirical data through modeling of the different components and their interfaces to a demonstrator implementation.

Demonstrator Architecture

Dedtreff'04 & LogInf'04



Example Utterance

Dedtreff'04 & LogInf'04

T1: Bitte zeigen Sie : $K((A \cup B) \cap (C \cup D)) = (K(A) \cap K(B)) \cup (K(C) \cap K(D))$
!

S1: (correct) nach deMorgan-Regel-2 ist $K((A \cup B) \cap (C \cup D)) = (K(A \cup B) \cup K(C \cup D))$

T2: Das ist richtig !

Abbrev.	Meaning	Example
STDIN	standard input	<i>“nach deMorgan-Regel-2 ist $K((A \cup B) \cap (C \cup D)) = (K(A \cup B) \cup K(C \cup D))$”</i>
LM	linguistic meaning	s : @h1(holds \wedge <Norm>(d1 \wedge deMorgan-Regel-2) \wedge <Patient>(f1 \wedge FORMULA1))
LU	proof language with underspecification	(input (label 1_1) (formula (= (complement (intersection (union a b) (union c d))) (union (complement (union a b)) (complement (union c d))))) (type ?) (direction ?) (justifications (just (reference DeMorgan-2) (formula ?) (substitution ?) (role:from))))
LS	system-oriented proof language (+ evaluation)	((KEY 1_1) \longrightarrow ((Evaluation (expStepRepr (label 1_1) (formula (=complement(intersection(union(A B) union(C D))) union(complement(union(A B)) complement(union(C D))))) (type inference) (direction forward) (justification ((reference demorgan-2) (formula nil) (substitution ((X union(A B) Y union(C D)))) (role nil)))) (StepCat correct))) (ProofCompleted false) (completeProofs ... see next poster ...))
DMS	dialog move specification	{ fwd = “Assert”, bwd = “Address_statement”, commm = “”, taskm = “”, comms = “”, task = “Domain_contribution” }
PSI	proof step information	{ domConCat: “correct”, proofCompleted: false, proofstepCompleted: true, proofStep: “”, relConU: true, hypConU: true, domRelU: false, iRU: true, relCon: “” “+(char)8745”, hypCon: “” “+(char)8746”, domRel: “”, iR: “deMorgan-Regel-2” }
TMS	tutorial move specification	{ mode= “min”; task= (signalAccept; {proofStep= “”; relCon= “”; hypCon= “”; domRel= “”; iR= “”; taskSet= “”; completeProof= “”}) }
SDMS	system dialog move specification	{ mode = “min”; fwd = “Assert”; bwd = “Address_statement”; task = (“signalCorrect”, {proofStep= “”, relCon= “”, hypCon= “”, domRel= “”, iR= “”, taskSet= “”, completeProof= “”}); comms = “”; commm = “”; taskm = “” }
STD_out	textual representation of NL output	<i>“Das ist richtig”.</i>

- Experiment design, empirical test environment, and Wizard of Oz tool.
- First experiment in naive set theory domain, with written dialog input and output.
- Preliminary corpus investigation and subsequent formal annotation at several levels of interpretation: deep semantic structure, dialog moves, and tutorial task aspects.
- Coarse grained architecture and specification of refined modules for input analysis, proof management, and tutorial dialog moves (especially hinting).

- Realization of input analysis using a deep dependency-based grammar, focusing on uniform interpretation of informal interleaved natural language and mathematical formulae.
- Proposal of a user-oriented proof language with underspecification.
- Realization of the proof manager: proof representation languages for the proof manager, interfacing to the underlying domain reasoner (the Omega theorem prover); agent-based assertion reasoning that can enumerate proof step suggestions.
- Design and development of a demonstrator.

- Experiment & WOZ Tool & Corpus
- NL Analysis
- Tutoring Aspects
- Dialog Management
- **Proof Manager**
- **Proof Step Evaluation**
- Modeling of System & Demonstrator

Overview, Papers, Corpus, etc.: see

<http://www.ags.uni-sb.de/~chris/dialog/>

- Resolution of
 - ▶ Ambiguities
 - ▶ Underspecification
- Proof Step Evaluation
 - ▶ Soundness: Can the proof step be verified by a formal inference system?
 - ▶ Granularity: Is the granularity (i.e., 'logical size' or 'argumentative complexity') of the proof step acceptable?
 - ▶ Relevance: Is the proof step needed or useful in achieving the goal?

Discourse:

- (1) From previous observations we know that A or B.
- (2) The former implies D by Lemma X.
- (3) Similarly, from the latter follows C.

Alternative user utterances with underspecification:

- (a) From **this** follows D since C implies D by Lemma Y.
— *this may refer to (1)+(2)+(3), to (3), or even (1) or (2) with wrong justification*
- (b) It holds D since C implies D by Lemma Y.
— *no underspecified anaphoric reference but **ambiguity at domain reasoning level***

Ambiguities and Underspecification

Dedtreff'04 & LogInf'04

<i>Example</i>	<i>Where does ambiguity arise?</i>	<i>Ambiguity resolution means</i>
(1) $x \in B$ und somit $x \subseteq K(B)$ und $x \subseteq K(A)$ wegen Voraussetzung (2) A enthaelt B	linguistic meaning level; attachment, coordination	linguistic means; type checking in (2)
(3) $P((A \cup C) \cap (B \cup C)) = PC \cup (A \cap B)$ (4) $K((A \cup C) \cap (B \cup C)) = KC \cup (A \cap B)$	linguistic meaning level; informal character of discourse	type checking for (3); domain reasoning for (4)
(5) T1: Bitte zeigen Sie: $K((A \cup B) \cap (C \cup D)) = (K(A) \cap K(B)) \cup (K(C) \cap K(D))$! S1: nach deMorgan-Regel-2 ist $K((A \cup B) \cap (C \cup D)) = (K(A \cup B) \cap K(C \cup D))$	underspecified proof step	domain reasoning

Assertions already introduced

(A1) $A \wedge B$.

