Modelling solution step discussions in tutorial dialogue

Mark Buckley

Dissertation

zur Erlangung des akademischen Grades eines

Doktors der Philosophie

der Philosophischen Fakultäten

der Universität des Saarlandes

Juli 2010

Vorsitzender der Promotionskommission:	Prof. Matthew W. Crocker
Berichterstatter:	Prof. Manfred Pinkal
	Prof. Johanna D. Moore

Abstract

This thesis is concerned with intelligent tutoring systems with natural language interfaces, which combine research from the fields of dialogue modelling and pedagogical science. Their development is motivated by the discovery that the learning gains achieved by human one-on-one tutoring are much higher than those achieved by traditional classroom instruction. The mechanisms which have been proposed as the cause of this difference include deep explanatory questioning by the tutor and self-explanation by the student, both of which target the student's deep understanding of domain concepts. Such interactions have been found to exhibit recurring dialogue patterns known as dialogue frames.

A further pattern in conversational communication is that of grounding, the process by which conversational partners reach mutual understanding and repair communication errors. The store of their mutual beliefs which is built up through grounding is known as the common ground. Just as the tutor requires the student to show evidence of understanding of domain concepts, so too does the speaker expect the hearer to indicate in the course of grounding that the utterance content has been understood. Our research will use concepts from grounding and the structure of dialogue frames to investigate the tutor's choice of whether to accept or reject solution steps or to ask explanatory questions which request evidence of understanding.

We consider the following hypotheses: First, by modelling the structure of discussions of solution steps, we can maintain a representation of what the student has shown evidence of having understood, based on the outcomes of previous such discussions. And second, knowing that discussion elicits self-explanation, we can then use this representation to predict whether to enter into a discussion of the latest contribution, using only features drawn from the previous dialogue and from the analysis of the steps the student just contributed. The computational model we will propose to investigate these questions will account for subdialogues which are solution step discussions, and is based on a concept which we will call task-level grounding. Task-level grounding is the process by which contributions to the current solution are proposed, possibly discussed and finally accepted or rejected. The object of this process

is the student's deep understanding of domain concepts.

We evaluate the model in two stages. First we train a classifier from annotated data to predict whether to request evidence of understanding from the student or not, thereby entering into a solution step discussion. We find that the classifier takes advantage of the information maintained in our task-level grounding model, which confirms our first hypothesis. Although it performs well for acceptance and rejection of steps, the performance for the decision whether to request evidence of understanding is low. This result is mitigated by the second evaluation, in which we elicit ratings of the model's output in context from human experts. The model's output is rated as similarly appropriate compared to the corpus, which confirms our second hypothesis. We also find that the task of detecting when to perform requests for evidence of understanding is difficult even for human experts.

Zusammenfassung

Diese Arbeit befasst sich mit intelligenten Tutorsystemen, bei denen Forschungen aus den Bereichen Dialogmodellierung und Pädagogik vereint werden. Die Motivation hinter der Entwicklung solcher Systeme beruht auf der Feststellung, dass bei Einzelunterricht durch Menschen höhere Lerneffekte erzielt werden als bei herkömmlichem Gruppenunterricht. Zu den Mechanismen, die dahinter vermutet werden, gehören unter anderem tiefgehende Erklärungsfragen durch den Tutor sowie die Erklärung der eigenen Lösungen durch den Lernenden. Ziel beider Mechanismen ist das tiefgehende Verständnis des Lernenden für Konzepte in der Domäne. Solche Dialoge weisen wiederkehrende Muster auf, die *dialogue frames* genannt werden.

Ein weiteres Muster in der menschlichen Kommunikation ist *grounding*, der Prozess, durch den Dialogpartner versuchen, einen Zustand des gegenseitigen Verständisses zu erreichen und Kommunikationsfehler zu beheben. Der sogenannte *common ground* enthält die Gesamtheit ihrer gemeinsamen Annahmen. So wie der Tutor einen Beweis für das tiefgehende Verständis der Domänenkonzepte von dem Lernenden verlangt, erwartet beim *grounding* auch der Sprecher, dass der Hörer zeigt, dass der Inhalt seiner Äußerungen verstanden wurde. Unsere Forschung verwendet Konzepte aus dem *grounding* und die Struktur von *dialogue frames*, um die Entscheidung von Tutoren zu untersuchen, ob ein Lösungsschritt akzeptiert oder abgelehnt werden soll, oder ob eine Erklärungsfrage gestellt werden soll.

Wir betrachten folgende Hypothesen: (1) Durch das Modellieren der Struktur von Diskussionen über Lösungsschritte ist es möglich, aufgrund des Verlaufs bisheriger Diskussionen eine Repräsentation dessen zu verwalten, was der Lernende bisher verstanden hat. (2) Da Diskussionen zu Erklärungen der eigenen Lösungen führen, ist es möglich, aus dieser Repräsentation heraus die Entscheidung des Tutors für oder gegen eine Diskussion vorherzusagen, und dabei nur Merkmale aus dem bisherigen Dialog und der Analyse des aktuellen Lösungsschrittes heranzuziehen. Das operationale Modell basiert auf unserem Konzept des *task-level grounding*. Dies bezeichnet den Prozess, in dem Lösungsbeiträge vorgeschlagen, möglicherweise diskutiert und anschließend entweder akzeptiert oder abgelehnt werden können. Gegenstand dieses Prozesses

ist das Verständis des Studenten der Domänenkonzepte.

Schließlich evaluieren wir das Modell in zwei Phasen. Zunächst wird ein Klassifizierer anhand annotierter Daten trainiert, um vorherzusagen, ob von dem Lernenden ein Beweis für das Verständnis verlangt werden soll. Dies bedeutet auch das Eröffnen einer Diskussion über den letzten Lösungsschritt. Wir stellen fest, dass der Klassifizierer die Informationen nutzt, die von dem Modell des *task-level grounding* verwaltet werden. Damit wird unsere erste Hypothese bestätigt. Obwohl der Klassifizierer gute Resultate für Akzeptanz und Ablehnung von Lösungsschritten erzielt, sind die Erbegnisse im Hinblick auf die Entscheidung, ob ein Beweis des Verständnisses verlangt werden soll, nicht zufriedenstellend. Diese Resultate werden jedoch gemäßigt durch eine zweite Evaluierung, bei der die Vorhersagen des Klassifizierers durch menschliche Experten bewertet werden. Die Vorhersagen werden von den Experten im Vergleich zum durch das Korpus dargestellten Goldstandard als ähnlich angemessen angesehen. Dies bestätigt unsere zweite Hypothese. Wir stellen zudem fest, dass die Aufgabe, zu erkennen, wann nach einem Beweis verlangt werden soll, auch für menschliche Experten sehr schwierig ist.

Acknowledgments

My *Doktorvater* was Prof. Manfred Pinkal, who gave me the chance to come to work at Coli and gave me time to find my research topic in the field I wanted to work in. Over the years it was the simple questions that were the hardest, but they were always the ones I benefited the most from. Thank you Manfred for your expertise, for your interest, and for your guidance.

My main collaborator in nearly all of the work during this PhD and even before was Magdalena Wolska. Every time I had a dark day and said "none of this makes any sense", Magda was able to remind me that it did. So thank you Magda for your endless good ideas and your genuine enthusiasm for our work, without which I certainly would never have gotten nearly as far as I have now.

My research was funded by the International Research Training Group for Language Technology & Cognitive Systems at Saarland University. As part of this program I had the opportunity to spend seven months at the School of Informatics at the University of Edinburgh, supervised by Prof. Johanna Moore. Thank you Johanna for taking the time to host my stay and giving me a new perspective on my work.

Many other people have also made contributions to this thesis, some small, some larger. Thank you all of you (in no particular order) whose brains I've picked, who have listened to me explain my research problems, or who have shared your expertise in my field of work or in yours: Chad Brown, Bart Cramer, Dominik Dietrich, Hagen Fürstenau, Oliver Lemon, Verena Rieser, Marvin Schiller, David Schlangen, Maria Staudte, Juliane Steinberg, David Traum, and Claus Zinn. Thank you also to Boris Fersing who did the external annotation of our dialogue data.

Whether it's doing your maths homework in primary school or writing a PhD, parents seem able to find just the right combination of "you're nearly there, keep going" and "aren't you finished yet?" Somehow both forms of motivation work equally well, even though no-one likes admitting their parents knew better. So to Evan and Rosemary, thank you both for many many years of support in seemingly never-ending education.

On the other side of the family, thank you to Renate and Paul, who kept me very well fed in the final few weeks of writing up, and gave me the coolest

room in their house during the warmest two weeks of the year.

To Yvonne, for so much of all of the above, and so much much more: Thank you.

Contents

List of Tables

List of Figures

1
Introduction

Education is part of everyone's lives, starting from an early age and usually lasting for many years. In systems of formal schooling the use of computer-based learning technology has become the norm across the full range of school subjects. Using computers in the classroom takes advantage of multimedia to engage students in challenging and entertaining tasks [58]. Formal subjects such as mathematics or physics are particularly suited to being taught using these tools because there are well-defined curricula and because the material lends itself to automatic analysis. Some teaching systems go beyond the provision of study material and presentation of exercises and use more natural interfaces, offer adaptive feedback and tailor their behaviour to a student model. Such systems are referred to as *intelligent tutoring systems* (ITSs) [58, 156, 180].

Bloom [22] and Cohen et al. [54] have demonstrated the benefits of one-on-one tutoring in comparison to classroom teaching. Students who were given individual tutoring on average achieved a learning gain of 0.4 to 2.3 standard deviations above those who had covered the same material in a traditional classroom setting [22]. Cohen et al. [54] find that even when tutors are knowledgeable about the material but otherwise untrained as tutors, this benefit still exists. Unfortunately there is a clear resource bound when it comes to providing such teaching methods in the school system, because it is too expensive to provide dedicated tutors for every student. The potential benefits of one-on-one tutoring have therefore motivated a vein of research on the provision of ITSs which mimic the behaviour of human tutors [87, 2, 129, 217, 72, 77]. Their common goal is to develop models and systems with which students can interact using natural language dialogue.

Instruction by humans through natural language has been shown by Moore [142] to be more effective than automatically generated natural language, and studies of human tutoring have identified certain individual features of tutorial dialogue which can account for its effectiveness as measured by learning gain. These include for instance directed questions to elicit explanations [86], student initiative [176, 60], collaborativity [85, 68, 187] and self-explanation [42, 3]. These studies indicate that it is the interaction with the system rather than the mere exposure to the material which causes the direct positive effect on the student's learning [199].

Tutoring has been investigated across a wide range of domains, from well defined closed world simulations [163] to less well defined open domains such as ethics [99]. One domain which is formally well defined but whose theories are open ended is mathematics, and as such it is a suitable candidate for ITS research [2, 217]. This is especially true of mathematical theorem proving. Although it is less well investigated for tutoring than other fields of mathematics and offers strong challenges, it holds promise for intelligent tutoring systems because as Hersh [98] argues, it is the explaining aspect of proofs that leads to learning. Further, there are a number of automated reasoners for mathematics which can support the system by analysing the student's input.

Graesser et al. [86] have shown that questions cause students to explain, so motivated by the learning effect of explanations, ITSs should try to ask the student questions about the content which is to be learned. But under what conditions should the tutor provoke explanations by engaging the student in a discussion about the content at hand? Tutors ask questions depending on what they believe the student's current knowledge state to be [46], so whether the student has shown evidence of having understood certain concepts will be one of these conditions.

A second direction from which it is possible to analyse discussions in tutorial dialogue is from the point of view of grounding. Successful conversational communication depends strongly on the coordination of meanings and background assumptions as to the state of the world [49, 183, 188]. Dialogue participants try to achieve a situation in which they mutually believe that their utterances are being interpreted as intended and that their assumptions as to the shared knowledge, the common ground, agree. To this end, they engage in a process called grounding [51, 191], whose purpose is to ensure explicit alignment of beliefs. Grounding can serve to avoid or recover from communication failures arising from problems which may range from low level signal-related issues through the interpretation of the propositional content up to the communicative intentions of speech acts [174].

Tutoring is inherently prone to misalignment of beliefs beyond the level of

the communicative intentions of speech acts: namely at the level of deep understanding of the tutored domain. This is partly because tutoring interactions are characterised by an asymmetry of the knowledge possessed by the tutor and the learner [83, 146, 119]. There is also an uncertainty on the part of the tutor as to the learner's deep understanding and the overall knowledge state. Indeed Chi et al. [46] have shown that tutors tend to have difficulties in estimating the learner's deep understanding. The posing of questions in tutoring can nevertheless facilitate the maintenance of the belief states of the student and tutor [84], particularly concerning the student's deep understanding of the concepts in the tutoring domain. The tutor's beliefs about the student's understanding will form a second condition influencing the tutor's decision whether to discuss contributions or not.

Our work will take advantage of theories from dialogue modelling, such as grounding, in the context of ITSs for mathematics in order to model the circumstances in which tutors ask explanatory questions in reaction to contributions of solution content from the student.

1.1 A model for solution step discussions

This thesis proposes a model for solution step discussions, which are subdialogues in tutorial dialogue in which an individual contribution to the current solution is discussed by the student and tutor. Tutors enter into such discussions in order to elicit explanations from the students, either because they suspect the student may not know the concepts, or simply with a view to the learning effect of the explanation. Students, when engaged in such discussions, must offer explanations of their solution steps, thereby demonstrating that they have understood the concepts necessary to perform such steps. This interaction leads to the student experiencing a learning effect and to the tutor having a better model of the student's knowledge state.

Our model will take advantage of two theories of grounding in dialogue which maintain and align the content of the common ground [51, 191]. The sequences of actions which they account for result in typical subdialogues which are pervasive in communication [166] and which Chi et al. [46] have found also occur in tutorial dialogue.

Another account of subdialogues comes from an analysis of the collaborative nature of tutorial dialogue by Graesser et al. [85]. They find recurring local patterns known as *dialogue frames*, which describe the structure of discussions between the student and the tutor about the student's answers. They consist of five phases: a question is posed by the tutor, and is followed by an answer from the student. The tutor gives feedback on this answer, and then may

enter into a phase of collaborative improvement of the answer, where elaborations of the answer, hints or traces of explanations can take place. Finally the tutor assesses the student's understanding of the answer. The collaborative improvement phase is what sets tutoring apart from classroom instruction, and it is here that the learning effects of tutoring are achieved.

In considering how to combine previous work on dialogue modelling and pedagogical science to account for discussion of solution steps the following research questions have emerged:

- By modelling the structure of the discussion in the dialogue frame of a student's contributed solution steps, can we maintain a representation of what the student has shown evidence of having understood, based on the outcomes of previous such discussions?

- Knowing that discussion elicits self-explanation, which in turn leads to learning, can we then use this representation to predict whether to enter into a discussion of the latest contribution, using only features drawn from the previous dialogue and from the analysis of the steps the student previously contributed?

The computational model we will propose to investigate these questions will account for subdialogues which are solution step discussions. It is based on a concept which we will call *task-level grounding* [30, 31], a notion inspired both by grounding and by dialogue frames. Task-level grounding is the process by which contributions to the current solution are proposed, possibly discussed and finally accepted or rejected. What is being grounded is the student's deep understanding of domain concepts.

Sequences of task-level grounding actions follow patterns similar to those found in dialogue frames. A task-level grounding interaction begins with the contribution of a solution step by the student. It may be discussed in order for the tutor to ascertain that the student in fact understands the concepts involved in the step, but only when the tutor is convinced about the student's understanding will the step be accepted. When such an interaction ends with the acceptance of the solution step then it has the side-effect that this content is added to the common ground of the dialogue—in other words, the student and the tutor come to mutually believe that the student has successfully demonstrated mastery of this content. This way the common ground serves as a rudimentary model of the student's knowledge state, to which concepts that the student successfully uses are continually added. The content of the dialogue state representation is used to choose the tutor's future actions, therefore the model allows the student's knowledge state to be taken into account in deciding whether to enter into discussions in the future.

1.2 Methodology

The methodology we will follow in investigating our hypotheses begins with the qualitative analysis of a corpus of tutorial dialogues on mathematical theorem proving. We will analyse the corpus to try to find evidence of the occurrence of dialogue frames, and to categorise the types of actions that tutors and students perform in this kind of tutorial dialogue. With this categorisation in mind we will propose task-level grounding, a model for dialogue frames which is similar in style to Traum's grounding acts model [191] but in which the object of the grounding process is the student's deep understanding of domain concepts.

Based on the categorisation of actions and their occurrence within the dialogue frame, we will then develop our computational model of task-level grounding to describe solution step discussions. It will be a finite-state model of the legal sequences of actions in the dialogue frame, and will maintain a dialogue state representing the student's knowledge state in the common ground. We will use the information state update approach [193], in which a dialogue model is characterised as a set of transitions between states which are triggered when dialogue participants perform dialogue moves. The information state can include a representation of the common ground [135].

A further line of previous research which we will build upon is machine learning for dialogue systems. Walker [204] has shown for instance that the choice of the system's action can be successfully modelled using machine learning, whereby the action is predicted given a representation of the state of the previous dialogue and the most recent user utterance. We perform a series of experiments whose goal is to learn a classifier which predicts the tutor's task-level grounding actions, in particular, whether the tutor should ask an explanation-eliciting question after a solution step has been proposed. The classifier is trained on features which the dialogue model maintains in its dialogue state representation. Such a classifier can play the role of action selection algorithm in an implemented ITS. Our goal with the classifier is to demonstrate the usefulness of the dialogue model: If the classifier uses features from the dialogue state representation and predicts the tutor's actions well, then we can conclude that it is useful to maintain these features in an ITS with dialogue capabilities. A prerequisite for such machine learning experiments is the existence of labelled data. To this end we will annotate our corpus with the concepts we introduced in the model of task-level grounding, namely the task-level grounding actions, and with the mathematical content of the student's solution steps.

Finally we perform a further validation of the model with a subjective hu-

man expert rater study, in which we ask expert tutors to rate the output of the model and the actions in the corpus in context. We surmise that some of the classifier's misclassifications may in fact be appropriate actions in context, because a tutor typically has the option to perform any one of a number of possible actions. This evaluation will allow us to compare how appropriate different possible actions are in the same context. Our hypothesis here is that the model's predictions will be rated as well as the corpus actions. We also elicit ratings of an accept/reject baseline in order to measure the difficulty of the prediction task for human experts.

We envisage a number of potential benefits of the model whose development this thesis reports. Firstly, the recognition and assignment of dialogue structure can help interpret what a dialogue participant intends. For instance, obligations can be a source of expectations for what the student may do [194], which offers a constraint for the interpretation of intentions which might otherwise be ambiguous [35]. Secondly, our model explicitly handles the questions which elicit self-explanation, which leads to learning [44], and the questions themselves (or their answers) help tutors to model students' knowledge states [86]. Finally, having a representation of the student's knowledge state to act as a rudimentary student model helps an ITS choose an appropriate action, in our case it helps decide when to ask questions about the latest contribution. Our model facilitates this by augmenting the representation of the student's knowledge state when the student has shown evidence of understanding of content. This in turn informs the future decisions about whether to discuss this content again or not.

1.3 The contributions of this thesis

We see the following contributions emerging from this thesis:

- We offer an analysis of students' and tutors' behaviour in tutorial dialogues on mathematical proofs, which focuses on actions which affect their belief states.

- We propose a model of dialogue frames for this type of tutorial dialogue. It borrows concepts from Traum's grounding acts model, and like Traum's model it is computational, in the sense that it can be implemented and used in a run-time system.

- We contribute an annotation of the corpus with the concepts drawn from our model and with mathematical concepts, as well as the schemata and guidelines developed for this task.

- We show that machine learning can be used to predict task-level grounding actions, and the classifier's predictions are similarly appropriate when compared to the corpus as a gold standard. This result also shows us which information from the previous dialogue is important in choosing such actions.

1.4 Thesis structure

This thesis is structured as follows. We begin in Chapter 2 with a review of the contributing fields of pedagogical science, dialogue modelling and intelligent tutoring systems. We present theoretical and empirical results as well as implementations of dialogue-based ITSs, including the DIALOG project, the research project within which this work was conducted. Our model is motivated by a corpus of one-on-one tutoring sessions which was collected by the DIALOG project. Chapter 3 presents our qualitative analysis of this data, which is a categorisation of the types of actions that students and tutors perform with regard to the discussion of solution steps. We show that the dialogues exhibit structures which correspond to the definition of dialogue frames of Graesser et al. [87].

Chapter 4 details the concept of task-level grounding and our proposed model of solution step discussion which is based on it. We describe the similarities between task-level grounding and Traum's grounding acts model, and argue that task-level grounding accounts for the same subdialogues in our corpus as the dialogue frame model does. We give the details of the action types and the finite-state model which describes their possible sequences, and present a series of examples.

Having developed a rule-based model for solution step discussion we would like to validate it in two ways. First we will perform a quantitative analysis of the occurrence of the actions we identify in Chapter 3. Second we will use the information that the model maintains for a series of supervised learning experiments. To support both of these goals we develop annotation schemata on three levels to perform an annotation of the corpus, which is reported in Chapter 5. The annotation levels are task-level grounding actions, dialogue moves following an adaptation of the DAMSL schema [5] and the mathematical content of the solution steps.

In Chapter 6 we present the results of the supervised learning experiments and describe how the experiments confirm our hypotheses about the use of dialogue features in choosing task-level grounding actions. We show how we derived the data set from the annotated corpus and present five experiments in turn, followed by a misclassification analysis.

The further evaluation of the classifier is reported in Chapter 7. Here we take a subset of the misclassified cases from the previous chapter and use them to generate stimuli for a rating experiment. We introduce the approach of using expert human raters, and present our experimental design to have the classifier's choices rated against the gold-standard corpus. We then present the results and a discussion. Chapter 8 offers some conclusions and an outlook.

2

Dialogue modelling for effective ITSs

The research direction which we will follow in this thesis combines results and theories from dialogue modelling, pedagogical science and intelligent tutoring systems. This chapter presents previous work in each of these three fields. The overarching theme is the belief states of tutors about their students with respect to deep knowledge about the domain of teaching, and the dialogue structures which arise in relation to this.

In the field of dialogue modelling we will review developments culminating in general models of dialogue agency, in which an agent's actions are influenced by its beliefs, goals and intentions. We focus of matters of mutual belief as well as its attainment and maintenance through the process of grounding. We also present briefly some approaches to implementation, including the information state update approach.

We then review a series of studies in the area of educational psychology which show that natural language tutoring is highly effective compared to traditional classroom instruction. We present a number of features which have been proposed to explain this, including the mechanism of self-explanation, which has been shown to lead to strong learning effects. Self-explanation prompted by explanatory questions is found within dialogue structures called dialogue frames, which occur frequently in tutorial dialogue. We also present some views on the use and utility of belief states and common ground in tutoring.

Finally we look at the state of the art in ITSs which use dialogue interfaces.

We first highlight some of the practical issues involved in an end-to-end automated tutorial dialogue system, as well as some aspects of teaching mathematics in this context. We then review a series of ITSs which are relevant to our research, either because they implement sophisticated dialogue models or because they teach difficult mathematical domains. Following this we give an outline of the DIALOG project, the research project within which the work reported in this thesis has been conducted.

2.1 Issues and approaches in dialogue modelling

Tutorial dialogue is a form of task-oriented dialogue, and for modelling task-oriented dialogue many different approaches have been proposed. This section reviews general-purpose dialogue modelling, with a particular focus on how the beliefs of dialogue participants are handled.

2.1.1 Dialogue modelling

A dialogue model describes patterns of interaction between dialogue participants. It encodes legal sequences of utterances and maintains a representation of the state of the interaction. Dialogues are typically modelled at the level of speech acts, which are defined by Austin [11] as the action a speaker performs by saying something. A speech act is made up of a locutionary act, which is the act of speaking the words, an illocutionary act, which is the act the speaker wants to perform with the utterance, and a perlocutionary act, which is the effect that the utterance has on the listener. The term speech act has come to denote only the illocutionary aspect. In more recent dialogue systems research the concept of speech act has been extended to the concept of dialogue act, or dialogue move [153], as used for example in the DAMSL taxonomy [5]. A dialogue move uses dimensions to encode the different functions of the utterance, and these summarise the intentions of the speaker. The backward-looking function encodes the relationship of the utterance to the preceding discourse, and the forward-looking direction constrains the future beliefs and actions of the participants, and affects the discourse. The forward-looking function corresponds to the purpose of the utterance, and is close to Austin's speech act. An example is (2.1), which has as its backward-looking function the reference to a previous utterance in the discourse, and has the forward-looking function of imposing an obligation on the hearer to give an explanation.

(2.1) Could you explain that please?

A dialogue system embeds a dialogue model and uses it to choose the next move that the system should perform. Early research on dialogue modelling

investigated the regularities of local sequences of actions. The notion of turn-taking was encoded by Sacks et al. [168] with a set of rules which apply at points in the dialogue where the turn can change hands. A more advanced notion related to turn-taking is that of adjacency pairs [122, 170]. These are pairs of adjacent utterances produced by different speakers in which the first speaker offers the turn to the second. They are ordered into a first and a second part, and the first part restricts the type of utterance which can occur in the second part, such as question-answer or greeting-greeting.

Later models focus on dialogue as an interaction between agents, and try to capture what it is that makes these agents perform utterances. They use to a greater or lesser degree the concepts of beliefs, desires and intentions (BDI) of an agent. An agent's beliefs are what it holds to be true about the world, its desires are the goals it wishes to achieve, and its intentions are the actions that it will execute to achieve those goals. The agent's beliefs include beliefs about other agents' beliefs, for example "A believes that B knows a fact P". Since dialogue is a collaborative process [52], agents will cooperate to achieve their goals, and will form joint intentions, which are "joint commitments to perform a collective action while in a shared mental state" [55, pg 3].

Cohen and Perrault [56] propose an early BDI model by formalising speech acts in a planning framework. They see the intentions which underlie speech acts as plans and the performer of the speech acts as the agent carrying out such plans. An agent's beliefs about itself or other agents are captured by the modal operator BELIEVES. Planning operators representing the speech acts for requesting and informing are formalised in a STRIPS-like system.

Litman and Allen [127, 128] also propose a formalisation speech acts using planning. Their approach builds on work by Grosz and Sidner [92] which showed that the structure of a task-oriented dialogue mirrors the structure of the task under discussion, and that a dialogue can be structured into segments according to the intentions being followed by the conversational partners in those segments. They advance Cohen and Perrault's formalisation by differentiating between domain plans, which express knowledge about the task at hand, and discourse plans, which are generated by discussing a domain plan. This allows the dialogue to contain for instance clarification subdialogues and topic changes. The motivation is the fact that subdialogues such as clarifications do not have a corresponding element in the task-level plan. They are thus part of the discourse plan but not the domain plan. Plans are generated from the goals and intentions of agents and stored on a stack of executing, suspended and completed plans which represents one agent's view of the joint plan that it and the other agent are constructing and executing.

Poesio and Traum [152] have proposed a model which unifies models from the BDI tradition with discourse representation theory [111], a model of the semantics of anaphora. Their representation of the context of the dialogue, which they call the discourse situation, includes information about agents' mental states, including their beliefs.

Complex dialogue genres such as negotiation [200] or collaborative planning [95] exhibit phenomena such as mixed initiative, topic changes and possible differing conflicting goals. These require solutions which can handle the agents' planning process explicitly. Blaylock and Allen [21] have proposed a model of dialogue based on a formal model of collaborative problem solving for agents. It is motivated by two kinds of plan-based dialogue models: those that model planning dialogue as joint activity, for instance SharedPlans [90, 91, 131], and those that model plan execution, for instance Cohen et al. [57]. The model covers both planning and execution in the same dialogue by abstracting to the level of agent-based collaborative dialogue, in which planning and execution can take place in an interleaved way, and different agents jointly choose and execute different parts of the same plan.

2.1.2 Maintenance of belief in dialogue

An important element of the dialogue models based around agency is the beliefs an agent holds, because they constitute the agent's view of the world and of the state of its conversational partner. However beliefs not only inform an agent's choice of action, they are also affected in turn by the actions that that agent and others perform. This also holds for mutual belief, in other words what an agent believes that another agent believes, and vice versa [48], and successful conversational communication depends strongly on the coordination of meanings and mutual background assumptions as to the state of the world [49, 183, 188]. As the dialogue progresses the store of mutual beliefs held by the dialogue participants grows. This store is called the common ground [182], and in the following we review a number of dialogue theories have been developed which focus on this.

Many approaches to dialogue modelling, including the BDI theories above, work under the assumption that assertions are simply added to the common ground, but this is an idealised situation. Utterances are not always correctly understood, and non-understanding can be signalled by using a token like "Pardon?", meaning that the content of an utterance could not be added to the common ground. The process by which the common ground is established and maintained during a dialogue is known as grounding [50, 51]. In performing grounding, the shared beliefs of the dialogue participants are con-

stantly aligned as new information is added to the dialogue. It is also a way to recover from the communicative failures associated with misunderstandings. The term grounding can refer to both the performance of a grounding utterance like "I see..." and the act of assimilating commonly held information.

Clark and Schaefer [51] see grounding as the process of adding information to the common ground by way of contributions. In their Contribution Model a contribution made by a speaker is considered to be grounded when the hearer has shown to have understood the propositional content which was uttered. In the presentation phase the speaker presents a contribution for the hearer's consideration. In the acceptance phase the hearer demonstrates having understood the contribution by one of five methods, including continued attention, making a relevant next contribution, and backchannel acknowledgement of the utterance. Traum [191] proposes the Grounding Acts model, a computational approach which avoids the problem in Clark and Schaefer's theory that acceptances also have to be accepted. A set of grounding acts manipulate the content and status of the discourse unit, which represents the content which is to be grounded. They include initiation, acknowledgment, and repair. A finite state machine models the sequences of grounding acts which lead from the speaker's initiation to the hearer's acknowledgment, thereby adding the content to the common ground of the dialogue. Traum's model has been implemented in the TRAINS system [190].

The common ground is part of the broader dialogue context, which represents the state of the interaction between the dialogue participants. In making an analogy between a baseball game and a dialogue game, Lewis [124] proposes the conversational scoreboard as a representation of the current state of the dialogue. The scoreboard is a list of the values of all contextual parameters which describe the dialogue. What can occur in the dialogue is constrained by the values in the conversational scoreboard, and its values evolve in a rule-governed way as the dialogue progresses.

Stalnaker [182] proposes a context which represents the commonly accepted information at a given point. At some time point t there is a set of assumptions which are commonly held at t. When an utterance is made, its descriptive content can be added to the context if it is not inconsistent with the context. However, as Ginzburg [80, 79] argues, this view of context, due to its lack of an inner structure, does not account for the discursive potential of the dialogue. In Stalnaker's account new things have as a precondition the totality of what has been hitherto accepted into the common ground, which is not always the speaker's intention.

Ginzburg proposes a notion of context which, in addition to the common ground, explicitly represents what is being discussed in the dialogue at time t.

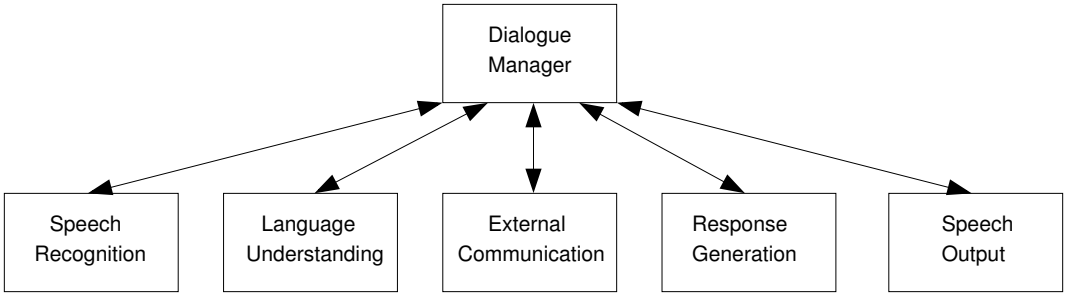

Figure 2.1: General architecture of a spoken dialogue system (from [137], page 113).

He extends the discourse context with LATEST-MOVE in order to introduce an aspect of locality into the context. LATEST-MOVE stores the syntactic and semantic content of the newest utterance of the dialogue. However not all utterances relate to the utterance directly preceding them, and for this reason Ginzburg proposes a more general account of what is being discussed in a dialogue, known as questions under discussion (QUD). This is a partially ordered set of questions which are currently being discussed, and its maximal element is the current topic of discussion. New questions are added to the top of the QUD as new topics of discussion. A question can be removed from the QUD if information is added to the common ground which decides the question, or which indicates that no information about the question can be provided.

The concept of belief in dialogue modelling will be important in this work because we want to model what the tutor believes the student to currently know. We will see in Section 2.2.2 that the achievement of common ground between the tutor and student can be one of the mechanisms behind questions in tutorial dialogue.

2.1.3 Embedding a dialogue model in a dialogue manager

In a dialogue system a dialogue model is implemented within a dialogue manager, which connects modules of the system and controls their execution. The dialogue manager allows information to be passed from one module to the next through its representation in the dialogue model. A general architecture for spoken dialogue systems proposed by McTear [137] is shown in Figure 2.1. The external communication in the diagram refers to back end systems with which a dialogue system may communicate. For instance an information delivery dialogue system will access a database to find the information that the user is seeking. For a tutorial dialogue system the external communication is with an expert system which can analyse the student's task-related input, or

with a store of pedagogical knowledge, or with a student model.

The techniques used for the implementation of a dialogue model in this setting depend on the complexity of the dialogue genre. For simpler tasks such as command and control of machines, finite state methods are often suitable [138]. Here the dialogue is encoded statically in a finite state machine in which nodes represent dialogue states and edges represent actions, or dialogue moves. Such models are however inflexible and cannot vary for instance initiative. Information seeking tasks like flight timetable information [97] can use form-based models of dialogue [10]. The system tries to fill slots in the form which represent the information that is needed to perform a database lookup. Slots can be filled out of order, and overanswering is possible.

More recently the information state update (ISU) approach [193] has been developed within the TRINDI project. It is motivated by the need to be able to formalise different theories of dialogue management to allow evaluation and comparison. From an engineering perspective, the ISU approach is motivated by the fact there are no hard and fast rules governing the design of dialogue systems, leading to bad support for reusability. The ISU approach proposes a unifying view of dialogue management based around the information state, in which domain independent theories can be implemented in a reusable foundation. The information state of a dialogue is "the information necessary to distinguish it from other dialogues" [193, pg 3]. It represents the cumulative effect of previous actions in the dialogue, and provides a context for future actions. It is similar to concepts such as the conversational score [124] or discourse context [92].

The ISU approach provides a method for specifying a theory of dialogue, defined by the following components:

- An *information state*
- *Representations* for the information state
- A set of *dialogue moves*
- A set of *update rules*
- An *update strategy*

The information state is the description of the state of the discourse and its participants which is maintained by the dialogue manager. It stores dialogue level knowledge, such as the common context, linguistic and intentional structure, or aspects of beliefs or obligations, depending on what theory of dialogue it formalises. This context then forms the basis for the choice of action of the

dialogue manager. For each aspect of dialogue context that should be modelled, a representation must be chosen. This can range from a simple data structure like a list or a string to a more complex representation such as an attribute-value matrix, a term in a lambda calculus, or a discourse representation structure.

Dialogue moves provide an abstraction away from utterances and other dialogue actions to a description of their function. When a dialogue move is performed its content may result in a change being made to the state of the dialogue. Which dialogue moves a dialogue theory includes is influenced by the theory itself and the domain of the dialogue. As a dialogue progresses, the information state which describes it must be updated to reflect the effect that actions of the dialogue participants have on the dialogue context. How these updates take place is governed by update rules. Update rules fire in reaction to observed dialogue moves, and are specified by applicability conditions and effects. If the conditions are satisfied by the information present in the information state, then the effects of the rule can be carried out. The effects are changes that will be made to the information state. Thus update rules can be seen as transitions between information states. In order to control how updates are made to the information state, an update strategy must be declared. This is an algorithm which decides which update rules should be allowed fire. Options for this algorithm include allowing the first applicable rule to fire, allowing all applicable rules to fire, or choosing between rules based on probabilistic information. The information state update approach has been used in a number of research dialogue systems [116, 135, 23] as well as for example the tutorial dialogue system BEETLE [59] or the Witas project [120]

2.2 Dialogue-based one-on-one tutoring

We now review research from the field of pedagogical science which has investigated the use of dialogue in one-on-one tutoring. The role of the student in the interaction is to solve an exercise or series of exercises, or to answer questions from the tutor. The tutor guides the student through the solution construction process, offering feedback and remediation if needed. One-on-one tutoring is normally differentiated from classroom situations because in the classroom the interaction between the student(s) and the tutor does not take place to the same extent [85]. One-on-one tutoring is also different from peer learning, or collaborative learning, in which multiple learners interact [68, 94, 187]. In this case there is no tutor present, and learning is an outcome of the knowledge exchange between the peers. In the following we will simply use the term tutoring to refer to one-on-one tutoring.

Tutoring has been shown by Bloom [22] to be more effective than classroom instruction. This study compared groups of learners who received instruction either in normal classroom situation, in a classroom situation but with feedback, and in a tutoring situation. Students who were given individual tutoring on average achieved a learning gain of 0.4 to 2.3 standard deviations above those who had covered the same material in a traditional classroom setting. Cohen et al. [54], in a meta-analysis of school tutoring studies, also found benefits of tutoring over classroom teaching, and showed that this benefit still exists even when the tutors are knowledgeable about the material but not necessarily trained as expert tutors. Lu et al. [133] however found differences between expert and non-expert tutors which led to the expert tutors being more effective, such as using demonstration, summaries and prompts. Tutoring has also been shown by VanLehn et al. [199] to be more beneficial than studying text in certain conditions. They found that the target audience of the text affected whether it was less effective than tutoring or not. Texts tailored to novices or intermediates were equally effective as tutoring, whereas reading of intermediate texts by novices was less effective than tutoring.

The effectiveness of natural language explanations in tutoring by humans in comparison to automatically generated text has been shown by Moore [142], who compared explanations generated by an ITS to explanations generated by human tutors in the same dialogue state. This raises the question of what characteristics of natural language tutoring cause a learning effect. As an aside, since our work will only consider the text modality, we will not concentrate on the effects of spoken dialogue tutoring. Although for instance Litman et al. [130] have shown spoken dialogue to be more effective in human-human interactions, most tutorial dialogue systems still rely on typed communication. The work we review here does not concern characteristics of tutorial dialogue which are specific to either the spoken or written modality.

There have been many studies suggesting and demonstrating different facets of tutorial dialogue which may be responsible for its effectiveness, such as Chi et al. [45] or Merrill et al. [139]. The features which are proposed include the collaborative nature of tutorial dialogue [85, 94], student initiative [176, 60], directed questions from the tutor to elicit explanations [86], self-explanation by the student [42, 3] and tailored feedback and hinting from the tutor [104, 214].

Tutorial dialogue is collaborative in the sense that the tutor and the student are working on the common goal of solving an exercise, or more abstractly, the goal of learning new content. Their interaction allows tutoring to be seen as a collaborative activity in the sense of Clark [48]. Graesser et al. [85], whose work is presented in more detail in the next section, find that patterns of col-

laborativity lead to learning. In a study of face to face human-human tutoring they recognise cooperation between the tutor and student to solve the task, for instance when the tutor fills gaps in the student's knowledge. The collaborative nature of the dialogue manifests itself in the high number of turns per question in comparison to classroom instruction. Hausmann et al. [94] investigate co-construction of knowledge as a learning device in collaborative learning groups. They find that collaborativity facilitates co-construction, which leads to more generated knowledge by learners.

Student initiative is attributed to contributions which change the course of the tutorial session [175]. Core et al. [60] have measured student initiative in a comparison of didactic and socratic tutoring strategies, hypothesising that didactic dialogue would exhibit more tutor initiative and socratic dialogues would have mixed initiative. They annotated initiatives in a corpus of human-human dialogues and found no relation between initiative and learning in the didactic condition, but the socratic condition was more interactive, and interactivity correlated with learning. Shah et al. [176] have categorised initiatives in a corpus and find the student's initiatives include requests, challenges, refusals to answer and conversational repair.

In a study of the questions asked by students in a corpus of human-human dialogues, Graesser and Person [84] find that students ask questions 240 times as frequently as students in classroom instruction. Questions are categorised by quality, and the study finds that the quality of questions correlates with student achievement, whereas the quantity does not. Deep reasoning questions correlate highly with achievement. A follow up study [86] looks at the questions asked by the tutor, and argues that why, how and what-if questions elicit explanatory reasoning from students, such as justification.

Demonstrations by the student of explanatory reasoning are a form of self-explanation when they are performed in the context of the student's previous answer. Students studying examples or textbook text learn with greater understanding when they explain the study materials to themselves [43]. Chi [42] has examined self-explanations in dialogue interactions. She considers three possibilities for the effectiveness of tutoring: that the tutor's actions cause learning, for instance asking questions, that the student's answers are responsible, in this case self-explanations, or that it is a combination of the two, which results in the co-construction of knowledge. In a case study she finds evidence for the latter two hypotheses. VanLehn et al. [197] have implemented the effects of self-explaining in a computational cognitive model which acquires both domain and derivational knowledge.

Interacting with the student while the student is solving an exercise gives the tutor the opportunity to offer the student tailored feedback when mistakes

are made. Hume et al. [104] see hinting as prompting the student to recollect information or to make an inference needed to solve a problem, and as such they are seen as being part of a socratic approach to tutoring. Hints can for instance take the form of complete or partial steps in the solution, a concept which should be used in the next step or pointers for the student on how to approach the problem.

2.2.1 Dialogue frames in tutoring

In an examination of what features of one-on-one tutoring make it more effective than classroom instruction, Graesser et al. [85] analyse a corpus of dialogues between students and non-expert tutors. Their data is drawn from transcripts of two groups of learners: a group of 27 undergraduate students learning research methods, and a group of seventh grade pupils learning algebra. Tutoring took place face to face.

Graesser et al. highlight the features they found in the data which contribute to the learning effects of tutoring. They observe processes of collaborative problem solving, which has been identified by Roschelle [167] as an element of collaborative learning, as well as being a more general aspect of communication [51]. It is most salient when the tutor and student work on problems and answer questions, where the median number of turns in the answer to questions was five and ten for the two groups respectively. They identify a pervasive dialogue pattern in collaborative exchanges which they call the *dialogue frame*. It consists of the following five phases:

Phase 1 Tutor asks a question, which can be repeated or specified further if the student signals that the question was not understood.

Phase 2 The student offers an answer, which in the data was often found to be incomplete or only semi-coherent.

Phase 3 Tutor gives feedback on the answer. This feedback was classified as either positive, negative or neutral, and some feedback was non-verbal.

Phase 4 Tutor and student collaboratively improve the quality of the answer, and there are many strategies which the tutor can follow to achieve this. The tutor can for instance elaborate on the answer, give hints, pump the student for more information, or trace an explanation or justification.

Phase 5 The tutor assesses the student's understanding of the answer, for instance by explicitly asking whether the student understood, although the authors refer to the fact that the answers to such questions are not necessarily reliable.

1. Tutor: Now what is a factorial design?
2. Student: The design has two variables.
3. Tutor: Uh-huh.
4. Tutor: So there are two or more independent variables and one *PAUSE*
   Student: dependent variable.
5. Tutor: Do you see that?
   Student: Uh-huh.

Figure 2.2: Example of a dialogue frame, numbered with phases, from [85].

Phases four and five are do not occur in classroom instruction, therefore the benefit of tutoring must at least in part lie here. Graesser et al. identify two further features of tutoring which typically occur in phase four of the dialogue frame. The first is question answering [84], which is identified as leading to the exploration of deeper levels of comprehension. Tutors asked deep explanatory questions with a much higher frequency than in classroom interactions, and such questions were highly correlated with deeper levels of cognition. The second is student explanation, where a correlation was found between correct answers and achievement.

An example of a dialogue frame taken from their data is given in Figure 2.2. Here the tutor gives initial positive feedback on the student's answer, but follows it with a completion question, which the student answers correctly. The tutor finally assesses the student's understanding of the material with a straightforward comprehension-gauging question. Dialogue frames have been used in subsequent work to inform studies of explanation and questioning in tutoring, for instance by Chi [42] and VanLehn et al. [199].

2.2.2 Belief and knowledge states in tutorial dialogue

We saw in the previous section that belief is a crucial part of how agents act in dialogue, and the same is true of students and tutors in tutorial dialogues. It is characterised by a strong asymmetry of knowledge between the tutor and student [83, 146, 119]. They enter the dialogue with differing levels of expertise in the domain at hand, and knowledge should pass from the tutor to the student. Lee and Sherin [119] argue that this inequality makes it difficult for the tutor to interpret students' explanations correctly. It also influences their roles in the dialogue: the tutor carries a certain authority, and can for example ignore the student's requests. The student, on the other hand, has no such freedom, and is obliged for instance to answer questions when asked, and to justify claims which the tutor would be free to state without justification.

Since tutoring involves imparting knowledge to a student, tutors should ideally monitor their students' knowledge states. According to Good [83] the

tutor needs to know what knowledge the student does not have in order to offer additional knowledge of the right size and complexity. However tutoring is prone to misalignment of belief at the level of deep understanding of the domain; in other words, tutors and students do not always have the same view of their knowledge states. Chi et al. [46], by measuring how well tutors detect different kinds of misunderstandings, show that tutors tend to have difficulties in estimating the learner's deep understanding and that human tutors do not seem to maintain an accurate model of student's knowledge level during the tutoring process. Similarly, Putnam [157] finds that experienced tutors did not attempt to form detailed models of the students' knowledge before attempting remedial instruction.

Some studies do point to tutors' ability to infer what their students know. Nückles et al. [147] show that tutors who are better informed on the learners' prior knowledge can better adapt their feedback. An investigation of student questions and answers by Person et al. [151] finds that the quality of their answers allows the tutor to infer the student's understanding, but also finds that answers to comprehension-gauging questions are very misleading.

Common ground in tutoring is referred to in many studies related to the collaborative aspect of tutoring. Even though previous work indicates that tutors do not build exact models of students' knowledge, Graesser and Person [84] argue that the posing of questions in tutoring can nevertheless facilitate the maintenance of the belief states of the student and tutor. Their categorisation of action types includes question-generation mechanisms, one of which is monitoring the common ground between participants, otherwise known as grounding. An explicit model of common ground can feed into a module that monitors the student's performance [136, 140].

A number of researchers have considered the role of grounding and grounding acts in the pursuit of pedagogical goals in tutorial dialogue. Chi et al. [46] uses an annotation of grounding acts to recognise when tutors have detected knowledge deficits. The performance of an accepting grounding act indicates no deficit was diagnosed, whereas the performance of a repair, request for repair or request for acknowledgment indicates some deficit was found. Baker et al. [13] argue that learning from grounding is the basis of collaborative learning, and Pata et al. [150] show how student grounding acts serve to inform tutoring scaffolds.

