Determining tutorial remediation strategies from a corpus of human-human tutoring dialogues

Charles B. Callaway and **Johanna D. Moore**
Human Communication Research Centre
University of Edinburgh
2 Buccleuch Place
Edinburgh, EH8 9LW, United Kingdom
{ccallawa,jmoore}@inf.ed.ac.uk

Abstract

We present an empirical approach to adaptively selecting a tutoring system's remediation strategy based on an annotated corpus of human-human tutorial dialogues. We are interested in the remediation selection problem, that of generating the best remediation strategy given a diagnosis for an incorrect answer and the current problem solving context. By comparing the use of individual remediation strategies to their success in varying contexts, we can empirically extract and implement tutoring rules for the content planner of an intelligent tutoring system. We describe a methodology for analyzing a tutoring corpus and using the resulting data to inform a content planning model.

1 Introduction

The goal of an Intelligent Tutoring System (ITS) is not simply to confer knowledge to a student or explain how to solve problems, but to give students help and feedback as they solve problems. Dialogue-based ITS's principally contribute to student learning by providing targeted remedial feedback when the student gives an incorrect answer. The success of a remediation depends both on the content of a remediation and on selecting the right strategy at the right time – for instance, a remediation strategy without content ("No") may be best. Additionally, unlike explanation generation, an ITS shouldn't tell the student everything it knows since the goal is to get the student to do the problem solving, not have the tutor do it for them.

There are many competing theories about what leads students to learn. For instance, VanLehn et al. (2003) propose that students learn when they experience an impasse, such as getting stuck, are correct but uncertain, or make a known type of error. Posner et al. (1982) believe that cognitive dissonance occurs when a student is forced to confront an inconsistency between a strongly held but inaccurate expectation and an actual outcome. Wolf et al. (2005) credit students with generating "accountable talk" when they use valid examples or reasoning strategies from within the domain of interest to back up their claims, and when they make appropriate connections between elements of the domain. Meanwhile Thomas and Rohwer (1993) explain learning gains by the extent to which students apply cognitive effort when processing the new material.

Just as important as the error diagnosis itself then is understanding how to present that diagnosis to the learner, which requires knowing what options are available for varying remedial feedback. Each type of error the student makes can be remediated using many styles varying along social, motivational, content and contextual dimensions even though the diagnosis from the domain reasoner is the same.

One way to discover which of these multiple potential remediation strategies would perform best after an incorrect answer in a given context is to systematically analyze a corpus of tutoring dialogues for (1) remediation strategies, (2) the student's performance after the remediation, and (3) any contextual factors that may have influenced which strategy was selected by the human tutor. Indeed annotation of tutoring dialogue corpora is becoming more common (Cabrera et al., 2007).

But even corpus analysis is difficult due to factors such as data sparsity: (1) there is a large range of potential contextual factors leading to low counts of many phenomena for each specific context, (2) individual remediation strategies don't occur in all domains making it harder to share annotation schemes,

(3) remediation strategies that do occur across domains may have different success rates for different domains, and (4) few dialogues can be collected since a large amount of time is needed to organize and conduct corpora collection between scarce human experts and students. For instance, in the corpus described below, 25 hours of dialogue were required to obtain 198 instances of remedial feedback (i.e., 8 per hour of collected dialogue).

To provide an empirical basis for selecting remediation strategies, we have explored the use of remedial feedback in our tutoring system in the domain of symbolic differentiation (Callaway et al., 2006). By annotating remediation dialogue acts, adjacent dialogue acts related to remediation, along with features such as problem type, we hope to find evidence of patterns in existing human-human dialogues that can be correlated with measures of problem-solving success. To measure the degree of success, we defined a performance metric to compare remediation strategies with their local outcomes, rewarding remediations that led to the student overcoming an impasse and penalizing cases where the tutor's remediations were ineffective or the tutor was forced to "bottom out" by supplying the correct answer.