(A2) $A \Rightarrow C$.

(A3) $C \Rightarrow D$.

(A4) $F \Rightarrow B$.

(G) $D \vee E$.

Alternative proof step directives.

(a) Aus den Annahmen folgt D.

(b) B gilt.

(c) Es genügt D zu zeigen.

(d) Wir zeigen E.

<i>Criterion</i>	<i>Task (first approach)</i>	<i>Requirements for theorem prover</i>
Soundness	$E \vdash_C^? D \vee E$	'Yes' or 'No' answer; any theorem prover resp. calculus C
Granularity	proof-steps($E \vdash_C^? D \vee E$)	adequate abstract-level theorem prover resp. calculus C; measure 'shortest' proof; take tutorial constraints into account; proof planning or assertion level reasoning?
Relevance	$ \begin{array}{l} A \wedge B \\ A \Rightarrow C \\ C \Rightarrow D \quad \vdash_C^? E \\ F \Rightarrow B \end{array} $	recognize detours; compare with other 'shorter' proofs; take tutorial constraints into account; forward case more challenging

What are the right provers?

Dedtreff'04 & LogInf'04

???

To support flexible, tutorial NL dialog on mathematical proofs we need to

- adapt our deduction systems to support specific tasks
- e.g., to reason about user uttered proof steps
- some of these tasks require abstract-level reasoning systems

From a perspective of cognitive science:

Very fascinating application domain for deduction systems.

Assertions already introduced

(A1) $A \wedge B$.

(A2) $A \Rightarrow C$.

(A3) $C \Rightarrow D$.

(A4) $F \Rightarrow B$.

(G) $D \vee E$.

Alternative proof step directives.

(a) Aus den Annahmen folgt D.

(b) B gilt.

(c) Es genügt D zu zeigen.

(d) Wir zeigen E.

Soundness verification of utterance (a) boils down to proving the theorem:

$$P1 : (A \wedge B), (A \Rightarrow C), (C \Rightarrow D), (F \Rightarrow B) \vdash D$$

Analogously, for the backward reasoning step given in (d) we get:

$$P2 : E \vdash (D \vee E)$$

No specific requirements imposed on the proof system \vdash (any kind of first-order theorem prover)

Requires analyzing the 'complexity' or 'size' of proofs: For utterances (a) and (d) above, it thus boils down to judging about the complexity of the proof tasks (P1) and (P2).

For illustration consider Gentzen's ND as proof system \vdash .

Define *argumentative complexity*: number of \vdash -steps in the smallest \vdash -proof.

Judgements:

- argumentative complexity of (a) is bigger than that of (b)
- argumentative complexity of (a) is above a (tutorially) motivated complexity threshold

Granularity Evaluation (contd.)

Dedtreff'04 & LogInf'04

Natural deduction calculus probably not appropriate: two intuitively very similar user proof steps may actually expand into natural deduction proofs of completely different size.

Question: What is an appropriate choice of a proof system \vdash ?
It should reflect the argumentative level of human reasoning.
Empirical studies are possible and planned.

Alternative approaches listed according to increasing difficulty:

- Statically choose one or a few “golden proofs” and match the uttered partial proofs against them.
- Generate from the initially chosen golden proofs larger sets modulo, for instance, (allowed) re-orderings of proof steps and match against this extended set.
- Dynamically support relevance analysis with domain reasoning. For this, we test whether a proof can still be obtained from the new proof situation (using an abstract-level proof system). Resource-bound enumeration of possible proofs and proof step matching is additionally required.
- Stimulate research in proof theory: compact and tractable representation of the proofs in the proof space.

Backward reasoning case (c): $D \vee E$ is refined to goal D .

Relevance question: can a proof still be generated? The task is thus identical to proof task (P1) as before.

A backward proof step that is not relevant according to this criterion is (d) since it reduces to:

$$P3 : (A \wedge B), (A \Rightarrow C), (C \Rightarrow D), (F \Rightarrow B) \vdash E$$

for which no proof can be generated. Thus, (d) is a sound refinement step that is not relevant, in contrast to utterance (c).

Approach needs to be refined: exclude detours and take tutorial aspects into account (teaching particular styles of proofs, particular proof methods, etc.).

More challenging forward reasoning case is discussed next.

Example (a):

$$P4 : \quad (A \wedge B), (A \Rightarrow C), (C \Rightarrow D), (F \Rightarrow B), D \vdash (D \vee E)$$

The question whether D is relevant reduces to the question whether there exists a proof for the given task that employs D and which is shorter than the best proof that can be obtained when deleting D from the available knowledge. According to this approach, utterance (b) describes an non-relevant proof step.

Note that we do not just ask about the existence of an arbitrary proof but about the existence of a proof with particular properties. This requires techniques such as (resource-bound and heuristic guided) enumeration of proofs.