In summary, this section has presented studies about what features of tutorial dialogue cause it to be so effective compared to classroom instruction. They give an indication of what kinds of behaviour ITSs should try to emulate if they want to reproduce the learning gains shown by students who have taken part in one-on-one tutoring.

2.3 Tutorial dialogue systems

Research in the area of implemented dialogue-based ITSs has been motivated by the promise of recreating the kind of interaction which has been shown to lead to learning effects in human-human tutoring. In this vein many aspects of the research reviewed in Section 2.2 above have been implemented in systems which combine pedagogical theories with natural language dialogue interfaces. Dialogue-based ITSs have a history going back at least to Carbonell's SCHOLAR system in 1970 [36]. Since then many different systems have been proposed covering a variety of domains and targeting different aspects of dialogue-based interaction. In this section we review some of the more recent developments of ITSs which have tackled some of the same problems as our research, namely those which deal with formal domains such as mathematics and which implement sophisticated dialogue models. We also present the goals of and research carried out in the DIALOG project in more detail, within which our research was carried out.

The designer of a tutorial dialogue system faces a number of tasks in addition to dialogue modelling. The input from the student must be processed by an interpretation module which can determine the student's intention and the task-relevant content, if any, of the utterance. Although allowing the student full expressive freedom would be in keeping with the assumption that interactivity leads to learning, such freedom makes utterance interpretation more difficult. When precise understanding is needed, tutorial systems either use menu- or template-based input, or use closed-questions to elicit short answers which exhibit little syntactic variation [81].

The complexity of the domain of instruction puts high demands on the domain reasoning abilities of an ITS. As we introduced in Section 2.1.3 above, a tutorial dialogue system accesses an expert system to analyse the domain content of students' inputs. This analysis, containing for instance a measure of the correctness of the input, is necessary to offer appropriate feedback [215]. The domain reasoner should also maintain a model of the task based on a general representation of domain concepts, so that contributions can be assessed in the context of the task solution built so far. The approach of tracing through preauthored solutions can often be sufficient for simple domains [96], but more complex domains require a deeper semantic analysis, such as that of Makatchev and VanLehn [134].

Related to domain reasoning is student modelling. The primary role of the student model is to encode the student's knowledge state, or mastery of concepts. As such the student model can take advantage of the same representation of the domain as the domain reasoner. Additionally the student model

may contain further measures such as errors the student has committed, or cognitively motivated properties such as affect or motivation.

An ITS should follow some teaching strategy to inform its feedback. This can include what hints are given when, how errors are dealt with or how the system reacts to the student's affective state [154]. Corbett et al. [58] highlight the question of when and in what form advice should be presented to the student. A system should make this choice based on the student and task models, thereby adapting its feedback to the student's current performance [78]. For instance hints containing relevant concepts should be chosen so that the concepts help the student solve the current problem and are not too far away from the student's current knowledge state.

2.3.1 Teaching mathematics

Developers of ITSs which teach mathematics, especially proving, face further challenges, primarily in the area of domain reasoning. The domain reasoner must work with a formalisation of the mathematical theory which is being tutored, and mathematics formalisation is a non-trivial task which itself has developed into a fully-fledged field of research (see for example Kamareddine et al. [110]). Complex tasks such as theorem proving required deep sophisticated analysis in order to provide the correct feedback to the student. One of the characteristics of mathematical proofs is that there is not one unique solution, in other words there is typically a selection of correct solutions to a given problem which the student might propose. This means that approaches based on pre-authored answers are not suitable [67]. A step must be analysed as part of the previously constructed proof, therefore its correctness is only derivable in context.

A number of systems for teaching mathematics have been built upon pre-existing, general purpose provers, including for instance the CMU proof tutor [171], Proofweb [109] and WinKE [61] for propositional and first order logic, or ETPS [8] for higher-order logic. These systems focus on pure logic and proof construction. Proof construction works essentially as follows: The student selects a formula to be modified and can then try to apply a deduction rule, mostly without having to state the result of the application of the rule. A drawback of these types of system is that students must first learn the commands and input syntax to be able to interact with the system. Furthermore, students are able to construct a proof by randomly applying rules, without getting a deeper understanding of the meaning of these rules. In this regard it has also been shown that logic alone does not help humans do more abstract proofs [71]. Consequently some tutorial systems support teaching mathemat-

ics at a more abstract level, comparable to the type of mathematics taught in schools. These include the EPGY Theorem Proving Environment [181], the Geometry Tutors [113] and Tutch [1].

There is also evidence that teaching proving is amenable to the same effects of natural language tutoring as other domains. Hersh [98] sees proofs in the classroom as complete explanations for students. By concentrating on the detail in proofs these explanations become more informative for the students, which leads to better understanding. Yackel [213] argues that explanations and justifications serve communicative functions, which suggests they will fit well into dialogue-based tutoring.

2.3.2 Implemented ITSs

In this section we review some of the important systems which implement dialogue interfaces for tutoring. They are relevant to our work either because they teach mathematics or because they use advanced dialogue management techniques.

The same group of researchers who proposed the dialogue frames model, presented here in Section 2.2.1, have implemented this model in **Autotutor** [87, 89]. It teaches Newtonian physics and computer literacy through a multimodal interface using text, diagrams and an animated talking head. A problem is presented to the student, who offers an answer of about one paragraph in length. The answer is then discussed with the system in a mixed initiative dialogue. The student's answer is analysed using latent semantic analysis, a probabilistic method which measures the similarity of the answer to a correct answer.

Autotutor follows the explanation-based constructivist theories which we introduced in Section 2.2. Its set of possible feedback actions includes asking a question, for instance to repair an error or to request a clarification, giving a hint or an example, answering a student question, affective feedback, for instance commenting on the student's ability, as well as positive, neutral and negative feedback. Its dialogue model which chooses these actions is an implementation of the dialogue frame. It is a finite state model in which actions are chosen based on conditions on the edges, for example the result of the analysis of the answer or the student's knowledge goal state. A drawback of this approach is that it can not engage in multi-turn dialogues to handle one particular remediation. Autotutor has been successfully evaluated, showing learning gains for deep levels of comprehension.

One of the more advanced dialogue models among all ITSs is used by the **Beetle** system [218, 217]. It teaches conceptual knowledge in the area of basic

electricity and electronics using a combination of multiple choice questions and exercises in a simulated electronics laboratory. The development of the Beetle system has focused on models for dialogue management. Its dialogue manager is implemented using the information state update approach, and is based on the EDIS system [135]. The dialogue manager connects an interpretation module, which processes input text and user interface actions, an update module, which maintains the dialogue state and encodes conversational expertise, and a response generation module, which uses a three tier architecture [59, 216]. In the first tier a deliberative planner synthesises abstract plans at the start of the session and after planning failures. The second tier is a sequencer which refines the abstract plans, and finally on the third tier a controller generates concrete actions which realise the plan. Tutorial actions such as feedback are chosen at the second tier, and the choice of action depends on the current dialogue state, including for instance the content of the student's latest utterance and any active obligations. The dialogue state also includes the common ground, however this is only updated, and is not used in the process of choosing actions.

BeeDiff [32] shares a number of the components of Beetle [33], but teaches mathematical differentiation and basic algebra through natural language. It uses a text interface with formula entry so that formulas can be embedded in utterances. The dialogue with the system is mixed initiative. From a corpus analysis a task model and a set of feedback types were derived [32]. Beetle and BeeDiff use the same dialogue manager, parsing and reference resolution modules, but have separate interpretation and generation subsystems. The tutorial strategy implemented in BeeDiff targets local pedagogical aims. It accepts new tasks from the student, gives hints when requested, and gives diagnoses in response to new terms offered as solutions.

Whereas pedagogical actions in Beetle are encoded directly in the planning operators, BeeDiff is more adaptive due to its flexible domain reasoner [215]. The domain reasoner is responsible for the diagnosis of students' terms representing steps in the derivation and for the content of hints. It generates an extended representation of terms and differentiation rules from which a solution model can be built for a new task. Terms are matched against the model and hints are generated from possible next steps in the model.

PACO [163] is a tutorial dialogue system based on a model of collaborative discourse, motivated by the fact that tutorial dialogue is a collaborative interaction. It teaches procedural tasks in simulated environments, such as repairing a gas turbine. It is developed using the COLLAGEN framework [162], a partial implementation of the tripartite model of discourse [92, 91]. COLLAGEN agents use the discourse state to generate a set of utterances and domain actions

and then choose one of these to perform. PACO encodes procedural knowledge (domain dependent recipes) in COLLAGEN's declarative language. The procedural knowledge is then used to construct plan trees, which form the intentional structure of the discourse. Plan trees are a partial implementation of SharedPlans. This plan representation can be used to determine next steps in the procedure being taught. A discourse representation algorithm updates the discourse state after each utterance using near-miss plan recognition. Tutorial behaviour is encoded in the discourse acts that PACO can perform. The choice is based on a ranking of acts as well as their conditions on the discourse state. For example giving the initiative back to the student , which PACO realises by saying "You take it from here", is ranked higher than teaching the step at hand.

Although tutorial dialogue does indeed seem to be a collaborative undertaking, Dzikovska et al. [69] argue that collaborative problem solving models do not account for 28% of the student utterances that occur in tutorial dialogue. Their study is based on the corpus collected by the LeActiveMath project, which consists of human-human dialogues on procedural differentiation problems. While CPS models are able to account for planning future actions, which can consist of proposing actions, evaluating options, agreement, they do not address actions that have been performed in the past, which are found to be evaluation, explanation and knowledge statement.

Evens et al. [72] have developed the **CIRCSIM** system, which teaches medical students about the human circulatory system and focuses on tutor behaviour and the generation of hints. The system asks the student to predict the effects of changes to certain physiological parameters. The domain is formalised as a concept map linking dependent variables which measure physiological states. After the student offers an answer the system enters into a dialogue to remedy any errors that the student has made, for instance where the student has forgotten to mention a dependent variable.

The dialogue model in CIRCSIM is a tree structure which approximates the intentional structure of Grosz and Sidner [92], and is constructed using a tutorial planner [76]. Each correction mechanism, which is a realisation of a remediation tactic, is expressed as a dialogue schema. Dialogue schemata are derived from the analysis of transcripts of human tutors and are made up of sections, where each section is a plan operator. The output of the planner is a set of discourse goals from which the system's utterances are generated. The discourse goals include hints, elicitations and prompts. The plan can be updated during the dialogue in the case of problems, such as when the student fails to produce the correct answer to a question.

The first PACT Geometry tutor, which is one of the Cognitive Tutors [7],

has already been shown to achieve learning gains by requiring students to explain each element of their answer [2]. Aleven et al. [3] present the **Geometry Explanation Tutor**, which adds natural language dialogue capability to the PACT Geometry tutor and also targets self-explanation. In the interaction with the system students are required to give fully accurate and complete explanations of their answers. If an initial explanation is incomplete, the system will give feedback guiding the student towards a complete explanation. The feedback is based on a categorisation of the explanation at that point, for instance in terms of completeness, what rules were referenced and whether they were used wrongly.

In an evaluation the learning gain of students who gave natural language explanations was not found to be significantly better than those who gave menu based explanations, although the authors note that the subjects already had high pretest scores, and therefore may not represent typical students [4]. All explanations in the Geometry Explanation Tutor must be given in full regardless of content, and the system does not allow any flexibility to accept partial explanations, for example from knowledgeable students.

Longer explanations of answers in the domain of qualitative physics are analysed by the **Why-Atlas** system [198, 108]. Statistical methods such as latent semantic analysis are sufficient for short answers such as those handled by Autotutor above, but here students offer multisentential essay-like answers which contain chains of reasoning. These longer answers require more exact analysis in order to support appropriate feedback. They are first transformed into a logical form before being compared with pre-authored standard answers [134], however authoring the answers for a new domain is highly time-intensive. The dialogue model in Why-Atlas is a finite-state machine with reactive planning, in which complex topics are handled with subnetworks. The system has been evaluated against a reading only condition, but learning gains were found only for fill-in-the-blanks post test questions.

2.3.3 The DIALOG project

The research reported in this thesis has been carried out as part of the DIALOG project[1] [14, 15, 16, 20]. The goal of the project was to investigate the issues and challenges involved in natural language tutoring of mathematical proofs, and to work towards a conversational tutoring system helping students to construct proofs of mathematical theorems. The envisaged scenario was that students would construct mathematical proofs in communication with a text-

[1]The DIALOG project was a collaboration between the Computer Science and Computational Linguistics departments of the Saarland University within the Collaborative Research Centre on Resource-Adaptive Cognitive Processes, SFB 378 (`http://www.coli.uni-sb.de/sfb378`).

based dialogue system. Proof steps are embedded in the student's utterances, and may be verbalised. The dialogue system in turn interfaces with an automated theorem prover. The role of the automated theorem prover is to evaluate and verify the proof steps which the students propose. During the interaction the student is given feedback after each utterance, for instance concerning the correctness of the proof step, or offering hints and remediation.

Areas of investigation

The project investigated a number of specific areas relevant to natural language tutoring. Because little data was available on the characteristics of the language used in tutorial dialogue on mathematical proofs, two Wizard-of-Oz experiments [62] were conducted to gather dialogue data. In each experiment a group of students interacted over a chat interface with a human tutor, who guided them in constructing proofs of a series of theorems. In the first experiment the proofs were in the area of naive set theory, in the second experiment in binary relations. The result was two corpora of computer-mediated tutorial dialogues. We will reported on the data collection experiments and the corpora in more detail in the next chapter. The analysis of the data identified issues in natural language understanding, tutoring strategy, mathematical reasoning and dialogue management.

The analysis of the language used by students in the corpus has been investigated by Wolska and Kruijff-Korbayová [209], Horacek and Wolska [101], and Wolska et al. [212]. Such language is characterised by a range of interleaving of mathematical formulas and natural language. Utterances may consist completely of formulas, completely of natural language expressions, or may mix the two, as shown in (2.2). The expressions are often imprecise and informal, for instance in (2.3), in which the word "contain" can conceivably mean either the subset or element relation. Referring expressions are used to refer to mathematical objects and to previously uttered formulas and their constituent parts [210]. Many different types of errors occur, including syntactic errors in formulas, type errors between operators and objects, or semantic errors [102].

(2.2) B contains no $x \in A$

(2.3) ...then all A and B must be contained in C

The task of the input analysis module is two-fold. It should construct a representation of the utterance's linguistic meaning and it should identify and separate the parts of the utterance which constitute meta-communication with the tutor (for instance "I don't understand what the task is"), which are not to be processed by the mathematical reasoner, from parts which convey domain knowledge, that should be verified by the reasoner. The result of the analysis

is a representation of the linguistic meaning of the utterance, a set of dialogue moves that the utterance realises, and in case of solution steps, a formal representation of the content of the step [209], which can be further processed by the mathematical reasoner.

To support the generation of appropriate feedback each proposed proof step needs to be analysed by the system in the context of the partial proof developed so far. Issues involved in processing such mathematical content have been identified by Benzmüller and Vo [16]. They highlight the declarative style used by students to present proof steps, that is, very often the students only 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 without language capabilities where proof steps are invoked by calling rules. Proof steps can be underspecified, in other words part of their content can be left unstated, and there is much ambiguity in the expressions used to present the steps.

The analysis of the proof step takes place in the context of the proof that has been constructed by the student so far. The representation of the proof is maintained by a mathematical theorem prover. The theorem prover used in the DIALOG project is the Omega system [177], which was chosen because it reasons at a level close to that at which humans reason [103]. Proof steps are verified by comparing the formula which the step derives to all possible continuations of the current proof object [12, 67]. If the formula matches one of these continuations, then the step was correct and the proof representation may be extended accordingly. If there is no such continuation the step is considered incorrect. The proof object contains the rules and premises which were used to derive the step, and these can be returned to the dialogue system in order to form part of the feedback offered to the student, or to inform a student model.

Benzmüller et al. [19] and Schiller et al. [173] have investigated further properties of proof steps including granularity. Granularity evaluation requires analysing the complexity or size of a proof step rather than just its correctness. An approximate measure of step size could be for instance the number of inference rule applications necessary in a given calculus to derive the step. Schiller and Benzmüller [172] have cast granularity analysis as a classification problem, where the granularity of a proof step ranges from too coarse-grained to too fine-grained. Overall the proof manager offers an analysis of the proof step in the context of the proof so far which includes the correctness of the step, the rules and premises used to derive it, and a categorisation of its granularity status.

The tutorial manager embodies a pedagogical theory. The first Wizard-of-

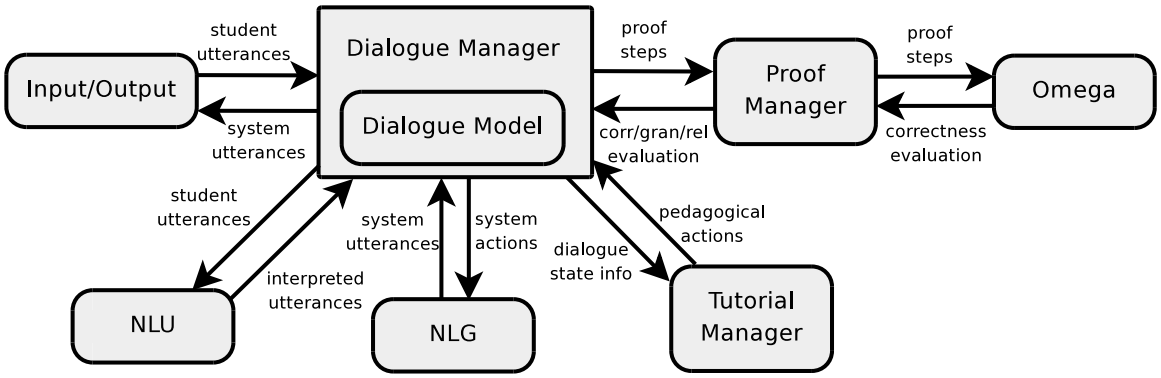

Figure 2.3: The envisaged architecture of the DIALOG system

Oz experiment compared the effectiveness of a didactic and a socratic teaching strategy with respect to a minimum feedback baseline. Work within the project [196, 75, 74] has followed this to propose a pedagogical module which relies internally on an ontology of mathematical concepts and a taxonomy of hints, such as the elicitation or giving away things like concepts or inference rules. Based on a simple student model, previously generated hints, and the categorisation of the current step, a hinting algorithm may generate a hint to be presented to the student.

The flexible analysis of students' statements requires a tight interleaving of natural language and mathematical analysis. This is facilitated by the dialogue manager, which enables intercommunication between other system modules. Initial work in the project focused on models for dialogue management [26], using an agent-based blackboard architecture to allow modules to share information. In [28] we have proposed a model of how pragmatic information, proof state information and tutoring state information can be combined into a single representation of dialogue state within the dialogue manager.

The different aspects of processing are brought together in the system architecture illustrated in Figure 2.3. The modules which embody the research directions outline in this section share information through the dialogue manager. For instance the proof content of the interpreted utterances from the NLU module is passed to the proof manager for analysis. The tutorial manager receives much information about the dialogue state, including the actions performed by the student and the analysis of the steps by the proof manager. The dialogue manager embeds a dialogue model whose state representation stores this information centrally. This design has been used for the implementation of a demonstration system [27], which illustrated the interaction of the modules for a small number of example dialogues. We will see in the coming chapters how the work reported in this thesis fits into this proposed architec-

ture, and how it takes advantage of the information supplied by each of the modules, primarily the interpretation of the utterance by the NLU module and the analysis of its proof step by the proof manager.

2.4 Summary

In this chapter we have seen how results from dialogue modelling and pedagogical science contribute to the provision of effective ITSs with natural language interfaces. We presented theories which see dialogue as an interaction between conversational agents who are motivated by their beliefs, and saw how the process of grounding allows them to align their beliefs with their dialogue partners. We summarised a number of studies on what mechanisms lead to the effectiveness of one-on-one tutoring, among them self-explanation. We also reported evidence for the use of belief about student knowledge states by tutors and grounding in tutorial dialogue. We presented a theory of dialogue structure in which the collaborative aspects of tutoring occur, namely dialogue frames.

We then reviewed a number of ITSs which implement some of these theories. These use a wide breadth of techniques in dialogue modelling, from quite rigid, such as Autotutor, to more flexible, such as Beetle and BeeDiff. PACO implements a sophisticated dialogue model, but has been found to not cover the full range of actions in tutorial dialogue. Few systems target self-explanation, and those that do, such as the Geometry Explanation Tutor, follow strict dialogue patterns. Task modelling, a particularly important issue in ITSs for mathematics, is handled by purpose-built systems, whereas except for the DIALOG project, no dialogue based ITSs use general purpose mathematical reasoners.

In the rest of this thesis we will build on the results presented in this chapter. Our proposal will combine aspects of grounding and dialogue frames in a model for solution step discussions, in which the tutor may decide to pose explanatory questions concerning the student's answers.

3
Dialogue phenomena

Our investigation into the phenomena found in tutorial dialogue on mathematics and our development of a model to account for them will both be based on an analysis of a corpus of human-human tutorial dialogue. The corpus was collected in the context of the DIALOG project using the Wizard-of-Oz paradigm and involves tutors guiding students through proofs on binary relations over a text-based chat interface. This chapter is a qualitative analysis of the data in that corpus.

Our hypothesis for the analysis of the corpus is that it will exhibit the same local structures as those that have been highlighted by Graesser et al. [85] and presented in Section 2.2.1, namely dialogue frames. In order to investigate this we must consider the task that the student and the tutor are solving, which is the construction of a solution to a mathematical exercise, and the actions that they perform in doing this. For instance the student makes contributions to the solution under construction, and the tutor gives the student feedback on these contributions. We will show that each of these actions has a role to play in the completion of a dialogue frame.

We begin this chapter with a presentation in Section 3.1 of the corpus collection experiment and an outline of the data which was collected. Section 3.2 gives a characterisation of the data by example. We first look at its mathematical aspects and the proofs that the students construct and then consider the range of actions performed by the students and tutors, respectively. We also present the specific phenomena of informationally redundant utterances. Section 3.3 shows excerpts from the corpus annotated with dialogue frames in order to illustrate their occurrence and the kinds of actions that make up

the different parts of the frame. Finally in Section 3.4 we review other corpus analyses in tutorial dialogue and argue that our categorisation of the range of possible actions is comparable, before summarising the chapter.

3.1 Corpus collection

Our work is based on the second of two corpora of human-human dialogues collected by the DIALOG project [14]. The goal of the first corpus collection experiment [211] was to gain data to support an initial investigation into the tasks which arise in building a natural language tutorial system for mathematics. These included the analysis of mixed mathematical and natural language [101] and the analysis of students' proofs as sequences of proof steps [16]. The second experiment was carried out in order to collect more data on mixed-language interaction in a more complex domain of mathematics [18]. A further goal was to investigate the effect of variation in the presentation of the learning material from terse mathematical to verbose linguistic style. This section presents the second experiment and the resulting corpus.

3.1.1 Experimental setup

The experiment was carried out using the Wizard-of-Oz paradigm [62]. The user, here the student, is led to believe that the system with which he or she is interacting is fully automated, however in fact it is just a front-end, and the role of the system is being played by a human expert at a different location. In the case of this experiment human tutors provided the system's utterances. The 37 participants were university students who had previously taken at least one lecture course in mathematics at university level. They were informed that they were interacting with a conversational tutorial system with natural language capabilities. Four expert tutors alternated in the role of wizard. The tutors' background with respect to teaching mathematical proofs was the following: One was a senior lecturer with several years of experience in lecturing a course Foundations of Mathematics, the second was a trained mathematics teacher with a few years of teaching experience, the third was a recent graduate with a degree in teaching mathematics, and the fourth was a doctoral student in Institute of Theoretical Mathematics at Saarland University with several years of experience as a teaching assistant in various mathematics courses. All those involved were paid for their participation.

The procedure of the experiment was as follows. First the students were given printed material introducing the concepts and definitions of logic, set theory and binary relations, and were allowed time to study this. One group

W. $(R \circ S)^{-1} = S^{-1} \circ R^{-1}$
A. $(R \cup S) \circ T = (R \circ T) \cup (S \circ T)$
B. $(R \cup S) \circ T = (T^{-1} \circ S^{-1})^{-1} \cup (T^{-1} \circ R^{-1})^{-1}$
C. $(R \cup S) \circ S = (S \circ (S \cup S)^{-1})^{-1}$
E. Assume R is asymmetric. If R is not empty (i.e. $R \neq \emptyset$), then $R \neq R^{-1}$

Figure 3.1: The theorems to be proved in the experimental session

was given a formal presentation of the material, the other group was given a presentation using natural language descriptions. The study material is described in Section 3.1.2 below. The students then began their interaction with the dialogue system. The system appeared to the students as a chat-like interface [17], in which they could see the previous utterances in the dialogue. All communication with the tutor was typed, and the interface allowed the use of buttons and Latex commands to input mathematical symbols such as $\cup$. The students could copy and paste content from previous utterances. The language of the interaction was German and each experimental session took 115 minutes.

The students were asked to prove up to four theorems, which are listed in Figure 3.1. In all theorems R and S stood for binary relations on a carried set M. Theorem E was given to students who had difficulty with the other tasks. Each theorem was presented to the student after the previous one had been completed. Each theorem that the student proved could be used as an identity in the subsequent proofs. The students were instructed to enter individual proof steps rather than complete proofs and to think aloud, and both students and tutors were free to express themselves in any way they liked. The tutors were given instructions on what constitutes Socratic teaching, but were not restricted to any teaching strategy.

The tutors were asked to annotate the mathematical content of the students' steps, if present. There were three dimensions: correctness, which referred to the logical correctness of the step in the range "correct", "incorrect", and "partially correct", granularity, which is a subjective measure of step size ranging from "too fine-grained" to "acceptable" to "too coarse-grained", and relevance to the current proof, in the range "relevant", "partially relevant" and "irrelevant". For each dimension a "not applicable" choice was allowed. The students did not see these annotations.

3.1.2 Study material

The study material, which was taken from Bronstein and Semendjajew [25], contains the mathematical knowledge which students were to use in solving the exercises. It was presented to students in two forms: A formal version

which used mathematical notation in its concepts and definitions was given to 20 of the participants, and a verbose version which used natural language explanations to the remaining 17. Their content was otherwise the same. An example of the difference in expression is shown in this explanation of the subset relation between sets:

Formal:
If A and B are sets and

$$\forall x(x \in A \Rightarrow x \in B)$$

holds, then A is called a subset of B. We write $A \subseteq B$.

Verbose:
If A and B are sets and it is the case that every element of A is also an element of B, then A is called a subset of B. We write $A \subseteq B$.

The study material begins with a refresher on propositional and predicate logic, including the definitions of the logical connectives and quantifiers and identities such as the DeMorgan laws, modus ponens and the distributivity properties of $\vee$ and $\wedge$. The basic concepts of set theory are presented, which define set element, extensionality and subset. This is followed by the set operators such as union and intersection as well as cross product. Finally the definitions for binary relations are given: for a relation R on a set M, it defines $R \circ S$, the relation product or composition of relations, R^{-1}, the inverse of a relation, and a set of further properties of relations such as reflexivity and symmetry. It is these definitions concerning binary relations that the students are expected to apply when solving the exercises.

In the rest of this thesis we will make use of the list of definitions taken from the study material, therefore we now give the full list in Table 3.1. We have grouped the definitions concerning propositional logic in "prop", and those concerning predicate logic including quantifiers in "quant". The set operator properties in "set_prop" include for instance distributivity, associativity or commutativity of set operators in relation to one another. We have included in this list the first three of the four theorems that the students are asked to solve. This is because these theorems are available to students as identities after they have been solved, and can be applied in proof steps as the other definitions can be.

3.1.3 The resulting corpus

In addition to the dialogue transcript the experimental setup also gathered an audio recording of the students thinking aloud, a video of the subject using the software, a screen recording, and the wizard's audio commentary. We will only concern ourselves with the written dialogue data. For each turn

Group	Name	Content
logic	prop	propositional logic
	quant	predicate logic
set theory	subset	definition of subset
	psubset	definition of proper subset
	exten	set extensionality
	set_prop	set operator properties
	union	definition of union
	intersection	definition of intersection
binary relations	composition	composition of relations
	inverse	relation inverse
identities	identity1	$(R \circ S)^{-1} = S^{-1} \circ R^{-1}$
	identity2	$(R \cup S) \circ T = (R \circ T) \cup (S \circ T)$
	identity3	$(R \cup S) \circ T = (T^{-1} \circ S^{-1})^{-1} \cup (T^{-1} \circ R^{-1})^{-1}$

Table 3.1: The set of mathematical definitions in the study material

	No. utterances									
	1	2	3	4	5	6	7	8	9	10
Student turns	822	83	16	2	1	0	0	2	0	0
Tutor turns	550	445	69	14	1	0	0	1	3	2
Total turns	1372	528	85	16	2	0	0	3	3	2

Table 3.2: Turns by number of utterances for student and tutor

the transcript contains the timestamp of the turn, the text of the utterance and, for the tutor's turns, their annotations of the correctness, granularity and relevance of the previously entered step.

The corpus contains 37 transcripts with a total of 2011 turns. Student turns accounted for 926 of these, tutor turns 1085. The number of utterances per turn is shown in Table 3.2. The table shows that the large majority of turns, 1372 in total, contained only one utterance, and only 5.6% of turns contained more than two utterances. The longer turns of 5 utterances or more were nearly all a result of either recapitulations of the complete proof by the tutor, or of students entering complete proofs in one turn, which they were explicitly asked not to do. There were an average of 54.3 turns per session. Most students attempted at least two exercises, as shown in Table 3.3, and 15 students were able to attempt all four.

	No. exercises			
	1	2	3	4
No. students	4	11	7	15
Cumulative	37	33	22	15

Table 3.3: Number of exercises attempted by how many students

3.2 Characterisation of the data

In this section we describe by example the kinds of dialogues in our corpus and the different actions that we observe the students and tutors performing. A typical example of a full exercise is shown in (3.4)[1]. The tutor sets the exercise and then the student contributes a sequence of formulas. The formulas are the conclusions of proof steps, which are the building blocks of proofs. A proof step consists of a newly-derived formula along with the rules used to derive it and the premises it was derived from. The rules and premise need not be explicitly stated, in this case they are underspecified. A sequence of proof steps from axioms to the theorem constitutes a complete proof of that theorem. In the first step for instance, S8, the identity from exercise B is applied to the left-hand side of the theorem in order to derive the formula $(R \cup S) \circ T = (R \circ T) \cup (S \circ T)$. Here both the rule and premise used remain underspecified.

The tutor gives feedback on each step that the student contributes. In this example the feedback is very minimal—the tutor simply accepts or rejects each of the steps in turn. When the tutor sees that the proof is finished he indicates this to the student, in utterance T13. Having finished the exercise the student is presented with the next one.

(3.4)
T8: Please prove $(R \cup S) \circ T = (T^{-1} \circ S^{-1})^{-1} \cup (T^{-1} \circ R^{-1})^{-1}$
S8: $(R \cup S) \circ T = (R \circ T) \cup (S \circ T)$
T9: That's correct!
S9: $(R \circ T) \cup (S \circ T) = (T^{-1} \circ R^{-1})^{-1} \cup (T^{-1} \circ S^{-1})^{-1}$
T10: How did you get this? Use smaller steps!
S10: $(R \circ T) \cup (S \circ T) = ((R \circ T)^{-1})^{-1} \cup ((S \circ T)^{-1})^{-1}$
T11: That's right!
S11: $((R \circ T)^{-1})^{-1} \cup ((S \circ T)^{-1})^{-1} = (T^{-1} \circ R^{-1})^{-1} \cup (T^{-1} \circ S^{-1})^{-1}$
T12: That's also right!
S12: $(T^{-1} \circ R^{-1})^{-1} \cup (T^{-1} \circ S^{-1})^{-1} = (T^{-1} \circ S^{-1})^{-1} \cup (T^{-1} \circ R^{-1})^{-1}$
T13: This exercise is also correct!

The tutor can also give more detailed and helpful feedback on the wrong steps that the student is contributing than in the example above. In (3.5) the

[1] Although our corpus is in German, for simplicity of presentation we have translated the examples for this chapter into English. We will include German to English glosses in any cases where this is necessary or helpful. Sx and Tx label the xth student and tutor turn respectively, and Sx-y is used for the yth utterance within the xth turn.

tutor uses hints to guide the student to the right next proof step by highlighting the errors in each of the student's contributions. In utterance T8 the tutor refers to a syntactic error in the formula and in T10 to a wrong instantiation of the definition of composition.

(3.5) **S8:** Then $(R \cup S) \circ T$ is $\{(x,y) \mid \exists z(z \in M \wedge ((x,y) \in R \vee (x,y) \in S) \wedge (y,z) \in T)$.
T8: not correct. The error is presumably in the last "and"-expression.
...
S10: Then $(R \cup S) \circ T$ is $\{(x,y) \mid \exists z(z \in M \wedge ((x,y) \in R \cup S) \wedge (y,z) \in T)\}$
T10: not correct. Your variable names are not consistent.

A more complex example, in (3.6), shows that the feedback that tutors give to students is not restricted to the just the subsequent utterance. Here the tutor and the student enter into a discussion addressing the deficiencies of the latest step. The tutor asks the student to contribute more detail in order to show how the step was concluded. This example, and this kind of action by the tutor, will be taken up again later in this chapter, in Section 3.3.

(3.6) **S8:** $\exists x$, so that $(a,x) \in (R \cup S)$ and $(x,b) \in T$
T8: What does this follow from?
S9: That follows from the definition of relation product for binary relations: $(a,x) \in R$ and $(x,b) \in S$, then $R \circ S$
T9: More precisely: it follows from $(a,b) \in (R \cup S) \circ T$

These examples are typical for the corpus, and show one of the important characteristics of the data: student initiative. After having presented the exercise that the student should solve, the tutor leaves the student to lead the solution construction process. This agrees with Merrill et al. [139], who argue that expert tutors should leave student in control and free to reason through problems themselves in order to achieve learning. Only when the tutor must ask questions or give the student directions, as in T8 in (3.6) above, does the initiative change hands.

3.2.1 Mathematical aspects

One characteristic of the mathematical content of the corpus which is illustrated by the examples above is the terseness of the students' utterances. In their analysis of the corpus, Benzmüller et al. [18] find that the 40% of the utterances in the formal study material group and 15% in the vebose group contain only symbolic content. They also find that the average number of natural language tokens per dialogue session is 36.3 (with a standard deviation of 19.43) at an average of 51 turns. Considering that these averages include the tutor's turns, whose expressions are much more verbose, this indicates that the students mostly use mathematical expressions. In the utterances which do include both mathematical expressions and natural language tokens there

is often a complex overlap between the two. Benzmüller et al. [15], investigating a similar corpus of tutorial dialogues on mathematical proofs, find recurring structural phenomena whereby mathematical and natural language is interleaved, such as in (3.7) and (3.8). Wolska and Kruijff-Korbayová [209] have proposed models for parsing these types of utterances.

(3.7) **S4:** B contains no $x \in A$

(3.8) **S9:** If $(y,z) \in R$ and $(z,x) \in S$, then $(z,y) \in R^{-1}$ and $(x,z) \in S^{-1}$

Benzmüller and Vo [16] find that students in such dialogues use a declarative style to express proof steps, and usually only present the result of the proof step, in other words the formula which has been derived. Possible other elements of the proof step, such as the premises, axioms or rules used to derive the step are very often left underspecified.

The structure of the proofs that the students construct is usually quite linear, due to the simple nature of the theorems that the students are asked to prove. A rewriting style is often used, in which in order to prove $A = B$, the steps in the proof are rewrites of the left hand side of the form $A = A'$, followed by $A' = A''$ and eventually $A''' = B$. This is illustrated by the proof style used in (3.4). Each equation follows from the previously stated equation. Another approach, which is occasionally used explicitly, is to use the definition of set extensionality to split an equality between two sets into subproofs, namely the subset and superset relations between the sets, and is introduced for instance with the proof step in (3.9). Other standard proof structures such as proof by induction or contradiction are not required by the theorems the students were asked to prove. Overall the linear structure used by the students means that each proof step usually follows from the previously contributed proof step. Most often the relationship is that the formula derived by the previous proof step forms one of the premises for the new proof step.

(3.9) **T0** Please prove $(R \circ S)^{-1} = S^{-1} \circ R^{-1}$
S0: Proof using $\subseteq$ and $\supseteq$
T1: Right, you can do it like that.

3.2.2 Categorisation of students' actions

In this section we give a categorisation by example of the kinds of actions that students perform in the corpus. The most common action that the student performs is the contribution of mathematical content in order to advance the current solution, an example of which is the utterance in (3.10). The student has given the premise and the formula which has been derived from it.

(3.10) **S4:** If $(a,b) \in (R \circ S)^{-1}$, then it holds that $(b,a) \in (R \circ S)$

The contribution of a new solution step is a claim that the formula contained in the step is true, in other words derivable from the current state of the solution which has been constructed so far. The step has some relation to the material under discussion, that is, it follows logically or by some chain of proof argument. There are a number of ways this content is contributed to the solution. The simplest way is when the student merely states the formula that the solution step derives, as in all of the student's contributions in (3.4) above. The student leaves the further elements of the step and its intended relation to the solution so far implicit. However even though the intention of the utterance with respect to its content is not stated, in this corpus the tutors assume that the bare statement of a formula is in fact a solution step because of the proving task that the student has been set.

A more verbose way to contribute to the current solution occurs when students use linguistic markers to indicate the intended effect. In (3.11) the words "It follows that" show that the following content is being proposed as a formula which is true. Other cues which are found in the corpus can be more specific, including "hence", "it holds" and "from this". These indicate that the formula has been derived from some previously stated formula in the dialogue. Some linguistic cues are necessary for the correct interpretation of the step, for instance assumptions are always marked, as in (3.12).

(3.11) **S4:** It follows: $(x,y) \in S^{-1} \circ R^{-1}$

(3.12) **S0:** Let $(x,y) \in (R \circ S)^{-1}$.

As we outlined in the previous section, solution steps have a number of elements, including the derived formula and the rules used to derive it. In addition to the derived formula the students may additionally state the rule which was used, such as the definition of relation composition in (3.13), or the premise, such as the $(a,b) \in (R \circ S)^{-1}$ in (3.10) above. A combination of both is also possible. The cues that are used to identify rules include "by the definition of" or "according to exercise W". Premises are identified for instance with "it follows from", or "since X holds, we conclude".

(3.13) **S4:** Then (z,b) is in R^{-1} and (a,z) in S^{-1}
T4: Correct.
S5: According to the definition of $\circ$ it follows that (a,b) is in $S^{-1} \circ R^{-1}$

Closely related to the contribution of new solution steps is the augmentation of existing, previously contributed solution steps with additional content. This normally happens after the tutor has asked for the student to supply this additional content, in which case the link to the previous step which is being extended is clear. The elements of the step which are added, most often premises and derivation rules, are introduced with the same linguistic cues as are used when these elements are included directly in the initial contribution

of the step. An example is shown in (3.14). The student has performed a large proof step in deriving the formula in S8, and the tutor asks for a justification for the step. The student supplies the main rule used, which augments the step.

(3.14)
S7: But this means: $(z,y) \in R^{-1}$ and $(x,z) \in S^{-1}$
T7: Now it is correct
S8: Therefore it follows: $(x,y) \in S^{-1} \circ R^{-1}$, which was to prove.
T8: Correct. Give a (simple) justification for the last step of the proof.
S9: This follows directly from the definition of relation product.
T9: Correct. You have solved the exercise.

Not all utterances which contain mathematical content are intended to be contributions to the solution. Students also state mathematical knowledge for other reasons, although these are less common than solution contributions. One reason is to refer to a part of a step which is about to be performed, as in (3.15), in which the student repeats the definition of relation product. Such an action makes the knowledge salient in the dialogue and tells the tutor what concepts or goals or approaches will be used in the coming step. The student can state a solution strategy they are about to follow, such as a proof by cases, or the use of set extensionality, as in (3.9) above. They can also restate a previously proved identity, either just by stating the formula or saying that it has been previously proved (3.16).

(3.15)
S0: The relation product of two binary relations R, S in a set M is defined as $R \circ S := \{(x,y) \mid \exists z(z \in M \wedge (x,z) \in R \wedge (z,y) \in S)\}$
T0: That's right.

(3.16)
S16: According to exercise A $(R \cup S) \circ T = (R \circ T) \cup (S \circ T)$
T16: Yes, you have proved this already.
S17: Since $(R \circ T)^{-1} = T^{-1} \circ R^{-1}$, it also holds that $R \circ T = (T^{-1} \circ R^{-1})^{-1}$
T17: You have drawn a correct conclusion.

Students ask various questions seeking information about the task at hand. Since students typically have the initiative, they can ask these at nearly any point in the dialogue. They ask for very general hints when they are stuck or do not know what the right next step in the proof should be, using open questions like "what should I do now/next?", or "please give me a hint". This often occurs after the student has had several contributions in a row rejected by the tutor. Such requests for help are made more concrete when the student asks for a specific kind of help. Students for instance ask tutors to reveal the next step in the proof, as shown in (3.17), or ask for the tutor to explain certain concepts or to explain how they are used, as in (3.18). They also ask the tutor to evaluate steps that have been contributed, for instance as shown in (3.19).

(3.17)
S0: How should I start?
T0: You have to show the equality of two sets. Please look up how to do this.

Type	Action
Solution step contribution	New step Augment existing step
Knowledge statement	Future step Strategy Concept Previously proved
Request	Hint Next step Explain concept Evaluation Task state
Task state	Describe

Table 3.4: Range of student actions by type

(3.18) **S4:** What does $(R \circ S)$ mean?
T4: $(a,b) \in (R \circ S)$ holds if and only if there is a $z \in M$ with $(a,z) \in R$ and $(z,b) \in S$.

(3.19) **S13:** Okay. So: $(R \cup S) \circ T = \{(x,y) \mid \exists z(z \in M \wedge ((x,z) \in R \wedge (z,y) \in T) \vee ((x,z) \in S \wedge (z,y) \in T))\}$ Is that right?
T12: Yes, very good. And now?

The final type of action is realised by utterances in which students talk about the state of the proving task. For instance they sometimes describe what in their view is the state of the task, namely that the task is done, as shown in (3.20). They also make explicit reference to the state of the task when they need to restart or go back to a previous state, as in (3.21). The students also ask questions asking the tutor to give them this information about the state of the task, such as "Am I done?"

(3.20) **S13:** Thus $(a,b) \in R \circ T \cap S \circ T$ q.e.d.

(3.21) **S26:** Start again. $(S \circ (S \cup R)^{-1})^{-1} = (S \cup R) \circ S^{-1}$
T26: Good.

To summarise we list the categorisation of actions that students perform in Table 3.4. Note that we have restricted ourselves here to task or domain related actions, so we do not consider off-topic utterances.

3.2.3 Categorisation of tutors' actions

The most common thing a tutor has to do in our corpus is give direct feedback on a solution step that the student has just contributed. That the tutor is mainly concerned with feedback has to do with the fact that the student

has the initiative and leads the solution construction process. There is a broad range of actions available to the tutor in this situation. The simplest are to accept or reject the step without giving any further comment, such as the acceptances in utterances T9 and T11 of (3.4) on page 38. This is not very informative feedback, because it does not tell the student why the step is being accepted or rejected, but nevertheless it is the most common way the tutors react. Similarly brief ways the tutors use to express acceptance include "correct", "that's right", or "go on", and for rejection "wrong", or "that's not right".

Another way the tutor can react to a contributed step and implicitly indicate rejection is to draw attention to a flaw, as has been analysed for this data by Horacek and Wolska [102]. The tutor can simply inform the student that the step was erroneous ("you've made an error"), or the information can be more specific, for instance telling the student that there is a syntactic problem like a missing bracket, that a variable is not quantified, or that some symbol has been mixed up, as illustrated in (3.5). The range of possibilities is limited only by the range of types of error that the tutor can recognise.

The tutor can give the student directed hints in order to remediate deficiencies that have been recognised in the student's solution or skills in general. A taxonomy of possible hints has been developed for tutoring mathematical theorem proving by Tsovaltzi et al. [196], who analysed the first DIALOG corpus. In order to help the student continue constructing the solution, the tutor can give away for instance full solution steps, concepts which are relevant in the current solution state, or strategies that the student should follow. Although the tutor's choice of hint and the content of hints will not be dealt with in this thesis, we will look more closely at one kind, namely elicitation, because requests will be relevant to the model we will propose in the coming chapters.

Eliciting information from the student is an important part of the tutor's range of options for remediation. Requests in tutorial dialogue, when they come from the tutor, are different to requests in other forms of task-oriented dialogue. When a tutor makes a request, the aim of the request is not to find out the answer, as would be the case in other dialogue genres. Instead the tutor already knows the answer but wants to ascertain whether the student also knows it. Its function is to check understanding rather than to request actual information. It is a way of finding out the student's knowledge state and how well he knows the concepts that are being tutored. (3.22) is an example of a series of such questions. The tutor asks the student a number of questions in a row to find out at what concept level the remediation should start. Only when the tutor has found out that the student does not know the basic concept of cross product does he begin to explain the series of concepts.

(3.22) **S17:** Backward application of the relation product: $(a, b) \in (R \circ T) \cup (S \circ T)$

T17: That's much too fast. Do you understand the concept of a relation?
S18: No
T18: Do you know the concept of cross product?
S19: No
T19: Ok. You have a set M. The ordered pairs from this set $M \times M := \{(a,b) \mid a \in M \wedge b \in M\}$ are the elements of the cross product. Did you understand that?
S20: yes
T20: A relation R is a subset of $M \times M : R \subseteq M \times M$
S21: Okay, and how does the proof go on from here?

As with hints in which information is given away to the student, there are various elements that the tutor can elicit from the student. For instance, he can ask the student whether he understands a particular concept, as illustrated in (3.22) above for the concepts of relation and cross product, although this relies heavily on self-assessment by the student, and is not necessarily reliable [123].

The tutor can also use elicitation-type actions in the feedback on specific steps, for instance to ask the student to explain the reasoning used to derive a new solution step, as in (3.23). While this kind of request does elicit domain content from the student, it is not a hint in the usual sense because it is concerned with a step that has been performed rather than one that is to come. It is a way of requiring the student to self-explain, which is a tactic which has been shown to lead to a learning effect [43]. The tutor can be more specific with this kind of elicitation and ask the student to fill in parts of a step that were not explicitly stated in the initial contribution of the step. This is another way to elicit a self-explanation from the student, because it requires that reasoning be demonstrated explicitly.