We then statistically analyzed the resulting data in order to provide advice to an intelligent tutoring system on which strategy to use in a given context. We hope to be able to empirically answer four questions: (1) what is the variation of success of individual remediation strategies, (2) do multiple remediations have better results than single remediations, and (3) which remediation strategies are correlated with particular types of problems (such as polynomials or trigonometric functions). The resulting information can be directly used to help decide which remediation strategy is best to use when the student answers incorrectly in a particular context.

We begin by examining related work in dialogue generation and tutoring, then introducing our tutoring domain of symbolic differentiation and the corpus we analyzed, describing the annotation scheme and evaluation methodology, presenting and analyzing the resulting empirical data, and discussing its implications for NLG.

2 Related Work

Adding generated natural language dialogue to a tutorial system is a complex task whether using templates or deep generation since interactivity allows for a wide range of local variation in context. Many existing tutorial dialogue systems rely on preauthored curriculum scripts (Person et al., 2000) or finite-state machines (Rosé et al., 2001) without detailed knowledge representations. These systems are easier to design for curriculum providers, but offer limited flexibility because the writer has to predict all possible student responses. Representations of domain knowledge and reasoning, along with a record of past student problem solving behavior and misconceptions, is vital for adaptively interacting with students via natural language.

Newer generations of tutoring systems have concentrated more on the tutor's utterances than on being able to understand free natural language input. CIRCSIM is a tutor in the cardiac physiology domain (Michael et al., 2003) that parses student input via finite state machines, arrives at a diagnosis, and then selects and realizes a response for the student, notably with the systematic use of discourse markers. This project also used annotation as a means of identifying key domain phenomena, but without relating it to a success measure (Kim et al., 2006).

The BEETLE1 system (Moore et al., 2004) describes a tutor for teaching basic electricity concepts and components in circuits. The focus of this work was to explore how affective factors should effect the response given. The DIAG-NLP2 system (Eugenio et al., 2005) in the domain of appliance repair takes menu-based input for determining students' actions in a schematic environment and employs high-level abstract revision operations when creating tutorial feedback to make the tutor's responses sound more natural. A formal evaluation showed that a version with revision significantly improved learning gain over a version without it.

In addition to CIRCSIM, annotation has been used in the generation community to attempt to discover relationships or prove effectiveness. Litman and Forbes-Riley (2006) annotated a large array of factors that might potentially affect learning and used χ-square tests over sequences of dialogue moves to discover which of those factors had the greatest influence on learning gain. The GNOME project (Poesio, 2004) created annotation schemes of noun phrases and their co-referring pronouns in order to be able to utilize them for evaluating pronominalization algorithms.

3 Background

We are attempting to semi-automatically formulate remediation strategies using a corpus of human-

Tutor: Differentiate sin(2x)
Student: cos(x)
Tutor: Again we have to use the chain rule.
Tutor: So the answer you gave isn't right.
Student: cos(2x) (x)
Student: sorry
Student: it is cos(2x) (2)
Tutor: Yes, well done.
Tutor: And well done for spotting your mistake.

Figure 1: Extract of a tutoring dialogue

human tutoring sessions in the symbolic differentiation domain. At an abstract level each tutoring session consists of a short subdialogue where a preparatory exchange occurs, followed by a series of math problems proposed by the tutor and solved by the student whenever possible, until a preset time limit is reached. A segment of a typical tutoring dialogue is shown in Figure 1.

Each individual math problem consists of a problem proposal followed by attempts by the student to determine the next correct substep (or the final solution). When a student answers incorrectly, the tutor may immediately respond with remedial feedback to help the student learn from that mistake or wait for the student to ask for help before providing the remediation. Tutors then advance to a subsequent problem either when the student has produced a correct answer, the student is unable to solve the problem and the tutor is forced to produce the answer (a "bottom out"), or the tutor tries a simpler, but related, differentiation problem.

Symbolic Differentiation Domain: The subject area for our human-human corpus data is the differentiation of polynomials and other functions, which involves significant use of the Chain Rule:

$$\frac{d}{dx}\left[f(g(x))\right] = \frac{d}{dg}\left[f(g(x))\right] \cdot \frac{d}{dx}\left[g(x)\right]$$

Its application involves several steps that can be formalized in a task model describing the step-by-step task of building a derivative in terms of simpler, partially ordered sub-tasks, helps to identify which parts of the task a student is currently attempting to solve, and can aid a tutoring system in deciding how to address student actions and provide corrective feedback on the current problem.