(3.23) **T28:** Why does this (correct) relationship hold?

Requests for explanation are triggered by the tutor's suspicion that the student may not fully understand the concepts that are required to derive a step. For instance if the step is too long the tutor may not accept that the student understood each individual substep, especially if he suspects the student is weak. Requests for augmentation or explanation in such a situation act as requests to show evidence of understanding of concepts and reasoning. When the student does show this, the tutor can conclude that the student understands these concepts and the tutor can maintain a better model of the student's expertise. We will use this notion of evidence of understanding in the model which will be presented in Chapter 4.

The answers that students give to the different types of elicitations follow the categorisation of actions from the previous section. For instance, the typical response to an elicitation of a concept definition will be much like a statement of a concept. When the tutor asks the student to fill in parts of an already

Type	Action
Give feedback	Accept Reject
Highlight error	Highlight error
Hint	Give away step Give away concept Give away strategy Elicit step Elicit concept Elicit strategy
Request	Explain reasoning Missing step element

Table 3.5: Range of tutor actions by type

contributed step, the answer from the student will be a step augmentation. We summarise the categorisation of the tutor's possible actions in Table 3.5. Note that due to the fact that we will not concentrate on hinting, this categorisation is far from exhaustive. However it does serve to put the actions which we will focus on later, namely feedback and requests, into the context of the broader range of possible actions.

3.2.4 Informationally redundant utterances

One characteristic of tutorial dialogue which applies to both student and tutors is their use of informationally redundant utterances (IRUs) [112, 201]. Sometimes it is necessary to repeat information that has already appeared in the dialogue, for example to make a known fact salient again to support an argument [202]. Such utterances are informationally redundant, which means that the proposition they express is entailed, presupposed or implied by a previous utterance in the discourse. We have found that such utterances can contain linguistic markers which signal that they are IRUs, for instance, "as you know" or "of course". Such marking signals to the hearer that the speaker expects that the hearer knows the information in the utterance already, and emphasises the reminding function of the utterance. This is particularly relevant in the context of giving the student hints.

If an IRU does not include this marking then it is an indication that the dialogue participants may have misaligned beliefs about what information has been uttered so far, due perhaps to short-term memory bounds or a previous misunderstanding. In order that this does not happen unintentionally, the tutor attaches marking to utterances when these would otherwise falsely indicate

that misalignment has taken place. We will look at IRUs in detail again as a case study to demonstrate the use of the model which will be presented in the next chapter, and therefore we describe the phenomena with some examples here. In (3.24) the content of utterance S10, namely that the formula embedded in the utterance holds, is repeated in utterance S18 without any marking indicating that it is informationally redundant. This could indicate that the student is no longer aware that this content had been uttered, so to realign the student's beliefs the tutor explicitly reminds the student that the proof step had already been performed. In (3.25) the tutor acts in a similar way, this time during the recapitulation of the proof. The formula which was successfully derived in S2 is used again, and the tutor makes this repetition explicit with the particle "of course".

(3.24)
S10: It holds that $(R \cup S) \circ T = \{(x,y) \mid \exists z(z \in M \wedge (x,z) \in (R \cup S) \wedge (z,y) \in T\}$
T10: That's right!
...
S18: By definition it holds that $(R \cup S) \circ T = \{(x,y) \mid \exists z(z \in M \wedge (x,z) \in (R \cup S) \wedge (z,y) \in T\}$
T18: That's right! You've already performed this step.

(3.25)
S2: $A \cap B = \emptyset$
...
T4: Right. Now what?
...
T8: ... The justification could for instance be: Let x be an arbitrary element of B, then it can't be in A (since of course $A \cap B = \emptyset$) ...
(*German: ... (da ja* $A \cap B = \emptyset$ *) ...*)

The data shows that misalignment between student and tutor can be observed in the case of informationally redundant utterances. Unmarked IRUs (such as S18 in example (3.24) should trigger strategies for realignment, and conversely, when IRUs are to be generated as part of pedagogical strategies (such as T8 in example (3.25)), these should be marked as such in order to avoid the student falsely concluding that misalignment has occurred.

3.3 Occurrence of dialogue frames

Dialogue frames [85] were presented in Section 2.2.1 as a model of the local structure of interactions between students and tutors in one-on-one tutoring. In this section we show by example that dialogue frames also occur in our corpus, and list the actions from the previous section which may occur in the dialogue frame phases. We first identify some parallels between the five phases and their expected realisation in our corpus, motivated by the characteristics of our tutoring scenario as outlined in Section 3.2.

Phase 1, in which the tutor asks a question, is seldom realised in our data. The reason for this lies in the proving task. The theorem which is to be proved is the only question per se. The initiative in constructing the solution is left with the student, and in effect each contribution could be seen as an answer to an implicit question from the tutor "What is the next proof step?". However such a question is rarely posed. Phase 2 of the dialogue frame, in which the student offers an answer, is realised in our data by utterances in which the student either contributes a new step or augments an existing step.

In phase 3 the tutor provides initial feedback on the student's answer. In our corpus this phase will nearly always be realised by the next utterance from the tutor after the student has contributed a solution step, since the tutors give direct feedback after every step. Phase 4 in the dialogue frame is a possible multi-turn sequence of collaborative improvement of the answer the student has offered. Into this category fall many of the different kinds of hints that tutors can perform in our scenario, such as pumping as a remediation tactic, or giving away or eliciting explanations or justifications. Phase 5, the assessment of the student's understanding of the answer, would be realised by a request from the tutor as to whether the student understands a particular concept which was used in the answer. However this occurs very seldom in our data.

According to [85], all phases but phase 2 may remain unrealised in a dialogue frame, and indeed we do not expect many occurrences of phases 1 and 5. We also expect to sometimes see an overlap between the feedback phase and the collaborative improvement phase. This will occur when the tutor's initial feedback on a contributed step is an elicitation. The elicitation is intended to improve the content of the step, but it may also implicitly include an element of feedback on the step itself. For instance we saw in (3.23) that the tutor can realise both of these actions in one utterance. In such cases phase 3 is conflated with the first utterance of phase 4. Phase 4 is also where evidence of understanding may be elicited from the student. Asking the student to augment the step has the parallel effect of requiring the student to demonstrate understanding of the concepts, as described in Section 3.2.3.

We can now examine some examples from our corpus annotated with the phases in the dialogue frame. In (3.26) we see two simple cases of the student's contributions being accepted by the tutor. The contribution in S0 corresponds to phase 2 and the tutor's acceptance in T0 to phase 3. Because the tutor is immediately satisfied that the student has understood the answer, phases 4 and 5 do not take place and the dialogue frame is closed. Such a sequence, consisting of a contribution followed by an acceptance or rejection of the step by the tutor, is the simplest and most common realisation of a dialogue frame in our data. S1 and T1 form a new dialogue frame which is the same as the

first except that phase 4 is realised in T1-2 by the tutor, who elaborates on the answer to tell the student that the step could have been performed more simply.

(3.26)

		Phase
S0:	$(R \circ S)^{-1} = \{(x,y) \mid (y,x) \in (R \circ S)\}$	2
T0:	correct	3
S1:	$(R \circ S)^{-1} = \{(x,y) \mid (y,x) \in \{(x,y) \mid \exists z(z \in M \wedge (x,z) \in R \wedge (z,y) \in S\}\}$	2
T1-1:	okay,	3
T1-2:	but you could have done that more simply	4

The example in (3.6) above exhibits a longer phase 4 of collaborative improvement. We repeat it here in (3.27) with the dialogue frame annotation. The student, in S8, has applied the definition of relation composition to simplify the left-hand side of exercise B ($(R \cup S) \circ T$), however without having previously introduced the pair (a,b). The appropriate step in between would have been "Let $(a,b) \in (R \cup S) \circ T$." The tutor does not give direct feedback about the correctness of the step, which would have constituted phase 3 of the dialogue frame, but rather begins phase 4 by querying the missing elements of the step in order to ascertain whether the student understood the step or not. The student correctly supplies the rule that was used along with its definition. This satisfies the tutor that the student has understood the use of the rule, but the tutor then goes on and improves the answer further by stating the premise which was still missing from the step.

(3.27)

		Phase
S8:	$\exists x$, so that $(a,x) \in (R \cup S)$ and $(x,b) \in T$	2
T8:	What does this follow from?	4
S9:	That follows from the definition of relation product for binary relations: $(a,x) \in R$ and $(x,b) \in S$, then $R \circ S$	4
T9:	More precisely: it follows from $(a,b) \in (R \cup S) \circ T$	4

The interaction in (3.28) is more complex and begins with the student's contribution in S19, phase 2 of the dialogue frame. It consists of two contributions, however only the first one (S19-1) is discussed. Although the identity the student has applied is stated, the contribution is incomplete in that it leaves implicit the premise from which the formula in S19-1 was derived. The identity in question is exercise W, which states that $(R \circ S)^{-1} = S^{-1} \circ R^{-1}$. The tutor is not satisfied with the incomplete step and responds with a request to elaborate the answer in T25, which begins phase 4 of the dialogue frame. This is an example of when phase 3, the tutor's initial feedback, is implicitly realised by the first utterance of the collaborative improvement phase 4. In S20 the student applies the definition of inverse to the identity from exercise W to derive

$(R \circ S) = (S^{-1} \circ R^{-1})^{-1}$. The tutor accepts this in T26 but still wants the student to justify the initial contribution. S20 and T26 continue the collaborative improvement of the student's step by adding elements to the step. Finally in S21 the student provides the right missing premise, which is the instantiation of the identity for the sets R and T. The tutor is now satisfied that the student understands the component parts of the step which was begun in S19-1, and accepts the step, closing phase 4.

(3.28)

		Phase
S19-1:	$(R \circ T) = (T^{-1} \circ R^{-1})^{-1}$ (by exercise W),	2
S19-2:	then it must also hold that $(S \circ T) = (T^{-1} \circ S^{-1})^{-1}$	2
T25:	Why does this follow from exercise W?	3/4
S20-1:	$(R \circ S) = (S^{-1} \circ R^{-1})^{-1}$ (according to exercise W),	4
S20-2:	then it must also hold that $(S \circ T) = (T^{-1} \circ S^{-1})^{-1}$ and $(R \circ T) = (T^{-1} \circ R^{-1})^{-1}$	4
T26-1:	All other steps are appropriate,	4
T26-2:	but the justification for $(R \circ T) = (T^{-1} \circ R^{-1})^{-1}$ is still missing.	4
S21:	$(R \circ T)^{-1} = (T^{-1} \circ R^{-1})$ (by exercise W)	4
T27:	Yes.	4

From these examples it is clear that we can recognise dialogue frames in this data. The realisation of phase 4 can become quite long if the discussion of a particular step extends over many utterances, however we will see in the model proposed in Chapter 4 that an internal structure can be assigned by categorising the actions that the student and tutor perform.

3.4 Comparison with other corpus analyses of tutorial dialogue

As we introduced in Chapter 2 there have been many other analyses of corpora of tutorial dialogue. Here we review some of their findings on the range of actions that are typically performed by students and tutors. Dzikovska et al. [69], in an analysis of a corpus of tutorial dialogue on the subject of differentiation as part of the LeActiveMath project, find many of the same action types as we have presented here. They find help requests, which correspond to our requests for concepts and hints, and progress evaluation and task progress references, which are equivalent to our references to the task state. They find that explanations are used in both the forward and backward direction. Forward explanations match our references to steps or to concepts which will be used in future steps. Backward explanations, which are performed in answer

to requests from the tutor such as "Why did you do that?", would be augmentations of solution steps in our categorisation. They find a category of knowledge statement which corresponds to ours, as well as language edits, which we have not included but which have been analysed for our corpus [209].

A further study of maths differentiation data by Porayska-Pomsta et al. [155] looks at students' affective states and the kinds of actions that influence them. They categorise student actions into 10 broad types including queries for steps, concepts or hints, corrections, self-knowledge statements, and reflection, which is sometimes realised as an explanation of reasoning, like our step augmentation. The tutor's actions include positive and negative feedback, corrective feedback, in other words drawing the student's attention to an error, requests for clarification, hints and hints followed by gauging questions such as "do you understand that?". In an analysis of spoken dialogue data in the domain of physics tutoring, Litman and Forbes-Riley [126] propose a set of dialogue acts in order to investigate learning gain correlations. They subdivide questions and answers into three and four categories, respectively, and categorise the tutor's actions into positive and negative feedback, restatement, recap, request, bottom out (which reveals the complete answer), hint and expansion, which reveals new details about the answer.

Graesser et al. [85], analysing the same data from which the dialogue frames account was developed, categorise the tutor's actions that they found. They include positive, negative and neutral feedback, pumping, which corresponds to our tutor requests, prompts, hinting and elaborations. Chi et al. [45] collected a corpus of face-to-face tutoring on the subject of the human circulatory system. They find that the tutors' actions included giving explanations, giving positive and negative feedback, answering content questions from the student, prompting, and asking comprehension-gauging questions. They note that because the dialogues were predominantly controlled by the tutors, the range of student actions was smaller. This contrasts with our strongly student-led dialogues. The student actions they identify are self-explanation, asking and answering questions, responding to scaffolding prompts, and reflecting. This categorisation has been used as the basis for further corpus analyses such as [133].

Overall we see an overlap with similar action definitions between these studies and the categorisation we have presented in this chapter, although there is some divergence where the goal of the study is different, for instance the investigation of affect in [155]. This indicates that our corpus exhibits phenomena typical of other tutorial dialogue corpora, both within and outside of the domain of mathematics, and suggests that the model which we will propose for this data should be applicable to the data collected in other studies.

3.5 Summary and discussion

The qualitative analysis in this chapter has demonstrated that dialogue frames occur in tutorial dialogue on mathematical theorem proving. Our starting point was a corpus of 37 such human-human dialogues, which we presented in Section 3.1. In Section 3.2 we analysed the mathematical content of the dialogues and presented the categorisations we have developed for the different actions that students and tutors can perform. We also looked at the specific example of informationally redundant utterances, whose occurrence can trigger linguistic marking. We then presented examples in Section 3.3 annotated with dialogue frame phases and showed which actions are part of which phases. The dialogue structures we found indeed reflect phases 2 though 4, with individual proof steps being proposed (phase 2) the tutor giving feedback (phase 3) and subsequently optionally elaborating the content (phase 4). Phase 4 includes the tutor's requests for evidence of understanding during the collaborative improvement of the answer. The tutor can use such requests to check the students' mastery of concepts and to better monitor the student's knowledge state. Phase 1 does not occur due to the student's initiative, and phase 5 is seldom realised.

We argue that dialogue frames systematically recur in the dialogues in our corpus. As an informal validation of the categorisation we have proposed, we reviewed the results of other analyses of tutorial dialogue corpora with respect to the action categories they find in Section 3.4.

The existence of dialogue frames in our data means that we can use them as a structural mechanism within a model for tutorial dialogue. Such as model will be proposed in the next chapter. We believe it is useful to model dialogue frames explicitly because the discussion and collaborative action that they contain encourage students to self-reflect and self-explain, which has been shown to lead to knowledge discovery [43]. The questions that tutors ask in the elaboration phase also allow them to infer the knowledge status of the student, which Chi et al. [46] have shown tutors have difficulty doing.

In the next chapter we will use this corpus analysis to motivate a rule-based model to account for the sequences of actions in our corpus which constitute dialogue frames. In Chapter 5 it will inform the development of an annotation schema for the corpus. Chapter 6 will propose a model to predict whether to enter into the elaboration phase 4 or not given the student's step in the context of the previous dialogue.

4

A model of solution step discussions

In the previous chapter we introduced the concept of dialogue frames in tutorial dialogue, proposed by Graesser et al. [85], and showed that they also occur in our corpus. In this chapter we present our model of solution step discussion subdialogues, which is an alternative, operational account of dialogue frames, conceived in the style of Traum's Grounding Acts theory [191]. It is a computational model which describes the possible sequences of actions by tutors and students in such discussions. The model tracks and updates the state of the tutor's beliefs about the knowledge state of the student. The situation in which an update to this state happens is the end of a solution step discussion. At this point the student has used, successfully or unsuccessfully, certain domain concepts and has thereby shown to have either understood or not understood them. The tutor can conclude that the student knows these concepts and the model extends the belief state accordingly. The architecture and representations in the model follow previous work on dialogue systems in the information state update tradition, as we have introduced in Chapter 2.

We begin in Section 4.1 by introducing Traum's Grounding Acts theory and arguing that dialogue frames can indeed be accounted for by a grounding-style model. We show the structural similarities that we have recognised and will exploit. Section 4.2 presents the architecture of the model. It consists of representations for the basic concepts of solutions steps, actions and evidence of understanding, a representation of dialogue state including the common ground, and a finite-state model of possible sequences of actions. In Section 4.3

we illustrate the operation of the model with a number of examples and uses before giving a short discussion in Section 4.4. Much of the material presented in this chapter has been published in [30] and [31].

4.1 Dialogue frames as grounding exchanges

In the model of dialogue frames proposed by Graesser et al. [85] (henceforth GPM) students and tutors discuss the content of answers that students contribute. In grounding subdialogues speakers and hearers discuss the content of utterances that speakers contribute. In this section we first introduce the concept of grounding and show that the subdialogues described by GPM's dialogue frames and those modelled by Traum's Grounding Acts theory have strong conceptual and structural similarities. It is these similarities which motivate and form the basis of the design of our model, including the types of dialogue actions and their order.

4.1.1 Modelling grounding

In order to communicate efficiently, participants in a dialogue take into account the information believed to be mutually known to them. They can use it to form assumptions about what else their interlocutor might know and can tailor their utterances to the current audience. The collected information is known as the common ground and the process of achieving mutual understanding in conversation is known as grounding [50, 51, 191]. During grounding dialogue participants try to collaboratively reach a situation in which the hearer has shown to have understood the propositional content which had been uttered by the speaker; the content is then considered to have been grounded. Grounding is a method by which participants can align their common ground, and is also a way to recover from the communicative failures associated with misunderstandings. Grounding is therefore crucial for successful communication.

One of the first detailed models of grounding was proposed by Clark and Schaefer [51]. In their Contribution Model, grounding is a two-phase collaborative process in which dialogue participants first present and then accept contributions. The combined effect of presentation and acceptance is to augment the common ground with the content of the contribution. In the presentation phase the speaker presents an utterance for the hearer's consideration. The assumption is that if the hearer can give sufficient evidence of having understood the utterance, then the speaker can believe that the hearer has understood the utterance. In the acceptance phase the hearer accepts the utterance by offering

	Evidence type	Description
1	Display	B displays verbatim all or part of A's presentation
2	Demonstration	B demonstrates what he has understood A to mean
3	Acknowledgment	B nods or says "yeah" or the like
4	Initiation of relevant next contribution	B starts the next contribution that would be relevant at a level as high as the current one
5	Continued attention	B shows that he is continuing to attend and therefore remains satisfied with A's presentation

Table 4.1: Types of evidence of understanding, as proposed by the Contribution Model

evidence of having understood it. He assumes that the speaker, upon registering this evidence, will come to believe that the hearer correctly understood the utterance.

Successful acceptance, and therefore mutual belief that the content was successfully understood, is reached in the acceptance phase when the grounding criterion has been met. It is the situation in which "the contributor and the partners mutually believe that the partners have understood what the contributor meant to a criterion sufficient for the current purpose" [51, p. 262]. Different kinds of evidence can be offered by a hearer to demonstrate understanding; these are listed in Table 4.1 for a presenter A and a hearer B. Displaying the content verbatim is considered to be the strongest form of evidence, whereas mere continued attention is the weakest. During the course of the grounding process multiple conversational turns may be performed. For instance the hearer may ask for a repair of the original presentation; the speaker may then offer this repair or not. The hearer also must not necessarily accept the presentation immediately.

In proposing a computational model of grounding, Traum [191] highlights a number of deficiencies of the Contribution Model. The first is the imprecise termination of the grounding process. Acceptances are simultaneously also presentations of new content and must be understood before they can fulfil their accepting function. However this means that they themselves must also be accepted in order to be understood, which leads to a theoretically endless chain of acceptances of acceptances. The model does not propose for instance a primitive acceptance which does not need to be accepted.

A further problem is the strength of the types of evidence which are proposed, because it is not clear that these in fact prove that an utterance has been successfully understood. A backchannel, which would be classed as the evidence type "acknowledge", can be given even when the presented content was not completely understood, for instance to show that acoustic but not necessarily intentional content was understood. Similarly a hearer may manage to

Grounding Act	Description
Initiate	Begins a new DU
Continue	Adds related content to an open DU
Acknowledge	Shows evidence of understanding of previous material
Repair	Corrects misunderstanding of the content of the DU
ReqRepair	Shows lack of understanding
ReqAck	Asks for the other interlocutor to acknowledge the DU
Cancel	Abandons the current DU without grounding it

Table 4.2: The set of acts in Traum's Grounding Acts model

perform a relevant next contribution even if the previous contribution had not been understood correctly.

Finally the Contribution Model is not suited to operationalisation. Traum argues that there is no easy way to tell the "state" of the current contribution while engaged in conversation. It is not always possible to detect whether an utterance by the initiator of the original contribution is a continuation of that contribution or a completion, so the hearer can not know if a contribution is complete and is ready to be accepted. This makes the Contribution Model unsuitable for implementation in a conversational agent.

To tackle these problems, and with a view to developing a computational model, Traum has proposed the Grounding Acts model [191, 192]. The content about which interlocutors try to reach mutual belief is contained in a Discourse Unit (DU). The performance of *grounding acts* manipulates the content and status of discourse units. These are listed in Table 4.2. *Initiate* is the equivalent of the presentation of a new contribution in Clark and Schaefer's model and *acknowledge* is the equivalent of their acceptance. An explicit *continue* act makes it possible to differentiate between the continuation of an existing DU and the initiation of a new one, similarly for *repair*, solving one of the difficulties of the Contribution model. In a major simplification, *acknowledge* collapses the full range of types of evidence listed in Table 4.1. Acknowledgments in this model can therefore be of varying strength.

Grounding acts take the DU from initiation through possible repair to acknowledgment and groundedness. The problem of recognition of the state of the current DU is solved by having an explicit finite-state model of the stages that the DU can go through. It defines the (possibly embedded) sequences of grounding acts which lead interlocutors to mutual belief. This way a conversational agent can know whether the DU has been repaired, is awaiting repair, or is awaiting acknowledgment. There is no need to be able to look ahead to interpret the grounding effects of an utterance. The finite-state model is shown as a set of transitions in Table 4.3, in which *I* refers to the initiator of the DU and *R* to the responder. F is a final state in the model and D is a dead

	In State						
Next Act	S	1	2	3	4	F	D
initiate^I	1						
continue^I		1			4		
continue^R			2	3			
repair^I		1	1	1	4	1	
repair^R		3	2	3	3	3	
ReqRepair^I			4	4	4	4	
ReqRepair^R		2	2	2	2	2	
ack^I				F	1	F	
ack^R		F	F			F	
ReqAck^I		1				1	
ReqAck^R				3		3	
cancel^I		D	D	D	D	D	
cancel^R			1	1		D	

Table 4.3: Discourse Unit state transition diagram (from [191])

state, which is reached when the DU is cancelled. In the subdialogue in example (4.1), taken from [191], the content of the initiating utterance I1-1 and the continuation I1-2 has been successfully grounded by the acknowledgment Grounding Act in R1. The state transitions which the DU goes through are S, 1, 1, F.

(4.1) **I1-1** Move the box car to Corning — init^I
I1-2 and load it with oranges — cont^I
R1 ok — ack^R

4.1.2 Dialogue frames are a kind of grounding

We argue that dialogue frames can be modelled in a similar way to the grounding subdialogues which are described by the Traum's Grounding Acts theory. The key parallel which we recognise and exploit is in the object of the grounding process: Both models are concerned with reaching agreement about understanding of content. In Grounding Acts this content is contained in the DU, in our model it is in the solution step. There is however a difference in the nature of this content. Whereas Traum's model is concerned with ensuring correct understanding of the content of an utterance and its intention, the object of the grounding process in our model is the student's deep understanding of a domain concept. Nevertheless the similarities warrant the reuse of the notion of evidence of understanding.

A further similarity between grounding and dialogue frames is that contributions, evidence of understanding and acceptance are associated with distinct roles played by the dialogue participants. In grounding these actions are per-

formed by an initiator and a responder. The initiator initiates the DU which is to be grounded, and may continue to add content to it. The responder must demonstrate evidence of having understood the DU by acknowledging it. In a dialogue frame the salient roles are those of tutor and student. The student makes contributions of content and may subsequently add to these. However it is also the student who is obliged to demonstrate understanding of the content under discussion. There are two reasons for this obligation. First, the content under discussion is part of the solution to the task at hand, which the student must solve. Second, by holding a position of authority the tutor is in general not obliged to explain any implicit reasoning, whereas the student must do so if asked. Clark and Schaefer's Contribution Model offers a direct parallel on this issue: In their acceptance phase there is an assumption that the speaker will believe that the hearer has understood the presentation if the evidence the hearer offers is strong enough. In a dialogue frame the tutor will believe that the student has deeply understood the domain content if the evidence the student offers is sufficiently strong.

The suitability of a grounding-type model for dialogue frames is illustrated in example (4.2) from the corpus.

(4.2) ... ...
S8: From this it follows that: $(x, y) \in S^{-1} \circ R^{-1}$, which was to be proved
T8: Correct. Please give a simple justification for the last step of the proof
S9: It follows directly from the definition of relation composition
T9: Right. You have completed the exercise

The first utterance is the contribution from the student of a new step in the current proof. We see this as being conceptually similar to an initiation in the Grounding Acts theory. In T8 the tutor confirms the correctness of this step, but also asks for further evidence of understanding from the student. This intention is similar to a request for acknowledgment in the Grounding Acts theory. The student fulfils this request by supplying the missing inference rule, demonstrating understanding of the content under discussion. Now the tutor considers the evidence supplied so far to be sufficient and accepts the complete step.

4.2 Elements of the model

The model we propose has at its core the concept of *Task-level Grounding* (TLG). TLG is the process by which tutor and student reach a state of mutual belief about the student's understanding of task-level concepts. The process is realised by the actions that the tutor and the student perform in the course of a subdialogue discussing a solution step. We use the term *grounding* because

TLG works in a similar way to Traum's grounding model, which we will refer to from now on as communication-level grounding. We use the term *task level* because our model is concerned with objects and content which belong to the task domain of the tutoring session.

At the local level, that is, at the level of individual subdialogues, the model uses a finite-state automaton to describe the possible sequences of utterances which constitute a discussion about a solution contribution. This is the equivalent of the state transition model in communication-level grounding. We will categorise the types of actions that students and tutors can make within such subdialogues, and describe the effect they have on the continuing dialogue and on the belief states of the interlocutors. Over the course of the whole dialogue the model maintains the belief state of the tutor with respect to the student, in other words, what the tutor believes that the student knows or understands. Based on the outcome of a discussion subdialogue the global model updates the common ground of the dialogue, a collection of agreed-upon facts. This allows the common ground to function as a simple student model, and its content can be used to inform the tutor's choice of action in the future. Overall the model defines a complete tutorial dialogue as a sequence of discussion subdialogues. Its foundation is a shared representation of dialogue state, including the common ground and information about the current and previous utterances and their content.

Our realisation of the dialogue model uses the Information State Update (ISU) approach [193], which we have introduced in Section 2.1. This is a general purpose approach to dialogue modelling which is characterised by an information state representing the state of the dialogue, a set of dialogue moves, and a set of rules which update the information state depending on its content and the dialogue moves that an interlocutor has performed. The update rules model the effects that utterances have, and define the possible transitions between information states.

In summary the elements of our model, which we will introduce in this section, are: a dialogue state in the ISU tradition; a representation of mathematical facts in the common ground; a set of *Task-level Grounding Actions* (TLGAs); a set of update rules; and a finite-state machine encoding sequences of dialogue states, whose transitions are annotated with preconditions and effects.

4.2.1 Contributing modules and assumed capabilities

As we have seen in Chapter 2, a dialogue system is dependent on a number of back-end modules which provide the domain-specific information that it

needs to complete the task at hand. Tutorial dialogue systems are dependent on a number of expert systems, and in developing our model of solution step discussions we assume that certain modules with certain capabilities can provide information to the dialogue manager as required. The most important of these is the mathematical domain reasoner, however we also assume the availability of modules for natural language analysis and generation, as well as a tutorial manager. We have already seen the schematic design of such a system, as envisaged by the DIALOG project, in Section 2.3.3.

The mathematical domain reasoner is an expert system which processes the content of the solution steps that the student inputs to the system. Because the dialogues in our data set are concerned with proofs in set theory, our mathematical domain reasoner is more specifically an automated theorem prover (ATP). When we introduced the architecture of the DIALOG project in Section 2.3.3 we described how Omega [177] is the envisaged ATP for the system due to its flexible human-level reasoning and sophisticated proof object. These features are not available from standard logic level provers. However within the lifetime of the project it did not prove possible to connect Omega to our dialogue model to analyse proof steps at run time. Despite this our model is conceived with the functionality of Omega in mind. We have extended its functionality for the purposes of this research to allow it to verify proof steps in the context of the proof object built so far [66, 67]. The ATP uses an internal formalisation of the mathematical theory, here binary relations, to analyse the steps that the student performs. It builds and maintains a tree-like proof data structure, which represents the current state of the proof and forms the context of the analysis of further steps. At its root is the theorem to be proved, at its leaves are tasks, or formulas which have yet to be fully proved. When there is no task left in a leaf then this leaf has been closed, and when all leaves are closed the theorem has been proved. The edges in the tree are transformations of tasks caused by the application of inference rules, in other words they represent derivations of new formulas. Information about proof steps, such as which or how many inference rules were applied, can be read directly from the proof object.

The ATP receives a sequence of proof steps as the dialogue progresses. The verification of each step is carried out by first expanding the tree, starting from the current task, using a depth-limited breadth-first search over inference rules. The student's step is compared to the newly-created nodes, and if it matches then it is considered correct. The non-matching nodes are pruned from the tree. By inspecting the newly created nodes the ATP can tell us which inference rules were applied how often, which premises were used, and what is left to prove. This approach has been tested on a subset of the DIALOG corpus,

and was able to correctly verify 113 out of 116 proof steps [67]. The benefits of this approach are that there is no need for preauthoring possible proofs in the style of model tracing approaches, and there is no required order of steps. The student is free to build any correct proof of the current theorem. Further, the Omega system reasons at a human-oriented level, which means its set of inference rules is designed to closely match the rules that mathematicians use. We further assume for our model the availability of an analysis of the relevance and granularity of proof steps. Granularity refers to the size of a proof step, and has been investigated within the Omega project [173, 19]. They analyse granularity as being dependent on the student, tutor and the state of the current task. We were again unable to add the granularity analysis to our model within the scope of the DIALOG project, however our assumptions of the available functionality are realistic and are supported by results from the Omega project.

The tutorial manager embodies a theory of tutoring. Work within the DIALOG project [74] has proposed a pedagogical module which relies internally on an ontology of mathematical concepts and a taxonomy of hints, such as the elicitation or giving away things like concepts or inference rules. Based on a simple student model, previously generated hints, and the categorisation of the current step, a hinting algorithm may generate a hint to be presented to the student. A natural language understanding module should analyse the student's input and extract the solution step it contains, if present. We assume that the dialogue manager receives a solution step and the dialogue moves that were performed. A natural language generation should verbalise the dialogue moves which our model outputs.

4.2.2 Concepts from the data

In our analysis of the corpus of tutorial dialogues in Chapter 3 we identified a number of concepts which we will now use in our formal model of task-level grounding. We present in turn our definitions of solutions steps, the task-level grounding acts, and the types of evidence of understanding students can offer.

Solution steps

As we introduced in Section 3.2.1, a *solution step* is the building block of the solution that a student is preparing. In tutoring the performance of a solution step is a knowledge demonstration, that is, a successful solution step is an indication of mastery of the concepts that it contains. A sequence of solution steps constitutes a possibly partial solution to an exercise. We use an explicit concept of solution step in our model because it is the object which

the student and tutor discuss, and it is the equivalent of the discourse unit in communication-level grounding. For mathematical theorem proving, the solution steps are steps in the proof of a theorem, and are typically derivations of new valid formulas. The components of such solution steps in our module are: the newly derived formula, a set of inference rules applied to derive the formula, and a set of premises from which the formula was derived.

For the possible values of solution step components in the theorem proving scenario, we orient ourselves to the study material which was given to the students in the data collection experiment presented in Section 3.1. The range of inference rules are those listed in Table 3.1 on page 37, and each one is associated with the number of times it was applied in the solution step. The types of formulas which may be part of solution steps are those which are syntactically well-formed using the standard operators and connectives from mathematical logic, set theory and binary relations, which are also taken from the study material. From mathematical logic we have $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$, $\forall$ and $\exists$. From set theory we have $\in$, $\cup$, $\cap$, $\subset$, $\subseteq$, $\supset$, $\supseteq$, as well as the notation for pairs (x, y) and the set construction expression $A := \{x | x = \ldots\}$. From binary relations we have the connectives $\circ$, meaning relation composition, and R^{-1} for the inverse of a relation R. Premises must be formulas which are part of the partial proof constructed so far.

Solution steps must not necessarily contain all of these components explicitly. When they do not, they are said to be underspecified. For instance, a solution step may be performed by stating only its inference rules and the derived formula. The premises are still part of the proof step in the logical sense, but the student's expression of the step left them unstated. The inference rules may also remain underspecified. If only a subset of either the inference rules or the premises are stated, then the remaining ones are said to be underspecified. Underspecification can be partially or fully resolved by explicitly stating previously missing components, thereby adding them to the step. This is referred to as augmentation. For instance the solution step under discussion in example 4.2 on page 58 would be represented after the first utterance by only the formula "$(x, y) \in S^{-1} \circ R^{-1}$"; the justifications and premises remain underspecified. After the student's second utterance, which is an augmentation, the step would be represented by the formula, the justification "composition", and no specified premises.

Task-level grounding acts

For a solution step to become grounded it must first be proposed by either the tutor or the student and subsequently accepted by the tutor, provided that the tutor has sufficient evidence to believe that the student has deeply understood

Action	Description
Propose	Proposes a solution step
Accept	Accepts the solution step and that the student has understood it
Reject	Rejects the step
ReqEv	Requests evidence to show understanding of the current step
SuppEv	Gives evidence showing understanding of the current step
Abandon	Abandons the discussion of the step without accepting or rejecting it

Table 4.4: Task level grounding actions

the concepts used in the step. To reach this state the student may be obliged to supply evidence of having understood the step, and this evidence can be of varying strength. It is these actions, used in the course of grounding a solution step, which we model, and we will refer to them as *task-level grounding actions* (TLGA). We list the set of TLGAs in Table 4.4.

The *Propose* action begins a new discussion subdialogue with the contribution of a new solution step. It contains the initial content which is to be discussed, and is the equivalent of *Initiate* in communication-level grounding. A proposal may be made by either dialogue participant, but in our data is typically made by the student. *Accept* and *Reject* are both intended to be performed by the tutor. They inform the student that the solution step was acceptable (or not acceptable, respectively) as an addition to the current partial solution, and bring the discussion to an end. Acceptability in this definition depends on whether the tutor believes that the student did in fact understand the solution step and its components at a deep domain level.

The actions *ReqEv* and *SuppEv* are concerned with ascertaining whether such understanding is in fact present, with a view to determining the acceptability of the step. *ReqEv*, or "request evidence", is performed by the tutor to ask the student to explicitly demonstrate deep understanding of the solution step in its current state. It could be performed when the tutor believes that the content provided by the student so far is not sufficient to show that the student has understood everything, for instance when the step is too complex or too underspecified for the student's current level of expertise. *SuppEv*, or "supply evidence", is the student's action in response to this request, and offers evidence of having understood the step. Because it is concerned with providing evidence of understanding, *SuppEv* is closest in function to *Acknowledge* in communication-level grounding. To paraphrase, the performer of a *SuppEv* states "yes, I have understood the content which we are currently discussing", just as the performer of an *Acknowledge* does. It is also a function similar to that of *Continue*, because it adds related content to a previously proposed solution step. Analogously *ReqEv* corresponds most closely to *ReqAck*. Finally

Type	Description
Augment	An elaboration of the current step
Reword	Paraphrase of the current step
Claim	Positive answer to "do you understand?"
Verbatim	Repeat back the step verbatim

Table 4.5: Types of evidence of understanding, from strongest to weakest

Abandon ends a solution step discussion subdialogue without either accepting or rejecting the step.

Evidence of understanding

Whether a solution step is acceptable or not depends directly on the level of evidence that the student can supply to show that he has understood it fully. For some steps (given the current state of the tutor's beliefs about what the student knows) the mere statement of the step is sufficient, but for others, further evidence of understanding may be required. Just as Clark and Schaefer [51] identify different types of evidence of understanding (which we have listed in Table 4.1 on page 55), the action *SuppEv* encompasses a number of different ways of showing understanding of a solution step. From our analysis of the data, we propose the four categories listed in Table 4.5 from strongest to weakest. Although verbatim repetition of the content being grounded is the strongest evidence type in Clark and Schaefer's communication level grounding model, at the task level it is the weakest form, since it does not show any understanding beyond recognition of the original utterance. Claiming understanding is an indication of self-reflection on the student's own belief state, and is considered a weak form of evidence. For instance Person et al. [151] find students' answers to comprehension-gauging questions to be misleading and that students are often unaware of their knowledge deficits. Rewording is a strong indication of understanding, but does not add anything to the current content which is being grounded. The strongest and most prevalent type of evidence is an augmentation of the current solution step with further information. This action shows that the student knows even those components of the step which were not explicitly stated in the original proposal.

In keeping with the observation within the Contribution Model that evidence must be "sufficient for the current purpose" [50, p. 136], the tutor's decision of whether to consider this evidence sufficient to assume understanding of the current content and to licence accepting the step depends at least on the content itself, the state of the dialogue and of the solution constructed so far. In general it will also depend on some student model and a teaching

Speaker	Utterance	Step	TLGA
Tutor	Now what is a factorial design?	1	–
Student	The design has two variables	2	*Propose*
Tutor	Uh-huh	3	*Accept*
Tutor	So there are two or more independent variable and one *PAUSE*	4	*ReqEv*
Student	dependent variable	4	*SuppEv* (*Augment*)
Tutor	Do you see that?	5	*ReqEv*
Student	Uh-huh	5	*SuppEv* (*Claim*)

Figure 4.1: Annotated example from [85]

Speaker	Utterance	TLGA
Student	From this it follows that: $(x, y) \in S^{-1} \circ R^{-1}$, which was to be proved	*Propose*
Tutor	Correct. Please give a simple justification for the last step of the proof	*Accept*, *ReqEv*
Student	It follows directly from the definition of relation composition	*SuppEv* (*Augment*)
Tutor	Right. You have completed the exercise	*Accept*

Figure 4.2: Annotation of example (4.2)

strategy. We will return to this topic in Chapter 6, in which we discuss the selection of the tutor's action in a given dialogue state.

We now have an inventory of solution steps, TLGAs and types of evidence, and to illustrate their use we now show the annotation of two examples. Figure 4.1 shows the annotation of a dialogue excerpt presented in [85], in which we see that TLG offers an alternative account of the same unit of dialogue structure. The question posed by the tutor at the start of the excerpt is not annotated with a TLGA because we assume that solution step discussions begin with the contribution of the step itself. Direct questions are rare in our data because of the nature of the task, which leads the student to take the initiative. Steps 4 and 5 are both accounted for as pairs of *ReqEv*, *SuppEv* because they both lead to the tutor being able to add to a private model of what the student has shown to understand. The example finishes before an final explicit acceptance is performed.

Figure 4.2 shows the annotation of example (4.2). As we showed above, the tutor asks for further evidence after the initial proposal of the solution step, and after an augmentation of the step is able to accept it.

CG	*common ground*
PENDING	*solution steps under discussion*
LU	*latest utterance*
HISTORY	*history of utterances*

Figure 4.3: The top-level structure of the information state

4.2.3 Dialogue state

As yet our model describes the function of individual actions within task-level grounding subdialogues. We must still add two further components which will extend it into an operational model: a data structure to represent the dialogue state, which will contain representations of the concepts we introduced in the previous section, and a finite state machine defining the acceptable sequences of TLGAs.

We model dialogue state in the style of traditional information state-based theories [193]. These theories, such as those implemented in GoDis [118, 117] or EDIS [135], typically represent some of the following concepts in their dialogue state: a dialogue history, some model of interlocutors' beliefs, plans or goals, the latest utterance, as well as any task- or theory-specific components that are required. Our information state will contain representations for the following elements, which will describe in turn: the common ground (CG), the latest utterance, solution steps and those that are under discussion, and a dialogue history. The top-level structure of our IS is shown in Figure 4.3.

We model CG as being similar to that of DeVault and Stone [64], who build on Lewis' [124] notion of context as a conversational score. Their *objective normative context* is a product of the actions taken by the dialogue participants in the preceding interaction. In our case, actions in the dialogue result in the dialogue participants having beliefs about the truth (or falsity) of the propositions that have been contributed by the student and evaluated by the tutor. This is combined with the knowledge in the study material that the students are given before the tutorial session. We assume that it is part of the CG at the start of the dialogue. The CG is populated by the facts which have been made true by the actions of the dialogue participants. In our model this includes the facts that propositions were uttered, the evaluations of those utterances by the tutor, and the facts that the student knows about the domain as a result of preparatory study.

Our notion of CG is also similar to that of Matheson et al. [135]. The GND (abbreviation for "grounded") element in their dialogue state contains material which has been grounded. Within GND a history of dialogue moves is stored in GND/DH and the propositions to which the dialogue participants are

committed are stored in GND/SCP. We will also maintain a set of propositions in the common ground (in CG/PROPS), namely those which have been agreed upon in the course of the dialogue. The common ground additionally includes a stack of discourse obligations in CG/OBL, which are the actions the dialogue participants are obliged to perform in the future. This topic is taken up in more detail in Section 4.2.4. Our representation of the common ground is the following:

$$\begin{bmatrix} \text{PROPS} & \{\texttt{uttered(student, solnstep1)}, \ldots\} \\ \text{OBL} & \langle \texttt{obl(tutor, address)}, \ldots \rangle \end{bmatrix}$$

The slot CG/PROPS is a set, and supports element of, add and delete operations. There are three types of propositions which can appear in it. The fact that a solution step s was uttered by a speaker A is modelled as $\texttt{uttered}(A, s)$. If we have $\texttt{uttered}(A, s) \in$ CG/PROPS we know only that the event took place, however we do not yet know anything about the truth or acceptability of the solution step. For this we use the proposition type holds(). Depending on whether they were accepted or rejected by the tutor, for each solution step s that has been uttered the common ground will contain either $\texttt{holds}(s)$ or $\neg\texttt{holds}(s)$ respectively. The negation of holds() is not intended to express that the step was wrong, rather it means it was not right. This distinction is important for irrelevant steps, which may be logically correct but are still rejected by the tutor in the context of the current solution.

For previous knowledge that the student is assumed to have at the outset of the tutorial session, we use the proposition prev(). Its argument is a mathematical concept that the student knows, taken from the set of concepts we introduced in Section 3.2.1. For instance if prev(set_union) $\in$ CG/PROPS then the student is judged to know the concept of simple set union.

The representation for solution steps reflects our description of their constituent parts as presented above. A solution step consists of all of the information we know about it after it has been analysed by the mathematical domain reasoner. Its structure is illustrated in Figure 4.4, which is the representation of the solution step from example (4.2), after the student's first utterance and after the evaluation has been provided by the domain reasoner. Each solution step will be assigned an identification string when it is first uttered in the dialogue. The IDs are used to refer to the solution steps in other parts of the dialogue state, such as in the PENDING and CG/PROPS slots, to avoid having to copy the complete structures. The FORMULA slot contains the formula that was derived in the step. Similarly the PREMISES slot contains a list of such formulas to encode the premises from which the formula was derived. The inference rules which were used in the derivation of the step are encoded in the

$$\begin{bmatrix} \text{ID} & \texttt{solnstep1} \\ \text{FORMULA} & (x,y) \in S^{-1} \circ R^{-1} \\ \text{PREMISES} & \langle\rangle \\ \text{RULES} & \langle\rangle \\ \text{EVALUATION} & \begin{bmatrix} \text{CORR} & \texttt{correct} \\ \text{GRAN} & \texttt{too\_coarse} \\ \text{REL} & \texttt{relevant} \end{bmatrix} \end{bmatrix}$$

Figure 4.4: Representation of solution steps in the dialogue state

$$\begin{bmatrix} \text{ID} & \texttt{utterance\_s1} \\ \text{SPEAKER} & \texttt{student} \\ \text{DMOVES} & \{\mathit{Propose}\} \\ \text{SOLN-STEP} & \texttt{solnstep1} \end{bmatrix}$$

Figure 4.5: Representation of utterances in the dialogue state

RULES slot as a list of rule names, taken from the same list we presented in Section 3.2.1. The evaluation of the solution step, assuming it is already available, is encoded along the three dimensions provided by the mathematical domain reasoner: correctness, granularity and relevance. The possible values of the correctness slot are "correct", "incorrect" and "partially correct". The granularity slot can have the values "appropriate", "too coarse-grained" and "too fine-grained". The relevance value can be "relevant", "irrelevant" or "partially relevant". All three may carry a "not applicable" value.

We use a separate top-level slot to store the solution steps which are currently under discussion, which we call PENDING. This slot functions like the CDU element of the dialogue state in [135], where CDU stands for current discourse unit. CDU contains the discourse unit which has been initiated but not yet grounded. Similarly solution steps which have been proposed but not yet accepted or rejected, or their IDs, will be stored in PENDING.

The top-level slot LU contains a representation of the latest utterance in the dialogue; Figure 4.5 shows the representation of the first utterance in example (4.2). Traditionally the LU slot stores the speaker of the latest utterance and a list of the dialogue moves which were realised by the utterance. In our model the value of LU/SPEAKER is either "student" or "tutor", and the value of LU/DMOVES is a set of TLGAs. To these two we add a solution step, if the utterance in question contained one, and an identification string. The slot LU/SOLN-STEP contains a structure as shown in Figure 4.4 above.

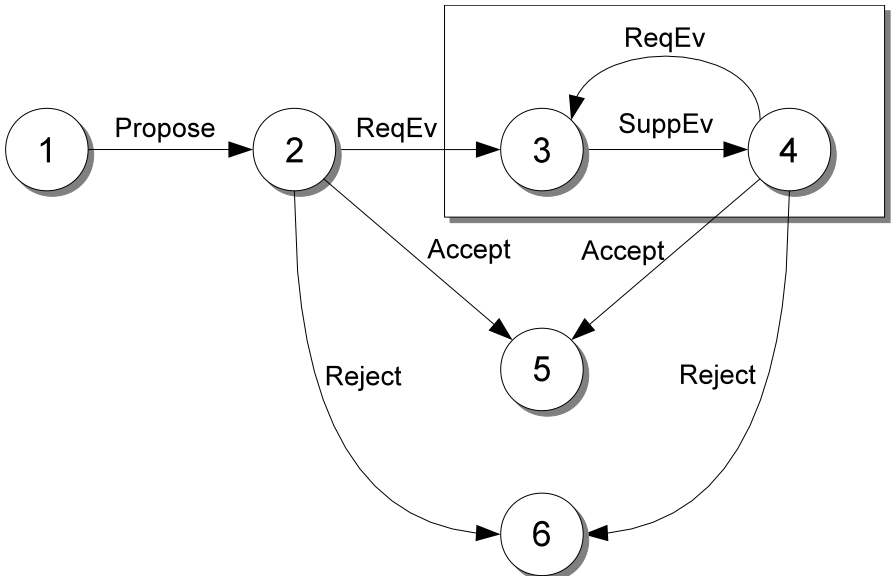

Figure 4.6: The FSA describing task-level discourse units

4.2.4 Finite-state model of subdialogue structure

Like in Traum's model of grounding, the legal sequences of utterances in a solution step discussion subdialogue will be governed by a finite-state machine. The states in the finite-state machine are dialogue states, whose representation we have introduced in the previous section. The edges are thus transitions between dialogue states, and are traversed in reaction to the recognition of dialogue moves that the tutor or student perform. The traversal of an edge triggers an update to the dialogue state. The combination of the finite-state machine and the dialogue state representations is an operational model of solution step discussion subdialogues. From the finite-state machine we will extract a set of update rules which operate on the dialogue state, and we will use these in the implementation of the model in an ISU framework.

The finite-state machine is shown in Figure 4.6. In state 1 no solution step has yet been proposed or is under discussion. The only legal TLGA is *Propose*, which begins the subdialogue with an ungrounded solution step. This is the equivalent of GPM's phase 2, the contribution of an answer from the student. The tutor now has three options to respond with: either *Accept*, *Reject* or *ReqEv*. These constitute GPM's phase 3, in which the tutor gives initial feedback to the student's answer. If the tutor accepts the step, the subdialogue ends with it being grounded in state 5. If the tutor rejects it, the subdialogue ends in state 6 without the step having been grounded. In both of these cases phases 4 and 5 are not realised.

If the tutor decides to request evidence of understanding, the subdialogue enters GPM's phase 3 — collaborative improvement of the quality of student's

contribution. The finite-state machine is in state 3, and now the student should respond by supplying the requested evidence. In state 4, the tutor has the same range of choices as in state 2, only now the student has supplied evidence of understanding. The tutor may accept or reject the step, ending the subdialogue in state 5 or 6 respectively, or may remain in the collaborative improvement phase by requesting further evidence from the student. A series of such requests and answers will eventually allow the tutor to either accept or reject the step. In any state in the model either dialogue participant can perform an *Abandon*, which ends the subdialogue in a final state 7. The solution step under discussion is removed and does not become grounded. In the interest of clarity in the diagram, these edges are not drawn.

The link between the dialogue state introduced above and the finite-state model of solution step discussions is realised in the definition of the transitions between states in the finite-state model. The conditions which determine whether a transition can be made are modelled as constraints on the content of the dialogue state, and transitions have effects which can change the content of the dialogue state. This design choice has been made with a view to implementing the model using the information state update approach.

In order to restrict which TLGAs may be performed in which dialogue states, we use a rudimentary model of discourse obligations. We follow previous work [194] in which discourse obligations are a representation of what an agent should do, induced by a set of social conventions. Their account of obligations makes no assumptions about shared knowledge, and offers an account of uncooperative or adversarial dialogues. This property makes obligations attractive for modelling tutorial dialogue, because the tutor's role is not one of complete co-operativity. For instance a tutor may respond to an information request from the student with "You tell me" or even "I'm not telling you". Such a response would fulfil the obligation to respond that the request introduced, but without actually answering the request itself.

Agents deliberate based on both their goals and their obligations, the crucial difference being that obligations must be fulfilled whereas goals must not necessarily be followed. Obligations encode social norms, such as that assertions should be addressed and requests for action acted upon. For the norms associated with the tutoring scenario we introduce $SuppEv_o$ and $Address_o$ to denote the obligation on the student to supply evidence to the tutor and the obligation on the tutor to address the student's contributions respectively. The introduction and discharge of obligations is associated with the performance of TLGAs as listed in Table 4.6. Obligations are annotated with a subscript *o* to differentiate them from dialogue moves. We will use the notation `obl`(*speaker*,$type_o$) to denote that an obligation of a particular type is imposed

Type	Introduced by	Discharged by
$SuppEv_o$	*ReqEv*	*SuppEv*, *Abandon*
$Address_o$	*Propose*, *SuppEv*	*Accept*, *Reject*, *ReqEv*, *Abandon*

Table 4.6: Obligation types

on a speaker.

Following [114], who develop an ISU-based dialogue model with discourse obligations, we put the current set of obligations into the common ground (CG/OBL). This reflects the fact that since obligations are derived from performed utterances, then as long as the student and tutor follow the same social conventions, they will always have an aligned set of obligations. By using obligations we do not have to explicitly name the states from the finite-state machine in either the definition of the transition rules or the dialogue state, rather they are implicitly encoded in the current list of obligations. Again, this design choice has been made with a view towards the implementation.

The definitions of the transition rules are listed in Table 4.7. The table contains some notation and operations which we use to write our rules. Equality of the value of an information state slot is tested with "=", and membership of a set object with $\in$. We use local variables to store objects for the course of the execution of a rule with the "let" construct. An object is added to the top of a list with "push", and "add" and "del" are used to add to and remove from a set.

We now go through the intended meaning of each rule in turn. Rule A is the transition from state 1 to state 2 in Figure 4.6. If the speaker of the last utterance was the student, and a *Propose* was performed, then get the evaluation of the solution step from the mathematical domain reasoner, put the step in the PENDING slot, and introduce an obligation on the tutor to address this proposal. Rules B and C cover the situation in which the tutor either accepts or rejects the current solution step. They fire if the last speaker was the tutor, if an *Accept* or a *Reject*, respectively, was performed, and if the tutor currently has an obligation to address the student's latest contribution to the solution step. In both cases the current step is removed from the PENDING slot and the obligation to address is deleted. In the case of acceptance the proposition `holds`(s) is added to the common ground, for a rejection $\neg$`holds`(s) is added.

The transition in Rule D happens when the tutor requests evidence of understanding. Like the previous two rules, the performance of a *ReqEv* discharges the obligation to address, so this obligation is deleted. The other effect of this rule is to introduce an obligation on the student to supply evidence. Rule E covers the case in which the student supplies the requested evidence. In doing so the obligation to supply evidence is discharged, and a new obligation is

Rule		Definition
A	if	LU|SPEAKER = student *Propose* ∈ LU|DMOVES Let *s* := LU|SOLN-STEP
	then	*update s with evaluation* PENDING := *s* push(obl(tutor, *Address*$_o$), CG/OBL)
B	if	LU|SPEAKER = tutor *Accept* ∈ LU|DMOVES Let *s* := PENDING obl(tutor, *Address*$_o$) ∈ CG/OBL
	then	PENDING := ∅ del(obl(tutor, *Address*$_o$), CG/OBL) push(holds(*s*),CG|PROPS)
C	if	LU|SPEAKER = tutor *Reject* ∈ LU|DMOVES Let *s* := PENDING obl(tutor, *Address*$_o$) ∈ CG/OBL
	then	PENDING := ∅ del(obl(tutor, *Address*$_o$), CG/OBL) push(¬holds(*s*),CG|PROPS)
D	if	LU|SPEAKER = tutor *ReqEv* ∈ LU|DMOVES obl(tutor, *Address*$_o$) ∈ CG/OBL
	then	del(obl(tutor, *Address*$_o$), CG/OBL) add(obl(student, *SuppEv*$_o$), CG/OBL)
E	if	LU|SPEAKER = student *SuppEv* ∈ LU|DMOVES Let *s* := PENDING obl(student, *SuppEv*$_o$) ∈ CG/OBL
	then	del(obl(student, *SuppEv*$_o$), CG/OBL) push(obl(tutor, *Address*$_o$), CG/OBL) *update pending step*
F	if	*Abandon* ∈ LU|DMOVES PENDING ≠ ∅
	then	PENDING := ∅ CG/OBL := ∅

Table 4.7: Transition rule definitions

introduced on the tutor to address this latest contribution. If the *SuppEv* utterance added content to the current solution step, this update is also made, for instance evidence of type *Augment* resolves underspecification by adding elements to the solution step such as justifications or premises. Finally an *Abandon* may be performed by either dialogue participant at any time as long as there is a solution step under discussion. Any pending step or obligations are deleted, which ends the subdialogue without changing the common ground.

4.3 Examples

To illustrate the operation of the model we now present some examples. The simplest solution step discussion subdialogue is one in which a proposal from the student is followed directly by either an acceptance or a rejection of that proposal by the tutor. This is the trivial case in that no actual discussion takes place, and in terms of dialogue frames this constitutes skipping step 3, collaborative improvement. The majority of interactions in the corpus are of this type. They are accounted for by the model as a sequence of rule A followed by rule B, which leads to the update of the common ground with `holds`(s). A more illustrative example is one in which some discussion takes place. We will go step by step through example (4.2), whose annotation was given in Figure 4.2.

The performance of a *Propose* by the student to start the discussion causes rule A to fire. This updates the dialogue state with the evaluation of the step, puts it on PENDING, and adds the tutor's obligation to address it. The full dialogue state resulting from this update is shown in Figure 4.7. The tutor's response is a *ReqEv*, which triggers rule D. The pending step remains the same, but the obligation to address is discharged and the content of CG/OBL becomes

$$\left[\text{CG} \quad \left[\text{OBL} \quad \left\langle \texttt{obl(student, SuppEv}_o\texttt{)} \right\rangle\right]\right]$$

The dialogue is now in state 3 of the finite-state machine, that is, in GPM's answer improvement phase. The student performs a *SuppEv* with evidence type *Augment*, which triggers rule E. The result is that the obligation to supply evidence is discharged and a new obligation obl(*tutor*, $Address_o$) to address the contribution is imposed on the tutor. The solution step is also augmented by adding the rule name "`composition`" to the RULES slot. The final utterance in the example is the tutor's acceptance of the step under discussion, which triggers rule B. The pending step is removed, the obligation is discharged, and the fact that the step holds is added to the common ground. The final dialogue state is shown in Figure 4.8.

$$\left[\begin{array}{ll}
\text{CG} & \left[\begin{array}{ll} \text{PROPS} & \{\texttt{uttered(student, solnstep1)}, \ldots\} \\ \text{OBL} & \langle \texttt{obl(tutor, Address}_o\texttt{)} \rangle \end{array}\right] \\
\text{PENDING} & \left\langle \left[\begin{array}{ll} \text{ID} & \texttt{solnstep1} \\ \text{FORMULA} & (x,y) \in S^{-1} \circ R^{-1} \\ \text{PREMISES} & \langle\rangle \\ \text{RULES} & \langle\rangle \\ \text{EVALUATION} & \left[\begin{array}{ll} \text{CORR} & \texttt{correct} \\ \text{GRAN} & \texttt{too_coarse} \\ \text{REL} & \texttt{relevant} \end{array}\right] \end{array}\right] \right\rangle \\
\text{LU} & \left[\begin{array}{ll} \text{ID} & \texttt{utterance_s1} \\ \text{SPEAKER} & \texttt{student} \\ \text{DMOVES} & \{\textit{Propose}\} \\ \text{SOLN-STEP} & \texttt{solnstep1} \end{array}\right] \\
\text{HISTORY} & \langle \ldots \rangle
\end{array}\right]$$

Figure 4.7: Dialogue state after the student's *Propose*

4.3.1 Using the common ground: Detecting misalignment

The benefit of maintaining common ground in a dialogue model for tutorial dialogue lies in supporting the action selection algorithm. We will deal with the question of selection in more detail in Chapter 6, but here we will show examples concerned with misaligned context. If a tutorial dialogue system knows more about the current knowledge state of the student, it is better equipped to tailor its output and handle possible problems. One example of using common ground to draw conclusions about the student's knowledge state is in detecting misaligned context. In general if grounding fails then dialogue participants may reach the state of having differing beliefs about the state of the common ground. Their CGs are said to be "defective" [183] or misaligned, and this situation can lead to miscommunication. Tutoring is particularly prone to such problems because of the inherent asymmetry of knowledge between the student and the tutor.

A clear manifestation of misaligned context in dialogue is the occurrence of informationally redundant utterances (IRU), examples of which we have presented in Section 3.2. An utterance is informationally redundant if the proposition it expresses is entailed, presupposed or implicated by a previous utterance in the discourse [201]. If an unmarked IRU is performed then the information it contains, which has already been grounded, is being repeated without the speaker indicating that this repetition is being done on purpose.

$$\begin{bmatrix} \text{CG} & \begin{bmatrix} \text{PROPS} & \{\texttt{holds(solnstep1)}, \texttt{uttered(student, solnstep1)}, \ldots\} \\ \text{OBL} & \langle\rangle \end{bmatrix} \\ \text{PENDING} & \langle\rangle \\ \text{LU} & \begin{bmatrix} \text{ID} & \texttt{utterance\_t2} \\ \text{SPEAKER} & \texttt{tutor} \\ \text{DMOVES} & \{\mathit{Accept}\} \\ \text{SOLN-STEP} & \texttt{n/a} \end{bmatrix} \\ \text{HISTORY} & \langle \texttt{utterance\_s1}, \texttt{utterance\_t1}, \texttt{utterance\_s2}, \ldots \rangle \end{bmatrix}$$

Figure 4.8: Final dialogue state after the tutor's *Accept*

Predicate	Definition
grounded(s)	holds(s) $\in$ CG/PROPS $\vee$ $\neg$holds(s) $\in$ CG/PROPS
iru(u)	grounded(solnstep_of(u))
misaligned()	(iru(LU) $\wedge$ LU/MARKED$-$) $\vee$ ($\neg$iru(LU) $\wedge$ LU/MARKED$+$)

Table 4.8: Predicates to detect misalignment of the common ground

This indicates to the hearer that this information was not in what the speaker believes to be the CG of the dialogue. The hearer must then conclude that the CG has become misaligned. The occurrence of IRUs in our corpus has been highlighted by Karagjosova [112].

We detect misalignment in our model by defining a predicate on the common ground and the latest utterance. For this we will assume that the natural language analysis module can recognise whether a utterance was marked for informational redundancy, for example with a particle such as "of course". We extend our representation by adding the slot LU/MARKED with the range $\{+, -\}$ to the latest utterance. We then define the set of predicates listed in Table 4.8. Informational redundancy is defined in terms of the solution step that an utterance contains. If the solution step was either accepted or rejected by the tutor in the previous dialogue, then the step is in the CG, and grounded(s) is true. An utterance is an IRU if its embedded solution step (accessed by solnstep_of()) is grounded. The dialogue context is in turn misaligned if either the latest utterance is an IRU which has not been marked for informational redundancy, or if the latest utterance is marked for informational redundancy but is not an IRU. Now that we can detect misalignment in the dialogue model, we are in a position to indicate this to the pedagogical module, which can take appropriate action to attempt to remedy the problem.[1]

[1]Note that the definition of grounded(s) implies some test of formula equality, and we are assuming only a simple string representation. This is clearly a strong approximation, and more sophisticated representations

We now consider again two of the examples of misalignment from Section 3.2. For the purpose of example (3.24), repeated here as (4.3), we let s_1 stand for the solution step embedded in utterance S10.

(4.3) **S10:** It holds that $(R \cup S) \circ T = \{(x,y) | \exists z(z \in M \wedge (x,z) \in (R \cup S) \wedge (z,y) \in T\}$
T10: That's right!
...
S18: By definition it holds that $(R \cup S) \circ T = \{(x,y) | \exists z(z \in M \wedge (x,z) \in (R \cup S) \wedge (z,y) \in T\}$
T18: That's right! You've already performed this step.

The *Propose* in S10 and the *Accept* in T10 trigger the update rules A and B, leading to s_1 becoming grounded. The resulting CG is

$$\left[\text{CG}\left[\begin{array}{ll}\text{PROPS} & \{\texttt{holds(s\_1)},\ldots\} \\ \text{OBL} & \langle\rangle\end{array}\right]\right]$$

When the *Propose* in S18 is performed the predicate `misaligned` becomes true of the current dialogue state because S18 is an IRU (the solution step it contains is a match of s_1, and is therefore grounded) but is unmarked for informational redundancy. We can conclude that misalignment has taken place and inform the tutoring module. In the example the tutor decided to remedy the misalignment by reminding the student that the solution step in question had already been performed.

If the tutorial dialogue system were to generate an unmarked IRU then the student could mistakenly conclude that the common ground has become misaligned. Therefore detecting IRUs in the output of the dialogue system is also an important aspect. It is illustrated in example (3.25), repeated here as (4.4). As a result of the *Propose* in S2 and the *Accept* in T4 the CG includes $\texttt{holds}(s_3)$, where s_3 is the formula $A \cap B = \emptyset$.

(4.4) **S2:** $A \cap B = \emptyset$
...
T4: Right. ...
...
T8: ... The justification could for instance be: Let x be an arbitrary element of B , then it can't be in A (since of course $A \cap B = \emptyset$) ...
(*German: ... (da ja* $A \cap B = \emptyset$ *)* ...)

When the system recapitulates the solution in T8, one of the utterances contains a formula which matches s_3. That means that $\texttt{grounded}(s_3)$ holds and that this utterance is an IRU, and to generate it without marking would be

would be possible, for example a context-sensitive definition of formula equality supported by the domain reasoner.

a false indication of misalignment. So that the student does not mistakenly conclude that misalignment has taken place, the tutor adds the marking "of course" to indicate informational redundancy.

4.4 Summary and discussion

In this chapter we have presented a computational model of solution step discussion dialogues based on the concept of task-level grounding actions. The model accounts for the same subdialogues as described by Graesser et al. in an analysis of tutorial dialogues as dialogue frames. We were able to show that dialogue frames could be described in a grounding-style model, and the model we have presented is similar in design to Traum's Grounding Acts theory. Our model maintains the common ground in a tutorial dialogue, which is updated to reflect the tutor's beliefs about the student's task-level knowledge state. The common ground functions as a simple student model. Updates to the common ground are triggered by task-level grounding actions. For instance when the tutor accepts a solution step that the student has proposed, this fact becomes common ground. The decision of whether to accept or reject a solution step is made at least partially based on what evidence of understanding has been offered by the student — if the evidence is deemed insufficient the tutor can demand that the student provides further evidence to show understanding of the step.

The model is characterised by a set of task-level grounding actions, a dialogue state, and a finite-state machine defining transitions between dialogue states. The dialogue state contains representations of concepts found in the data, such as solution steps and evidence of understanding. The TLGAs and the finite-state machine are analogous to grounding acts and the finite-state model in the Grounding Acts theory. The notion of graded evidence of understanding is taken from Clark and Schaefer's Contribution Model.

The combination of grounding and tutorial dialogue has been advocated by Baker et al. [13] who argue that learning from grounding is the basis of collaborative learning, but admit that this does not guarantee "deeper" understanding. However few tutorial dialogue systems explicitly use a model of common ground. Rickel et al. [164] use a general dialogue model in a tutoring system which combines pedagogical expertise with collaborative discourse theory and plan recognition. Their approach models the knowledge state based on steps that the student has been exposed to, however without considering whether the student in fact fully understood these. Zinn et al. [218] present a tutorial dialogue system which maintains common ground in the dialogue model, however they do not make use of grounding status to structure the dialogue

locally and the choice of tutoring actions is not informed by the state of the CG.

We have made certain simplifying decisions in the design of the model. Although it would be useful to be able to refer to objects in the discourse explicitly, we do not use a discourse model. This restricts our ability to refer to solution steps or their parts, or to resolve such references used by the student. We use the common ground as a student model, however it is a rudimentary one. The representation of mastery of a concept is limited to whether that concept has been successfully used so far or not. Although the dialogue model makes no further claims about this, the common ground does form the basis for deliberation about the student's knowledge state in the pedagogical module.

In the corpus there are situations in which there is more than one solution step under discussion at a time. To handle this situation it would be possible to run two instances of the finite-state machine and have two pending solution steps on PENDING. Like downdating of questions under discussion, one TLGA could then conceivably ground multiple solution steps. However without a discourse model it would be difficult to attribute actions to the right instance of the finite-state machine.

Despite these simplifications the design of the model does facilitate certain possible extensions. A given dialogue model or theory of some other dialogue phenomenon should be able to interact with ours due to our use of obligations to implement the finite-state machine. Whenever there is an active obligation on CG/OBL, but a non-TLGA is performed, the obligation simply remains, effectively leaving the model in the same state as before. This way we could for instance combine communication-level grounding such as [135] and task-level grounding: whenever a communication-level grounding act is recognised, the TLG model is put on hold until the intended meaning of the utterance can be grounded. This way problems at the communication level would be solved before problems at the task level.

Overall we have a continuously updated model of common ground which can inform a selection algorithm for the tutor's actions. Much further information can be computed from the contents of the common ground, for instance what rules have been used by the student how often, whether they were of suitable correctness, granularity and relevance, and so on. In Chapter 6 we investigate selection in a series of machine learning experiments, and see that the content of the common ground can contribute to and improve this choice.

5
Corpus annotation at three levels

The operational model for solution step discussions which we have presented in the previous chapter was developed based on a qualitative analysis of the phenomena found in a corpus of tutorial dialogues. In the course of this analysis we proposed two dimensions of categorisation of utterances in the corpus. The first, in Chapter 3, is a categorisation of the actions that students and tutors can perform in relation to the exercise solving task. The second, in Chapter 4, is a categorisation of the utterances' function in task-level grounding. This chapter takes the insights from the qualitative analysis and uses them to inform a quantitative analysis of the content of the corpus.

We will present an annotation of the corpus on three levels. The first is the task-level grounding action (TLGA) level, for which we develop annotation guidelines based on the definitions of the TLGAs from Section 4.2.2. The second level is the dialogue move level. Here we take the general-purpose DAMSL dialogue move encoding scheme [5] and extend it to account for the types of actions we highlighted in Chapter 3. Finally we perform an annotation of the mathematical content of the student's proof steps, with the goal of approximating the output which a mathematical domain reasoner would provide.

There are two main goals which motivate us to carry out these annotation experiments. The first, which particularly concerns the TLGA and dialogue move level, is to provide quantitative evidence for the occurrence of the action types which we have proposed. The second is to use the corpus as a data set for a series of supervised learning experiments, which we will report on in Chapter 6. We will also use the corpus for the evaluation of our model, which will be presented in Chapter 7.

5.1 Annotation levels

We now present the three annotation levels in turn. For each one we outline the motivation behind it and the intuitive concepts it is designed to encode. We present the methodology we applied to prepare the definitions and annotation guidelines, and how the annotation was performed. We give quantitative results as well as measures of inter-rater agreement. Inter-rater agreement is calculated using Cohen's Kappa (K) [53]. It is a statistical measure of the extent of agreement between two raters for categorical annotations which takes into account chance agreement. It is expressed as

$$\mathrm{K} = \frac{P(A) - P(E)}{1 - P(E)}$$

where $P(A)$ is the observed agreement between raters and $P(E)$ is the probability of chance agreement, which depends among other things on the number of categories. At $\mathrm{K} = 1$ agreement is perfect and $\mathrm{K} = 0$ means no better than chance agreement. Negative values are also possible. A value above 0.8 is usually considered very good and above 0.6 is good [37]. All annotations were carried out using the annotation tool MMAX2 [145], which includes an implementation of the K statistic.

5.1.1 Task-level grounding actions

In Chapter 4 we proposed a model for task-level grounding along with a set of action categories. The categories were inspired by Traum's Grounding Acts model [191], and capture the different actions that students and tutors can perform in order to reach mutual belief about the student's understanding of mathematical content. We will now use these categories and their definitions as the basis of an annotation schema for TLGAs. The goal of this annotation is twofold: First, our presentation in the previous chapter was illustrated with examples, but we would now like to analyse the frequency of occurrence. Second, we plan to use the corpus as a data set for supervised learning, more specifically, we plan to learn to predict the tutor's TLGAs from a dialogue context representation which includes the previously performed TLGAs. For this task the annotated corpus will serve as a labelled data set.

This is a one-class annotation task. The markables are the turns in the corpus, of which there are 2011 (mean per dialogue = 54.4, sd = 20.1). We introduce two further labels to the original set of TLGAs which are intended to describe possible combinations of actions at this level. We use *AcceptReqEv* and *RejectReqEv* to capture the combination of *Accept* with *ReqEv* and *Reject*

TLGA	Definition
Propose	This action proposes or augments a contribution to the current solution.
Accept	This action is performed by the tutor to take the solution step which is currently under discussion on board for the current solution, when the tutor is satisfied about the student's understanding.
Reject	Like *Accept*, except that the solution step under discussion was not acceptable and should not become part of the solution.
Abandon	This action can be performed by either student or tutor in order to abandon the step which is currently being discussed.
ReqEv	This action is performed by the tutor to try to find out whether the student really knows how to perform the step that he has just performed, in other words, whether he really understands it. The tutor may specifically ask for a type of evidence.
SuppEv	This action is the offer of evidence of understanding as requested by *ReqEv*
Other	This label is used for actions which have a function in the discussion and acceptance of solution steps, but whose function is not covered by the labels listed above.
None	This label should be given to any utterance which does not contribute to task-level grounding.

Table 5.1: Definition of categories for TLGA annotation

with *ReqEv*, respectively. Their introduction is motivated by utterances such as (5.5), which is simultaneously an acceptance and a request for evidence of understanding. We also added a *None* type for utterances which have no relation to the process of task-level grounding, and an *Other* act for utterances which do have such a relation, but which is not captured by the existing categories.

(5.5) **T28:** Why does this (correct) relationship hold?

Methodology

We used the definitions from Chapter 4 as the basis of the annotation guidelines. They explain the scenario of dialogue-based tutoring and outline the concept of task-level grounding. Each TLGA definition is listed along with examples. The definitions given in the guidelines are paraphrased in Table 5.1. The annotator is instructed to follow a decision tree to choose the right TLGA for a turn, and is encouraged to keep in mind the previous utterance in the dialogue when interpreting the intention of the next utterance. The guidelines also stress the fact that TLGAs may be implicitly realised, for instance an expression of encouragement can be an implicit acceptance of the latest proof step. The decision trees for the student and tutor utterances are shown in Figures 5.1 and 5.2 respectively. In both figures the annotator must first consider

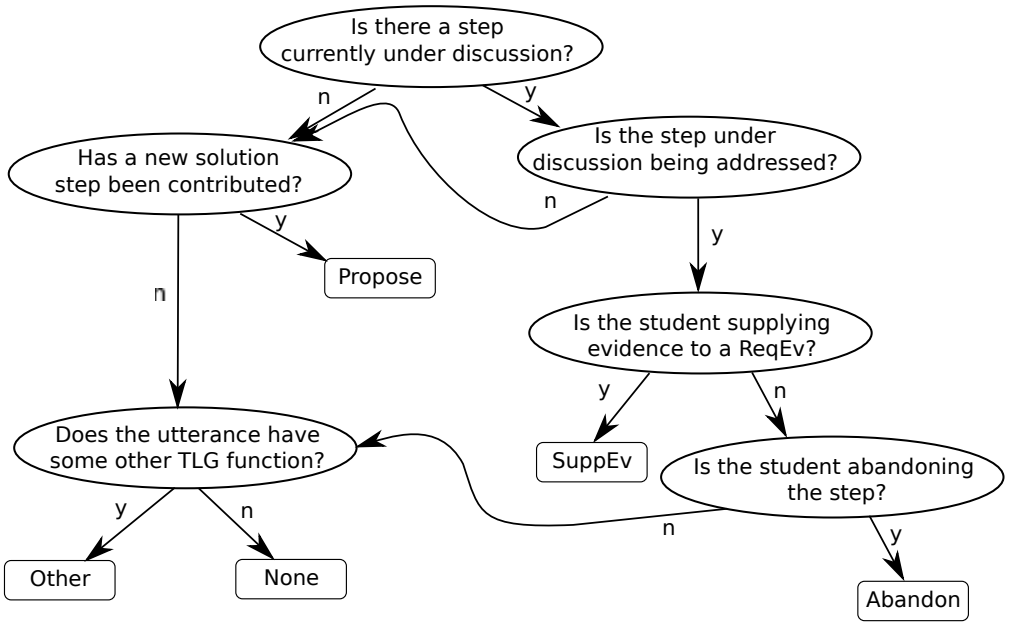

Figure 5.1: Annotation decision tree for student utterances

whether some step was already under discussion before the current utterance was performed. If there is such a step, then a solution step discussion is continuing, and the annotator must decide whether that step is being addressed by the current utterance. If not, the current utterance may be *Propose*, *Other* or *None*. If the step is being addressed, then the annotation will be one of the other moves according to their definitions. For utterances labelled as *Other* the annotator was asked to add a comment describing the case. Following these guidelines, the full corpus was annotated by this author.

To verify the validity of the schema, a validation set of one dialogue session containing 46 turns was annotated by an independent annotator, an undergraduate student of computational linguistics who had also taken lectures in pragmatics. He was familiar with the data in the corpus but not with the concepts of task-level grounding or its annotation. The dialogue was chosen at random from those which had at least one occurrence of a *ReqEv* act. This restriction was made in order to find a dialogue in which it was likely that a variety of TLGAs appear.

Results

The K value for the validation set of 46 turns is 0.51, which is below the threshold for tentative agreement [37]. In examining the parallel annotations the problem seems to concern the *Propose* and *None* categories. Rater 1 (this author) categorised 14 utterances as proposals, 13 of which were also categorised as proposals by rater 2. Rater 2 however categorised a further 9 utterances

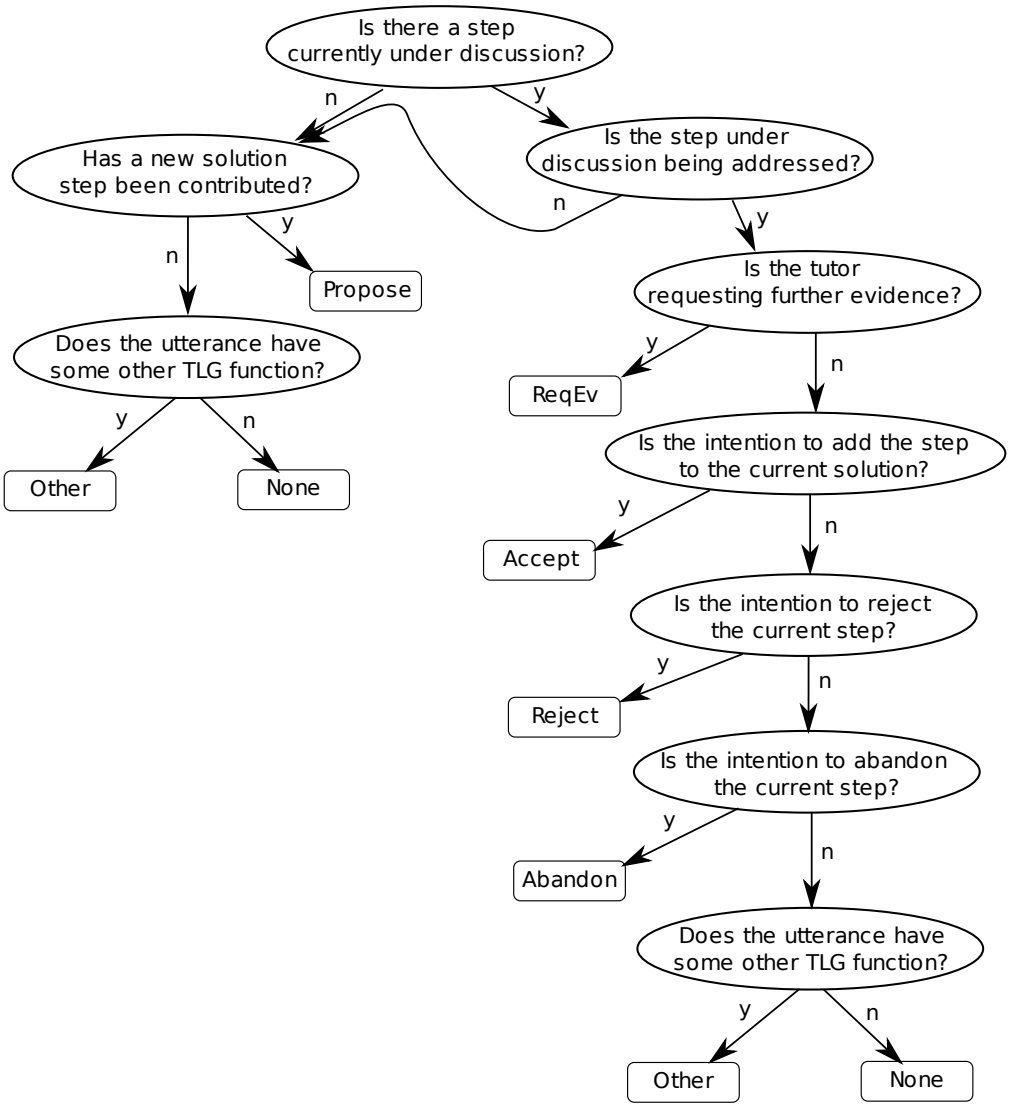

Figure 5.2: Annotation decision tree for tutor utterances

as proposals which were categorised as *None* by rater 1. This difference had a strong effect on the overall K value because propose is by far the most common category. We list the by-class K values in Table 5.2. For *Propose* and *None* these were 0.54 and 0.13 respectively, and the cause of this seems to lie with content statements by the tutor. Where rater 1 considered content statements by the tutor (for instance the statement of the task, statements of concepts, referrals to study material) as not having any contribution to task-level grounding, rater 2 classed these as *Propose*. The remaining classes either show a good or very good K value (*Accept*, *Reject* and *AcceptReqEv*) or occurred very seldom.

The distribution of TLGAs is given in Table 5.3. We find a very small number of *Other* acts by both student and tutor, which indicates that the available categories are covering the kinds of actions that contribute to task-level grounding. 584 of 2011 turns (29%) have no TLG role. These include for in-

TLGA	Total annotations	K
Propose	36	0.54
Accept	18	0.86
Reject	6	0.64
Abandon	0	n/a
Reqev	0	n/a
Suppev	4	0.47
AcceptReqEv	6	0.64
RejectReqEv	1	-0.01
Other	2	-0.02
None	19	0.13

Table 5.2: K by class, validation set

TLGA	Student	Tutor
Propose	693	1
Accept	0	516
Reject	0	147
Abandon	0	0
ReqEv	0	18
SuppEv	20	0
AcceptReqEv	0	11
RejectReqEv	0	10
Other	3	8
None	210	374

Table 5.3: Distribution of TLGAs

stance posing the theorem, task management, and much of the remediation, such as direct content questions from the student or explanations from the tutor. Although we decided in our model that tutors can also add content to the solution under construction, we only see one *Propose* from the tutor. It may be that contributions from the tutor do occur but have been categorised in this dimension into *None*, and indeed the tutor's role in a tutorial dialogue is such that his contributions are not proposals in the strictest sense, since they do not need to be accepted by the student. Other than the single *Propose* the actions are completely partitioned into *Propose* and *SuppEv* for the student and *Accept*, *Reject*, *ReqEv*, *AcceptReqEv*, *RejectReqEv* for the tutor.

5.1.2 Dialogue moves

The annotation of dialogue moves is intended to encode the general pragmatic function of utterances as well as their function in relation to the task at hand. In our case the annotation we propose in this section has been developed to describe utterances in tutorial dialogues. The set of moves and their definitions are based on the categorisation of action types which we performed in Chapter 3 and on a further analysis of another corpus of tutorial dialogues on mathematical theorem proving. To ensure the taxonomy of moves we propose is generally applicable we will design it as an extension of a widely used dialogue move schema, DAMSL (Dialogue Act Markup in Several Layers) [5]. We will also apply the annotation schema to an excerpt from a third corpus of tutorial dialogues.

DAMSL provides a top level structure for an ontology of dialogue moves, and has as its dimensions forward-looking function, backward-looking function, communicative status (whether the utterance was intelligible and successfully completed) and information level (the semantic content of the ut-

terance). The forward dimension captures utterances which are either assertions, requests or commands. The backward dimension captures utterances which relate to the previous discourse, such as agreeing or disagreeing, addressing questions, signalling the understanding status of previous utterances, or which stand in some information relation to previous utterances. DAMSL uses an information-level dimension to characterise the content of the utterance, which can be related to the task, task management or communication management. Communication management refers to conventional phrases like salutations or references to the communication channel, and is not further specified here. Utterances which contribute to the task at hand are of type task, those which contribute to the task solving process are task management. DAMSL provides for a fine-grained description of dialogue moves but is intended to be refined and extended to account for specific phenomena in a given dialogue genre.

The material presented in this section has been published in Buckley and Wolska [29] and Wolska and Buckley [208].

Methodology

The development and validation of the taxonomy of dialogue moves uses three corpora of tutorial dialogues. We analyse the first corpus which was collected in the context of the DIALOG project, which we will refer to as Corpus 1 [211], as well as the corpus which was introduced in Chapter 3, which we will refer to here as Corpus 2. Corpus 1 was also collected using the Wizard-of-Oz experimental paradigm. Students solved proofs in naive set theory under the guidance of an expert human tutor playing the role of the system. The dialogues were conducted in German using the keyboard and a graphical user interface.

The tutoring strategies in Corpus 1 were restricted according to three experimental conditions: minimal feedback, didactic and socratic. Subjects in the control group (eight subjects) were tutored according to the minimal feedback strategy, in which the tutor's reactions were limited to informing the student of the correctness and completeness of their contributions. In the didactic group (seven subjects) the tutor's strategy focused on disclosing partial solutions to the student in case of lack of progress. In the socratic group (seven subjects) the tutor was encouraged to lead the student toward the solution through hinting. The verbosity of the minimal feedback tutors was limited, while in both other conditions as well as in the second experiment, the subjects and the tutors were unconstrained in terms of the linguistic realisation of their turns. Corpus 1 contains 775 turns (332 student and 443 tutor turns).

Finally as part of the validation we will use an excerpt from the corpus col-

lected by the LeActiveMath project (http://www.leactivemath.org). The LeActiveMath corpus consists of 33 transcripts of tutoring sessions on differentiation, conducted via a chat interface. The tutors were five experienced mathematics instructors and the subjects were 28 first-year mathematics or science undergraduate students, of whom five participated twice. Mathematical expressions were entered using a formula editor, and text and formulas could be interleaved. The corpus contains 1650 utterances, 5447 words and 559 formulas. Further details on this data can be found in Callaway and Moore [34] and Porayska-Pomsta et al. [155].

The development of our annotation schema for dialogue moves proceeded as follows. In order to build the initial taxonomy we first analysed a development set of 18 dialogues containing 299 utterances from Corpus 1. The purpose of this analysis was to ascertain to what extent the DAMSL annotation scheme can be used for tutorial dialogue. As DAMSL is a taxonomy with general applicability we expected the non-genre specific elements to be suitable. We also wanted to identify any features of relevant dialogue moves for tutorial dialogue that were not present in the original taxonomy, motivated by the set of possible actions arising from our analysis in Chapter 3. We identified an initial set of task-level moves, and wrote draft annotation guidelines containing definitions of each.

To test this initial draft of the taxonomy we used it to annotate four dialogues containing 108 utterances taken from both Corpus 1 and Corpus 2 which had not been part of the development set. The annotation was performed independently by this author and a co-developer of the taxonomy. We compared and discussed the differences in this initial test set, after which we both extended the taxonomy and refined some existing category definitions.

In order to test the coverage of the final tutoring taxonomy we randomly selected two subsets of Corpus 1 and Corpus 2 with a total of 64 utterances, which had not been part of the first or second step of development. In choosing this validation set we avoided utterances consisting of formulas only in order to encourage the occurrence of a wider variety of types of actions. The validation set was annotated independently by this author and the co-developer. To investigate whether the taxonomy transfers to data outside of but close to our corpus, we applied it to 74 utterances from the LeActiveMath corpus.

The taxonomy

We now present the resulting taxonomy of dialogue moves. At the general dialogue level we follow the DAMSL taxonomy and categorise the functions of utterances according to their relationship with the previous dialogue and their effect on the dialogue to follow. For these functions we use the forward

and backward dimensions, respectively. In general, we try to accommodate the DAMSL categories in order to build as much as possible on existing generally accepted work on dialogue moves. The main differences between DAMSL and our categorisation within these two dimensions are the following: We combine DAMSL's Action-directive and Info-request in a higher-level category of Requests, and in place of DAMSL's Answer, we introduce a more general Address category in the backward dimension with subcategories Answer, Deflect, and Neutral, where Deflect accounts for avoiding answering and Neutral refers to those utterances which simply address a previous information request without answering it or a previous action directive without acceding to it. The remaining DAMSL categories were left unchanged.

We further specify the task and task management dimensions in order to enrich the DAMSL taxonomy to cover tutorial dialogue. Utterances in the task category have the function of altering the state of the task solution, for instance by performing a step in the solution, or talking about the task solution without altering it, for instance making statements about previously performed steps. We divide the task related actions into those which address the task directly and those which address the solution construction process, and capture these in the task and task management categories, respectively.

In the previous chapters we have identified the solution step as the building block of tasks in tutorial dialogue in formal domains. The notion of what is a contribution in the given dialogue genre, what task-level functions are relevant and what types of task-level contributions there are, is determined by both the type of the dialogue and participants' goals. As mentioned above, in tutoring dialogues, the task-level contributions refer to the solution to a posed problem. The participants' roles are those of a student and a tutor. The student's goal is to solve a problem (e.g. prove or compute), while the tutor's goal is to teach (e.g. guide towards a solution by giving hints). In other dialogue genres, the participants' roles, their goals, and so the definition of a contribution are different and depend on the dialogue purpose. In problem solving dialogues, exemplified by Heeman and Allen [95] or Skuplik [179], the participants' roles are those of planning agents and their goals are to construct a plan for the problem at hand. The main dialogue contribution is thus a step in a plan. In information seeking dialogues (such as time-table enquiries), the participants' roles are those of information "seeker" and "giver" and the goals are to find out and provide information, respectively. The contributions focus on enquiry components.

In Table 5.4 we present the top level taxonomy with explanations of the labels and examples. The Task category has three subcategories, which will be detailed further below: Contribute domain content covers utterances which

Label	Explanation	Example
Forward Dimension		
Assert	makes a claim about the world	"It holds that *P*"
Reassert	repeat a claim about the world	"It holds that *P*"
Request	introduces an obligation to answer	
Action-directive	the obligation is that an action is performed	"Please show the following "
Info-request	request for a piece of information	"What is the definition of...?"
Open-option	suggestion of future action without obligation	"You could do . . . "
Backward Dimension		
Agreement	acceptance or rejection of propositions	
Accept	accepts a proposal	"Ok, that's right"
Reject	rejects a proposal	"that's incorrect"
Address	responses to requests	
Answer	answers a previously posed info-request	"yes" or "no"
Deflect	shows inability or unwillingness to answer	"I can't answer that"
Neutral	addresses without answering or deflecting	"Why do you ask?"
Information relation	relation to an antecedent utterance	
Understanding related	refers to problems understanding the speaker	
request clarification	asks to clarify a previous utterance	"what do you mean by X?"
request rephrase	asks for a repeat/rephrase of an utterance	"could you repeat that?"
signal non-understanding	catch-all for understanding problems	"pardon?"
Information-level Dimension		
Communication management	maintaining communication/contact	"Hello!"
Task	performing the task	
Contribute domain content	refers to content for the current solution	"let $x \in A$"
Address domain content	Discuss a performed step	"Good idea!"
Request	ask for help or information	"What's the next step?"
Task management	addressing the task-solving process	
Start task	starts the solution construction process	"please prove $P = Q$"
Finish task	indicates end of solution construction	"I'm done", "q.e.d"
Restart task	indicates solution being started again	"start again"
Give-up task	abandons the current solution attempt	"I give up"
Task solution status	references to solution progress	"Your solution's not finished"
Check solution adoption	Check is solution is acceptable	"So shall we do that?"
Other		

Table 5.4: The full taxonomy, with explanations and examples

bring new domain content into the dialogue, such as new or continued solution steps. Address domain content labels utterances which talk about previously introduced solution steps, for instance evaluations. Request covers a number of task utterances in which speakers ask for or about concepts in the domain. The Task management category contains dialogue moves which start, finish, restart or give up the task, as well as moves which refer to the status of the task, for instance whether it is complete or not.

Table 5.5 presents the details of the task category of the taxonomy. The dialogue moves listed here encode the actions which our qualitative analysis in Chapter 3 highlighted. Domain content is realised by solution steps and strategies. New steps can be performed and existing steps can be augmented with missing elements. In addition, the category Provide domain content is used for domain content which does not perform a solution step, such as references to objects which are part of the general state of affairs. After having been performed, steps can be addressed and discussed, for instances they can be evaluated as being correct or incorrect. Further information about a step

Dialogue move
Contribute domain content
Perform step
new solution step
solution step augmentation
Provide domain content
State of Affairs
State strategy
state strategy
state future step
Address domain content
Evaluate step
correct
incorrect
elicit further step information
hint
Request
req explanation
concept def
symbol
req worked example
req domain content
step
solution strategy
step augmentation
req reformulate

Table 5.5: Dialogue moves in the task dimension of the taxonomy

can be elicited or a hint can be given. There are many types of requests in tutorial dialogue; our taxonomy lists those which occur in the data, but this list is not necessarily exhaustive. Both students and tutors can request explanations of domain concepts or worked examples. Students can also request that the tutor supply steps or strategies as a source of help.