The high-level task description for solving derivatives of a given function $y = f(g(x))$ consists of (1) rewriting y to a form $\overline{y}$ which the student already knows how to solve; (2) identifying the component elements of $\overline{y}$ as two nested functions $f(g(x))$; (3) identify the "inner" ($z = g(x)$) and "outer" ($w = f(z)$) layers of $\overline{y}$; (4) differentiate each of the layers, $\frac{dz}{dx}$ and $\frac{dw}{dz}$; (5) combine the results appropriately $\frac{dy}{dx} = \frac{dw}{dz} \cdot \frac{dz}{dx}$; and (6) use algebra to convert the result to a canonical form.

In these dialogues, tutors proposed problems drawn from six fundamental types: polynomials ($3x^4 - 2x$), trigonometric functions ($sin(5x^2)$), square roots ($\sqrt{4x + 6x^5}$), logarithms ($log(7x)$), inverses ($1/sin(4x)$) and combinations ($\sqrt{sin(3x^2)}$).

Corpus: The human-human corpus of tutored differential calculus for this study was conducted at the University of Edinburgh as part of an effort to associate subjective situational factors with learning in the domain (Porayska-Pomsta et al., 2007) and implement the newly discovered principles in an intelligent tutoring system (Callaway et al., 2006)

The data consists of 33 transcripts of tutoring sessions conducted via a chat interface and lasting about 40 minutes each. During each session, the tutor gave the student a sequence of problems to solve until they ran out of time, regardless of the number of problems completed. Five experienced mathematics instructors (as tutors) and 28 first-year mathematics or science undergraduate students who were learning differentiation in a calculus course at the University of Edinburgh were paid to participate.

The data collection environment separated students from tutors physically. They could, however, communicate via a chat interface where the two interlocutors could send each other their typed utterances. Complex mathematical expressions could be entered using a special editor, and text and formulas could be intermixed. The tutor could observe the student's actions in real-time on a second screen. Students and tutors were trained to use the interfaces prior to the data collection session. The resulting corpus consists of 33 dialogues (5 students returned twice) and contains 1650 utterances, 5447 words and 559 formulas.

Domain Reasoning: A domain reasoner supports a model that describes correct actions and relationships within that domain. To support a tutoring system, a domain reasoner must be capable of determining whether a student's answer is correct or not, and if not, what is the most likely explanation for the error. This is usually accomplished by model-tracing using both correct and buggy rules (Brown and Burton, 1978). For symbolic differentiation,

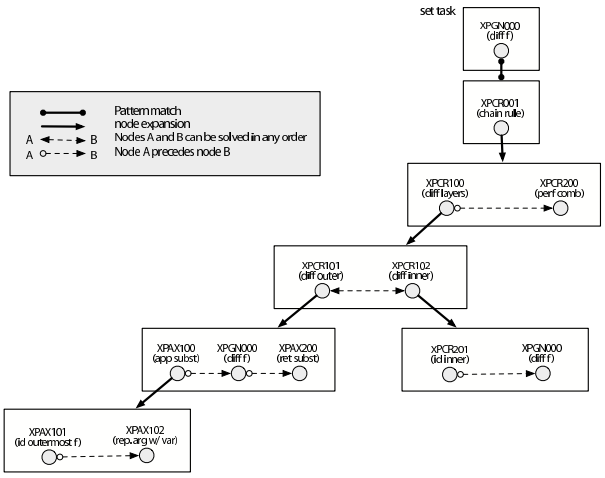

Figure 2: A graphical depiction of the task model

this translates to correct and buggy rules not only for differentiation itself, but also the algebraic rewriting techniques necessary for problem solving.

In tutoring systems based on chat-style text interfaces, the tutor's dialogue acts are the sole means for the system to exhibit adaptivity. Of the possible range of tutor utterances, remediations and proactive hints are most likely to introduce adaptive behavior that will lead to increased student problem solving performance.

Remediations and hints are based directly on the task model itself, where the domain reasoner plays an important role in ensuring that both student and computer tutor are in step when solving problems. A domain reasoner for symbolic differentiation for instance should minimally be capable of judging the correctness of both intermediate steps and final solutions, using buggy rules to be able to interpret the most likely reason for student errors, knowing how far the student has progressed through the current problem, and producing diagnoses that allow the tutoring system to provide feedback to the student.

Given a student solving the differentiation problem x^{-3} by answering $-3 * x^{-2}$ instead of the correct $-3 * x^{-4}$, our domain reasoner would produce:

```
[bg-pl111, type(buggy), rule(power_rule),
 tmc(basic,null), expr(x^(-3)),
 student_answer(-3*x^(-2)),
 wrong_part(exponent, -2),
 corrected_part(exponent, -4)]
```

A non-adaptive template system would match this with a single tutorial utterance which would be the same in every context. However here we are trying to use the dialogue context and problem solving history to vary tutor feedback where the diagnosis serves as the starting point but not the final determinant of remediation content.

4 Annotation Scheme

As mentioned earlier, we are interested in using the corpus to identify which remediations are most likely to lead to improved student performance in a given context. We have ignored clarification dialogues and in-depth follow-ups to mistakes, as they are not easily classifiable and dialogue interpretation components would have a difficult time distinguishing their intent. We have instead concentrated on (1) student answers, (2) tutor responses to those answers, and (3) utterances involved in cognitive impasses, such as help requests. Given a corpus annotated for remediation, we can then use empirical techniques to determine the success or failure of particular remediation strategies in given contexts and adjust the tutorial content planner's high level generation rules accordingly.

The following annotation scheme describes the types of utterances (out of a larger annotated set) that we assume have a strong bearing on remediation and adaptivity of tutorial utterances in general:

Giving and Repairing Answers: Once the tutor has proposed a problem or presented feedback from a previous substep, the student is expected to either provide the next substep or the final solution, or else declare to the tutor that he can't advance by requesting help. Answers are either correct or wrong (we coded partially correct answers as wrong), while step type indicates whether the answer is considered an intermediate step or final solution in terms of the domain model.

We coded correct answers differently depending on whether they occured within normal problem solving or during remediation. In the latter case, answers are instead termed repairs, and rather than being marked for correctness (since repairs are correct by definition), they were instead annotated according to whether the answer was given without an intervening tutor turn (*immediately*) or if there was a discussion of the error before the repair (*delayed*). Figure 3 shows examples of answers in lines 4, 6, 12 and 19 and repairs in lines 8 and 17.

Local Remediation Strategies: If instead the student's answer is incorrect, the tutor must first make the student aware of the mistake and then decide how best to get the student to understand and then fix it. In procedural problem solving domains like

```
Student-01:  Hi.                              No-Annotation
Tutor---02:  Welcome back.                    No-Annotation
Tutor---03:  Try sqrt(3x^2)                   Propose-Problem/Sqrt
Student-04:  (3x^2)^(1/2)                     Give-Answer/Correct/Substep
Tutor---05:  good                             Accept-Answer/Move-On
Student-06:  then (3x^2)^(-1/2)               Give-Answer/Wrong/None
Tutor---07:  you're missing the factor        Remediation/Hint-Error-Location-General
Student-08:  1/2 (3x^2)^(-1/2)                Repair-Answer/Immediate/Substep
Tutor---09:  right                            Accept-Answer/Move-On
Student-10:  then i'm not sure                Help-Request
Tutor---11:  use the chain rule here          Hint-Not-Remediation/Hint-Relevant-Rule-Name
Student-12:  1/2 (6x)^(-1/2) 3x^2             Give-Answer/Wrong/None
Tutor---13:  no, not quite.                   Remediation/Rejection-Only
Tutor---14:  (3x^2)^(-1/2) was right          Remediation/Bottom-Out-Substep
Student-15:  oh, the other way around?        No-Annotation
Tutor---16:  yep                              No-Annotation
Student-17:  1/2 (3x^2)^(-1/2) (6x)           Repair-Answer/Delayed/Substep
Tutor---18:  right                            Accept-Answer/Move-On
Student-19:  (6x)/[2*sqrt(3x^2)]              Give-Answer/Correct/Solution
Tutor---20:  Great!                           Accept-Answer/Positive
```

Figure 3: Condensed dialogue demonstrating typical dialogue moves and their annotation

differentiation, these options are limited to techniques like rejections, hints, and bottoming out. Remediations can occur *singly* (line 7 of Figure 3) or in *multiples* (lines 13–14).