Results

Table 5.6 shows the inter-rater agreement values for the annotation of the validation set described above which was taken from Corpus 1 and Corpus 2. In the annotation we did not consider the category information-relation because no definition is given by the original DAMSL taxonomy. These results can be considered very good for the forward dimension and the task management category, good for the task category, and low for the backward dimension. Among the categories with the lowest agreement were Neutral at 0.11 and Step-augmentation at 0.37. In this preliminary evaluation our strategy was

Dimension	κ
Forward	0.87
Backward	0.47
Task	0.75
Task Management	0.91

Table 5.6: Inter-rater agreement values by dimension

Utterance	Forward	Backward	Info-level
S: It holds that $P(C \cup (A \cap B)) \subseteq P(C) \cup P(A \cap B)$	assert		solution-step:new
T: Really?	info-request	reject	signal-incorrect
S: no it's not,		answer	
S: the other way around	assert		solution-step:new
T: that's right at last	assert	accept	signal-correct
S: $R \circ S := \{(x,y) \mid \exists z(z \in M \wedge \ldots$	assert		task:contr:perform:new soln step
T: That's right!	assert	accept	task:addr:eval:correct
S: now i want the inverse of that	assert		task:contr:strategy:state-future-step
T: yes?		neutral	task:addr:hint
S: $(R \circ S)^{-1}$	assert	neutral	task:contr:perform:new soln step
T: = ?	info-request	request-clar	task:req:explanation
S: How will the system answer?	info-request	neutral	
T: What's the question?	info-request	neutral	
S: Can the system conclude $(R \circ S)^{-1}$ from $R \circ S$	info-request	neutral	
T: yes	assert	answer	
T: But try it yourself!	action-dir		hint

Figure 5.3: Annotated examples from Corpus 1 and Corpus 2

not to use a category "other" for utterances which did not appear to belong to any existing category, but rather to try to fit the annotation to the categories as they are.

For the second part of our validation we now give examples of annotated data from each of the three datasets introduced above. The examples in Figure 5.3 illustrate some of the types of problematic utterances which the corpora contain. For instance the utterance "Yes?" is a question and could appear to be an Information request, but in fact acts more like a prompt to continue, for which we had no category. Similarly the functions of the questions in sequence in the second example are difficult to abstract. We have tagged these as Neutral, since they discharge the obligations introduced by the questions before them, but the link between consecutive interrogative utterances is elusive.

In Figure 5.4 we show the information level annotation for an example from the LeActiveMath corpus on differentiation. In annotating the dialogues on differentiation with this taxonomy, we find they exhibit largely the same phenomena as the theorem proving data, that is, the same dialogue moves seem to

Utterance	Info-level
T: try this: $y = 1/(6x^2 - 3x + 1)$	taskmng:start
S: $dy/dx = -12x + 3/(6x^2 - 3x + 1)$	task:contr:perform:new soln step
T: Bracket problem again	task:addr:eval:incorrect
...	
S: $dy/dx = (-12x + 3)/(6x^2 - 3x + 1)$	task:contr:perform:augment step
T: Good	task:addr:eval:correct
and now what about the power of $(6x^2 - 3x + 1)$?	task:req:req augment step
...	
T: yes well done	task:addr:eval:correct
T: time to stop	taskmng:finish

Figure 5.4: Annotated example from the LeActiveMath corpus

capture the data well. There are some differences, such as the more common occurrence of step augmentations, possibly due to the computational nature of the task, and the more common occurrence of task management utterances, probably because of the shorter exercise length.

As in the case of *Accept* and *ReqEv* at the TLGA level, in annotating the dialogue moves we have found that tutors typically perform utterances which contribute to many different goals — for instance they can simultaneously reject proposed solution steps while giving hints on how to continue in the task. The purpose of multidimensional dialogue move taxonomies is to handle this very multifunctionality, and while this is successful to a point, conflicts in the annotation experiment have highlighted some dual functions within the same category. For instance, utterances simultaneously rejecting steps and requesting explanations of the errors in the steps were found a number of times.

Some possible categories have emerged that may need to be added to the current taxonomy to allow it to cover tutorial dialogue more completely. As discussed above, a prompt type in the forward dimension seems necessary to handle cases in which the tutor does not want to give any specific feedback, but would like the student to continue solving the exercise. Such an action could be part of a pedagogical strategy which lets students learn by continuing after mistakes in order to come to contradictory conclusions. We would also foresee a backward category which corrects a previous utterance. Some similar categories are proposed by Tsovaltzi and Karagjosova [195], and may be taken up.

With a view to our second goal in preparing the annotated corpus, using it to support a series of supervised learning experiments, we have decided to not yet proceed and annotate the full corpus with dialogue moves. The reason for this that we have found a strong correlation between the TLGA annotations

and the task-level dialogue moves. Nearly all utterances which have no annotation at the task-level are also annotated with the TLGA *None*. Utterances which are labelled as "new solution step" are always TLGA *Propose*, similarly "signal correct" and *Accept*, "signal incorrect" and *Reject* as well as "request for augmentation" and *SuppEv*. In the forward dimension of the dialogue move annotation nearly all utterances are assertions, and those which are not are either TLGA *None* or *Other*. This mean that in the context of supervised learning, having the task-level dialogue move annotation adds very little to the discriminatory power of a data point.

5.1.3 Mathematical content

The mathematical level annotation describes the content and properties of the proof steps that the students contribute and uses in its encoding the mathematical concepts which we have introduced in the previous chapters. In Section 4.2.1 we presented the Omega system and its functionality, which includes the ability to verify proof steps of the sort that occur in our corpus. Omega constructs a formal proof object which can be inspected and from which information describing the step can be read, such as the rules that were applied, their instantiations or the premises that were used. As we however explained, for technical reasons we were not able to use the Omega system to process the corpus automatically. Our goal then with this annotation is to approximate the information that such a domain reasoner would have been able to provide by performing a manual annotation of mathematical content. The role of the annotator is to read and to try to follow the proof that the student offered. At each proof step the annotator has to reconstruct the components of the step, ideally recognising the same step as the student intended.

Three kinds of attributes will be annotated. We will annotate the rules that were applied to derive the step, whether those the rules were applied correctly, and how explicitly the proof step was presented, in other words, whether the rules or premises were stated by name or not. To this we add the annotations of correctness, granularity and relevance which were added by the tutors during the corpus collection experiment. Overall the attribute set we use is intended to reflect the information which it is reasonable to expect that a mathematical domain reasoner can compute automatically. This way any model we derive from the annotation will not make unrealistic assumptions about the information it must receive from other parts of a hypothetical system.

Methodology

The markables at the maths level were chosen based on the TLGA annotation. Any turn which is annotated at the TLGA level as either *Propose* or *SuppEv* is a contribution of a proof step, so therefore it should be annotated for mathematical content. There are a total of 714 maths markables in the corpus (mean number of maths markables per dialogue session = 19.3, sd = 9.9).

The initial attribute set contained the following attributes. We used one binary attribute for each of the rules in our standard set from Table 3.1 on page 37, which in turn was drawn from the study material, as presented in Chapter 3. These binary attributes encode the fact that the rule was used in the current step or not. Similarly we have one such attribute for each of the three identities which are available to the student after they have been proved. A three-way attribute represents whether the rules in the step were applied correctly. The three cases are: the right rule was applied correctly; the right rule was applied wrongly; or the wrong rule was applied. Two binary attributes encode the explicitness of the expression of the step, one for when at least one rule was stated and one for when at least one premise was stated.

The meanings of each of these attributes were documented in a set of annotation guidelines. The guidelines introduce the scenario and present the set of rules which form the basis of the annotation, before defining each of the attributes. The annotator is encouraged to "try to understand what solution the student is constructing for the current proof". The motivation was to have the annotator mentally building the same proof as the student so that hopefully the same rule applications are recognised.

The annotation was carried out by an external annotator. He was an undergraduate student of mathematics and computational linguistics, and had had a number of lectures in mathematics and mathematical logic. He was paid for his work. After an initial brief discussion of the task he was given the annotation guidelines and a single dialogue session to annotate. This subset of the data consisted of one complete dialogue session of 60 turns containing 27 markables at the maths level. It was annotated independently by the annotator and this author.

This annotation of the initial data set was then discussed with the annotator in order to align on the interpretation of the annotation guidelines and to further develop the attribute set. As a result of this discussion a number of changes were made to the annotation schema. The attributes encoding the application of rules were changed from binary to numeric to encode the number of times a rule was used in the step. The goal here was to be able to reconstruct a measure of step size from the total number of rule applications from the annotation. Their range was set at $[0,5]$, because we had yet to observe any

steps in the data with more than five applications of one rule. Unused rules are annotated with zero applications. For each rule that was applied, we added an attribute stating whether the rule was applied correctly. This superseded the attribute for wrong application of a rule above, because the fact that some rule was wrongly applied can be read off rule by rule from the annotations of correct applications.

It was decided that parallel applications of the same rule should be counted twice, even if conceptually the same action has been performed twice, for example when two structurally similar operands are both rewritten the same way in a single step. An attribute was added for proof steps which do not apply any rules, but which do contain mathematical content. Since such steps are typically introductions of simplifying notation, such as "We refer to $(R \cup S) \circ T$ as F", or assumptions of facts, such as "let a be element of A", this attribute is called "declaration". Having reached agreement on the annotation schema, we changed the annotations of the first dialogue to reflect our decisions, and the annotator annotated the rest of the corpus.

Results

To test interrater agreement we chose at random a dialogue (44 turns, 19 maths markables) which had close to the mean number of maths markables and which was not the dialogue in the initial annotation data set. It was annotated independently by this author. The design of this annotation schema is such that it is not possible to calculate an overall K value. The reason is that the annotation of markables at the maths level is not a single-decision categorial annotation, as was the case with TLGA, but rather markables are annotated with values across multiple features. Proof steps typically contain many rules and vary across multiple dimensions such as explicitness, correctness etc.

There has been work on calculating agreement values for this kind of annotation task [115], but we will take a simpler approach and examine the K values per attribute. We can calculate K for each individual attribute, since these are categorical within themselves. The results are listed in Table 5.7. The attributes for which the there is no K value are those whose rules did not occur in the test annotation data set. These are annotated by both raters with all zeros, from which it is not possible to calculate a meaningful K value. The two attributes with the negative K values, quant and set_prop, received one and two non-zero ratings, respectively, from one rater. This means that the confusion matrix for these rules is heavily weighted towards the $(0, 0)$ cell. The low value is caused by the fact that although the observed agreement is high, so is the probability of chance agreement. Of the four rules which were used more often, composition, inverse, and identity1 exhibit K values which may be con-

Attribute	Range	K
prop	[0,5]	1.00
quant	[0,5]	-0.03
subset	[0,5]	n/a
psubset	[0,5]	n/a
exten	[0,5]	n/a
set_prop	[0,5]	-0.05
union	[0,5]	0.48
intersection	[0,5]	n/a
composition	[0,5]	0.87
inverse	[0,5]	0.80
identity1	[0,5]	0.88
identity2	[0,5]	0.62
identity3	[0,5]	n/a
declaration	[0,5]	1.00
rule_stated	[0,1]	n/a
premise_stated	[0,1]	n/a

Table 5.7: Agreement per attribute

Rule	Occ.	Wrong appl.
prop	49	6
quant	20	1
subset	3	0
psubset	1	0
exten	15	0
set_prop	8	0
union	122	23
intersection	0	0
composition	265	48
inverse	179	29
identity1	66	9
identity2	27	4
identity3	1	0
declaration	96	1

Table 5.8: Occurrences of rules

Step size	0	1	2	3	4	5	6	7	8	...	12
No. of steps	41	444	122	50	19	8	5	2	4	0	1

Table 5.9: Distribution of step sizes

sidered "very good" agreement, and identity2 exhibits "good" agreement [37]. We note that the choice of range of [0,5] may unfairly lower the K values because although it is a range of six possible categories, most rule applications do not occur so often. In the validation dialogue for instance, all but one proof step contained less than four applications of any one step.

We can now present a quantitative analysis of the maths annotation level. The occurrences of each mathematical rule are given in Table 5.8. Here we see that composition of relations, relation inverse, and the definition of union are the most frequently used rules. This reflects the fact that relations are the goal of the tutoring session and that these three operators occur most often in the theorems which are to solved. These three are also the most frequently wrongly applied rules, and their ratios of wrong to total applications are very similar: 18.1%, 16.2% and 18.1% for composition, inverse and union respectively.

The mean step size, calculated as the sum of the number of applications of all rules in the step, is 1.5 (sd = 1.2). The distribution of step sizes in Table 5.9 shows that the majority (62.2%) of steps contained only one application of one rule, but that a number of steps used up to eight rule applications. From the annotation of wrongly applied rules we find that 134 steps have at least one

Correctness	correct 532	incorrect 80	partially correct 47	n/a 51
Granularity	appropriate 579	too fine 17	too coarse 55	n/a 59
Relevance	relevant 605	partially rel. 39	irrelevant 8	n/a 58

Table 5.10: Distribution of correctness, granularity and relevance annotations

wrongly applied rule, which is 18.8% of cases. This is close to the 127 steps that were annotated as being either incorrect (80) or partially correct (47) by the wizards during data collection. In terms of explicitness, 72 steps name at least one rule which was applied, but 18 dialogues have no steps at all which name a rule. For the premises, 18 steps include at least one premise explicitly, but 30 of the 37 dialogues contain no such steps. These results reflect the "formulas only" style preferred by many students. Finally we give the distribution of correctness, granularity and relevance annotations in Table 5.10, which shows that the vast majority (74.5%) of steps were considered correct. Similarly most steps (81%) were of appropriate granularity, and most (84.7%) were relevant to the exercise. The high number of "n/a" annotations across all three attributes is due to annotations of utterances which did not contain proof steps.

5.2 Summary and discussion

In this chapter we have presented an annotation in three levels of the corpus of tutorial dialogues which was introduced in Chapter 3. The first level was the annotation schema for task-level grounding actions, which was derived from the concepts introduced by our computational model of task-level grounding in the previous chapter. We found an inter-rater agreement K value of only 0.51, however we were able to point to some systematic errors which may have caused this.

The second level of annotation was dialogue moves. Here we developed a broad taxonomy of dialogue moves for tutorial dialogue based on the categorisation of actions from Chapter 3. It is an adaptation and specialisation of the DAMSL dialogue move taxonomy and includes forward, backward and task dimensions. Here we achieved good K values, although the validation annotation was not carried out by an external annotator. Finally in this chapter we presented the annotation of mathematical content, such as rules and the correctness of their application, which was intended to approximate the function

of a general-purpose mathematical domain reasoner. In this validation stage we observed mixed K values depending on the frequency of appearance of the rules. The overall frequency of application of the rules and the match between the annotator's and the wizards' annotations of incorrectness also indicate the validity of this annotation level.

Any annotation task with this type of data will find some utterances which are difficult to categorise. This often has to do with the terseness of the students' expressions—for instance a proposal of a new step and a statement of a definition or identity are not always distinguishable in the absence of linguistic tokens. In general backward-looking functions have a stronger negative effect on inter-rater agreement because it is often not clear what the relationship of an utterance to its previous discourse is. This is shown by the low K values for the backward dimension, which at 0.47 is much lower than the other dimensions, as well as in the score of 0.37 for the Solution-step augmentation category in the task dimension, which is heavily dependent on previous context. This result may even point to a general characteristic of tutorial dialogue which makes computational modelling challenging.

Further independent annotation with larger test sets is certainly necessary for the each of the three schemata. A further iteration of discussion and adaption of the annotation guidelines would no doubt improve the K values for TLGA. An external annotation of the three dimensions of dialogue moves would also surely lead to further development of the dialogue move definitions and would allow us to present stronger evidence for the validity of the taxonomy. Following this the rest of the corpus could be annotated by external annotators.

The result of this chapter, apart from the development of the annotation schemata, is that the corpus has now been annotated with TLGAs and mathematical content. This annotation will allow us to create a data set for supervised learning from the corpus. We will use the corpus to learn to predict the tutor's choice of TLGA in a given dialogue context. This series of experiments is presented in the next chapter.

6

Predicting the tutor's task-level grounding actions

In this chapter we add a selection model for the tutor's task-level grounding actions to the dialogue model described in Chapter 4. It answers the question of what the tutor should do in a given situation. Faced with a certain input from the student in a certain dialogue context, which pedagogically appropriate response should the tutor give? Our proposal is to infer a model of this decision using machine learning on the data in the corpus, taking advantage of the fact that we have already performed a detailed annotation, as presented in Chapter 5. Machine learning has been used extensively and successfully in dialogue systems research in the past number of years. Our corpus and the selection task which we wish to solve lend themselves to this approach: our corpus annotation can be translated into a feature vector representation of dialogue contexts, and we have to choose between a relatively small number of classes, namely the tutor's TLGAs. This is a typical classification problem. Such a classifier, which maps from dialogue contexts to dialogue actions, slots neatly into a dialogue system in the place of a selection algorithm.

The goal of the series of experiments which we present in this chapter is twofold: First, we want to learn a classifier which we can use together with the implementation of the dialogue model in order to predict which TLGA should be performed in which situation. Second, we want to use this learning setup to investigate a set of hypotheses about the utility of maintaining a sophisticated model of common ground in a tutorial dialogue model. We will investigate the effects of varying the feature set used in the classifier, the size and distribution

of the data set, and we will consider a way to improve performance by splitting the classification problem into two simpler problems. Overall we hope that the experiments will show that the information maintained by the dialogue model presented in Chapter 4 provides the features to train a good classifier for the choice of TLGA.

This chapter is structured as follows. Section 6.1 introduces the basic concepts of machine learning along with the machine learning algorithms we will primarily use. We also review some previous applications of machine learning in dialogue research. In Section 6.2 we list the hypotheses which will be investigated in the remainder of the chapter. Section 6.3 presents how the data set has been derived from the existing annotations of the corpus. Our methodology and a description of each experiment in turn are detailed in Section 6.4. We finally summarise the results and discuss the work in Section 6.5.

6.1 Machine learning and dialogue systems research

Machine learning is the process of inducing from a set of data a function that models that data. If the set of data is a set of examples of a particular task, then the learned function is a solution to that task. In this chapter we will apply a technique called supervised learning, or classification, and in this section we first briefly introduce the relevant concepts, following [6]. A supervised learning problem is characterised by a set of examples $(\overline{x_i}, c_i)$, where $\overline{x_i} \in \overline{X}$ is a vector of features whose values describe the example, and $c_i \in C$ is the label or class to which the example belongs. $\overline{X}$ is the feature space of the problem and C is the set of possible classes. The machine learning process induces a function $f : \overline{X} \rightarrow C$ which classifies new examples $\overline{x}$ into some class c'. The example data are known as instances, and the induced function f is known as a classifier.

In supervised learning all instances are labelled with a class, in contrast with unsupervised learning, in which data is not labelled. Machine learning is a suitable tool for problems for which it is difficult to express a solution programmatically, for instance because the data set is too large for manual analysis, or because the knowledge required for the solution is otherwise implicitly applied by humans. The work presented in this chapter is an example of the second situation: since we can not ask tutors to operationalise their behaviour in dialogues with students, we will use machine learning to compute a function for the same task from examples of tutors' behaviour.

The learning algorithm which we will use extensively in this work is naive

Bayes [107]. It computes a probabilistic classifier based on the assumption of independent features, that is, that each feature contributes independently to the likelihood that an instance is in a particular class. We train a model for the a-posteriori probability $P(c|x_1 \dots x_n)$ of a class given a feature vector. With large numbers of features and large feature value ranges this computation can become infeasible. We therefore rewrite the probability using Bayes' theorem:

$$P(c|x_1 \dots x_n) = \frac{P(c)P(x_1 \dots x_n|c)}{P(x_1 \dots x_n)}$$

$P(c)$ is the prior probability of the class c, which is the distribution of the class in the data set. $P(x_1 \dots x_n|c)$ is the probability of the features given the class, which is the information that is learned from a data set. Given the assumption of the independence of features, this probability can be rewritten as

$$P(x_1|c)P(x_2|c)P(x_3|c) \dots P(x_n, c)$$

using the definition of conditional probability. $P(x_1 \dots x_n)$ is the evidence, or probability of the feature set, which is constant for a data set and can therefore be dropped. We can calculate the a-posteriori probability for each class $c \in C$, and from this derive a naive Bayes classifier f, which selects the class with the highest probability for a given set of feature values:

$$f(\overline{x}) = \arg\max_{c \in C} P(c)P(\overline{x}|c)$$

We will also compare the naive Bayes learner to a decision tree learner (for example [158]), which is a rule based approach to classification. A decision tree is a hierarchical data structure consisting of internal decision nodes and terminal classification leaves. To classify a previously unseen instance, a decision tree classifier begins at the root of tree and traverses a path to a leaf. At each decision node the value of a feature is tested, and the tree branches into as many subtrees as that feature has possible values. This process effectively repeatedly partitions the feature space into ever smaller regions. The leaf which is reached at the end of the path represents a region containing instances all of one class, and that class is the classifier's prediction for the current unseen instance.

A decision tree is constructed by recursively choosing a feature and partitioning the data set on that feature. In order to prefer smaller trees and shorter paths, the feature chosen is the one with the highest information gain, which measures how strongly a feature discriminates the instances. If any partition contains only instances of one class, then a leaf node is created and labelled

with this class. The remaining partitions are recursively partitioned on the feature which is most discriminative for the instances it contains, and so on. Decision trees are attractive in machine learning because they reduce the classification task to a small number of decisions in relation to the number of features, because the trees themselves are easily interpretable, and because they can be readily translated into an if-then representation for use in an application.

There are a number of applications of machine learning in the field of dialogue systems research. As is the case in this work, a data set is typically drawn from a corpus of human-human dialogues, and its instances represent utterances or dialogue contexts from the corpus. The feature vector describing the instances can contain for example information about the lexical items the utterance contains, its length, punctuation, or the dialogue act is has been assigned by an annotator. Higher level features are also possible, such as a measure of user frustration, or information about prosody in a spoken dialogue setting. The prediction tasks can include tagging utterances with dialogue acts, with or without features referring to their context, predicting dialogue acts in a given dialogue context, or predicting entire sequences of dialogue acts. In general the high cost of data collection means that machine learning work in dialogue often suffers from data sparsity, and for dialogue strategies there is always a danger that the model will overfit the data if the strategy space is large and the data set small in comparison.

The MapTask corpus [38, 189], whose annotation of 128 dialogues includes 12 dialogue acts, has been used as a basis for tagging utterances with dialogue acts by Louwerse and Crossley [132], who achieve a 58.08% accuracy using an word n-gram model. Stolcke et al. [185] combine a language model over dialogue acts with utterance features such prosody and lexical features. They achieve 71% accuracy on dialogue act tagging using the Switchboard corpus [82]. Using the BNC, Fernández et al. [73] classify *wh*-phrases into seven classes using 10 features representing lexical and syntactic aspects of the current and antecedent utterances. They achieve an accuracy of 90.32%.

The work presented here is an example of dialogue strategy learning, that is, given a dialogue context a classifier should predict what dialogue move should be performed next. In the VERBMOBIL system, n-gram dialogue act probabilities were used to compute the most likely next dialogue act [161], resulting in prediction accuracies of between 40% and 72%. Boyer et al. [24] use hidden Markov models for dialogue act prediction in tutorial dialogue. They use 11 tutoring-specific dialogue act tags, including Assessing Question and both Positive and Negative Content Feedback. Training on a corpus of around 4,800 dialogues, they achieve a maximum prediction accuracy of 57%.

More recent approaches have applied reinforcement learning (RL) [186]. The task is not to choose the correct classification, as in supervised learning, but rather to choose an agent's action in an environment in order to maximise an expected reward. A trial-and-error search through the action strategy space is guided by a reward function which is evaluated from observations of the effects of chosen actions on the environment. Strategies whose actions lead to a higher cumulative reward are preferred. RL has been applied extensively in task oriented dialogue [204, 169, 178, 165], but also to problems in the field of tutorial dialogue. For instance, Chi et al. [47] cast tutorial dialogue as a Markov decision process to evaluate the usefulness of 18 individual dialogue features in choosing whether to elicit content or tell the student that content. Dialogue contexts are mapped to states, the available dialogue moves to actions, and the reward function is the learning gain that the students show after the interaction.

RL is however not suitable in the case of tutorial dialogue because the sources of a reward function which have been used in previous research on RL for general task-oriented dialogue, such as dialogue length, task completion or number of clarification actions [178, 165], do not necessarily correlate with learning effects, which should be the measure of a successful strategy for tutorial dialogue.

6.2 Hypotheses to be investigated

The overall goal of the series of experiments in this chapter is to learn a classifier to predict the tutor's TLGA given a dialogue context. The experiments investigate a number of individual hypotheses about this classification problem, about the data set and about the feature set we derive from the annotations. First we compare four machine learning algorithms to find out which is best suited to this problem. Our second and third experiments are concerned with the class distribution of our data set. We will see that there is a strong skew between the smallest and largest class in the data set, which can lead to lower accuracy on the smaller classes. In order to mitigate this effect we hypothesise that it is possible to reduce the class distribution skew, and thereby improve the overall performance of the classifier, first by downsampling the larger classes, and second by splitting the classification problem into two simpler problems.

The fourth experiment is motivated by our original assumption that tutors use information from the previous discourse to tailor their feedback. We hypothesise that a classifier trained on local features, describing only the current step, will perform less well than a classifier trained on a feature set which addi-

tionally includes global features from the previous discourse. Our fifth experiment will apply standard attribute selection algorithms to investigate whether there may be a distinct feature subset which outperforms even the local and global feature set. Finally we will perform a misclassification analysis to illustrate the nature of the mistakes the classifier makes.

6.3 Preparation of the data set

The data set which we will use for the series of experiments in this chapter is based on the corpus of dialogues we presented in Chapter 3 and its annotation in Chapter 5. It contains TLGA annotations, the characteristics of the solution steps which the students contributed, and shallow annotations such as turn numbers. From the corpus we extract a set of data points which correspond to those dialogue contexts in which the tutor has to choose between task-level grounding actions. In this section we explain the process of choosing the data points and computing their full feature sets.

The features which we have chosen come from three different sources. Some features are taken directly from the annotation of the latest utterance, i.e. the student's input to which the tutor should respond. Some are derived from this annotation. Finally some features are aggregated from the individual utterance annotations over the course of the dialogue. We will refer to the first two kinds of features as *local*, because they pertain only to the last utterance in the dialogue context. Those features which are aggregations of information from whole previous dialogue we will refer to as *global*. The full feature set is the union of the local and global features.

6.3.1 Features from the utterance annotation

Most of the annotations are translated directly into features, and the list of these is given in Table 6.1. The feature <rule> refers to the mathematical concepts that were used in the solution step. It stands for 14 individual features, one for each of the mathematical concepts taken from the students' study material, which we introduced in Chapter 3. As in the annotation, correctness, granularity and relevance may have missing values.

Some of the annotations are not used as features because we have decided that they are not predictive for the classification that we will try to learn. For instance, since we are not trying to predict the idiosyncrasies of the four wizards in the original data collection experiment, we do not include the wizard identification number as a feature. Similarly the student id is not used. In both of these cases we are assuming that the participants acted in a more or less sim-

Feature	Type	Range
timestamp	date and time	
tlga	nominal	*Accept*, *Reject*, *ReqEv*, etc
turnnumber	numeric	1...
correctness	nominal	correct, incorrect, partially correct
granularity	nominal	suitable, too fine-grained, too coarse-grained
relevance	nominal	relevant, irrelevant, partially relevant
rule_stated	boolean	0,1
premise_stated	boolean	0,1
<rule>	boolean	0,1

Table 6.1: Local features taken directly from the corpus annotation

Feature	Type	Range
time_to_reply	numeric	0...
setDefinitionRuleUsedInCurrentStep	boolean	0,1
relationDefinitionRuleUsedInCurrentStep	boolean	0,1
identityRuleUsedInCurrentStep	boolean	0,1
numberOfInterestingRulesInCurrentStep	numeric	0...6
numberOfInterestingRuleApplicationsInCurrentStep	numeric	0...
numberOfUninterestingRulesInCurrentStep	numeric	0...8
numberOfUninterestingRuleApplicationsInCurrentStep	numeric	0...

Table 6.2: Local features computed from the corpus annotation

ilar way, and that any results we find are generalisations over all participants. The speaker annotation is also not used because the speaker at the data points which we will extract is always the student.

From these basic features we can compute further features which are meaningful for the task at hand. The additional local features are listed in Table 6.2. We collate the maths concepts into three groups: setDefinition, relationDefinition, identity. These reflect the split between basic knowledge (setDefinition, those rules which pertain to naive set theory), the knowledge which the student should be learning in the exercises (relationDefinition, the definitions of inverse and composition) and the mathematical concepts which the student must derive before using (identity, the theorems which the students prove and may use in later proofs). If at least one rule in a group has been used in the current step, the corresponding feature is set to 1. A further split is into what we call interesting and uninteresting rules. This refers to whether the rules should be used explicitly by the student or not and whether they are part of the goal of the tutorial session. The interesting rules are the relation definitions, the identities and the definition of set extensionality, the other rules are in the uninteresting set. For these two groups we additionally represent the number of applications made in the current step. The rules in each of these

Feature	Type	Range
propSuccessfullyUsedRules	numeric	0...1
someRuleFromStepPreviouslySuccessfullyUsed	boolean	0,1
allRulesFromStepPreviouslySuccessfullyUsed	boolean	0,1
acceptedConsecutiveContributions	numeric	0...
acceptancesSoFar	numeric	0...
rejectionsSoFar	numeric	0...
proportionOfAcceptances	numeric	0...1
proportionOfRejections	numeric	0...1
numberOfJustifiedSteps	numeric	0...
numberOfReqestedEvidence	numeric	0...

Table 6.3: Global features computed from the corpus annotation

groups are listed in Appendix A.

We omit some original annotations from the list of features because their meaning is better captured by a computed feature. For instance the timestamp annotation is not used, instead we calculate the time taken by the student to reply by comparing the current and previous timestamps. We also remove each of the annotations of the form <rule> because their meaning is captured by the rule groupings.

6.3.2 From utterances to instances

There are two tasks involved in deriving a suitable data set from the annotated corpus. First we must choose which places in each dialogue are decision points for the tutor's choice of TLGA and therefore should become the instances in the data set. If the TLGA of the student's utterance was either *Propose* or *SuppEv* and the tutor's response was not *None*, then an instance should be generated which contains the features of the dialogue context from the beginning of the interaction up to and including the student's utterance. The class of the instance is the TLGA of the tutor's response.

Second we must compute the values of the global features of the instance. Each of the instances in our data set corresponds to a dialogue context rather than an utterance. This is because the decision we are trying to learn is not how the tutor reacts to an utterance, but rather how the tutor acts in a situation. Global features describe the dialogue context up to the student's utterance. They are aggregations of the annotations of individual utterances in the dialogue so far, and as such approximate a dialogue history. The global features we will use are listed in Table 6.3. The features someRuleFromStepPreviouslySuccessfullyUsed and allRulesFromStepPreviouslySuccessfullyUsed are true when some or all, respectively, of the rules which were used in the lat-

Algorithm 1 Data instance creation

```
1: Input: the corpus as a sequence of annotated utterances
2: Initialise an empty DialogueContext dc
3: for all pairs of consecutive utterances (u1,u2) do
4:     Update dc for the effects of u1
5:     if (u1,u2) should be an instance then
6:         Copy dc + u2.tlga to set of instances
7:     end if
8: end for
```

est solution step had previously been used in a step which the tutor accepted. These features capture the case of the student reusing known concepts. The features acceptancesSoFar and rejectionsSoFar count the number of times the tutor has responded to a solution step with *Accept* or *Reject*, respectively. ProportionOfAcceptances and proportionOfRejections express these as a proportion of the total number of acceptances and rejections in the dialogue so far. The number of steps in which a justification was explicitly stated is stored in numberOfJustifiedSteps, and the number of times the tutor has requested evidence of understanding from the tutor is stored in numberOfReqestedEvidence.

We perform both of these tasks in one pass through the annotated corpus using the algorithm listed in Algorithm 1. A DialogueContext object maintains the current values of each of the global features. Each of these features may be updated for the effects of each utterance in the corpus (line 4 of the algorithm). For example an *Accept* performed by the tutor increments the value of acceptancesSoFar. If the current utterance pair should generate an instance according to the test above, then a new instance is created containing the current values of the global features. Computed features such as those in Table 6.2 or proportionOfAcceptances are computed at this point. The class of the instance is the TLGA annotation of the tutor's utterance. The implementation of this algorithm uses the MMAX2 [144] API to extract the annotations from the corpus, and the Weka [93] API to create the data set. The resulting data set is represented in Weka's ARFF format, a plain text file format for instances sharing a common attribute list.

Finally we removed a small number of outliers. Three complete dialogues, corresponding to a total of 11 instances, were removed because they had a very low ratio of instances to total turns, in each case less than 6%. This low ratio means the student was only very seldom contributing to the solution and the rest of the dialogue was made up of off-task remediation, which does not contribute to our model. Two further instances were removed because they had very high values for time_to_reply, namely 1049 and 1409 seconds, whereas

Class	Instances
Accept	505
Reject	141
ReqEv	17
AcceptReqEv	11
RejectReqEv	10
Total	684

Table 6.4: Class distribution of the data set

the other instances of time_to_reply are distributed between 0 and 616 seconds. The resulting data set has a class distribution as shown in Table 6.4. Here we see the strong class skew which we have mentioned previously, and which we will examine more closely in the context of experiments 2 and 3.

6.4 Experiments

In this section we present in detail the experiments aimed at finding a good classifier to predict the tutor's TLGA from the instances in our corpus. We first describe our basic methodology, namely ten-fold cross validation, and the evaluation metrics we will present. We will try to find the most effective combination of settings across a number of dimensions, which will be investigated in turn by the following experiments. First we consider which learning algorithm is most suitable. We then look at two possible solutions to combat the problem of the skewed class distribution. The first is downsampling the data set, the second is splitting the classifier into two simpler problems. We investigate whether the addition of global features improves performance of the classifier overall, and finally we test whether there might be a better automatically selected subset of features from the full feature vector. We discuss the classifier's misclassifications as well as the overall results of the series of experiments.

6.4.1 Methodology

A standard approach to training and validating a classifier such as the one we intend to develop is to hold out a subset of the data set, say 10%, as a test set. The remaining instances, the training set, are used to train the classifier, and its performance is measured by how well it predicts the cases in the test set. Both training and test set are stratified, that is, they are randomly chosen but in such a way as to preserve the same distribution of classes as the complete data set. This approach is however not suitable given the class distribution

of our data set (Table 6.4). Due to stratification, if we were to extract a 10% test set from this data we would be forced to choose, albeit randomly, a single instance each from the *RejectReqEv* and *AcceptReqEv* classes, and possibly two instances from the *ReqEv* class. With this number of test instances it is clearly not possible to derive any meaningful evaluation metrics.

A more suitable approach to evaluation is n-fold cross validation. Here the data set is first randomly partitioned into n stratified folds—we will use n = 10, a standard value. A classifier is then evaluated n times on each of the n folds in turn, having been trained each time on the union of the instances in the remaining $n-1$ folds. Cross validation is particularly suited to small data sets such as ours because each instance in the data set appears in a test fold exactly once. The results of the evaluation are the averaged results over all n folds. In order to further stabilise the results, we will perform 10 runs of 10-fold cross validation with the partitions chosen randomly and differently for each of the 10 runs. The results again are the averaged results of the 10 runs.

The learning algorithms we will consider in these experiments, primarily naive Bayes, are used as implemented in the Weka environment. We use Weka's functions for splitting the data set as described above into cross validation folds, for training a model and evaluating it on test instances, and for extracting evaluation metrics from the results of cross validation.

The typical single-figure evaluation metric for a classifier is its accuracy (or equivalently its error rate), which is the percentage of correctly predicted instances in the test set. As we will see in our first experiment below, simply relying on accuracy to compare how well two classifiers have performed is not suitable for the current learning task for two reasons. Firstly, the class distribution skew, with 5.5% of all instances in the three smaller classes, means that a classifier can predict all instances in these classes wrongly and still have an accuracy of up to 94.5%. This figure, although high, would certainly misrepresent the suitability of the classifier for a tutorial dialogue system. The second reason is that it is these three classes which we are concentrating on in our model, which means that we need a metric with which we can quantify the performance in these classes individually. We will use the f-score, essentially accuracy per class, which is the harmonic mean of precision and recall. The precision within a class is proportion of predictions into that class which are in fact instances of the class. The recall within a class is the proportion of instances of that class which were correctly predicted. F-score is defined as:

$$F = 2\frac{precision \times recall}{precision + recall}$$

When we compare the results of two different classifiers we are comparing a

vector of 10 results, one for each of the 10 runs of 10-fold cross validation. The two vectors are paired because for both classifiers, each ith cross validation run uses the same partitioning of the data into 10 folds. For most of our samples, especially in the three smaller classes, we find that the f-scores are not normally distributed according to a Shapiro-Wilk test. In the remainder of this chapter we will use the paired Wilcoxon signed rank test [207] to test for significance of the differences between vectors of f-scores and error rates because it can be applied to populations which cannot be assumed to be normally distributed. The paired Wilcoxon signed rank test ranks pairs according to lowest absolute difference in value. The test statistic V is either the sum of the ranks with negative difference or those with positive difference, whichever has the lower absolute value. For our experiments with 10 pairs the highest possible value of V is 55, meaning that the pairs either all exhibited a positive or all a negative difference.

The accuracy and f-score per class of the evaluation of a classifier are provided by functions in the Weka API. We use the statistical software package R [159] to perform statistical tests and to draw the graphs which appear in the rest of this chapter.

6.4.2 Choice of learning algorithms

Our first experiment compares the performance of a number of different learning algorithms on the task of predicting the tutor's TLGA. We consider a naive Bayes learner (NB), and a decision tree learner (J48, its implementation in Weka), as introduced in Section 6.1 above. We also explored the two further learning algorithms, k-nearest neighbour (kNN) and support vector machine [40] (SVM). The kNN classifier is based the distances between instances in features space, and categorises a new instance into the majority class of its k nearest neighbours. An SVM represents training instances in feature space and maps them so that there is a plane which maximally separates the instances of each category. New instances are categorised according to which side of the plane they fall on. For kNN the number of neighbours is set to five, for the other learners the default settings are used. The data set given to each learner is that presented in Section 6.3 above, after outlier removal. We use ten runs of ten-fold cross validation, as described above, and take the average across the ten runs of the accuracy as well as the F scores for each of the five classes.

Results

The accuracy scores for each learner are listed in Table 6.5. Although the differences between the different learners are not significant according to a cor-

Algorithm	NB	J48	SVM	kNN
Accuracy (%)	73.66	82.60	73.92	85.38

Table 6.5: Accuracy scores for learning algorithms

rected resampled paired t-test, these results seem to show that J48 and kNN are performing considerably better than NB and SVM. However, as we discussed above, only when we consider accuracy class by class can we see which learner is most suitable for this task. We present the f-scores per class in Figure 6.1. Here we see that both SVM and kNN have an accuracy of 0% in each of the three small classes, which is clearly not acceptable for this task. NB and J48 however show sensitivity to these classes despite their size, that is, they are predicting at least some cases correctly. The Wilcoxon signed rank test shows that J48 is significantly better than NB for the classes *Accept* ($V = 55$, $p <= 0.05$), *Reject* ($V = 55$, $p <= 0.05$) and *AcceptReqEv* ($V = 55$, $p <= 0.05$). NB is significantly better for the classes *ReqEv* ($V = 53$, $p <= 0.05$) and *RejectReqEv* ($V = 55$, $p <= 0.05$). These results show us that NB and J48 are comparably successful at the classification task, whereas SVM and kNN are not, due to their insensitivity to the small classes. We will continue with NB in the following experiments, and we will compare NB to J48 and to SVM and kNN again in the experiment in Section 6.4.4 to see which is the more suitable algorithm.

6.4.3 Combating class skew (1): Downsampling

One aspect of our data set which makes learning a good classifier difficult is the difference in size of the larger and smaller classes. This is problematic, as described by Weiss and Provost [206], because a classifier tends to prefer the majority class, whose instances it has seen more often in training and appear more often in testing. If the prior probabilities are considered by the learning algorithm then the minority class will be further disadvantaged. There are three types of approaches to getting around this. The first is to associate a higher misclassification cost with the minority class, either by integrating this into the learning algorithm or by altering the class distribution with duplicate instances. Unfortunately we can not use a misclassification cost function because without further experimental investigation we can not quantify how much worse the misclassification of a case in the small class is with respect to a misclassification of a case from the larger classes.

Two further approaches which do not require such information are upsampling and downsampling. Both reduce the skew in the class distribution by either increasing the number of instances in the minority class or decreasing the number of instances in the majority class. For our data set upsampling is

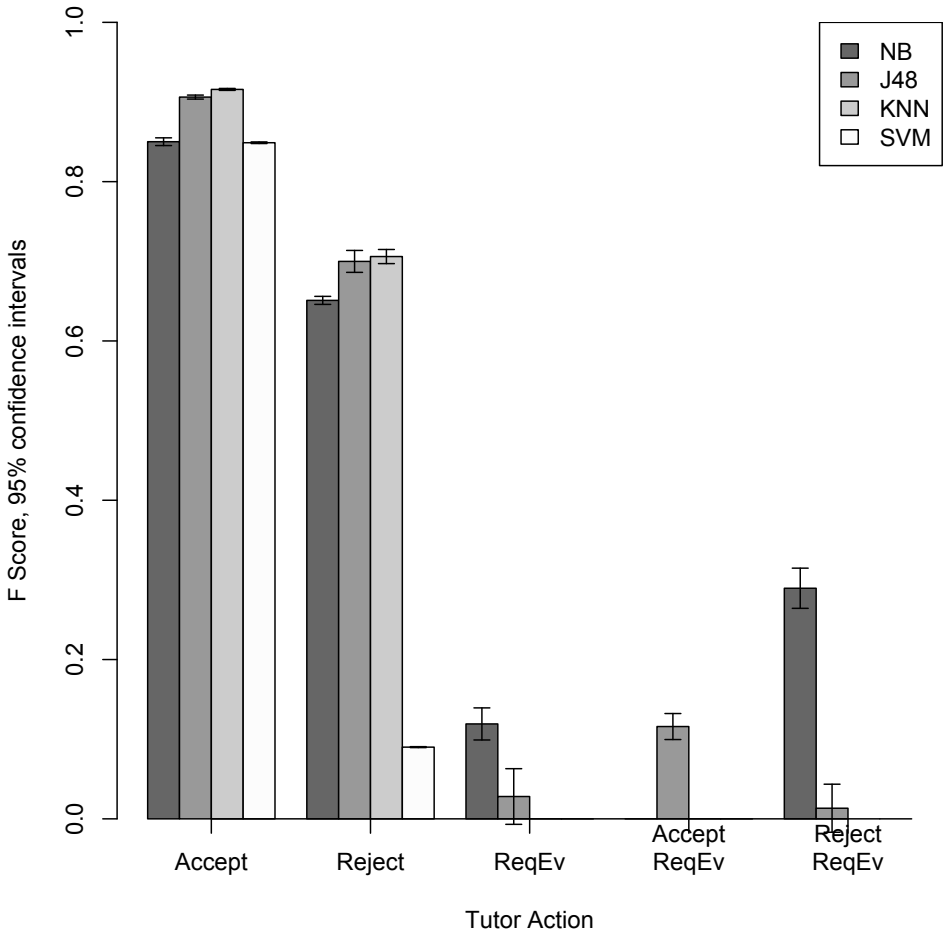

Figure 6.1: F-scores for each learning algorithm by class

not suitable: If instances are simply duplicated we will increase the likelihood that the model overfits the data. A better method of upsampling is to perturb rather than duplicate instances, such as in SMOTE [41]. This algorithm generates new instances on the line in feature space between two existing minority instances. For our data set this approach is also not suitable because the small classes are not only small in comparison to the large classes, they are also small in absolute terms, so that perturbed instances may be strongly misleading. We have therefore decided to simply downsample the larger classes. Although this throws away potentially useful instances, it has the advantage that we close the gap between the class sizes without introducing noise into the smaller classes. There is also research which indicates that downsampling is slightly preferable to upsampling [106]. This experiment investigates whether downsampling improves performance and what level is most beneficial.

The setup of this experiment is as follows: We begin with the full data set of 684 instances, which corresponds to a 17:1 ratio of largest to smallest class.

We downsample the data set in turn for each of the ratios 17, 15, 13, 11, 9, 7, 5, and 3:1 using the *SpreadSubsample* filter in the Weka environment. For a given ratio, this filter works by randomly discarding enough instances from any large class until the spread between the size of the smallest and largest class is at most in this ratio. We take the union of the three small classes as the basis for the calculation of the spread to the larger class, because these are the classes which include *ReqEv*. Similarly the larger class for downsampling is the union of the *Accept* and *Reject* classes, because these do not include *ReqEv*. With 38 instances in the three small classes, this means that for example at a ratio of 15:1 the downsampled data set contains 570 instances of either *Accept* or *Reject*, and a total of 608.

For each of the downsampled data sets computed this way, we use 10 times 10-fold cross validation to calculate the classifier's average error rate and average f-score per class. The learning algorithm used is Naive Bayes.

Results

Figure 6.2 shows the f-scores per class and overall error rate for each downsampling ratio. Each point in the graph is the average f-score for that class achieved by the classifier trained on the data set downsampled to that ratio. An informal inspection of the graph suggests that 7:1 could be a good choice. Up to this point the error rate has risen in comparison to the full data set from 22% to 30%, but at downsampling ratios lower than 7:1 it rises more sharply (to 40%, 46% and finally 59%). At 7:1 the f-scores for the small classes have all risen significantly in comparison to the complete data set (*ReqEv*: W = 1, p <= 0.001; *AcceptReqEv*: W = 5, p <= 0.01; *RejectReqEv*: W = 0.5, p <= 0.01, all using a Wilcoxon rank sum test with continuity correction). For the large classes the f-scores have also decreased significantly, but by much less in absolute terms: from 88% to 84% for *Accept* and from 70% to 66% for *Reject*.