Rejections: The tutor indicates the answer/repair is incorrect without elaborating why. Rejections can appear alone, forcing the student to discover the underlying cause, or in combination with a hint in the same utterance that provides more detail.

```
Tutor:  No.      Remediation, Reject-Only
Tutor:  No, but good try!
                 Remediation, Reject-Positive
Tutor:  That's not what I meant.
                 Remediation, Implied-Rejection
Tutor:  Very close!
                 Remediation, Almost-Complete
```

Hints: The tutor indicates how the student should proceed after the error (Zhou et al., 1999). Most hints in our differentiation domain concern the location of the error.

```
Tutor:  Check the numerator
        Remediation, Hint-Error-Location-General
Tutor:  Are you sure about the 4?
        Remediation, Hint-Error-Location-Specific
Tutor:  Use the power rule.
        Remediation, Hint-Relevant-Rule-Name
Tutor:  The power rule is n*x^(n-1)
        Remediation, Hint-Relevant-Rule-Form
Tutor:  Think of sin^3x as (sin x)^3
        Remediation, Hint-Rewrite
```

Bottoming Out: The tutor supplies the answer.

```
Tutor:  The chain rule gives you (3x^3-3)
             Remediation, Bottom-Out-Substep
Tutor:  No, the answer is 3(x^3-1)
             Remediation, Bottom-Out-Complete
```

Requesting Help If the student is blocked on a problem and prefers not to make an incorrect guess or provide an uncertain answer (hedging), he will often immediately ask for help. The student may also give no answer, and after a short time the tutor typically then supplies a hint to keep the tutoring session on track. Line 10 of Figure 3 shows a typical help request in our domain.

```
Tutor:    Let's try (x^3-3x)^2
                  Propose-Problem
Student:  Where do I start?
                  Help-Request
```

Using this scheme, two annotators coded three of the larger corpus dialogues, obtaining a Kappa of 0.88 considering only top level tags, and 0.78 when the additional lower-level dependent features described below were also taken into account. A single annotator then coded the remaining dialogues.

5 Data Analysis

When students provide incorrect answers, the natural reaction for a tutor is to remediate. There are then a small number of responses that students make to that remediation: (1) immediately repairing the error, (2) repairing with some delay after additional information has been given or requested, (3) explicitly making a request for help, (4) responding with another incorrect answer, and (5) not responding, forcing the tutor to provide further remediation or else directly supply the correct substep or solution.

The success of a remediation can thus be deduced from the subsequent outcome, where we might assume that an immediate repair indicates that an

Remediation Strategy	Immed. Repair	Delayed Repair	Help Request	Wrong Answer	Bottom Out Substep	Bottom Out Complete	Count Totals	Scores
Reject-Only	1 / 6	0 / 3	2 / 1	3 / 7	3 / 1	2 / 2	11 / 20	-21 / +4
Reject-Positive	0 / 2	1 / 2	0 / 1	0 / 3	0 / 0	1 / 0	2 / 8	-2 / +5
Implied-Rejection	2 / 3	0 / 5	1 / 1	3 / 2	0 / 2	1 / 0	7 / 13	-3 / +11
Almost-Complete	3 / 5	0 / 0	0 / 0	3 / 3	0 / 2	2 / 2	8 / 12	-2 / 0
H-E-Loc-General	5 / 8	5 / 3	0 / 1	4 / 9	0 / 6	0 / 3	14 / 30	+22 / -11
H-E-Loc-Specific	13 / 7	3 / 3	0 / 1	2 / 5	0 / 0	2 / 0	20 / 16	+46 / +23
Hint-Rule-Name	0 / 2	1 / 0	0 / 0	3 / 2	0 / 0	0 / 0	4 / 4	-4 / +4
Hint-Rule-Form	2 / 1	0 / 0	1 / 1	0 / 4	0 / 0	1 / 4	5 / 10	+3 / -25
Hint-Rewrite	1 / 2	3 / 4	1 / 0	1 / 2	0 / 0	0 / 0	6 / 8	+7 / +12
Totals	27 / 36	13 / 20	5 / 6	19 / 38	3 / 11	10 / 11	77 / 121	+46 / +23