At ratios lower than 7:1 the f-scores become erratic, falling strongly for the *Accept* and *Reject* classes. The total number of instances in the data set becomes very low. At 7:1 we have 304 instances, which is a data set comparable in size to other work in supervised learning for dialogue, such as Fernández et al. [73]. We can quantify the improvement in performance due to downsampling by inspecting the precision and recall values by class for the complete and downsampled data set, which are given in Table 6.6. Precision and recall are calculated as the average over the ten runs of cross validation. Both metrics fall slightly for the *Accept* and *Reject* classes in going from the full to the downsampled data set, but the other three classes exhibit a strong and uniform rise. The classifier is also able to detect a small number of *AcceptReqEv* cases which previously were not found. In absolute terms however, the results for the request classes

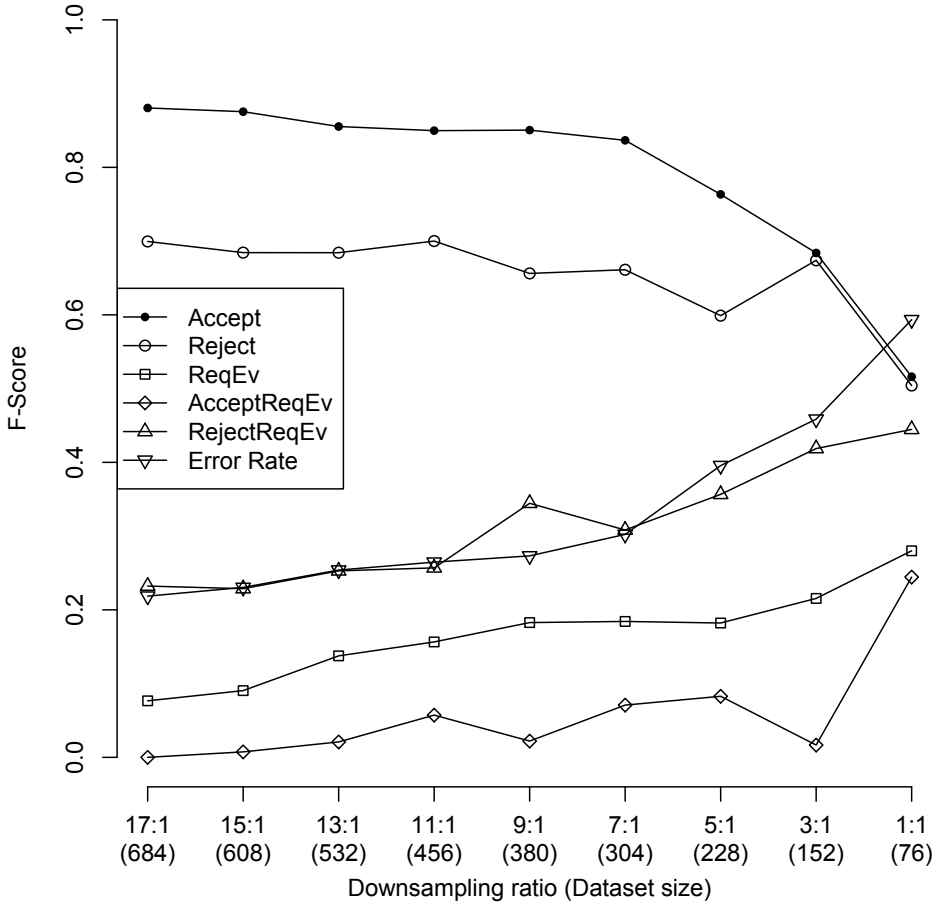

Figure 6.2: F-scores by downsampling ratio

may still be considered low. Overall the best trade-off between improvement of small-class performance, overall error rate and f-score stability appears to be at 7:1.

6.4.4 Combating class skew (2): Splitting the classifier

In this experiment we investigate a further way of minimising the effect of the class distribution skew. In short, we will translate our classification problem from a five-class problem into a combination of a two-class and a three-class problem, whereby the class skew in each of the two new classifiers is less than that of the original five-class classifier.

Even after we downsample the data set at a ratio of 7:1, we still must deal with a class distribution in which the largest class outweighs the smallest class by 207 to 10 instances. In addition it is the smaller classes which are more important for our overall task. We can however take advantage of the fact that

		Accept	*Reject*	*ReqEv*	*AcceptReqEv*	*RejectReqEv*
Complete	**Precision** (%)	92.1	66.1	06.8	0	18.8
	Recall (%)	84.4	72.6	13.5	0	30.0
Downsampled	**Precision** (%)	89.1	63.8	14.7	05.8	30.1
	Recall (%)	78.8	68.6	24.7	09.1	32.0

Table 6.6: Precision and recall values for the complete and downsampled data sets

	Request	**Not Request**	**Total** (C1 distr.)
Accept	11	207	218
Reject	10	59	69
Neither	17	0	17
Total (C2 distr.)	38	266	304

Table 6.7: Frequency distribution of the TLGA annotation of utterances into the classes accept or not and request evidence or not

there are generalisations which can be drawn across the classes. *AcceptReqEv* is conceptually a combination of accepting and requesting evidence, similarly *RejectReqEv*. *ReqEv* has no acceptance or rejection element. This generalisation allows us to redefine our classification problem into two parts: the first is the decision whether to accept or reject the student's step, the second is whether to request evidence of understanding from the student or not. For the former problem we combine *Accept* and *AcceptReqEv* into one class, and *Reject* and *RejectReqEv* in the other. For this classifier a third "neither" class is necessary for the *ReqEv* instances. For the latter we draw the classes *ReqEv*, *AcceptReqEv* and *RejectReqEv* together to form a "request" class, *Accept* and *Reject* then constitute the "do not request" class. The resulting class distributions are shown in Table 6.7. We will use C1 to refer to the acceptance classifier and C2 for the request evidence classifier. The first two columns in the table correspond to the class distribution of the downsampled five-class problem. The rightmost column and the bottom row are the distributions of classes for the classifiers C1 and C2.

When we have learned classifiers C1 and C2 then we have two predictions which we can combine to give us a prediction into the original five-class problem. This mapping is shown in Table 6.8. When C1 predicts Neither then the result is *ReqEv* regardless of the prediction from C2. We will refer to the classifier made up of the predictions of C1 and C2 as the "combined" classifier and we will refer to the classifier for the original five-class problem as the "simple" classifier.

C1	C2	Combined
Accept	*not ReqEv*	*Accept*
Reject	*not ReqEv*	*Reject*
neither	*not ReqEv*	*ReqEv*
Accept	*ReqEv*	*AcceptReqEv*
Reject	*ReqEv*	*RejectReqEv*
neither	*ReqEv*	*ReqEv*

Table 6.8: Combining the C1 and C2 predictions into the original problem space

	Request	Not Request
Accept	11	207
Reject	10	59

Table 6.9: Contingency table of C1 and C2 classes for dependence test

The experimental setup is as follows. We use the 7:1 downsampled data set from the previous experiment, which we have shown to be an improvement over using the full data set, with the full feature set, and we use 10 runs of 10-fold cross validation, but in a slightly different way to the standard method presented above. For each run and for each cross validation fold we train three classifiers: C1, C2 and the simple classifier. We then construct the combined classifier as in Table 6.8. Both the simple and the combined classifier predict the same five class range, so that we can then evaluate them on the test set of the fold. From this we calculate the error rate and f-scores as usual.

Independence of C1 and C2 Separating the problems of choosing whether to accept and choosing whether to request evidence requires that these are independent decisions. To check this we apply a chi-squared test to test the association between the classifications C1 and C2. Chi-squared measures the difference between the observed and expected frequencies. If there is no significant difference between these, then we can conclude that there is no association between the decisions. The Chi-squared test depends on the assumption that observations are independent. In our data this is the case because decision was made in a distinct dialogue context. That is, for each dialogue context, we have exactly one observation of the classification C1, and exactly one observation of the classification C2. One dependence which is present by design concerns the "neither" choice in C1. If an instance is found to be neither *Accept* nor *Reject*, then it is always *ReqEv*. Our test of independence must therefore not include these cases, because in effect the classification of these cases by C2 is ignored. The distribution of the remaining cases is given in Table 6.9. We

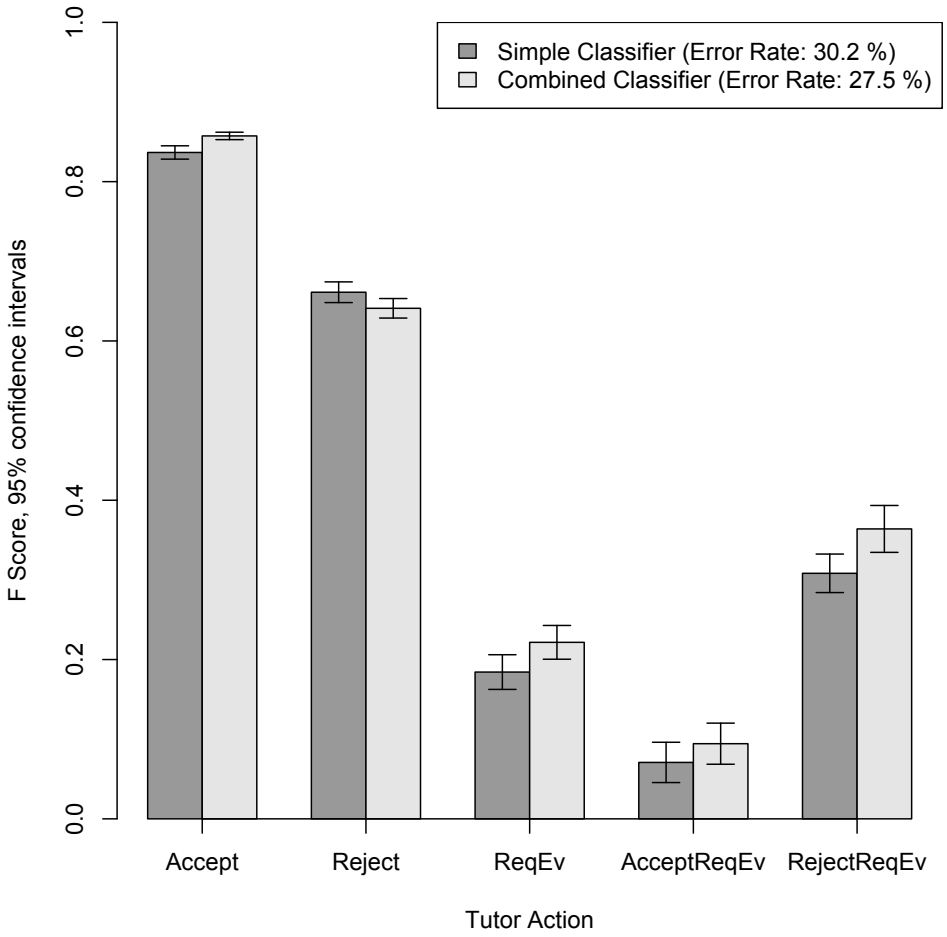

Figure 6.3: F-scores by class for the Simple vs Combined classifiers

apply Pearson's Chi-squared test with Yates' continuity correction to this distribution, which indicates there may be an interdependence between the two decisions (X^2(1, N = 287) = 5.5744, p <= 0.05). The significance of this test may be caused by a combination of the high frequency of both acceptances and non-requests. All other combinations are equally valid and more data may well improve the distribution. In the interest of simplicity we will continue with this assumption of the independence of C1 and C2.

Results

Figure 6.3 shows the f-scores in each class for the simple and combined classifiers. There are significant improvements in the performance of the classifier in four of the five classes: *Accept* (V = 0, p <= 0.01), *ReqEv* (V = 0, p <= 0.01), *AcceptReqEv* (V = 12, p <= 0.05) and *RejectReqEv* (V = 1, p <= 0.05). Only the *Reject* class shows a disimprovement (V = 55, p <= 0.01). The results show a reduction in error rate from 30.2% to 27.5%, which is significant

		Accept	*Reject*	*ReqEv*	*AcceptReqEv*	*RejectReqEv*
Simple	**Precision** (%)	89.1	63.8	14.7	05.8	30.1
	Recall (%)	78.8	68.6	24.7	09.1	32.0
Combined	**Precision** (%)	88.4	63.3	17.6	11.8	35.0
	Recall (%)	83.2	64.9	30.0	08.2	38.0

Table 6.10: Precision and recall values for the simple and combined classifiers

($V = 55$, $p <= 0.01$). Looking at the precision and recall results in Table 6.10, we see that for the *Accept* and *Reject* classes precision falls very slightly, as does recall for *Reject*, but the recall for *Accept* improves from 78.8% to 83.2%. The request classes show strong rises across the board, except for the recall value for *AcceptReqEv*, which falls by 0.9%.

We also repeated this experiment using the decision tree learner J48, because our first experiment (in Section 6.4.2) was not able to show clearly whether the naive Bayes or decision tree learner was more suitable. The results show that the naive Bayes learner is more appropriate. Although the combined J48 classifier has a lower overall error rate (24.8% as opposed to 27.5%), in the three small classes it performs slightly worse. The average f-score for the classes *ReqEv*, *AcceptReqEv* and *RejectReqEv* (21.4%, 13.9% and 12.8%, respectively) is 16%, lower than the average f-score of the naive Bayes learner on these classes, at 22.6%. We also found that the SVM and kNN learners again proved to be insensitive to the three small classes. Overall, splitting the classifier into two separate but simpler classification problems leads to improved performance, and the improvement is strong across the three classes which benefited most from the associated reduction in class distribution skew.

6.4.5 The influence of global features

One of the characteristics of tutorial dialogue which we model in this research is that tutors use some sort of internal model of the student's knowledge state in order to tailor feedback in a given dialogue situation. We approximate this model with the global features which appear in the description of a dialogue context, such as the number and type of rules the student has successfully used so far or the number of acceptances and rejections that have taken place. In this experiment we investigate the hypothesis that the performance of a classifier is improved by the addition of global features by comparing the performance of a classifier trained on local features only to one trained on the combination of local and global features. If the local and global feature set proves to be better, then this offers support for our dialogue model's use of dialogue context information.

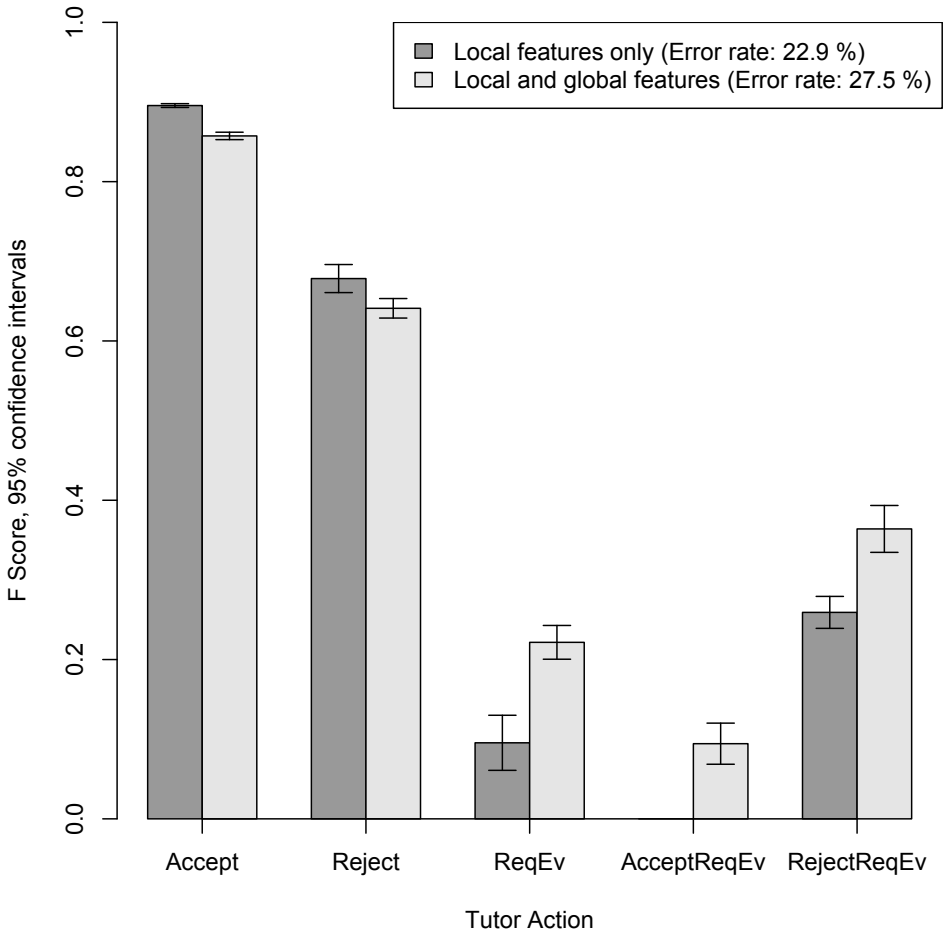

Figure 6.4: F-scores by class for local against local and global feature sets

The local feature set consists of the annotated features (Table 6.1) and the computed features (Table 6.2) which pertain only to the current utterance or the solution step it contains, a total of 18 features. The local and global feature set adds to this the features which are computed over the course of the dialogue, as presented in Section 6.3.2, resulting in a total of 27 features. As usual, for each feature set we use ten times ten-fold cross validation to compute the overall error rate and the average f-scores for each class. We use the combined classifier as described in the previous experiment, and the learning algorithm is naive Bayes.

Results

Figure 6.4 clearly shows large differences in f-score for each of the classes involving request evidence, which are significant according to a Wilcoxon paired signed rank test (all at $V = 0$, $p <= 0.01$). The decrease in f-score of the remaining two classes is smaller, but is also significant (*Accept*: $V = 55$, $p <=$

		Accept	*Reject*	*ReqEv*	*AcceptReqEv*	*RejectReqEv*
Local	**Precision** (%)	86.7	72.1	11.4	0	20.3
	Recall (%)	92.6	64.1	08.2	0	36.0
Local + Global	**Precision** (%)	88.4	63.3	17.6	11.8	35.0
	Recall (%)	83.2	64.9	30.0	08.2	38.0

Table 6.11: Precision and recall values for the local and local+global feature sets

0.01; *Reject*: V = 51, p <= 0.05). The difference in error rate, 22.9% versus 27.5%, is also significant (V = 0, p <= 0.01). The precision and recall results are shown in Table 6.11. For *Accept* the recall is lower although the precision is slightly higher, and for *Reject* a very small increase in recall is offset by a worse precision value. The request classes all show improvements in both precision and recall due to the addition of global features, whereby the recall value of *ReqEv* and the precision value of *RejectReqEv* both rise particularly strongly.

Overall we find that the classifier's performance for the classes involving request evidence is improved greatly by the addition of global dialogue context features, or equivalently, it decreases greatly when these features are removed. This result supports our hypothesis that global features are useful and important for deciding on feedback for the student and, more specifically, that they are important for deciding whether to engage the student in discussion or not.

6.4.6 Automatic attribute selection

The comparison in the previous experiment of the local and global feature set with the feature subset containing only local features was motivated by our underlying model of solution step discussions. There may however be other feature subsets which perform better than the full local and global feature set. To investigate this, we compare the performance of the classifier trained on feature subsets as chosen by a number of attribute selection algorithms. InfoGainAttributeEval ranks attributes according to their information gain with respect to the class, which is the approach taken by a decision tree learner. Information gain is the difference in entropy between the class and the class given the attribute. Similar to this is GainRatioAttributeEval, which uses an information gain ranking normalised by the entropy of the attribute. CfsSubsetEval uses a measure of the individual predictivity of attributes along with a measure of the degree of redundancy between them. A subset of attributes is preferred if it has a high correlation with the class but a low correlation between the attributes in the set. ChiSquaredAttributeEval ranks attributes by calculating their chi square statistic with respect to the class. A higher chi

InfoGainAttributeEval	cfsSubsetEval
time_to_reply	time_to_reply
correctness	correctness
proportionOfAcceptances	granularity
proportionOfRejections	ruleAppliedCorrectly
turnnumber	
ruleAppliedCorrectly	
acceptancesSoFar	
numberOfUninterestingRuleApplicationsSoFar	
numberOfInterestingRuleApplicationsSoFar	
granularity	

ChiSquaredAttributeEval	GainRatioAttributeEval
time_to_reply	correctness
proportionOfAcceptances	ruleAppliedCorrectly
proportionOfRejections	granularity
correctness	time_to_reply
turnnumber	relevance
numberOfInterestingRuleApplicationsSoFar	proportionOfRejections
numberOfUninterestingRuleApplicationsSoFar	proportionOfAcceptances
ruleAppliedCorrectly	tlga
acceptancesSoFar	turnnumber
numberOfInterestingRuleApplicationsInCurrentStep	acceptancesSoFar

Table 6.12: Feature subsets computed by four attribute selection algorithms

square value implies at the values of the attribute differ strongly for each class, making it more predictive for the learner. For each algorithm we use the implementations in the Weka environment.

The feature subsets computed by each selection algorithm are shown in Table 6.12, and were computed from the full data set of 684 instances. Together with the complete local and global feature set this gives us five feature sets to compare. For each one, we use the combined classifier and the naive Bayes learner and compute average f-scores from ten runs of ten-fold cross validation.

Results

Figure 6.5 plots the f-scores of each of the feature subsets in Table 6.12 for the five classes. The feature subsets perform slightly better than the complete data set for the *Accept* and *Reject* classes, but for the three *ReqEv* classes there is no clear best choice, and a number of f-scores are 0%. This may suggest that the features which are predictive of the two larger classes do not necessarily predict the smaller classes well, in which case it is necessary for our classifier to use the entire feature vector. We can also observe the presence of global features in the

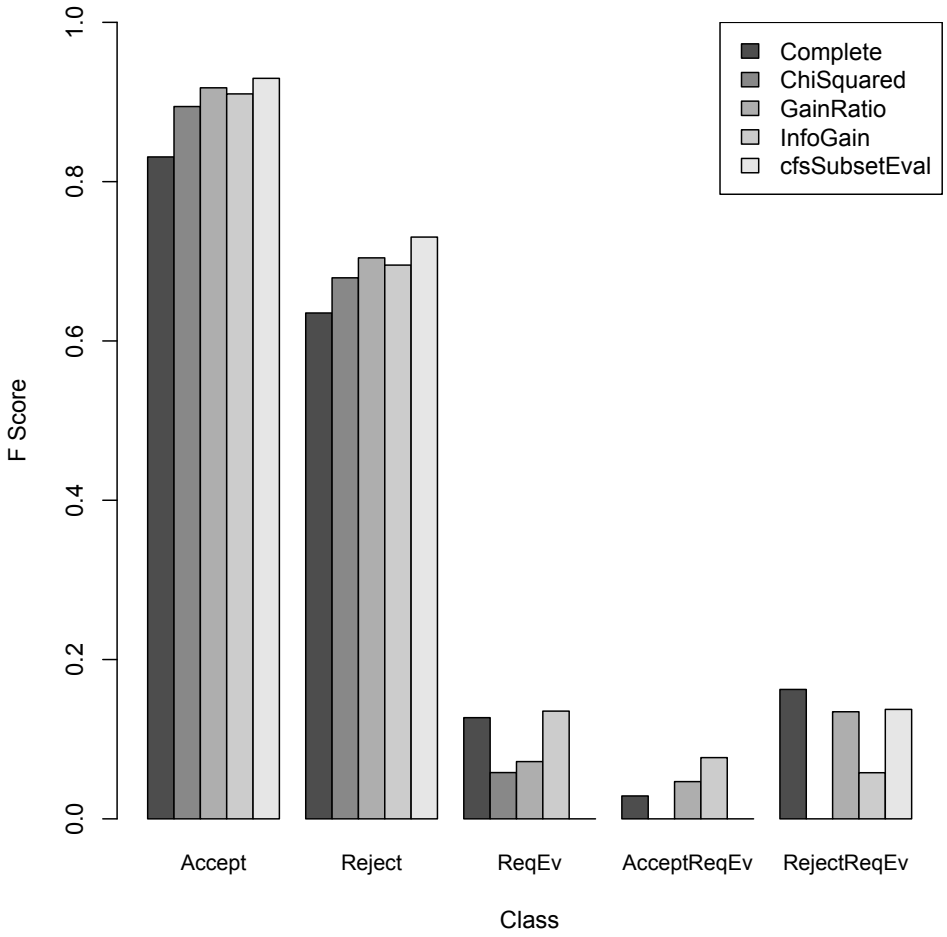

Figure 6.5: F-scores by class for each attribute selection algorithm

lists of highest-rated attributes. This offers support for the hypothesis tested in the previous experiment, that the global features improve performance and are useful—in this case, predictive—for the choice of TLGA. Finally we note that attributes which do refer to mathematical concepts, such as ruleStated, premiseStated or setDefinitionRuleUsedInCurrentStep, are less often among the highest ranked attributes. Instead the most predictive attributes seem to be those which describe the dialogue context at the level of the student's and tutor's actions, such as time_to_reply and proportionOfAcceptances.

6.4.7 Misclassification analysis

In this series of experiments we have been considering our task as a five-way classifier, and have presented our results as such. We can however view the results from the point of view of the generalisation of the individual classes which we used to split the classifier in two in Section 6.4.4. We argue that in

		Predicted				
		Accept	*Reject*	*ReqEv*	*AcceptReqEv*	*RejectReqEv*
Actual	*Accept*	172.3	16.4	9.8	5	3.5
	Reject	6	38.3	11	0.2	3.5
	ReqEv	6.4	2.8	5.1	2.6	0.1
	AcceptReqEv	9.2	0	0.9	0.9	0
	RejectReqEv	1	3	2.2	0	3.8

Table 6.13: Confusion matrix showing average number of predictions

the case of a misclassification, if the actual class and the predicted class share a concept, then the mistake the classifier has made is somehow less serious than other misclassifications. For instance, the classes *Accept* and *AcceptReqEv* both include the acceptance action, so if the classifier wrongly assigns an instance from the class *Accept* into *AcceptReqEv* or vice versa, then this wrong prediction may nevertheless be a reasonable action in the dialogue context.

To investigate how the classifier is misclassifying instances and whether they may be of this partially acceptable nature, we inspect its confusion matrix. A confusion matrix is a tabular representation of the predictions a classifier has made in a supervised learning problem. Each column represents the instances in a predicted class and each row the instances in an actual class. Each cell contains the number of predictions made into a class (its column) which were of a particular actual class (its row). The confusion matrix for a 100% accurate classifier contains zeros in all cells which are not on the main diagonal—all other cells represent misclassifications.

In Table 6.13 we present the confusion matrix for the classifier trained in the experiment in Section 6.4.4, that is, it is trained on the 7:1 downsampled data and uses the combined classification approach. It shows the average number of predicted instances in each class over the 10 runs of ten-fold cross validation[1]. We see for example that the majority of *Accept*, *Reject* and *RejectReqEv* instances are on the main diagonal. The fourth row the matrix shows that of the misclassifications of *AcceptReqEv*, none are misclassified into classes including a rejection aspect. Similarly on average only one instance of *RejectReqEv* is misclassified into a class including an acceptance aspect.

To illustrate better what proportions of each class are being misclassified into which classes we have graphed the data presented in Table 6.13 in Figure 6.6. In each bar the segments represent the percentage of actual instances of that class which have been classified into the different classes. We see that the the majority of *AcceptReqEv* instances are being classified into *Accept*, which suggests that for these instances the C1 sub-classifier for acceptance or rejec-

[1]The row totals may not exactly equal the actual class distribution due to rounding errors.

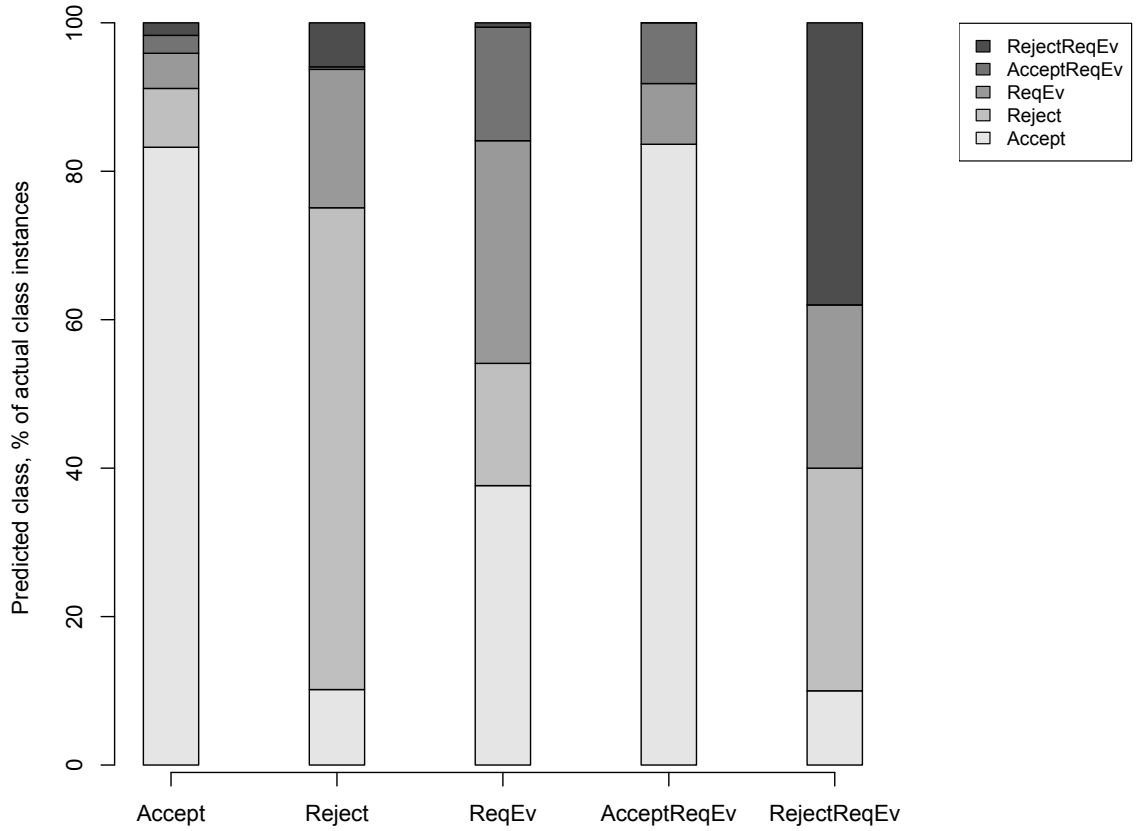

Figure 6.6: Confusion matrix expressed in percentages of the actual classes

tion (see Section 6.4.4) is correct but the C2 classifier is nearly always incorrect. The misclassifications of *ReqEv* are spread quite evenly over the four other classes. This would indicate that the classifier has not found features which distinguish *ReqEv* strongly from any other class, although this task is made more difficult by the characteristic of *ReqEv* that it can nearly always reasonably be performed.

To analyse this we can inspect the results of the ChiSquaredAttributeEval attribute selection algorithm for the two individual classifiers. For C1 the chi squared values, with a top ten range of $X^2 = 22528$ to $X^2 = 1397$, are much higher than those of the request/not classifier, which range from $X^2 = 10621$ to $X^2 = 606$ for the ten highest rated attributes. This shows that the feature vector discriminates better for the acceptance decision than the request evidence decision. The attributes themselves are similarly ranked, but one major difference is the placing of the attribute correctness. It is highly predictive for acceptance (second place, $X^2 = 11623.6$) but much less so for request evidence (13th place, $X^2 = 242.4$). On the other hand granularity is the only attribute which is more strongly predictive for request evidence (X^2

		Predicted		
		Accept	*Reject*	*Neither*
Actual	*Accept*	187.4	19.9	10.7
	Reject	7.2	48.6	13.2
	Neither	9	2.9	5.1

Table 6.14: Confusion matrix for the C1 classifier

		Predicted	
		ReqEv	*Not ReqEv*
Actual	*ReqEv*	15.6	22.4
	NotReqEv	33	233

Table 6.15: Confusion matrix for the C2 classifier

$= 3358.4$) than it is for acceptance ($X^2 = 3065$).

The difficulty of the *ReqEv* decision is further illustrated when we inspect the individual confusion matrices for the C1 and C2 classifiers, shown in Table 6.14 and Table 6.15 respectively. The table shows good results for C1, with an overall accuracy of 79.3%. Precision and recall for the *Accept* class are 92% and 87% respectively, and for the *Reject* class 68.1% and 70.4%. This shows that C1, and therefore also the combined classifier, is discriminating well between *Accept* and *Reject*. For C2 the accuracy result is also good at 81.8%, however this figure hides the poor performance of the classifier for the minority *ReqEv* class, which has precision and recall of 32.1% and 41.1%, respectively. This indicates that the C2 classifier is not able to predict *ReqEv* well, and it is this poor performance which causes the combined classifier to exhibit the low f-scores across the request classes which we saw in its results in Figure 6.3.

We may also consider what proportions of each class are being misclassified into classes which may in the context in fact be quite reasonable actions. As well as *Accept* and *AcceptReqEv* as described above, a misclassification in either direction between *Reject* and *RejectReqEv* may be considered reasonable since they share a common aspect of rejection. Classifying a non-*ReqEv* instance into *ReqEv* is also reasonable because *ReqEv* may follow many different categorisations of proof step. The tutor may elect to discuss a correct step in order to elicit a more detailed explantation exposing more domain content. Equally the tutor may elect to discuss an incorrect step in order to allow the student to learn by recognising a contradictory path of reasoning. *ReqEv* is therefore always an option open to the tutor. What is considered wrong in all contexts is a misclassification which crosses the boundary of acceptance and rejection. The confusion matrix for the C1 classifier in Table 6.14 showed that on aver-

	Right	Reasonable	Wrong
Accept	83.2	7.1	9.5
Reject	64.9	24.5	10.4
ReqEv	30.0	0.0	70.0
AcceptReqEv	8.2	91.8	0.0
RejectReqEv	38.0	52.0	10.0

Table 6.16: Grouped percentage of misclassified instances, by actual class

age 9.1% of *Accept* instances were misclassified as *Reject* and 10.4% the other way around. If we group the percentages of misclassified instances according to these definitions, as shown in Table 6.16, we see a relatively low proportion of wrong classifications, except in the case of *ReqEv*, for which we do not define reasonable misclassifications.

6.5 Summary and discussion

In this chapter we have developed a classifier for the tutor's task-level grounding actions. Given a dialogue context whose latest utterance is a solution step contribution, it predicts the TLGA that the tutor should perform. The features we have used to describe the dialogue context come either directly from the corpus annotation or are computed from it. As we argue in Chapter 5, the annotations consist of information which is also available to the dialogue model at run time. The dialogue model can therefore implement the same algorithm as we have used in Section 6.3.2 to calculate the values of the rest of the feature vector and this way the classifier can be used "live". We will consider this further in our evaluation in the following chapter.

In the course of developing the classifier we have investigated a series of hypotheses about how to achieve the best performance, and about how the performance of the classifier allows us to draw conclusions about our underlying dialogue model. In our first experiment we found that naive Bayes is the most suitable learning algorithm, and used this to train the classifiers in the subsequent experiments. The next two experiments attempted to deal with the skewed class distribution which our data set exhibits. We found that by downsampling the data set to a maximum spread of 7:1, and by splitting our task into two separate classification problems, we were able to strongly improve performance for the classes *ReqEv*, *AcceptReqEv*, *RejectReqEv* at the cost of a small decrease in the prediction accuracy of *Accept* and *Reject*.

Our fourth and fifth experiments focused on the question of which features from the complete feature set should be used to train the classifier for this task. Our dialogue model maintains a detailed representation of dialogue context

which is captured in the feature vector by the global features. These are the features which pertain to the dialogue history rather than just to the latest utterance. We compared the performance of classifiers trained with and without the global features, and found that the classifier trained with the global features is much more accurate across the *ReqEv* classes. We then considered a series of feature subsets generated by standard attribute selection algorithms. These did not lead to any clear improvement over the performance of the classifier trained using both the local and global features, but did show that global features are highly predictive for this task, and that domain independent features are more predictive than features specific to mathematical content.

The final classifier is thus trained on 7:1 downsampled data, uses the split classification approach, and is trained using the local and global feature sets. It has an overall error rate of 27.5%, which however masks a large variance in accuracy across the classes. *Accept* and *Reject*, the largest classes, are predicted well and the features are strongly predictive (as shown by high X^2 values) for this task. Across the three request classes we have an average f-score of only 22.6%, however this is a difficult task due to the small amount of data and the small proportion of instances in these classes. Our series of experiments has steadily improved the performance in these classes, and would hope that a larger, more balanced data set may continue this trend.

Finally, our misclassification analysis examined what kind of errors the classifier makes. We cautiously argue that some misclassifications, for instance choosing *AcceptReqEv* in the place of *Accept*, may in fact be quite reasonable actions in a given dialogue context. Seen in this light the number of wrong classifications is much smaller, around 10% or lower for *Accept*, *Reject*, *AcceptReqEv* and *RejectReqEv*. The question of whether these wrong predictions are in fact reasonable or appropriate in context will be one of the goals of the evaluation experiment which we will report in the next chapter.

By inspecting the individual confusion matrices for the C1 and C2 classifiers we saw that C1 is performing well for *Accept* and *Reject* but C2 is not performing well for *ReqEv*. This is the source of the low f-scores of the combined classifier for the request classes. There are two possible explanations for this poor performance: First, it may be that the classifier is not a suitable model of the decision whether to request evidence or not. In other words, despite the fact that we observe improved results when adding the global feature set to the local feature set, this feature set may still not be a sufficient description of the dialogue context in order to choose *ReqEv* correctly. The second possibility is that choosing *ReqEv* is an intrinsically difficult task, and that statistical methods can only reach a level of performance which is significantly worse than the gold standard represented by the wizards' actions in the corpus.

The results we have presented so far do not yet allow us to rule out either of these two explanations. We can however investigate further the question of whether choosing *ReqEv* is an inherently difficult task. If this hypothesis could be confirmed then the performance of our combined classifier would have to be seen in the light of this difficulty. An assessment of the difficulty of choosing request actions will be the second goal of the evaluation experiment in the next chapter.

In the evaluation we will ask human expert tutors to rate the classifier's output in comparison to a baseline and to the tutor's original action. Given a dialogue context and a model prediction which was not the same as the corpus action, if the expert tutors rate the model's output as highly as the corpus actions, then we have evidence that the model's prediction is equally reasonable in that context. In this case we can say that the model is choosing pedagogically appropriate actions in context, even if they do not match the original wizard's actions. If the baseline, which does not predict request actions, is rated comparably highly in relation to the model and corpus, then we have evidence that choosing request actions correctly is difficult even for a human, in which case we may consider the performance of the C2 classifier as acceptable for the task at hand.

7

Evaluation of the model

In Chapter 4 we presented a finite-state model of solution step discussions which maintains a representation of dialogue state. In Chapter 6 we developed a classifier which in turn uses this dialogue state information to predict the tutor's task-level grounding action (TLGA) in a given dialogue context. The evaluation in the previous chapter found an accuracy of 72.5% as well as a certain sensitivity to the *ReqEv*, *AcceptReqEv* and *RejectReqEv* actions. We will refer to the group of these three as the *request actions*. The classifier however proved to have difficulty discriminating requests from non-request actions. This aspect of the classification, embodied by the C2 classifier, had a precision and recall of only 32.1% and 41.1%, respectively.

In this chapter we report on a further evaluation of the classifier in which we investigate the misclassifications which were initially analysed in Section 6.4.7. In that analysis we reasoned that the misclassifications which do not cross the accept/reject boundary, for instance when *Accept* is misclassified as *AcceptReqEv*, are not necessarily bad errors, and that the predicted action may in fact be appropriate, depending on the context of its performance. Our first hypothesis in the evaluation which we present in this chapter is that the predictions of the model and the actions in the corpus in such cases will be equally appropriate in context even though they are not the same action.

Our second hypothesis concerns the question of the intrinsic difficulty of the task of choosing whether to request evidence or not. Having found that the classifier does not perform well for the request classes, we would like to investigate whether this is due to the design of the classifier or due to the basic difficulty of the task. If humans are equally bad at detecting when requests

should be made (as defined by the gold standard of the corpus), then we may be able to consider the performance of the classifier as not so bad after all.

The methodology we follow elicits ratings of appropriateness from human raters, mathematics experts with teaching experience, who view and rate the predictions in context. We will use a baseline which only predicts *Accept* and *Reject*, and each dialogue context will be rated for each of the three experimental conditions (model, corpus, baseline). By comparing the ratings across the conditions rather than just the predictions we are effectively smoothing the differences between accurate and inaccurate predictions, and abstracting the evaluation criteria from "same prediction" to "same appropriateness rating by an expert".

The experiment is a one-factor repeated measures design. If our first hypothesis, about the appropriateness of the model's predictions, is confirmed we will find that the ratings assigned in the model condition are not significantly different to the ratings in the corpus condition. If our second hypothesis, about the difficulty of the task, is confirmed we will find that the ratings of the baseline do not differ significantly from the corpus ratings. In other words, this result would show that in contexts in which the wizard requested evidence, a simple *Accept* or *Reject* is considered just as good, which shows that it is difficult to detect the specific conditions which make the *ReqEv* action necessary.

This chapter is structured as follows. In Section 7.1 we motivate the use of human rater evaluations as opposed to other approaches to dialogue system evaluation, and review some similar studies in the literature. Section 7.2 presents the experimental design. We outline the task the raters should perform, the choice and presentation of the stimuli, the restrictions on participants' experience, and the procedure which was followed in the experimental session. Our results are presented in Section 7.3, and Section 7.4 gives a summary of the experiment and a discussion of the results.

7.1 Evaluation with human expert ratings

Dialogue system evaluation has moved strongly towards evaluation based on usability metrics such as task completion, dialogue length or number of user actions [148, 149]. Such metrics can be extracted from the transcripts of dialogues between the system and the user and then compared to baseline values as a measure of how well the system is performing. The PARADISE framework [205] aims to model the general concept of user satisfaction to measure how well a system performs. Users complete a questionnaire after interacting with the system in which a set of ratings is collected, such as whether the system fulfilled the user's expectations, whether the system understood what was

said or whether the user would use the system again in the future. Combined with the objective measures from the transcript of the dialogue such as number of turns, a regression model can be fit which estimates the user ratings of future dialogues from the objective measures. Using this evaluation approach requires the system to be fully implemented and robust to real users, if not then parts of the system can be simulated in a Wizard-of-Oz setup.

PARADISE is suitable for dialogue genres such as information seeking, in which it is clear what the objective measures should be: shorter dialogues, less clarification requests and higher task completion rates lead to higher user satisfaction. For tutorial dialogue it is less clear what the equivalent objective measures would be. For instance Graesser et al. [86] argue that questions are a fundamental component in driving explanatory reasoning, which leads to learning, so less requests does not necessarily correlate with better tutorial dialogue. More typical in tutorial dialogue is evaluation by measuring the learning gain that students exhibit after interacting with the system [88, 9]. An evaluation may also involve finding correlations between the student's learning gain and some set of relatively objective measures of the dialogue, such as the dialogue acts performed or the content of utterances [126]. A fully implemented tutorial dialogue system can also be used to focus evaluation on one of its parts, for instance interpretation errors [70].

For this research the evaluation methods outlined above are not suitable. During the development of the DIALOG project, which focused on fundamental rather than applied research, it was not possible to conduct studies including pre- and post-tests. The approach of correlating features of the dialogues that the model produces with learning gain is therefore not available. This means we can not tell which kind of tutor behaviour or which choice of action in a given situation leads to a more effective dialogue, whereby effectiveness is equated with learning gain.

There has been extensive work done within the DIALOG project on interpreting the kind of language students use as exemplified by the corpus in Chapter 3 [100, 101]. However the scale of the research reported here is such that a fully implemented end-to-end system with sufficiently robust input analysis as well as expressive natural language generation is not possible. This rules out the option of evaluating the system in a "live" experiment with students. Indeed the results of such an approach would be strongly affected by how well the input and output modules work, because the surface realisation of tutors' utterances has been shown to affect their contribution to learning [65, 130]. This would also be the case for a semi-implemented Wizard-of-Oz scenario in which a human performs the function of input analysis and output generation.

Instead, the evaluation we plan to carry out uses the ratings provided by

human experts to compare the output of the model with the gold standard represented by the corpus. If the ratings given to the action predicted by the model and the original action in the corpus are the same, then the actions are equally good in that context, even if they are not the same action. This approach is an example of what Miliaev et al. [141] refer to as an "intrinsic" evaluation, as opposed to an "extrinsic" one, in which a system is evaluated indirectly by measuring some dependent value such as task success. Our gold standard is thus no longer the actual action that occurred in the corpus, but rather the rating of that action.

Human judgement evaluations are typically used when a formal measure of how good an output is is hard to define, or when the measure of success is subjective. For this reason it has often been used to evaluate natural language generation (NLG) tasks, where there may be many "correct" ways to verbalise a communicative intention, and these different verbalisations may be difficult to compare objectively to one another. Similarly, in dialogue systems there may also be many correct but incomparable responses in a given situation. We now briefly review some experimental designs using human judgements in the fields of NLG and dialogue modelling.

Walker et al. [203] evaluate a multi-modal information presentation system for tourist information in order to investigate the effect of tailoring output to the user and the presentation mode. Each subject "overhears", or hears a playback of, a dialogue from the corpus and hears at a given point two responses corresponding to with and without tailoring (the experimental manipulation). The subject rates both responses by how much he agrees or disagrees with "the system's utterance is easy to understand and it provides the information that I'm interested in". 16 subjects heard 4 dialogues, and in each dialogue 6 judgments were made. There was a main effect found of tailoring on the judgments.

A number of sentence planners are compared within an NLG task by Rambow et al. [160]. The subject sees part of a dialogue as context, and is then asked to rate six possible system responses for their "understandability, well-formedness and appropriateness" in that context. 60 subjects each saw five dialogues with 20 points at which the response possibilities were presented, resulting in 1200 judgements, and it was found that a trainable sentence planner received significantly higher scores.

The evaluation carried out by Lester and Porter [121] compares an explanation generation system for biology concepts from a knowledge base to human writers. The "two-panel method" is used. Both the system and a panel of experts generate explanations for the concepts in the evaluation. Each concept is explained once by the system and once by a human. This set of explanations

is sent to a second panel of experts to be rated with A-F grades. The mean and stderr are calculated to compare the system to the human-generated explanations. Cawsey [39] evaluates EDGE, an explanation generation system. The evaluation results were that a "small number" of users found the explanation "instructive and largely coherent". No comparative evaluation was done. The evaluation by Miliaev et al. [141] consists of a subjective assessment of text quality. Nine subjects saw nine texts, three from each of original text, generated text, and text generated from an edited plan. They rated the "quality and understandability". There were no significant differences on the Likert scale between the three versions, thus it was concluded that they are similar for the readers.

The study reported by Moore et al. [143], and Porayska-Pomsta [154] is an example of using human judgements to evaluate a component of a dialogue system which is not the NLG component. In this tutorial dialogue application the experimenters investigate how tutorial feedback is influenced by affect. A subject sees a dialogue fragment which ends with a student utterance which was either incorrect or partially correct. They then see three possible tutor responses: The actual response from the corpus, the preferred response from the model, and the dispreferred response from the model. Four subjects each saw 20 dialogues and rated how appropriate they thought each response would have been in that context. The human response and the preferred response were not significantly different, and the others were, which supports the effectiveness of the model.

Stent [184] evaluates a generation module developed within the TRIPS system. The goal of the evaluation is to find out how "natural" the generated output is. The reasons for using human judgements in this case included the fact that the TRIPS system was not robust enough to support an end-to-end evaluation, and that such a task-based analysis does not allow conclusions to be drawn about specific modules or features of a system. In addition, the property of "naturalness" is best measured in context by humans.