Table 1: Frequencies of 198 single/multiple remediation dialogue acts by strategy and outcome

"optimal" remediation was selected, while bottoming out indicates a poor remediation was selected. We are thus measuring performance (Aleven et al., 2002) rather than learning gain, where the former is a finer-grained metric. To do this we created an ad-hoc scoring metric that reflects this assumption: for positive outcomes, 4 points for an immediate repair and 2 points for a delayed repair; for negative outcomes, subtracting 1 point for a help request, 2 for a wrong answer, 3 for bottoming out of a substep and 4 points for bottoming out of the entire problem.

The 33 dialogues in the corpus contained 198 total remediations, 77 of which were *single* (occurring alone) and 121 in combination. Table 1 presents the number of each remediation type (rows) as described in Section 4 grouped by outcome (columns) for single/multiple remediations in the entire corpus. The final column lists the score for that remediation type using the scoring metric described above. A positive score thus indicates that that remediation type (when used without other remediations) on average led to the student successfully solving a substep or the entire problem, while a negative score indicates that that remediation type was in general not successful. A score near zero indicates that the outcome averaged out. Pearson's correlation shows remediation as a whole was slightly correlated with successful remediations (those followed by repairs) at $R = 0.13$, and the strongest remediation type was `hint-location-specific` ($R = 0.23$).

The frequency data also indicate which remediations are most often used by tutors. For instance, both `hint-error-location-general` and `hint-error-location-specific` score well, but the former is just as likely as not to involve a discussion with the tutor before the student arrives at the correct answer. (This is not to say that having discussions is a bad thing, we merely score delayed repairs less highly than immediate repairs, but both positively. Also, very specific hints give away more information and so may result in lower longer-term learning.) Negative conclusions can also be drawn, for example that direct rejections ("No.") are very rarely used as a sole remedial utterance.

The right half of each column in Table 1 contains frequencies for multiple remediations, where a student's incorrect answer is followed by two or more adjacent remediation strategies before an outcome is registered. Here, `hint-error-location-specific` continues to correlate strongly with positive outcomes, while `hint-error-location-general` has switched from a positive to a negative correlation.

We were also interested in seeing how problem type (e.g., polynomial) and remediation type correlates with outcome. Table 2 presents frequencies and scores for each remediation type, where for instance `hint-error-location-general` is shown to fare better on polynomials (+11) than square roots (-4) while `hint-error-location-specific` resulted in successful outcomes regardless of problem type. We conclude that the type of differentiation problem can thus have a sizeable impact on the types of remediations a tutor should select in this domain and that this factor should be considered when writing content planning rules.

The data shows that not only are certain tutor remediation strategies better than others overall, but these strategies are also correlated with problem solving performance when the type of differentiation problem is taken into account. Single re-

Remediation Strategy	Poly-nomial	Square Root	Trigono-metric	Logari-thmic	Combin-ation	Inverses	Count Totals
Reject-Only	7 – +2	6 – +5	7 – -11	3 – -1	1 – -1	7 – -11	31
Reject-Positive	4 – -5	1 – +4	2 – -4	1 – +2	1 – +4	1 – +2	10
Implied-Rejection	12 – +9	0 – 0	4 – +3	1 – +2	0 – 0	3 – -2	20
Almost-Complete	9 – -4	3 – -2	3 – -2	2 – +8	2 – +2	1 – -4	20
H-E-Loc-General	16 – +11	5 – -4	10 – -5	4 – +8	2 – +6	7 – -5	44
H-E-Loc-Specific	12 – +7	6 – +16	5 – +10	1 – +4	0 – 0	12 – +32	36
Hint-Rule-Name	3 – +6	1 – +2	2 – -4	1 – -2	0 – 0	1 – -2	8
Hint-Rule-Form	2 – -4	2 – -8	5 – -16	2 – +8	2 – +3	2 – -5	15
Hint-Rewrite	3 – +6	1 – -2	3 – +4	1 – +4	0 – 0	6 – +7	14
Totals	68 – +28	25 – +11	41 – -25	16 – +33	8 – +14	40 – +12	198

Table 2: Combined frequencies and scores for remediations split by problem type

mediations are also more highly correlated with performance than are multiple remediations. This may be because a low aptitude student will perform poorly regardless of whether one or more remediations are administered, but the tutor may feel the need to give more than one remediation type to such a student. Similarly, more specific hints (like `hint-error-location-specific` and `hint-rule-form`) may be given immediately, and the next step in the tutor's remediation sequence after a highly specific hint is to bottom out, resulting in a lower score. Both multiple remediations and highly specific hints may also be given depending on how difficult a step the tutor thinks the student is facing (rather than the overall problem difficulty).

Even more variation is seen when problem type is taken into account. With the differences apparent in Table 2, it becomes possible to write the remediation rules of a tutoring component directly from such data. For instance, we could extract a rule that says if the student is working on a logarithmic function, then select randomly among `almost-complete`, `hint-error-location-general+` and `hint-rule-form`, and otherwise default to selecting `hint-error-location-specific`. This method also points to baseline strategies to use in evaluating an implemented tutoring system against a more sophisticated strategy.

6 Tutorial Generation Component

We have implemented a text generation component (named BUG) in the symbolic differentiation domain that produces turns for the computer in its role as tutor. BUG generates a variety of verbalizations for system feedback within the current context and automatically pronominalizes, aggregates dialogue segments, and inserts discourse markers appropriately. The remediation content planner is included as part of a dialogue manager (Callaway et al., 2006) based on TRINDIKIT which manages the dialogue context and generates all dialogue acts for verbalization within a single turn. As the tactical generator, BUG accepts dialogue moves (sequences of dialogue acts interrelated by rhetorical relations) and information about the dialogue context.

To illustrate, if the dialogue manager decides it would be best to correct a student's wrong answer with a `hint-error-location-specific` remediation type, it gathers details from the diagnosis and constructs a dialogue move such as:

```
[[id1,[concession,id2,id3]],
 [id2,[assert,correct,diff_outer]],
 [id3,[join,id4,id5]],
 [id4,[assert,location,factor,missing]],
 [id5,[assert,location,factor,correct_value,2]]]
```

producing the tutorial utterance "You differentiated the outer layer correctly. However, you missed the factor: it should be 2." where the first clause corresponds to `id2`, the second to `id4`, and the third to `id5`. BUG first converts the dialogue move to deep linguistic representations based on a version of the STORYBOOK system (Callaway and Lester, 2002) modified for use in dialogue systems, and which includes modules for pronominalization, clause aggregation and discourse marker insertion. In this example, it pronominalizes a repeated use of "factor" as "it", revises `id4` and `id5` as indicated by the `join` relation, and inserts the discourse marker "However" into the resulting sentence to connect it to `id2` as indicated by the dialogue move above.

7 Conclusions

We have described natural language generation in the setting of intelligent tutoring systems, focusing

on empirically acquiring tutorial (content) planning rules directly from a corpus of human-human tutoring dialogues. We developed an annotation scheme and local performance metric for tutorial remediations within that corpus, and constructed a framework for analyzing the resulting data. The analysis supports an initial baseline and set of parameters that will be extremely useful when the tutorial feedback generator selects remediation rules when we formally evaluate our computer tutor.

References

V. Aleven, O. Popescu, and K. Koedinger. 2002. More on pilot-testing a tutorial dialog system that supports self-explanation: Good feedback on explanations helps, how we can generate more of it? In *ITS 2002 Workshop on Empirical Methods for Tutorial Dialogue Systems*.