In three dialogues from the TRIPS corpus the user utterances were replaced with utterances generated by the generation module. Three judges each rated 6 dialogues (three original and three modified as above). They were told that there were 7 dialogue participants in all, but not that one of those was the computer. The judges evaluated the modified dialogues using the originals as reference material. The evaluation consisted of subjective annotations on utterances, phrases, referring expressions, and so on, as to whether they were wrongly placed, incorrect, redundant etc. Additionally comments about odd or incomplete utterances were elicited. The judges were asked to say whether each of the dialogue participants was a native speaker or not. One judge noted

the lack of fluency of the computer-generated turns, the other two judges did not identify the system as the worst of the speakers in terms of naturalness and the computer utterances received less negative comments than the humans. Overall "the judges considered the system's contributions to be at least as coherent, informative and robust as those of the humans".

7.2 Experimental design

Our goal with this experiment is to elicit ratings of the appropriateness of the actions in context across three conditions: The prediction of the model, the action contained in the corpus, and the prediction of a baseline. For this we will ask raters who are expert tutors to read excerpts from the corpus which end with a student utterance. One possible response to this utterance for each of the three conditions is rated. We can then compare the ratings and hopefully see that for our first hypothesis the model and corpus conditions are similarly highly rated, and for our second hypothesis that the corpus and baseline conditions are similarly highly rated.

This section presents the experimental design. We detail what rating task the experts have to complete and how the materials were chosen and presented. We outline the required profile of the participants and the procedure followed in carrying out the experiment.

7.2.1 The rating task

The participant's task in this experiment is to rate the appropriateness of the tutor's TLGAs in a given dialogue context. The participant sees a piece of dialogue context which ends on a student proof step. It is followed by the actions from the three conditions, each of which is to be given a separate rating. The task is defined by the rating question which the participant answers at each stimulus: "How appropriate do you consider the following reaction from the tutor?". Appropriateness is been defined in the introductory text to the experiment as referring to "mathematical and didactic appropriateness", and the explanation is given that a reaction is appropriate when it suits the student's current knowledge and promotes a possible learning effect. The exact choice of words in this question will have an influence on the ratings, and other phrasings would have been possible. For instance, asking "How well would this action contribute to you better understanding what the student knows?" would focus strongly on what we assume is one of the motivations behind requesting evidence of understanding of concepts. We could also target potential learning gain by asking "How well would this action contribute

to the student learning the material/proof/proving in general?", but it is not clear that this would elicit different ratings.

We decided to use the "appropriateness" expression because it is the same description as was given to the wizards in the corpus collection experiment. By giving the participants in this experiment the same instructions and asking them to rate along the same dimension, we implicitly encourage them to act like the wizards did. Appropriateness is also the most neutral and unspecific instruction for the raters, and does not bias their interpretation of the task in any direction. The rating is given on a seven-point Likert scale [125], a standard technique for mapping subjective responses to scalar values. Each of the three conditions should be rated as independent options and the rater should not consider it a ranking task.

7.2.2 Materials

The materials used in the experiment are generated from the annotated corpus and from the predictions made by the classifier which was developed in the previous chapter.

Generating the stimuli

Ideally for an evaluation like this one our stimuli would be taken from a test set of dialogue contexts which had been kept apart from the training data during the development of the model. However, as we explained in Section 6.4.1, due to the low frequency of occurrence of requests for evidence of understanding, splitting the data into a training and test set is not possible. If we had used a 10% split, which is a typical value, we would have had only one or possibly two instances of *ReqEv*, *AcceptReqEv* and *RejectReqEv* in the test set, from which it would be impossible to derive meaningful ratings. Instead we will use ten runs of ten-fold cross validation to generate predictions for the corpus and baseline conditions for instances in the corpus, as we did in the course of the machine learning experiments in Chapter 6.

For the model's predictions we will use the classifier with the optimal parameters according to the series of learning experiments; this is the classifier which we trained in Section 6.4.4. It uses the downsampled data set at a ratio of 7:1, resulting in 304 instances, it is trained using the naive Bayes learner, and it uses the approach of splitting the classification problem into two simpler parts. Using the classifier's predictions means we are not strictly using the implementation of the dialogue model from Chapter 4 in this evaluation. However since the dialogue model can use the same update algorithm as we implemented for the creation of the learning instances, it maintains the same dialogue context

representation as the learner in our previous experiments had. Therefore evaluating the classifier's predictions is equivalent to evaluating the classifier which predicts based on the content of the dialogue model's state representation.

Baseline predictions We considered a number of possibilities of how to generate baseline predictions. A typical and simple approach is to use the majority class found in the corpus. In our case this would mean that all instances would be labelled as *Accept*. This baseline would have an accuracy of 77.9% because *Accept* is by far the majority class, with 505 of 684 cases in the complete learning data set. However for the cases in which the student's utterance was incorrect, and the actual tutor's action was for instance *Reject* or *ReqEv*, this baseline is clearly wrong.

A slightly stronger baseline is a classifier trained from the same data as above but using only the correctness annotation of the previous student utterance. The attraction of this baseline is that it is a fair approximation of a simple tutorial system, in other words one that acts based only on a shallow analysis of the most recent input and does not have any access to the previous dialogue. Although this classifier predicts both *Accept* and *Reject*, it suffers from the same problem as the majority class baseline, namely that it sometimes predicts *Accept* for incorrect steps and *Reject* for correct steps. Comments from a participant in a pilot study indicated that rating this combination of context and action is misleading for the raters.

The baseline we have decided to use is a simple mapping from correctness value to TLGA. Correct steps map to *Accept*, incorrect steps to *Reject* and all others, in other words partially correct and n/a, map to *Accept*, because this is the majority class. This baseline does not get the correct and incorrect steps wrong like the two above, but can not predict any of the other three TLGA categories.

Verbalised dialogue moves It is important for this experiment that the actions in each condition are presented such that the conditions are indistinguishable from one another. However we do not want the stimuli to contain only the name of the TLGA, because in this case raters would have to learn what each move meant and remember this definition for the duration of the experiment. We also can not use natural language expressions because this would given an unfair advantage to the corpus condition. The actions in the corpus are already verbalised whereas the actions in the model and baseline condition would have to be generated by some form of natural language generation. Here the expressivity and naturalness of the wizards' utterances would no doubt lead to them being considered better, and the differences in surface

TLGA	Verbalisation
Accept	"Tell the student that that was the right next proof step"
Reject	"Tell the student that that was not the right next proof step"
ReqEv	"Ask the student how the proof step was derived"
AcceptReqEv	"Tell the student that that was the right next proof step, but also ask how the step was derived."
RejectReqEv	"Tell the student that that was not the right next proof step and ask how the step was derived"

Table 7.1: Verbalisations of the five TLGAs

realisation would be a strong confounding factor.

We have chosen a solution between these two extremes which we will refer to as a "verbalised dialogue move". This is essentially a template for the description of a dialogue move, rather than for a surface realisation of an utterance with the same intention. It describes what the dialogue move means rather than showing a way of realising it. The verbalisations which we will use for the five TLGAs are given in Table 7.1. This approach has a number of advantages in the context of this evaluation. Rating the model's output with verbalised dialogue moves reflect the fact that it reasons at the dialogue move level rather than the natural language level. The use of templates avoids having to write arbitrary natural language realisations of actions, of which there could be many. There are of course many different ways to express the intention of an the TLGAs which fit in with our original definitions, however we considered "the right next proof step" to be a good compromise between capturing both the correctness of the step and the relevance of the step to the current task in the current context.

Dialogue context The motivation for our model of TLGA has stressed that it is crucial that a tutor builds up some internal model of the student's knowledge state. Only then can the tutor decide with certainty whether a solution step should be discussed or not. This decision depends on whether the tutor believes the student has in fact mastered the concepts that occur in the step, and this belief can only arise from observing the student's actions in the previous discourse. The raters in this experiment are effectively rating the appropriateness of entering such discussions, therefore it is essential that they have access to the same context. However we are forced to make some trade-offs.

We have to decide how much context we can realistically present to the raters. On the one hand a longer dialogue context would mean that the rater has the opportunity to develop a better internal model of the student's expertise, however it would be highly time-intensive to ask raters to read full

transcripts, some of which are over 50 utterances long, in order to elicit only one rating. On the other hand the context does have to be long enough for the rater to have had some chance of recognising what the student might or might not know from his demonstration of the use of mathematical concepts. We have decided that the previous context of a rating point should contain at least all turns going back to the beginning of the exercise in which the step occurs. If this is less than 8 turns, the preceding exercise is included in its entirety.

In the interest of receiving as many individual ratings as possible from each participant we decided to present the stimuli in pairs. For each dialogue excerpt the raters read, they initially see the dialogue transcript up to and including a solution step contribution from the student. They are then presented with the three actions, one for each condition, to be rated. After they have given the three ratings, they see the dialogue transcript as it actually continued up to the second stimulus, where they are again asked for three ratings. They are informed after the first rating that they are now reading the dialogue as it in fact continued. The second stimulus may occur soon after the first in the dialogue. In this case the minimum context restriction described above is fulfilled by the context of the first stimulus.

Choice of dialogue situations to be rated

From the data set of 304 instances we must choose a subset which will be used to generate the rating stimuli as described in the previous section, because this is too much to ask participants to rate. The model predicted 72.5% of these correctly (221 cases), as reported in Section 6.4.4. The experimental stimuli are chosen from the remaining 83 cases, because it is the ratings of the cases where the model's prediction differs from the corpus that are most informative about whether certain wrong predictions may in fact be appropriate actions in that context.

We chose the rating stimuli according to the following criteria. The turn number of the student's utterance must be at least 8. This is to ensure that there is enough previous context for the step, as described above. We chose all instances for which the tutor's action in the corpus was a request action, of which there were 31. This choice is due to the focus of our model on solution step discussions and the tutor's decision of whether to begin such a discussion or not. For the same reason we also chose the instances in which the model's prediction was a request action, of which there were a further 19 instances. In order to have all of the stimuli in pairs we added six dummy instances whose model and corpus predictions were changed so that they also included at least one request action.

These criteria resulted in a stimuli set of size 56. In all cases the model

	Accept	*Reject*	*ReqEv*	*AcceptReqEv*	*RejectReqEv*
Model	14	10	12	14	0
Corpus	13	6	14	9	8
Baseline	42	8	0	0	0

Table 7.2: Distribution of TLGAs by condition in the stimuli

condition and the corpus condition are different and at least one is a request action. By always including one request action we focus on the cases in which we believe the model may be able to predict a good action. This approach of choosing the data points with the most differentiating potential was also taken by Porayska-Pomsta [154], who selected the incorrect and partially correct instances rather than the correct ones in her evaluation. In the case that the baseline condition is the same as either the model or the corpus, the baseline is not displayed to the rater and instead is assigned the same rating as the condition it is the same as received from the rater.

The remaining wrong predictions cross the Accept/Reject boundary, that is, the model misclassified an *Accept* into *Reject* (7.9% of all *Accept*s) or the other way around (10.2% of *Reject*s were classified as *Accept*). We did not use these as stimuli because they could confuse the rater and because the prediction is so wrong that the rating would not be informative. In other words the rater could not possibly rate "Reject" well if the step was in fact correct, so the difference in ratings between the accept and the reject is not meaningful.

Table 7.2 shows the distribution of TLGAs in the 50 non-dummy stimuli. The distribution is balanced for the model and corpus conditions, except that no stimuli included a prediction of *RejectReqEv* from the model condition. There is a strong skew in the baseline condition towards *Accept*, which is caused by the high frequency of correct proof steps in the underlying data. We can now show an example dialogue excerpt as it is presented to the rater. Figures 7.1 and 7.2 show the first and second stimulus of one pair, respectively. The stimuli are in German, but here we are interested in illustrating their presentation rather than their content. In the first stimulus the baseline condition has been suppressed, in the second all are present.

7.2.3 Experimental procedure

The stimuli were split into four groups so that each participant sees 14 items in seven pairs. A pilot study with example stimuli indicated this was an amount of material which could be finished within 30 minutes. The total occurrences of the TLGAs were balanced in each group. Participants were allocated into groups in sequence according to the order they began the study.

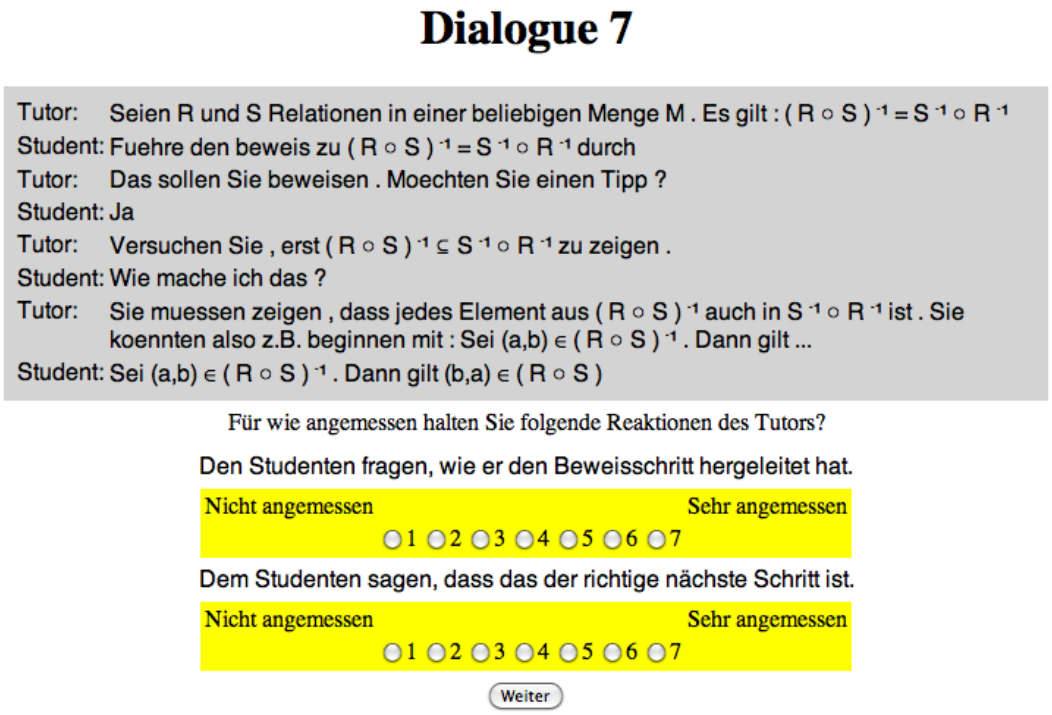

Figure 7.1: The first stimulus of a pair

To ensure the validity of the study we required that raters had experience as teachers or tutors. More specifically, we restricted the participants to those who had taught mathematics for at least one semester as either a schoolteacher, university lecturer or university teaching assistant. This way each participant had experience at least in a classroom setting, if not in one-on-one tutoring. The participants were all volunteers who were recruited by advertising the study across a number of mailing lists and internet forums for maths teachers, mailing lists of maths departments at German universities, and direct emails to members of the teaching staff of the Departments of Mathematics and Computational Linguistics at Saarland University. No other contact was made with participants, and they could complete the study any time they liked. The entire study was conducted in German.

The experiment was carried out online according to the following procedure. Participants were reminded of the restriction that only those with experience teaching mathematics may take part. They were then asked to fill out a short questionnaire about their teaching background. They were asked about their university degree (mathematics, mathematics didactics, a maths-related degree, or other), their teaching experience (schoolteacher, university lecturer or university teaching assistant), how long ago they taught maths for the last time measured in semesters, for how long they have taught mathematical foundations in total measured in semesters, and how much experience they have of teaching logic or proof theory (none, a little, middle, a lot).

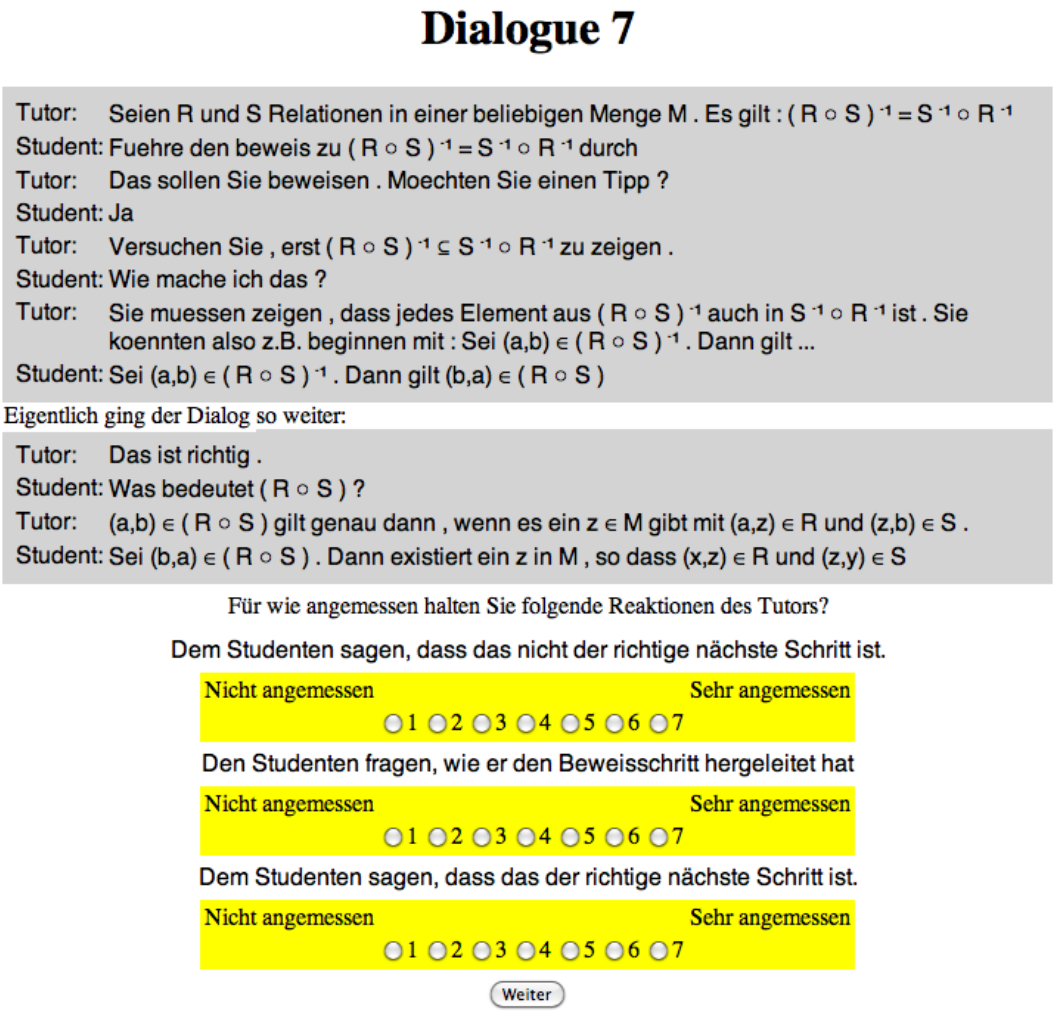

Figure 7.2: The second stimulus of a pair

In the next step the participants read an introductory text which explained the scenario of dialogue based tutoring and gave an example of a complete exercise as it appears in the corpus. The rating task was explained in detail. The instructions encourage the raters to put themselves in the position of the tutor and to try to understand what proof the student is constructing in order to best gauge the appropriateness of the proposed reaction. They were then shown the 14 stimuli in seven dialogue excerpts as illustrated above in random order and asked for their ratings of each one.

7.3 Results

The study was completed by ten raters. The website was visited 60 times, but the remaining 50 visitors did not continue after the introductory text, or rated only the first stimulus. The stimuli in groups one and four were rated by three raters, groups two and three were rated by two raters. Two raters in group four

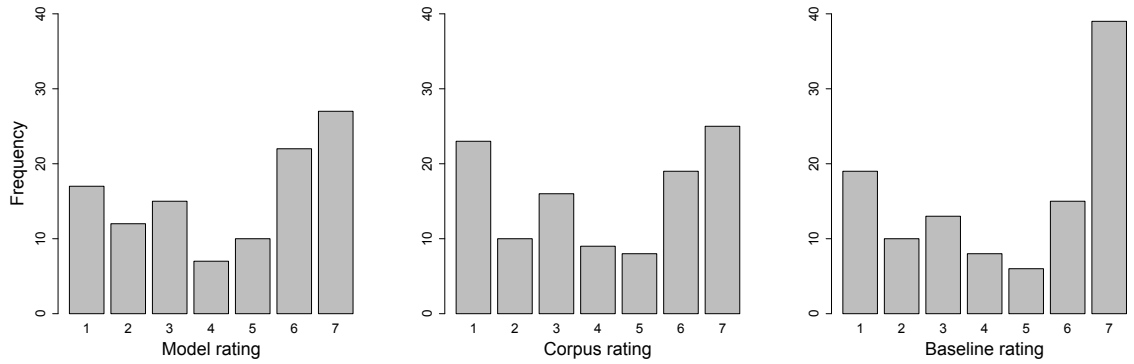

Figure 7.3: Frequency distributions of ratings by condition

rated only four and 11 stimuli respectively, and one rater in group one rated only seven stimuli. Including repeat ratings of the same item by different raters but excluding ratings given to dummy fillers, 110 stimuli in total were viewed and rated for each of the three conditions.

Among the participants were six who had studied mathematics and four who had studied mathematics teaching (the German *Lehramt* degree course). These were spread evenly over the four groups of stimuli. There were five university teaching assistants, two schoolteachers and three university lecturers, and all but one were currently teaching. Four participants had no experience teaching mathematical foundations and another four had two semesters' experience. The remaining two had taught foundations for 6 and 10 semesters respectively. The participants' experience teaching logic and proof theory was given as none by three participants, a little by a further three, two answered some and two answered a lot.

The frequency distributions of the ratings in each condition are shown in Figure 7.3. The graphs illustrate the skew towards the extreme values 1, 6 and 7 in all three conditions. We see that the baseline was rated with 7 more often than the other two conditions, as well as the unexpected result that the corpus was rated with 1 more often than the model and baseline predictions.

The use of parametric methods on Likert scale data is a matter for discussion because the data can be interpreted as being ordinal or interval [63, 105]. However we believe that our experiment generates interval data because the rating scale was expressed as a row of equidistant consecutive integers. In this case we expect the rater to consider the numbers to represent equal grades of categorisation, and thus the mean has a meaningful value. The mean rating in the model condition is 4.41 (sd = 2.22), for the corpus condition it is 4.15 (standard deviation = 2.29) and for the baseline condition it is 4.57 (sd = 2.36). The ratings are not normally distributed according to a Shapiro-Wilk normality test (Model: $W = 0.8647$, $p < 0.001$; Corpus: $W = 0.8645$, $p < 0.001$; Baseline:

$W = 0.8254$, $p < 0.001$). This result effectively means the interpretation of the Likert data as interval or ordinal remains moot, because due to the non-normally distributed data we will use non-parametric tests in the following analyses. The significance level will be set at 0.05.

We would like to investigate whether the rating that participants have assigned to actions varies by condition. Therefore we will carry out an analysis of variance with condition as the independent variable and rating as the dependent variable. We elicited a rating for all three conditions in each stimulus, therefore our experiment is a repeated measures design. Because the ratings are not normally distributed we will use the Friedman test, which analyses variance based on ranks rather than absolute values. The Friedman test does not indicate any variance in rating due to the experimental condition ($X^2(2) = 0.98$, $p = 0.612$).

This result supports both of our experimental hypotheses. The Friedman test shows there is no significant difference between the ratings given to the corpus and the model predictions, which confirms our hypothesis that they will be considered equally appropriate in context. It also shows there is no significant difference between the corpus and the baseline ratings, which indicates that finding the right circumstances for requests is difficult even for human experts.

In order to try to focus the analysis of the data on the question of entering solution step discussions, we investigated possible effects in subsets of the data which contain request actions. We first restricted the set of stimuli to those in which the action in the corpus was a request action, a set of 66 data points. The means now indicate the model condition being given even higher ratings on average (mean of model ratings = 4.53, corpus = 3.86, baseline 4.44), and a Friedman test again showed no significant effect of condition on the rating. For our second hypothesis this shows that when the cases are reduced to those where the wizards chose to request evidence, this request is still not being rated higher than the baseline by the experts. Similarly the subset of 59 data points in which the model predicted a request action also showed no effect of condition.

We also looked for differences in ratings in the data set reduced to those data points in which the student's most recent proof step had been annotated as incorrect by the wizard. This set contained 46 of the 110 instances. The motivation for this test was to see if the baseline still performs as well as the corpus in cases where a simple *Accept* is not a suitable answer. The baseline predicts a large majority of *Accept*, and these are highly rated, as shown in the frequency distribution in Figure 7.3. In this subset of the data the baseline receives 12 ratings of 1 and 10 ratings of 7, which contrasts strongly with the

distribution of the full data set. The means suggest that the baseline is being rated lower in this subset (mean rating of the model condition: 4.3, corpus: 4.02, baseline: 3.8) however a Friedman test again did not indicate any effect of condition on rating ($X^2(2) = 1.76$, $p = 0.419$).

Of the five actions, *Accept* received the highest rating most often, with 60 of the 151 occurrences, or 39.7%, being rated with 7. The next most frequently 7-rated action was *RejectReqEv* with five of 16 items, or 31.2%. *Reject*, *AcceptReqEv* and *ReqEv* followed with 22.2%, 18% and 8.5% respectively. This preference of the raters for high ratings of *Accept* explains in part why the baseline condition receives so many more 7 ratings than the other conditions, as shown in Figure 7.3.

We investigated whether type of teaching experience of the participants had an effect on the ratings they gave. The distribution of ratings by teaching experience of the participants showed 126 ratings from university lecturers, 171 from teaching assistants and 63 from schoolteachers. For the model and corpus conditions a Kruskal-Wallis test showed no significant effect of teaching experience on the rating. For the baseline condition there was a significant effect ($X^2(2) = 7.54$, $p < 0.05$). A set of post-hoc Wilcoxon tests within the baseline condition (mean rating from teaching assistants = 5.16, schoolteachers = 4.2, lecturers = 4.01) showed that teaching assistants gave significantly higher ratings than both schoolteachers ($W = 349.5$, $p <= 0.05$) and lecturers ($W = 706.5$, $p <= 0.05$). No other significant differences were found. No effect was found of the participants' university degree on their ratings.

As the final part of our analysis we considered whether we could analyse the full set of 304 predictions from which the stimuli were chosen. We gave artificial ratings to those data points which had been correctly predicted by the model and which therefore had not been considered for the generation of the experimental stimuli. We assigned all corpus actions the rating 4. When either the model or the baseline classifier predicted the correct TLGA, the model or the baseline condition was also given the artificial rating 4, in other words an equal rating compared to the corpus condition. If either the model or the baseline prediction or both were wrong then they received the rating 3. The actual number we use here is not important since we are using rank tests rather than absolute tests, so the magnitude of the difference is not considered.

This assignment of ranks gives us 255 new artificially rated instances. In 192 cases the model had predicted the correct TLGA, in 201 cases the baseline had predicted the correct action. We add this set to the 110 ratings from the experiment. A Friedman test shows an effect of condition on rating ($X^2(2) = 26.76$, $p < 0.001$), however the subsequent pairwise Wilcoxon tests are not powerful enough to find a significant difference between any two of the conditions. We

suspect the model condition exhibits significantly lower ratings than the other two in this set based on the means (model = 3.95, corpus = 4.04, baseline = 4.02), however because 255 of the 365 cases used artificial ratings, the mean values should not be interpreted.

7.4 Summary and discussion

The evaluation presented in this chapter was motivated by two hypotheses arising from the results of Chapter 6, in which we developed the classifier for TLGAs. Our first goal was to compare the TLGAs predicted by the model to those in the corpus. By asking human experts to rate the actions in context we rewarded cases where the model predicts an action which is different to that in the corpus but is considered equally appropriate. We hoped to find that the ratings given to the model condition were as good as the ratings given to the corpus condition. Our second goal was to show that choosing request actions is a difficult task for humans, motivated by the less than impressive results of the classifier for classes involving requesting evidence. To investigate this hypothesis the raters additionally rated a baseline which was a simple mapping from the correctness or incorrectness of the student's proof step to *Accept* or *Reject* respectively. If the task is indeed difficult for human experts then the baseline would be rated as highly as the corpus condition, showing that the raters were not able to detect the situations in which requesting evidence was the gold standard action. Our raters were mathematics experts with teaching experience, who rated a total of 110 stimuli.

Our first hypothesis was confirmed by the analysis of the experimental data. We found that the model condition was indeed rated equally highly in comparison to the corpus condition. This tells us that in cases where the model makes an inaccurate prediction, and considering we concentrated on request actions, the model's predictions are on average just as appropriate as the wizards' actual actions. Our second hypothesis was also confirmed. We found that the baseline ratings showed no significant difference in comparison to the corpus ratings. These two results persisted across two further sets of rated data, first the subset containing only those stimuli which were in fact request actions in the corpus, and second the superset of ratings including artificial ratings assigned based on accurate or inaccurate predictions by the model and baseline.

We were able to find some other significant results which were not due to the experimental condition. We observed that *Accept* received the highest rating more frequently in proportion to its total occurrences than three of the other TLGAs. Together with the distribution of baseline predictions which was strongly skewed towards *Accept*, this contributed to the baseline frequently

being awarded the highest rating. We also found that university teaching assistants awarded significantly higher ratings than both lecturers and schoolteachers in the baseline condition.

We can comment on some aspects of the experimental design which may have influenced the results. The experiment is valid because the raters who participated have the same mathematical expertise and teaching experience as the wizards who took part in the corpus collection experiment. Our stimuli were rated by a maximum of three experts, and there were higher ratings given by teaching assistants. However to show reliability we would need to have data from more participants. We have not ruled out possible differences in the preferred tutoring style of the participants. We do not know how these tutors would have handled the dialogue situations themselves, instead we must rely on their self-reporting of their behaviour through the rating task. Some tutors may favour a more socratic style, in which case they would rate request actions higher. Some may prefer more formal answers and would consider for instance missing solution step parts acceptable, and would not rate request actions as highly. Finally reading the transcript of a dialogue is faster than taking part in the dialogue itself, because the time taken by the students to think and type their replies was quite long. Also a reading-based task does not strictly require that what the student is saying is fully digested by the reader the way that taking part in the actual dialogue does. So we can not guarantee that raters took the time to construct the same mental models of their students' knowledge states as the tutors originally did.

Since the ratings in the experimental data are not normally distributed, we were forced to use non-parametric tests to analyse the data. This reduces the data to a ranking of conditions by items, which is exactly what we did not want our raters to do. The main drawback of this is that the magnitude of the differences between the conditions is lost, and it is possible that this may have obscured significant differences between the conditions.

This evaluation has extended the purely statistical evaluation of the classifier for TLGA in Chapter 6. The last chapter showed that a classifier trained on the data maintained by the dialogue model presented in Chapter 4 achieves an accuracy of 72.5% and while it is sensitive to the minority request classes, the precision and recall of the *ReqEv* decision are low, at 32.1% and 41.1%, respectively. In this chapter we chose a subset of the misclassifications and found them to be comparably appropriate in context, concentrating on the request actions. We also found that the baseline of only acceptance and rejection was rated as well as the other condition, which indicates that choosing *ReqEv* is a difficult task for human experts.

Overall we find that the model we have developed over the course of Chap-

ters 4 and 6 with its representation of dialogue context performs as well as can be expected for this task. The result that the ratings of the three conditions are not significantly different indicates that the model is doing just about as well as human experts at deciding when to request evidence from the student. The model achieves the results which can be achieved, first as measured by accuracy in Chapter 6, then by appropriateness rating in this intrinsic evaluation. Any claims over and above this we believe can only be substantiated by a full end-to-end evaluation experiment with learners, in which the performance of the model could be directly linked to learning gains.

8
Summary and outlook

This thesis offers a model for solution step discussions in natural language tutoring and for the conditions under which tutors pose explanatory questions to their students. At the core of the model is the notion of evidence of understanding: Solution steps are acceptable when the tutor believes the student has shown sufficient evidence of having understood the content necessary to perform those steps. Our work has drawn on elements of dialogue modelling, human communication research and pedagogical science.

We began by considering natural language tutoring and highlighting the fact that it has consistently been shown to lead to more effective learning gains than traditional classroom instruction. Among the reasons for this effectiveness which have been proposed and demonstrated in previous studies are the collaborative nature of tutoring and the opportunity during this collaboration both for tutors to ask direct questions and for students to answer them. The consensus is that questioning leads to explaining and self-explaining leads to learning. Our work is based on the influential theory of Graesser et al. [85] describing dialogue frames, the patterns in tutorial dialogue in which such collaborations occur. A dialogue frame includes a phase of collaborative improvement of the student's answer, in which the tutor can elaborate on the answer, give hints, pump the student for more information, or trace an explanation or justification.

The decision of whether to enter into the collaborative improvement phase depends on what the tutor believes the student's knowledge state to be, in other words the student's deep understanding of domain concepts. In the context of dialogue based tutoring this led us to consider models of belief and

agency in dialogue which have been proposed. An agent's actions are motivated by its beliefs and its beliefs about its conversational partners. They can reach agreement about their beliefs through the process of grounding, during which hearers are obliged to demonstrate evidence of understanding.

These two views on evidence of understanding suggested a possible benefit in combining them in a model for tutorial dialogue. Two research questions emerged with respect to the use of theories from dialogue modelling in ITSs with natural language interfaces. The first relates to modelling the student's knowledge state based on the tutor's feedback to solution steps:

- By modelling the structure of solution step discussions, can we maintain a representation of what the student has shown evidence of having understood, based on the outcomes of previous such discussions?

The second research question arises out of the observation that discussions elicit self-explanation, which in turn leads to learning:

- Can we use the representation of the student's evidence of understanding to predict whether to enter into a discussion of the latest contribution, using features drawn from the previous dialogue and from the analysis of the steps the student previously contributed?

We investigated these hypotheses according to the following methodology. We first analysed a corpus of human-human tutorial dialogues on mathematical theorem proving and categorised the kinds of actions that tutors and students perform. We found a broad range of types in and around solution step discussions, including statements of content, different kinds of feedback from tutors, hints, and requests. We also looked for a found evidence of the occurrence of dialogue frames. We concentrated in our qualitative analysis on how tutors ask questions to elicit self-explanations from students, and based on this we proposed the domain independent notion of task-level grounding. It is the process by which the tutor and the student reach mutual belief about the student's deep understanding of domain concepts. The student offers evidence of this understanding by successfully using these concepts in proof steps, and the tutor can demand such evidence by posing an explanatory question. Task-level grounding takes place in the course of a solution step discussion, in which a solution step is proposed, may be discussed, and is finally either accepted or rejected. Our model of solution step discussions offers an alternative, operational account of dialogue frames, conceived in the style of Traum's Grounding Acts theory [191].

In order to test our model, and in order to obtain labelled data to train a classifier with, we annotated our corpus at three levels. We annotated with a set

of task-level grounding acts, which are the actions performed by students and tutors in the process of task-level grounding. We also annotated the mathematical content of the solution steps in the corpus, and for a subset we annotated general dialogue moves. Using the annotated data we performed a series of supervised learning experiments. The goal was to train a classifier which could predict, given a dialogue context and the analysis of a solution step from the student, whether to enter into a discussion with the student about the step. The classifier predicted the actions *Accept*, *Reject*, *ReqEv*, and combinations of those.

The classifier's performance improved with the addition of features derived from the previous dialogue, which indicates that the information maintained by our dialogue model does indeed help decide whether to enter into solution discussions or not. We found that the performance of the classifier was good for acceptance and rejection, but for the detection of cases where evidence of understanding should be requested, the performance was unsatisfactory. This was in part due to the skewed distribution of classes, but we also hypothesised that the reason may lie in the intrinsic difficulty of the task. To test this we performed an evaluation experiment using ratings of appropriateness elicited from expert human tutors reading excerpts from the corpus. They rated predictions from the classifier, from an accept/reject baseline, and from the corpus itself. We found no significant difference between the three conditions, which indicates that the model is performing comparably well in relation to the gold standard corpus. The similarity of the baseline and corpus ratings indicates further that humans also have difficulty with the task of recognising situations in which a request for evidence of understanding should occur. The performance of the classifier model should be seen in this light.

The model we have proposed and our evaluation results allows us to consider our hypotheses largely confirmed. In terms of the first hypothesis, the fact that the classifier relies of features from the dialogue history shows that the model maintains information which is relevant to the choice of whether to enter into solution step discussions or not. Our second hypothesis is supported by the results of our evaluation experiment, in which we found that the model's predictions were seen as being equally appropriate compared to the actions that were taken by tutors in the corpus. The low precision and recall values of the decision whether to request evidence or not are mitigated by our finding that human experts also find this decision difficult.

While we have shown some potential benefits of using grounding and tutoring to structure solution step discussions, much work remains to be done, primarily due to a number of approximations we have made in our model. The model does not differentiate between types of request which a tutor may

make. We see the model as offering an initial decision, that a request should be made, but leaving the form of this request to a teaching strategy. A combination of these would allow more adaptive feedback. We also do not consider degrees of mastery of concepts, instead they are either known or not. A more sophisticated student model could reason over grades of evidence of understanding. As we mentioned in the discussion of the evaluation experiment, an end-to-end evaluation with students was not possible within the resources of this research project. We hope that such an experiment would confirm the utility of the model by linking it directly to observed learning gains. We thus see many opportunities for fruitful future projects.

ITSs are not yet ubiquitous in the schooling system, but there are already some notable success stories, such as Cognitive Tutors, which are widely applied and achieve good results. Considering the fact that dialogue systems in general are not in as widespread use as was once thought they would be by now, maybe it is no wonder that tutorial dialogue systems are lagging somewhat behind. Not only are they subject to the same problems as other dialogue systems, such as interpretation robustness and domain reasoning, it seems that tutoring is just a hard task for a computer. Nevertheless, the interdisciplinary field of natural language tutoring is showing consistent evaluation successes. We hope this thesis will contribute to the closer unification of approaches from the directions of both dialogue modelling and pedagogical science, and in turn to their eventual widespread success in the classroom.

A

Mathematical rule groupings

This appendix lists a number of subsets of the set of mathematical rules taken from the study material. They are used in the preparation of the corpus for the supervised learning experiments.

Subset name	Rules
Interesting	exten, composition, inverse, identity1, identity2, identity3
Uninteresting	prop, quant, subset, psubset, set_prop, union, intersection, declaration
Set def	subset, psubset, set_prop, union, intersection
Relation def	composition, inverse
Identity	identity1, identity2, identity3

B
Participant background questionnaire

The following is the text of the background questionnaire given to prospective participants in the evaluation study reported in Chapter 7.

Liebe(r) Bewerter(in),
danke, dass Sie sich für unser Experiment interessieren, das im Rahmen einer Doktorarbeit an der Universität des Saarlandes durchgeführt wird. Im Experiment geht es um binäre Relationen im Matheunterricht. Es gibt eine Voraussetzung für die Teilnahme:

- Haben Sie bereits mindestens 1 Semester Mathe in deutscher Sprache unterrichtet, z.B. als Übungsleiter oder Lehrer?

Wenn ja, können Sie bei uns mitmachen.
Bevor es losgeht, haben wir noch ein paar Fragen zu Ihrem Hintergrund:

- Ihr Studium:
  - Mathe
  - Mathe mit Didaktik, z.B. Lehramt
  - Mathenahes Studium, z.B. Physik oder Informatik
  - anderes Studium

- Ihre Erfahrung als Mathelehrer
  - Schule
  - Uni, als Übungsleiter
  - Uni, als Dozent
- Vor wie vielen Semestern haben Sie zum letzten mal Mathe unterrichtet?
- Für wieviele Semester haben Sie Grundlagen der Mathematik, z.B. Mengentheorie, Relationen, insbesondere das Beweisen, unterrichtet?
- Wie viel Erfahrung haben Sie im Unterrichten von formaler Logik oder Beweistheorie?
  - Keine
  - Ein wenig
  - Mittel
  - Viel

C
Introductory text

The following is the introductory text describing the task given to the participants in the evaluation study reported in Chapter 7.

Hallo Max Mustermann,
in diesem Experiment geht es darum, das Verhalten von Mathetutoren in Dialogen mit Studenten zu bewerten. Die Dialoge wurden in folgendem Szenario aufgezeichnet: Die Studenten, die im ersten Semester waren, sollten Beweisaufgaben aus dem Bereich der binären Relationen lösen. Die Tutoren haben auf die Eingaben von den Studenten jeweils Rückmeldung gegeben. Die Tutoren und Studenten konnten ausschließlich per Text miteinander kommunizieren und haben sich vor dem Unterricht nicht gekannt. Hier ist ein Beispiel für die Aufgabenstellung und den Ablauf einer Interaktion:

S0: was ist $\circ$

T0: Das Relationenprodukt, auch Komposition von Relationen genannt. Bitte schauen Sie sich die Definition unter Abschnitt 4 an.

S1: $(R \circ S)^{-1} = \{(z,x) | \exists y((x,y) \in R \wedge (y,z) \in S\}$

T1: Das ist korrekt.

S2: $R^{-1} = \{(x,y) | (y,x) \in R\}$

T2: Ebenfalls korrekt.

S3: Also ist $S^{-1} \circ R^{-1} = \{(v,x)|v \in S^{-1} \wedge x \in R^{-1}\}$

T3: Nein. Auch die inversen Relationen, S^{-1} und R^{-1} , sind binaere Relationen!

S4: Also ist $S^{-1} \circ R^{-1} = \{(v,x)|\exists z((v,z) \in S^{-1} \wedge (z,x) \in R^{-1}\}$

T4: Das ist korrekt.

S5: Das ist gleich $\{(v,x)|\exists z((z,v) \in S \wedge (x,z) \in R\}$

T5: Das ist korrekt. Aufgabe geloest!

Der Tutor hat mehrere Möglichkeiten, wie er auf die Eingabe des Studenten reagieren kann. Er kann sie zum Beispiel akzeptieren oder ablehnen, er kann Fehler beheben, Details verlangen, Hinweise geben, oder versuchen den Studenten zu motivieren.

Ihre Aufgabe

Ihre Aufgabe ist es, in einem Ausschitt aus einem solchen Dialog mögliche Reaktion des Tutors auf die letzte Eingabe des Studenten zu bewerten. Sie sollen angeben, für wie angemessen Sie diese Reaktion halten (mathematisch und didaktisch), auf einer Skala von 1 (sehr unangemessen) bis 7 (sehr angemessen). Eine Reaktion ist angemessen, wenn sie zu den aktuellen Kenntnissen des/der Studierenden passt und einen möglichen Lerneffekt befördert.

Versuchen Sie beim Lesen, sich in die Rolle des Tutors hineinzuversetzen und zu verstehen, welche Lösung der Student präsentiert. So können Sie die Angemessenheit der Reaktion am besten einschätzen. Sie werden sieben Dialoge sehen, in denen jeweils zwei Situationen bewertet werden sollen. Erst nachdem Sie die erste Situation bewertet haben, sehen Sie wie der Dialog eigentlich weiterging, um danach die zweite Situation zu bewerten.

Personenbezogene Daten werden nicht gespeichert. Danke dass Sie an unserem Experiment teilnehmen!

Bibliography

[1] Andreas Abel, Bor-Yuh Evan Chang, and Frank Pfenning. Human-Readable Machine-Verifiable Proofs for Teaching Constructive Logic. In Uwe Egly, Armin Fiedler, Helmut Horacek, and Stephan Schmitt, editors, *Proceedings of the Workshop on Proof Transformations, Proof Presentations and Complexity of Proofs*. Universitá degli studi di Siena, 2001.

[2] Vincent Aleven, Kenneth R. Koedinger, and K. Cross. Tutoring answer explanation fosters learning with understanding. In Susanne P. Lajoie and Martial Vivet, editors, *Proceedings of Artificial Intelligence in Education, Open Learning Environments: New Computational Technologies to Support Learning, Exploration, and Collaboration*, pages 199–206, Amsterdam, 1999. IOS Press.

[3] Vincent Aleven, Octav Popescu, and Kenneth R. Koedinger. Towards Tutorial Dialog to Support Self-Explanation: Adding Natural Language Understanding to a Cognitive Tutor. In Johanna D. Moore, Carol L. Redfield, and W. Lewis Johnson, editors, *Proceedings of the 9th International Conference on Artificial Intelligence in Education*, pages 246–255. IOS Press, 2001.

[4] Vincent Aleven, Octav Popescu, Amy Ogan, and Kenneth R. Koedinger. A Formative Classroom Evaluation of a Tutorial Dialog System that Supports Self-Explanation. In Vincent Aleven, Ulrich Hoppe, Judy Kay, Riichiro Mizoguchi, Helen Pain, Felisa Verdejo, and Kalina Yacef, editors, *Supplemental Proceedings of the 11th International Conference on Artificial Intelligence in Education, Vol. VI*, pages 303–312, 2003.

[5] James Allen and Mark Core. Draft of DAMSL: Dialogue act markup in several layers. *DRI: Discourse Research Initiative*, University of Pennsylvania, 1997.

[6] Ethem Alpaydın. *Introduction to machine learning*. MIT Press, 2004.

[7] John R. Anderson, Albert T. Corbett, Kenneth R. Keodinger, and Ray Polletier. Cognitive tutors: Lessons learned. *The Journal of Learning Sciences*, 4:167–207, 1995.

[8] Peter B. Andrews, Chad E. Brown, Frank Pfenning, Matthew Bishop, Sunil Issar, and Hongwei Xi. Etps: A system to help students write formal proofs. *Journal of Automated Reasoning*, 32:75–92, 2004.

[9] Elizabeth Arnott, Peter Hastings, and David Allbritton. Research Methods Tutor: Evaluation of a dialogue-based tutoring system in the classroom. *Behavior Research Methods*, 40:694–698, 2008.

[10] Harald Aust, Martin Oerder, Frank Seide, and Volker Steinbiss. The Philips Automatic Train Timetable Information System. In *Speech Communication*, volume 17, pages 249–262, 1995.

[11] John L. Austin. *How to Do Things with Words*. Harvard University Press, 1996.

[12] Serge Autexier, Christoph Benzmüller, Dominik Dietrich, Andreas Meier, and Claus-Peter Wirth. A Generic Modular Data Structure for Proof Attemps Alternating on Ideas and Granularity. In *Proceedings of Mathematical Knowledge Management*, pages 12–14, Bremen, Germany, 2005. Springer.

[13] Michael Baker, Tia Hansen, Richard Joiner, and David Traum. The role of grounding in collaborative learning tasks. In Pierre Dillenbourg, editor, *Collaborative Learning. Cognitive and computational approaches*, Advances in Learning and Instruction Series, pages 31–63. Pergamon, Amsterdam, The Netherlands, 1999.