John Seely Brown and Richard R. Burton. 1978. Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science: A Multidisciplinary Journal*, 2(2):155–192.

Anita Ferreira Cabrera, Johanna D. Moore, and Chris Mellish. 2007. A study of feedback strategies in foreign language classrooms and tutorials with implications for intelligent computer-assisted language learning systems. *International Journal of AI in Education*. In Press.

Charles B. Callaway and James C. Lester. 2002. Narrative prose generation. *Artificial Intelligence*, 139(2):213–252, August.

Charles Callaway, Myroslava Dzikovska, Colin Matheson, Johanna Moore, and Claus Zinn. 2006. Using dialogue to learn math in the LeActiveMath project. In *Proceedings of the ECAI Workshop on Language-Enhanced Educational Technology*, pages 1–8, August.

B. Di Eugenio, D. Fossati, D. Yu, S. Haller, and M. Glass. 2005. Natural language generation for intelligent tutoring systems: A case study. In *Proc. of the 12th International Conference on Artificial Intelligence in Education*, pages 217–224, Amsterdam, The Netherlands, July.

Jung Hee Kim, Reva Freedman, Michael Glass, and Martha W. Evens. 2006. Annotation of tutorial dialogue goals for natural language generation. *Discourse Processes*, 42(1):37–74.

Diane Litman and Kate Forbes-Riley. 2006. Correlations betweeen dialogue acts and learning in spoken tutoring dialogues. *Natural Language Engineering*, 12(2):161–176.

Joel A. Michael, Allen A. Rovick, Yujian Zhou, Michael Glass, and Martha Evens. 2003. Learning from a computer tutor with natural language capabilities. *Interactive Learning Environments*, 11(3):233–262.

Johanna D. Moore, Kaŝka Porayska-Pomsta, Sebastian Varges, and Claus Zinn. 2004. Generating tutorial feedback with affect. In *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Sociey Conference*, Miami Beach, FL, May.

N. Person, A. C. Graesser, D. Harter, and E. Mathews. 2000. Dialog move generation and conversation management in autotutor. In *Workshop Notes of the AAAI '00 Fall Symposium on Building Dialogue Systems for Tutorial Applications*.

Massimo Poesio. 2004. Discourse annotation and semantic annotation in the GNOME corpus. In *Proceedings of the ACL Workshop on Discourse Annotation*, Barcelona, Spain, July.

K. Porayska-Pomsta, M. Mavrikis, and H. Pain. 2007. Diagnosing and acting on student affect: the tutors' perspective. *User Modelling and User-Adapted Interaction, Special Issue on Affective Modelling and Adaptation*. Submitted.

G. J. Posner, K. A. Strike, P. W. Hewson, and W. A. Gertzog. 1982. Accommodation of a scientific conception: Towards a theory of conceptual change. *Science Education*, 66(2):211–227.

C. P. Rosé, P. Jordan, M. Ringenberg, S. Siler, K. VanLehn, and A. Weinstein. 2001. Interactive conceptual tutoring in Atlas-Andes. In *Proc. AI-ED 2001*.

J. W. Thomas and W. D. Rohwer. 1993. Proficient autonomous learning: Problems and prospects. In M. Rabinowitz, editor, *Cognitive Science Foundations of Instruction*, pages 1–32. Erlbaum.

K. VanLehn, S. Siler, C. Murray, T. Yamauchi, and W. Baggett. 2003. Why do only some events cause learning during human tutoring? *Cognition and Instruction*, 21(3):209–249.

M. K. Wolf, A. C. Crosson, and L. B. Resnick. 2005. Classroom talk for rigorous reading comprehension instruction. *Reading Psychology*, 26:27–53.

Yujian Zhou, Reva Freedman, Michael Glass, Joel Michael, Allen Rovick, and Martha Evens. 1999. Delivering hints in a dialogue-based ITS. In *Proceedings of the Sixteenth AAAI Conference*, pages 128–134, Orlando, FL.