[14] Chris Benzmüller, Armin Fiedler, Malte Gabsdil, Helmut Horacek, Ivana Kruijff-Korbayová, Manfred Pinkal, Jörg Siekmann, Dimitra Tsovaltzi, Bao Quoc Vo, and Magdalena Wolska. Tutorial dialogs on mathematical proofs. In *Proceedings of the IJCAI Workshop on Knowledge Representation and Automated Reasoning for E-Learning Systems*, pages 12–22, Acapulco, 2003.

[15] Chris Benzmüller, Armin Fiedler, Malte Gabsdil, Helmut Horacek, Ivana Kruijff-Korbayová, Dimitra Tsovaltzi, Bao Quoc Vo, and Magdalena Wolska. Language phenomena in tutorial dialogs on mathematical proofs. In Ivana Kruijff-Korbayová and Claudia Kosny, editors, *Proceedings of DiaBruck'03, the 7th Workshop on the Semantics and Pragmatics of Language*, pages 165–166, Saarbrücken, Germany, 2003.

[16] Christoph Benzmüller and Quoc Bao Vo. Mathematical domain reasoning tasks in natural language tutorial dialog on proofs. In Manuela Veloso and Subbarao Kambhampati, editors, *Proceedings of the 20th National Conference on Artificial Intelligence*, pages 516–522, Pittsburgh, Pennsylvania, USA, 2005. AAAI Press / The MIT Press.

[17] Christoph Benzmüller, Helmut Horacek, Ivana Kruijff-Korbayová, Henri Lesourd, Marvin Schiller, and Magdalena Wolska. DiaWOz-II - A Tool for Wizard-of-Oz Experiments in Mathematics. In *Proceedings of KI 2006: Advances in Artificial Intelligence, the 29th Annual German Conference on AI*, volume 4314 of *Lecture Notes in Computer Science*, pages 159–173. Springer, 2006.

[18] Christoph Benzmüller, Helmut Horacek, Henri Lesourd, Ivana Kruijff-Korbayová, Marvin Schiller, and Magdalena Wolska. A corpus of tutorial dialogs on theorem proving; the influence of the presentation of the study-material. In *Proceedings of the International Conference on Language Resources and Evaluation*, Genoa, Italy, 2006. ELDA.

[19] Christoph Benzmüller, Dominik Dietrich, Marvin Schiller, and Serge Autexier. Deep Inference for Automated Proof Tutoring? In *Proceedings of KI 2007: Advances in Artificial Intelligence*, volume 4667 of *Lecture Notes in Computer Science*, pages 435–439. Springer, 2007.

[20] Christoph Benzmüller, Helmut Horacek, Ivana Kruijff-Korbayova, Manfred Pinkal, Jörg Siekmann, and Magdalena Wolska. Natural Language Dialog with a Tutor System for Mathematical Proofs. In Ruqian Lu, Jörg Siekmann, and Carsten Ullrich, editors, *Cognitive Systems*, volume 4429 of *LNAI*, pages 1–14. Springer, 2007.

[21] Nate Blaylock and James Allen. A Collaborative Problem-Solving Model of Dialogue. In Laila Dybkjær and Wolfgang Minker, editors, *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*, pages 200–211, Lisbon, 2005.

[22] Benjamin S. Bloom. The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring. *Educational Researcher*, 13(6):4–16, 1984.

[23] Johan Bos, Stina Ericsson, Staffan Larsson, Ian Lewin, Peter Ljunglöf, and Colin Matheson. Dialogue dynamics in restricted dialogue systems. TRINDI Deliverable D3.2, 2000.

[24] Kristy Boyer, Rob Phillips, Eun Young Ha, Michael Wallis, Mladen Vouk, and James Lester. Leveraging hidden dialogue state to select tutorial moves. In *Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 66–73, Los Angeles, California, June 2010. Association for Computational Linguistics.

[25] Ilya N. Bronstein and Konstantin A. Semendjajew. *Taschenbuch der Mathematik*. Teubner, 1991.

[26] Mark Buckley and Christoph Benzmüller. An Agent-based Architecture for Dialogue Systems. In Irina Virbitskaite and Andrei Voronkov, editors, *Proceedings of Perspectives of System Informatics*, volume 4378 of *Lecture Notes in Computer Science*, pages 135–147, Novosibirsk, Russia, 2006. Springer.

[27] Mark Buckley and Christoph Benzmüller. System Description: A Dialogue Manager supporting Natural Language Tutorial Dialogue on Proofs. In David Aspinall and Christoph Lüth, editors, *Proceedings of the ETAPS Satellite Workshop on User Interfaces for Theorem Provers (UITP)*, pages 40–67, Edinburgh, Scotland, 2005.

[28] Mark Buckley and Dominik Dietrich. Integrating Task Information into the Dialogue Context for Natural Language Mathematics Tutoring. In Ben Medlock and Diarmuid Ó Séaghdha, editors, *Proceedings of the 10th Annual CLUK Research Colloquium*, Cambridge, UK, 2007.

[29] Mark Buckley and Magdalena Wolska. A Classification of Dialogue Actions in Tutorial Dialogue. In Donia Scott and Hans Uszkoreit, editors, *Proceedings of COLING 2008, The 22nd International Conference on Computational Linguistics*, pages 73–80, Manchester, UK, 2008. Coling 2008 Organizing Committee.

[30] Mark Buckley and Magdalena Wolska. Towards Modelling and Using Common Ground in Tutorial Dialogue. In Ron Artstein and Laure Vieu, editors, *Proceedings of DECALOG, the 2007 Workshop on the Semantics and Pragmatics of Dialogue*, pages 41–48, Rovereto, Italy, 2007.

[31] Mark Buckley and Magdalena Wolska. A Grounding Approach to Modelling Tutorial Dialogue Structures. In Jonathan Ginzburg, Pat Healey, and Yo Sato, editors, *Proceedings of LONDIAL 2008, the 12th Workshop on the Semantics and Pragmatics of Dialogue*, pages 15–22, London, UK, 2008.

[32] Charles Callaway, Myroslava Dzikovska, Colin Matheson, Johanna D. Moore, and Claus Zinn. Using Dialogue to Learn Math in the LeActiveMath Project. In *Proceedings of the ECAI Workshop on Language-Enhanced Educational Technology*, pages 1–8, Riva del Garda Italy, August 2006.

[33] Charles Callaway, Myroslava Dzikovska, Elaine Farrow, Manuel Marques-Pita, Colin Matheson, and Johanna Moore. The Beetle and BeeDiff tutoring systems. In *Proceedings of the SLaTE2007 Workshop*, 2007.

[34] Charles B. Callaway and Johanna D. Moore. Determining tutorial remediation strategies from a corpus of human-human tutoring dialogues. In *Proceedings of the 11th European Workshop on Natural Language Generation*, Schloss Dagstuhl, Germany, June 2007.

[35] Sandra Carberry. *Plan Recognition in Natural Language Dialogue*. MIT Press, 1990.

[36] J. R. Carbonell. AI in CAI: An Artificial-Intelligence Approach to Computer-Assisted Instruction. *IEEE Transactions on Man-Machine Systems*, 11(4):190–202, 1970.

[37] Jean Carletta. Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22:249–254, 1996.

[38] Jean Carletta, Amy Isard, Stephen Isard, Jacqueline C. Kowtko, Gwyneth Doherty-Sneddon, and Anne H. Anderson. The reliability of a dialogue structure coding scheme. *Computational Linguistics*, 23 (1):13–32, 1997.

[39] Alison Cawsey. Planning interactive explanations. *International Journal of Man-Machine Studies*, 38:169–199, 1993.

[40] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[41] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: Synthetic Minority Over-sampling TEchnique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[42] Michelene T. H. Chi. Constructing Self-Explanations and Scaffolded Explanations in Tutoring. *Applied Cognitive Psychology*, 10:33–49, 1996.

[43] Michelene T. H. Chi, Miriam Bassok, Matthew W. Lewis, Peter Riemann, and Robert Glaser. Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science*, 13:145–182, 1989.

[44] Michelene T. H. Chi, Nicholas de Leeuw, Mei-Hung Chiu, and Christian Lavancher. Eliciting self-explanation improves understanding. *Cognitive Science*, 18:439–477, 1994.

[45] Michelene T. H. Chi, Stephanie A. Siler, Heisawn Jeong, Takashi Yamauchi, and Robert G. Hausmann. Learning from human tutoring. *Cognitive Science*, 25:471–533, 2001.

[46] Michelene T. H. Chi, Stephanie A. Siler, and Heisawn Jeong. Can Tutors Monitor Students' Understanding Accurately? *Cognition and Instruction*, 22(3):363–387, 2004.

[47] Min Chi, Pamela Jordan, Kurt VanLehn, and Moses Hall. Reinforcement learning-based feature selection for developing pedagogically effective tutorial dialogue tactics. In Ryan S. J. de Baker, Tiffany Barnes, and Joseph E. Beck, editors, *Proceedings of the 1st International Conference on Educational Data Mining*, pages 258–265, 2008.

[48] Herbert H. Clark. *Using Language*. Cambridge University Press, 1996.

[49] Herbert. H. Clark, editor. *Arenas of Language Use*. University of Chicago Press, 1992.

[50] Herbert H. Clark and Susan E. Brennan. Grounding in communication. In Lauren B. Resnick, John M. Levine, and Stephanie D. Teasley, editors, *Perspectives on socially shared cognition*, pages 127–149. APA, Washington, DC, 1991.

[51] Herbert H. Clark and Emanuel F. Schaefer. Contributing to discourse. *Cognitive Science*, 13:259–294, 1989.

[52] Herbert H. Clark and Deanna Wilkes-Gibbs. Referring as a collaborative process. *Cognition*, 22(1):1–39, 1986.

[53] Jacob Cohen. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37, 1960.

[54] Peter A. Cohen, James A. Kulik, and Chen-Lin C. Kulik. Educational Outcomes of Tutoring: A Meta-Analysis of Findings. *American Educational Research Journal*, 19(2):237–248, 1982.

[55] Philip R. Cohen and Hector J. Levesque. Teamwork. *Noûs*, 25(4):487–512, 1991.

[56] Philip R. Cohen and C. Raymond Perrault. Elements of a Plan-Based Theory of Speech Acts. *Cognitive Science*, 3(1):177–212, 1979.

[57] Philip R. Cohen, Hector J. Levesque, José H. T. Nunes, and Sharon L. Oviatt. Task Oriented Dialogue as a Consequence of Joint Activity. In *Artificial Intelligence in the Pacific Rim*. IOS Press, Amsterdam, 1991.

[58] Albert T. Corbett, Kenneth R. Koedinger, and John R. Anderson. Intelligent tutoring systems. In Martin Helander, Thomas K. Landauer, and Prasad V. Prabhu, editors, *Handbook of Human-Computer Interaction*, chapter 37, pages 849–874. Elsevier Science, second edition, 1997.

[59] Mark G. Core, Johanna D. Moore, and Claus Zinn. Supporting constructive learning with a feedback planner. In *Proceedings of the AAAI Fall Symposium: Building Dialogue Systems for Tutorial Applications*, Falmouth, MA, 2000. AAAI Press.

[60] Mark G. Core, Johanna D. Moore, and Claus Zinn. The role of initiative in tutorial dialogues. In *The 10th Conference of the European Chapter of the ACL*, pages 67–74, Budapest, 2003.

[61] Marcello D'Agostino and Ulrich Endriss. WinKE: A Proof Assistant for Teaching Logic. In *Proceedings of the 1st International Workshop on Labelled Deduction*, 1998.

[62] Nils Dahlbäck, Arne Jönsson, and Lars Ahrenberg. Wizard of Oz studies: why and how. In *Proceedings of the 1st International Conference on Intelligent User Interfaces*, pages 193–200. ACM, 1993.

[63] René V. Dawis. Scale construction. *Journal of Counseling Psychology*, 34(4):481 – 489, 1987.

[64] David DeVault and Matthew Stone. Scorekeeping in an uncertain language game. In *Proceedings of Brandial, the 10th Workshop on the Semantics and Pragmatics of Dialogue*, pages 139–146, Potsdam, Germany, 2006.

[65] Barbara Di Eugenio, Davide Fossati, Susan Haller, Dan Yu, and Michael Glass. Be Brief, And They Shall Learn: Generating Concise Language Feedback for a Computer Tutor. *International Journal of Artificial Intelligence in Education*, 18(4):317–345, 2008.

[66] Dominik Dietrich and Mark Buckley. Verification of Proof Steps for Tutoring Mathematical Proofs. In Rosemary Luckin, Kenneth R. Koedinger, and Jim Greer, editors, *Proceedings of the 13th International Conference on Artificial Intelligence in Education*, volume 158, pages 560–562, Los Angeles, USA, 2007. IOS Press.

[67] Dominik Dietrich and Mark Buckley. Verification of Human-level Proof Steps in Mathematics Education. *Teaching Mathematics and Computer Science*, 6(2):345–362, 2008.

[68] Pierre Dillenbourg. What do you mean by collaborative learning? In Pierre Dillenbourg, editor, *Collaborative-learning: Cognitive and Computational Approaches*, pages 1–19. Elsevier, 1999.

[69] Myroslava O. Dzikovska, Charles B. Callaway, Matthew Stone, and Johanna D. Moore. Understanding student input for tutorial dialogue in procedural domains. In David Schlangen and Raquel Fernandez, editors, *Proceedings of the Brandial, the 10th Workshop on the Semantics and Pragmatics of Dialogue*, pages 10–17, 2006.

[70] Myroslava O. Dzikovska, Charles B. Callaway, Elaine Farrow, Johanna D. Moore, Natalie Steinhauser, and Gwendolyn Campbell. Dealing with interpretation errors in tutorial dialogue. In *Proceedings of the SIGDIAL 2009 Conference*, pages 38–45. Association for Computational Linguistics, 2009.

[71] Susanna S. Epp. The Role of Logic in Teaching Proof. *American Mathematical Monthly*, 110(10):886–899, 2003.

[72] Martha W. Evens, Stefan Brandle, Ru-Charn Chang, Reva Freedman, Michael Glass, Yoon Hee Lee, Leem Seop Shim, Chong Woo Woo, Yuemei Zhang, Yujian Zhou, Joel A. Michael, and Allen A. Rovick. Circsim-Tutor: An intelligent tutoring system using natural language

dialogue. In *Proceedings of the 12th Midwest AI and Cognitive Science Conference*, pages 16–23, Oxford, 2001.

[73] Raquel Fernández, Jonathan Ginzburg, and Shalom Lappin. Classifying Non-Sentential Utterances in Dialogue: A Machine Learning Approach. *Computational Linguistics*, 33(3):397–427, 2007.

[74] Armin Fiedler and Dimitra Tsovaltzi. Automating Hinting in Mathematical Tutorial Dialogue. In *Proceedings of the EACL-03 Workshop on Dialogue Systems: Interaction, Adaptation and Styles of Management*, pages 45–52, Budapest, 2003.

[75] Armin Fiedler and Dimitra Tsovaltzi. Automating Hinting in an Intelligent Tutorial System. In *Proceedings of the IJCAI Workshop on Knowledge Representation and Automated Reasoning for E-Learning Systems*, pages 23–35, Acapulco, 2003.

[76] Reva Freedman. Using a Text Planner to Model the Behavior of Human Tutors in an ITS. In Michael Gasser, editor, *Proceedings of the 1996 Midwest Artificial Intelligence and Cognitive Science Conference*, 1996.

[77] Reva Freedman, Carolyn Penstein Rosé, Michael A. Ringenberg, and Kurt VanLehn. ITS Tools for Natural Language Dialogue: A Domain-Independent Parser and Planner. In *Proceedings of the 5th International Conference on Intelligent Tutoring Systems*, pages 433–442, London, UK, 2000. Springer.

[78] Abigail S. Gertner, Christina Conati, and Kurt VanLehn. Procedural Help in Andes: Generating Hints using a Bayesian network student model. In *Proceedings of the 15th National Conference on Artificial Intelligence*, pages 106–111, Madison, Wisconsin, 1998.

[79] Jonathan Ginzburg. Interrogatives: Questions, facts and dialogue. In Shalom Lappin, editor, *The Handbook of Contemporary Semantic Theory*. Blackwell, Oxford, 1996.

[80] Jonathan Ginzburg. Dynamics and the semantics of dialogue. In Jerry Seligman and Dag Westerstahl, editors, *Language, Logic and Computation, Volume 1*, CSLI Lecture Notes, pages 221–237. CSLI, Stanford, 1996.

[81] Michael Glass. Processing Language Input in the CIRCSIM-Tutor Intelligent Tutoring System. In Johanna Moore, Carol Luckhardt Redfield,

and W. Lewis Johnson, editors, *Proceedings of the 9th International Conference on Artificial Intelligence in Education*, pages 210–212. IOS Press, 2001.

[82] John J. Godfrey, Edward C. Holliman, and Jane McDaniel. SWITCHBOARD: Telephone speech corpus for research and development. In *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 517–520, San Francisco, 1992.

[83] David Good. Asymmetry and accommodation in tutorial dialogues. In R. Beun, M. Baker, and M. Reiner, editors, *Dialogue and Instruction*, volume 142 of *NATO ASI Series F*, pages 31–38. Springer, 1995.

[84] Arthur C. Graesser and Natalie K. Person. Question Asking During Tutoring. *American Educational Research Journal*, 31(1):104–137, 1994.

[85] Arthur C. Graesser, Natalie K. Person, and Joseph P. Magliano. Collaborative dialogue patterns in naturalistic one-on-one tutoring. *Applied Cognitive Psychology*, 9:495–522, 1995.

[86] Arthur C. Graesser, William Baggett, and Kent Williams. Question-driven explanatory reasoning. *Applied Cognitive Psychology*, 10:17–31, 1996.

[87] Arthur C. Graesser, Katja Wiemer-Hastings, Peter Wiemer-Hastings, and Roger Kreuz. Autotutor: A simulation of a human tutor. *Cognitive Systems Research*, 1:35–51, 1999.

[88] Arthur C. Graesser, George T. Jackson, Eric C. Mathews, Heather H. Mitchell, Andrew Olney, Mathew Ventura, Patrick Chipman, Donald R. Franceschetti, Xiangen Hu, Max M. Louwerse, Natalie K. Person, and TRG. Why/AutoTutor: A Test of Learning Gains from a Physics Tutor with Natural Language Dialog. In Richard Alterman and David Hirsh, editors, *Proceedings of the 25rd Annual Conference of the Cognitive Science Society*, pages 1–5. Cognitive Science Society, 2003.

[89] Arthur C. Graesser, Shulan Lu, George Tanner Jackson, Heather Hite Mitchell, Mathew Ventura, Andrew Olney, and Max M. Louwerse. AutoTutor: A tutor with dialogue in natural language. *Behavior Research Methods, Instruments, & Computers*, 36(2):180–192, 2004.

[90] Barbara J. Grosz and Sarit Kraus. Collaborative plans for complex group action. *Artificial Intelligence*, 86(2):269–357, 1996.

[91] Barbara J. Grosz and Candace L. Sidner. Plans for discourse. In Philip R. Cohen, Jerry L. Morgan, and Martha E. Pollack, editors, *Intentions in Communication*, pages 417–444. MIT Press, Cambridge, MA, 1990.

[92] Barbara J. Grosz and Candace L. Sidner. Attention, intention and the structure of discourse. *Computational Linguistics*, 12(3):175–204, 1986.

[93] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[94] Robert G. M. Hausmann, Michelene T. H. Chi, and Marguerite Roy. Learning from collaborative problem solving: An analysis of three hypothesized mechanisms. In Kenneth D Forbus, Dedre Gentner, and Terry Regier, editors, *26th Annual Conference of the Cognitive Science Society*, pages 547–552, 2004.

[95] Peter A. Heeman and James Allen. The TRAINS 93 dialogues. Technical note 94-2, University of Rochester, Rochester, New York, 1995.

[96] Neil T. Heffernan and Kenneth R. Koedinger. Building a 3rd generation ITS for symbolization: Adding a tutorial model with multiple tutorial strategies. In *Proceedings of the ITS 2000 Workshop on Algebra Learning*, Montréal, Canada, 2000.

[97] Charles T. Hemphill, John Godfrey, and George R. Doddington. The ATIS spoken language systems pilot corpus. In *Proceedings DARPA Speech and Natural Language Workshop*, pages 96–101, Hidden Valley, PA, 1990. Morgan Kaufmann.

[98] Reuben Hersh. Proving is convincing and explaining. *Educational Studies in Mathematics*, 24:389–399, 1993.

[99] Rania Hodhod and Daniel Kudenko. Interactive Narrative and Intelligent Tutoring for Ill Defined Domains. In *Proceedings of the ITS-2008 workshop on Intelligent Tutoring Systems for Ill-Defined Domains*, pages 23–27, 2008.

[100] Helmut Horacek and Magdalena Wolska. Interpreting Semi-Formal Utterances in Dialogs about Mathematical Proofs. *Data and Knowledge Engineering Journal*, 58(1):90–106, 2006.

[101] Helmut Horacek and Magdalena Wolska. Interpretation of Mixed Language Input in a Mathematics Tutoring System. In *Proceedings of*

AIED-05 Workshop on Mixed Language Explanations in Learning Environments, pages 27–34, 2005.

[102] Helmut Horacek and Magdalena Wolska. Handling errors in mathematical formulas. In Mitsuru Ikeda, Kevin D. Ashley, and Tak-Wai Chan, editors, *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, volume 4053 of *Lecture Notes in Computer Science*, pages 339–348. Springer, 2006.

[103] Xiaorong Huang. Reconstructing Proofs at the Assertion Level. In Alan Bundy, editor, *Proceedings of the 12th Conference on Automated Deduction*, pages 738–752. Springer, 1994.

[104] Gregory D. Hume, Joel A. Michael, Rovick A. Allen, and Martha W. Evens. Hinting as a tactic in one-on-one tutoring. *Journal of the Learning Sciences*, 5(1):23–47, 1996.

[105] Susan Jamieson. Likert scales: how to (ab)use them. *Medical Education*, 38(12):1217–1218, 2004.

[106] Nathalie Japkowicz. Learning from Imbalanced Data Sets: A Comparison of Various Strategies. In Nathalie Japkowicz, editor, *Proceedings of Learning from Imbalanced Data Sets*, pages 10–15. AAAI Press, 2000.

[107] George H. John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In Philippe Besnard and Steve Hanks, editors, *Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Mateo, 1995. Morgan Kaufmann.

[108] Pamela Jordan, Maxim Makatchev, Umarani Pappuswamy, Kurt VanLehn, and Patricia Albacete. A Natural Language Tutorial Dialogue System for Physics. In Geoff C. J. Sutcliffe and Randy G. Goebel, editors, *Proceedings of the 19th International FLAIRS Conference*, pages 521–526, 2006.

[109] Cezary Kaliszyk, Freek Wiedijk, Maxim Hendriks, and Femke van Raamsdonk. Teaching logic using a state-of-the-art proof assistant. In Herman Geuvers and Pierre Courtieu, editors, *Proceedings of the International Workshop on Proof Assistants and Types in Education*, pages 33–48, 2007.

[110] Fairouz Kamareddine, Manuel Maarek, Krzysztof Retel, and Joe Wells. Digitised mathematics: Computerisation vs. formalisation. *Review of the National Center for Digitization*, 10:1–8, 2007.

[111] Hans Kamp and Uwe Reyle. *From Discourse to Logic*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1993.

[112] Elena Karagjosova. Marked informationally redundant utterances in tutorial dialogue. In Ivana Kruijff-Korbayová and Claudia Kosny, editors, *Proceedings of DiaBruck'03, the 7th Workshop on the Semantics and Prgmatics of Language*, Saarbrücken, Germany, 2003.

[113] Kenneth Koedinger and John R. Anderson. Reifying implicit planning in geometry: Guidelines for model-based intelligent tutoring system design. In Susanne P. Lajoie and Sharon J. Derry, editors, *Computers as Cognitive Tools*, pages 15–46. Erlbaum, 1993.

[114] Jörn Kreutel and Colin Matheson. Incremental Information State Updates in an Obligation-Driven Dialogue Model. *Logic Journal of the IGPL*, 11(4):485–511, 2003.

[115] Lawrence L. Kupper and Kerry B. Hafner. On Assessing Interrater Agreement for Multiple Attribute Responses. *Biometrics*, 45(3):957–967, 1989.

[116] Staffan Larsson. *Issue-based Dialogue Management*. PhD thesis, Department of Linguistics, University of Gothenburg, Sweden, 2002.

[117] Staffan Larsson, Robin Cooper, Elisabet Engdahl, and Peter Ljunglöf. Information states and dialogue move engines. *Electronic Transactions on Artificial Intelligence*, vol. 3, section D: Special Section on Knowledge and Reasoning in Practical Dialogue Systems, 1999.

[118] Staffan Larsson, Peter Ljunglöf, Robin Cooper, Elisabet Engdahl, and Stina Ericsson. GoDiS – an accommodating dialogue system. In *Proceedings of the NAACL'00 Workshop on Conversational Systems*, pages 7–10, Seattle, Washington, 2000.

[119] Victor R. Lee and Bruce L. Sherin. What makes teaching special? In *Proceedings of ICLS-04*, pages 302–309, 2004.

[120] Oliver Lemon, Anne Bracy, Alexander Gruenstein, and Stanley Peters. Information States in a Multi-modal Dialogue System for Human-Robot Conversation. In *Proceedings of Bi-Dialog, 5th Workshop on Formal Semantics and Pragmatics of Dialogue*, pages 57 – 67, 2001.

[121] James C. Lester and Bruce W. Porter. Developing and empirically evaluating robust explanation generators: The Knight experiments. *Computational Linguistics*, 23(1):65–103, 1997.

[122] Stephen C. Levinson. *Pragmatics*. Cambridge University Press, Cambridge, 1983.

[123] Magdeleine D. N. Lewa, W. A. M. Alwisa, and Henk G. Schmidt. Accuracy of students' self-assessment and their beliefs about its utility. *Assessment & Evaluation in Higher Education*, 35:135–156, 2010.

[124] David Lewis. Scorekeeping in a language game. *Journal of Philosophical Logic*, 8:339–359, 1979.

[125] Rensis Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 140(1), 1932.

[126] Diane Litman and Kate Forbes-Riley. Correlations between dialogue acts and learning in spoken tutoring dialogues. *Natural Language Engineering*, 12(2):161–176, 2006.

[127] Diane J. Litman and James F. Allen. A Plan Recognition Model for Subdialogues in Conversation. *Cognitive Science*, 11(2):163–200, 1987.

[128] Diane J. Litman and James F. Allen. Discourse Processing and Commonsense Plans. In P.R. Cohen, J. Morgan, and M. Pollack, editors, *Intentions in Communication*, pages 365–388. MIT Press, Cambridge, MA, 1990.

[129] Diane J. Litman and Scott Silliman. ITSPOKE: An Intelligent Tutoring Spoken Dialogue System. In *Proceedings of the Human Language Technology Conference: 4th Meeting of the North American Chapter of the Association for Computational Linguistics (Companion Proceedings)*, pages 1–4, Boston, MA, 2004.

[130] Diane J. Litman, Carolyn P. Rosé, Kate Forbes-Riley, Kurt Vanlehn, Dumisizwe Bhembe, and Scott Silliman. Spoken versus typed human and computer dialogue tutoring. *International Journal of Artificial Intelligence in Education*, 16(2):145–170, 2006.

[131] Karen E. Lochbaum. A collaborative planning model of intentional structure. *Computational Linguistics*, 24(4):525–572, 1998.

[132] Max M. Louwerse and Scott A. Crossley. Dialog act classification using n-gram algorithms. In G. Sutcliffe and R. Goebel, editors, *Proceedings of the International Florida Artificial Intelligence Research Society*, pages 758–763, Menlo Park, California, 2006. AAAI Press.

[133] Xin Lu, Barbara Di Eugenio, Trina C. Kershaw, Stellan Ohlsson, and Andrew Corrigan-Halpern. Expert vs. non-expert tutoring: Dialogue moves, interaction patterns and multi-utterance turns. In Alexander F. Gelbukh, editor, *Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Text Processing*, pages 456–467. Springer, 2007.

[134] Maxim Makatchev and Kurt VanLehn. Analyzing Completeness and Correctness of Utterances Using an ATMS. In Chee-Kit Looi, Gordon I. McCalla, Bert Bredeweg, and Joost Breuker, editors, *Proceedings of the International Conference on Artificial Intelligence in Education*, pages 403–410, Amsterdam, 2005. IOS Press.

[135] Colin Matheson, Massimo Poesio, and David Traum. Modelling grounding and discourse obligations using update rules. In Janyce Wiebe, editor, *Proceedings of the 1st Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1–8, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[136] David McArthur, Cathleen Stasz, and Mary Zmuidzinas. Tutoring Techniques in Algebra. *Cognition and Instruction*, 7(3):197–244, 1990.

[137] Michael F. McTear. Spoken Dialogue Technology: Enabling the Conversational User Interface. *ACM Computing Surveys*, 34(1):90–169, 2002.

[138] Michael F. McTear. Modelling spoken dialogues with state transition diagrams: Experiences with the CSLU toolkit. In *Proceedings of the 5th International Conference on Spoken Language Processing*, Sydney, Australia, 1998.

[139] Douglas C. Merrill, Brian J. Reiser, Michael Ranney, and J. Gregory Trafton. Effective Tutoring Techniques: A Comparison of Human Tutors and Intelligent Tutoring Systems. *The Journal of the Learning Sciences*, 2(3):277–305, 1992.

[140] Douglas C. Merrill, Brian J. Reiser, Shannon K. Merrill, and Shari Landes. Tutoring: Guided learning by doing. *Cognition and Instruction*, 13: 315–372, 1995.

[141] Nestor Miliaev, Alison Cawsey, and Greg Michaelson. Applied NLG System Evaluation: FlexyCAT. In Ehud Reiter, Helmut Horacek, and Kees van Deemter, editors, *Proceedings of the 9th European Workshop on Natural Language Generation*, pages 55–62, Budapest, 2003.

[142] Johanna D. Moore. What makes human explanations effective? In *Proceedings of the Fifteenth Annual Meeting of the Cognitive Science Society*, pages 131–136, Hillsdale, NJ, 1993.

[143] Johanna D. Moore, Kaśka Porayska-Pomsta, Sebastian Varges, and Claus Zinn. Generating tutorial feedback with affect. In Valerie Barr and Zdravko Markov, editors, *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Sociey Conference*. AAAI Press, 2004.

[144] Christoph Müller and Michael Strube. Mmax: A tool for the annotation of multi-modal corpora. In *Proceedings of the 2nd IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 45–50, Seattle, Wash., 2001.

[145] Christoph Müller and Michael Strube. Multi-level annotation of linguistic data with MMAX2. In Sabine Braun, Kurt Kohn, and Joybrato Mukherjee, editors, *Corpus Technology and Language Pedagogy: New Resources, New Tools, New Methods*, pages 197–214. Peter Lang, Frankfurt a.M., Germany, 2006.

[146] Roger H. Munger. Asymmetries of knowledge: What tutor-student interactions tell us about expertise. In *Proceedings of the Annual Meeting of the Conference on College Composition and Communication*, 1996.

[147] Matthias Nückles, Jörg Wittwer, and Alexander Renkl. How to make instructional explanations in human tutoring more effective. In Ron Sun, editor, *Proceedings of the 28th Annual Conference of the Cognitive Science Society*, pages 633–638, Mahwah, NJ, 2006. Erlbaum.

[148] Tim Paek. Empirical methods for evaluating dialog systems. In *Proceedings of the workshop on Evaluation for Language and Dialogue Systems*, pages 1–8. Association for Computational Linguistics, 2001.

[149] Tim Paek. Toward evaluation that leads to best practices: reconciling dialog evaluation in research and industry. In *Proceedings of the NAACL-HLT Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies*, pages 40–47, Morristown, NJ, USA, 2007. Association for Computational Linguistics.

[150] Kai Pata, Tago Sarapuu, and Raymond Archee. Collaborative scaffolding in synchronous environment: Congruity and antagonism of tutor/student facilitation acts. In Timothy Koschman, Tak-Wai Chan, and Daniel D. Suthers, editors, *Computer Supported Collaborative Learning 2005: The next 10 years*, pages 484–493. Kluwer, 2005.

[151] Natalie K. Person, Arthur C. Graesser, Joseph P. Magliano, and Roger J. Kreuz. Inferring what the student knows in one-to-one tutoring: The role of student questions and answers. *Learning and Individual Differences*, 6(2):205–229, 1994.

[152] Massimo Poesio and David Traum. Conversational actions and discourse situations. *Computational Intelligence*, 13(3), 1997.

[153] Andrei Popescu-Belis. Dialogue Acts: One or More Dimensions? ISSCO Working Paper 62, University of Geneva, 2007.

[154] Kaśka Porayska-Pomsta. *Influence of Situational Context on Language Production: Modelling Teachers' Corrective Responsesa*. PhD thesis, ICCS, School of Informatics, University of Edinburgh, 2003.

[155] Kaśka Porayska-Pomsta, Manolis Mavrikis, and Helen Pain. Diagnosing and acting on student affect: the tutor's perspective. *User Modeling and User-Adapted Interaction*, 18(1-2):125–173, 2008.

[156] Joseph Psotka, L. Dan Massey, and Sharon A. Mutter, editors. *Intelligent Tutoring Systems: Lessons Learned*. Psychology Press, 1988.

[157] Ralph T. Putnam. Structuring and Adjusting Content for Students: A Study of Live and Simulated Tutoring of Addition. *American Educational Research Journal*, 24(1):13–48, 1987.

[158] J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., 1993.

[159] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009.

[160] Owen C. Rambow, Monica Rogati, and Marilyn A. Walker. Evaluating a trainable sentence planner for a spoken dialogue travel system. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 426–433, 2001.

[161] Norbert Reithinger and Elisabeth Maier. Utilizing statistical dialogue act processing in VERBMOBIL. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 116–121. Association for Computational Linguistics, 1995.

[162] Charles Rich and Candace L. Sidner. Collagen: A collaboration manager for software interface agents. *User Modeling and User-Adapted Interaction*, 8(3/4):315–350, 1998.

[163] Jeff Rickel, Neal Lesh, Charles Rich, Candace Sidner, and Abigail Gertner. Using a Model of Collaborative Dialogue to Teach Procedural Tasks. In *Proceedings of the AIED Workshop on Tutorial Dialogue Systems*, 2001.

[164] Jeff Rickel, Neal Lesh, Charles Rich, Candace L. Sidner, and Abigail S. Gertner. Collaborative discourse theory as a foundation for tutorial dialogue. In Stefano A. Cerri, Guy Gouardères, and Fábio Paraguaçu, editors, *Proceedings of the 6th International Conference on Intelligent Tutoring Systems*, pages 542–551, London, UK, 2002. Springer-Verlag.

[165] Verena Rieser and Oliver Lemon. Learning effective multimodal dialogue strategies from Wizard-Of-Oz data: bootstrapping and evaluation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, 2008.

[166] Verena Rieser and Johanna Moore. Implications for Generating Clarification Requests in Task-oriented Dialogues. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, 2005.

[167] Jeremy Roschelle. Learning by Collaborating: Convergent Conceptual Change. *The Journal of the Learning Sciences*, 2(3):235–276, 1992.

[168] Harvey Sacks, Emanuel A Schegloff, and Gail Jefferson. A simplest systematics for the organization of turn-taking for conversation. *Language*, 50(4):696–735, 1974.

[169] Konrad Scheffler and Steve Young. Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning. In *Proceedings of the 2nd International Conference on Human Language Technology Research*, pages 12–19, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[170] Emanuel A. Schegloff. Sequencing in conversational openings. *American Anthropologist*, 70:1075–1095, 1968.

[171] Richard Scheines and Wilfried Sieg. Computer Environments for Proof Construction. *Interactive Learning Environments*, 4(2):159–169, 1994.

[172] Marvin Schiller and Christoph Benzmüller. Proof granularity as an empirical problem? In *Proceedings of Computer Science in Education*. INSTICC Press, 2009.

[173] Marvin Schiller, Christoph Benzmüller, and Ann Van de Veire. Judging granularity for automated mathematics teaching. In *LPAR 2006 Short Papers Proceedings*, Phnom Pehn, Cambodia, 2006.

[174] John R. Searle. *Speech Acts*. Cambridge University Press, New York, 1969.

[175] Farhana Shah. *Recognizing and responding to student plans in an intelligent tutoring system: CIRCSIM-Tutor*. PhD thesis, Illinois Institute of Technology, 1997.

[176] Farhana Shah, Martha Evens, Joel Michael, and Allen Rovick. Classifying Student Initiatives and Tutor Responses in Human Keyboard-to-Keyboard Tutoring Sessions. *Discourse Processes*, 33(1):23–52, 2002.

[177] Jörg Siekmann, Christoph Benzmüller, and Serge Autexier. Computer Supported Mathematics with ΩMEGA. *Journal of Applied Logic*, 4(4): 533–559, 2006.

[178] Satinder Singh, Diane Litman, Michael Kearns, and Marilyn Walker. Optimizing Dialogue Managment with Reinforcement Learning: Experiments with the NJFun System. *Journal of Artificial Intelligence Research*, 16:105–133, 2002.

[179] Kristina Skuplik. Satzkooperationen. Definition und empirische Untersuchung. Technical report, Bielefeld University, 1999. Report 1999/03 of SFB 360.

[180] D. Sleeman and J.S. Brown. *Intelligent Tutoring Systems*. Academic Press, New York, 1982.

[181] Richard Sommer and Gregory Nuckols. A Proof Environment for Teaching Mathematics. *Journal of Automated Reasoning*, 32(3):227–258, 2004.

[182] Robert C. Stalnaker. Assertion. *Syntax and Semantics*, 9:315–332, 1978.

[183] Robert C. Stalnaker. Common ground. *Linguistics and Philosophy*, 25 (5):701–721, 2002.

[184] Amanda Stent. *Dialogue Systems as Conversational Partners: Applying Conversation Acts Theory to Natural Language Generation for Task-Oriented Mixed-Initiative Spoken Dialogue*. PhD thesis, Computer Science Dept., University of Rochester, 2001.

[185] Andreas Stolcke, Noah Coccaro, Rebecca Bates, Paul Taylor, Carol Van Ess-Dykema, Klaus Ries, Elizabeth Shriberg, Daniel Jurafsky, Rachel Martin, and Marie Meteer. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26 (3):339–373, 2000.

[186] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[187] Ping-Kee Tao. Peer Collaboration in Solving Qualitative Physics Problems: The Role of Collaborative Talk. *Research in Science Education*, 29 (3):365–383, 1999.

[188] Rich Thomason, Matthew Stone, and David DeVault. Enlightened update: A computational architecture for presupposition and other pragmatic phenomena. In *Presupposition Accommodation*, 2006. (draft).

[189] Henry S. Thompson, Anne Anderson, Ellen Gurman Bard, Gwyneth Doherty-Sneddon, Alison Newlands, and Cathy Sotillo. The HCRC Map Task corpus: Natural dialogue for speech recognition. In *Proceedings of the workshop on Human Language Technology*, pages 25–30, Morristown, NJ, USA, 1993. Association for Computational Linguistics.

[190] David Traum. *A computational theory of grounding in natural language conversation*. PhD thesis, University of Rochester, NY, USA, 1994. Also available as TR 545, Dept. of Computer Science, University of Rochester.

[191] David Traum. Computational models of grounding in collaborative systems. In Susan E. Brennan, Alain Giboin, and David Traum, editors, *Working Papers of the AAAI Fall Symposium on Psychological Models of Communication in Collaborative Systems*, pages 124–131, Menlo Park, California, 1999. AAAI.

[192] David Traum and James Allen. A Speech Acts Approach to Grounding in Conversation. In *Proceedings of the 2nd International Conference on Spoken Language Processing*, pages 137–140, 1992.

[193] David Traum and Staffan Larsson. The Information State Approach to Dialogue Management. In Jan van Kuppevelt and Ronnie Smith, editors, *Current and new directions in discourse and dialogue*. Kluwer, 2003.

[194] David R. Traum and James F. Allen. Discourse Obligations in Dialogue Processing. In *Proceedings of the 32nd Annual meeting of the Association for Computational Linguistics*, pages 1–8, 1994.

[195] Dimitra Tsovaltzi and Elena Karagjosova. A View on Dialogue Move Taxonomies for Tutorial Dialogues. In *Proceedings of 5th SIGdial Workshop on Discourse and Dialogue*, Boston, USA, 2004.

[196] Dimitra Tsovaltzi, Armin Fiedler, and Helmut Horacek. A multi-dimensional taxonomy for automating hinting. In James C. Lester, Rosa Maria Vicari, and Fábio Paraguaçu, editors, *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, number 3220 in LNCS, pages 772–781. Springer, 2004.

[197] Kurt VanLehn, Randolph M. Jones, and Michelene T. H. Chi. A model of the self-explanation effect. *The Journal of the Learning Sciences*, 2(1): 1–59, 1992.

[198] Kurt VanLehn, Pamela W. Jordan, Carolyn P. Rosé, Dumisizwe Bhembe, Michael Boettner, Andy Gaydos, Maxim Makatchev, Umarani Pappuswamy, Michael Ringenberg, Antonio Roque, Stephanie Siler, and Ramesh Srivastava. The Architecture of Why2-Atlas: A Coach for Qualitative Physics Essay Writing. In Mitsuru Ikeda, Kevin D. Ashley, and Tak-Wai Chan, editors, *Proceedings of the 6th International Conference on Intelligent Tutoring Systems*, volume 2363 of *LNCS*, pages 158–167. Springer, 2002.

[199] Kurt VanLehn, Arthur C. Graesser, G. Tanner Jackson, Pamela Jordan, Andrew Olney, and Carolyn P. Rosé. When are tutorial dialogues more effective than reading? *Cognitive Science*, 31:3–62, 2007.

[200] Wolfgang Wahlster, editor. *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer, Berlin, 2000.

[201] Marilyn Walker. *Informational Redundancy and Resource Bounds in Dialogue*. PhD thesis, University of Pennsylvania, Philadelphia, PA, 1993.

[202] Marilyn Walker and Owen Rambow. The Role of Cognitive Modeling in Achieving Communicative Intentions. In *Proceedings of the 7th International Workshop on Natural Language Generation.*, pages 171–180, Kennebunkport, ME, 1994.

[203] Marilyn Walker, S. Whittaker, A. Stent, P. Maloor, J. Moore, M. Johnston, and G. Vasireddy. Generation and evaluation of user tailored responses in multimodal dialogue. *Cognitive Science*, 28:811–840, 2004.

[204] Marilyn A. Walker. An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email. *Journal of Artificial Intelligence Research*, 12:387–416, 2000.

[205] Marilyn A. Walker, Diane J. Litman, Candace A. Kamm, and Alicia Abella. PARADISE: A framework for evaluating spoken dialogue agents. In Philip R. Cohen and Wolfgang Wahlster, editors, *Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, pages 271–280, Somerset, New Jersey, 1997. Association for Computational Linguistics.

[206] Gary Weiss and Foster Provost. The Effect of Class Distribution on Classifier Learning: An Empirical Study. Technical Report ML-TR-44, Department of Computer Science, Rutgers University, 2001.

[207] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics*, pages 80–83, 1945.

[208] Magdalena Wolska and Mark Buckley. A Taxonomy of Task-related Dialogue Actions: The Cases of Tutorial and Collaborative Planning Dialogue. In Angelika Storrer, Alexander Geyken, Alexander Siebert, and Kay-Michael Würzner, editors, *Text Resources and Lexical Knowledge*, volume 8 of *Text, Translation, Computational Processing*, pages 105–114. Mouton de Gruyter, Berlin, 2008.

[209] Magdalena Wolska and Ivana Kruijff-Korbayová. Analysis of Mixed Natural and Symbolic Language Input in Mathematical Dialogs. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics*, pages 25–32, Barcelona, Spain, 2004.

[210] Magdalena Wolska and Ivana Kruijff-Korbayová. Modeling anaphora in informal mathematical dialogue. In *Proceedings of the 10th Workshop on the Semantics and Pragmatics of Dialogue (brandial-06)*, pages 147–154, Potsdam, Germany, 2006.

[211] Magdalena Wolska, Bao Quoc Vo, Dimitra Tsovaltzi, Ivana Kruijff-Korbayova, Elena Karagjosova, Helmut Horacek, Malte Gabsdil, Armin Fiedler, and Christoph Benzmüller. An annotated corpus of tutorial dialogs on mathematical theorem proving. In *Proceedings of the International Conference on Language Resources and Evaluation*, Lisbon, Portugal, 2004. ELDA.

[212] Magdalena Wolska, Mark Buckley, Helmut Horacek, Ivana Kruijff-Korbayová, and Manfred Pinkal. Linguistic Processing in a Mathematics Tutoring System: Cooperative Input Interpretation and Dialogue Modelling. In Matthew W. Crocker and Jörg Siekmann, editors, *Resource-Adaptive Cognitive Processes*, pages 267–289. Springer, 2009.

[213] Erna Yackel. Explanation, justification and argumentation in mathematics classrooms. In *Proceedings of the 25th Conference of the International Group for the Psychology of Mathematics Education*, pages 1–9, 2001.

[214] Yujian Zhou, Reva Freedman, Michael Glass, Joel A. Michael, Allen A. Rovick, and Martha W. Evens. Delivering Hints in a Dialogue-Based Intelligent Tutoring System . In Jim Hendler and Devika Subramanian, editors, *Proceedings of the 16th National Conference on Artificial Intelligence*, pages 128–134, Orlando FL, 1999. AAAI Press and MIT Press.

[215] Claus Zinn. Supporting Tutorial Feedback to Student Help Requests and Errors in Symbolic Differentiation. In Mitsuru Ikeda, Kevin D. Ashley, and Tak-Wai Chan, editors, *Proceedings of the International Conference on Intelligent Tutoring Systems*, volume 4053 of *Lecture Notes in Computer Science*, pages 349–359. Springer, 2006.

[216] Claus Zinn, Johanna D. Moore, and Mark G. Core. A 3-tier Planning Architecture for Managing Tutorial Dialogue. In James C. Lester, Rosa Maria Vicari, and Fábio Paraguaçu, editors, *Proceedings of the 6th International Conference on Intelligent Tutoring Systems,*, volume 2363 of *LNCS*, pages 574–584, Biarritz, France, 2002. Springer.

[217] Claus Zinn, Johanna D. Moore, Mark G. Core, Sebastian Varges, and Kaśka Porayska-Pomsta. The BE&E Tutorial Learning Environment

(BEETLE). In *Proceedings of Diabruck, the 7th Workshop on the Semantics and Pragmatics of Dialogue*, Saarbrücken, Germany, 2003.

[218] Claus Zinn, Johanna D. Moore, and Mark G. Core. Intelligent Information Presentation for Tutoring Systems. In *Intelligent Information Presentation*. Kluwer, 2005.