NATURAL LANGUAGE ANALYSIS AND GENERATION FOR TUTORIAL DIALOGUE

BY

JUNG HEE KIM

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science
in the Graduate College of the
Illinois Institute of Technology

Approved______________________________
Adviser

Chicago, Illinois
May 2000

ACKNOWLEDGMENT

Thanks be to God.

I am deeply grateful to my advisor, Prof. Martha W. Evens, for her great guidance and support during my whole graduate studies. She has been more than an advisor to me. And I thank Dr. Allen Rovick and Dr. Joel Michael for their expert advice.

I acknowledge Reva Freedman, for her suggestions and guidance of my research. Also, I thank Michael Glass, who always encouraged me as an instructor and a co-worker. And I appreciate the members of the CIRCSIM-Tutor research group. We spent many hours discussing our research topics.

I thank my Pastor Won-Ha Cho, Deacon Kwang Han, and Deacon Hyun K. Lee for their endless praying support.

Finally, I owe special thanks to my mother Hee-Ja Kim. She has always believed and encouraged me during whole my life. Also, I appreciate my sisters and brother, Eun-Hee, Yeon-Hee and Won-Yong Kim. Without their lovely care I could not finish this research.

This work was supported by the Cognitive Science Program, Office of Naval research under Grant No. N00014-94-1-0388, to the Illinois Institute of Technology, and the associated AASERT Grants to the Illinois Institute of Technology. The content does not reflect the position or policy of the government and no official endorsement should be inferred.

J. H. Kim

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

CHAPTER I

INTRODUCTION

1.1 Problem Statement

We are building a natural-language based Intelligent Tutoring System (ITS) CIRCSIM-Tutor (CST) designed to teach medical students to solve problems in cardiac physiology. CIRCSIM-Tutor communicates with students using natural language input and output, which is intended to carry on human-like tutoring dialogues. Providing human-sounding conversation is more difficult in a dialogue-based system than in most text generation systems, because the computer's utterances are interrupted by the student's utterances, which are not predictable.

In this thesis I especially want to focus on the text generation issues of CIRCSIM-Tutor. I have studied the language produced by tutors at three different stages of the generation process: discourse goals and strategies, turn tactics, and sentence structures. The major issues are tutoring goals, comparison of tutoring styles, cue words, sentence construction, grammar, and the markup of transcripts that is the basis of our analysis of these other issues.

Tutoring Goals. I have analyzed a large quantity of tutorial dialogue, formalizing the rhetorical strategies and tactics of the tutors. This analysis is detailed and rigorous enough to be used for construction of a planner. I have dissected tutorial patterns into methods, topics, and primitives, where the methods are the tutorial strategies such as "show contradiction" and "reason forward from previous result." This analysis is extensive enough that it has been used as the data for published machine-learning studies of discourse behavior and tutoring language issues. As part of this enterprise I have also categorized the varieties of student response.

SGML Markup of Human-Tutoring Transcripts. The transcripts of human tutors that I analyzed are used in common by all the CIRCSIM-Tutor researchers. I have put my analysis in machine-readable notation, using the Standard Generalized Markup Language (SGML). To this end, I constructed and formalized various content-based markup tags, which enable software to audit the markup for consistency. The marked-up files are available for software processing for my own research and others.

Comparison of Tutoring Styles. My counting and analysis of differences between expert tutors and inexpert tutors shows what are the better tutoring behaviors and language styles. This comparison gives us a model of effective tutoring style.

Analyzing Cue Words. To construct natural sounding dialogue, especially to make coherent turns, I applied machine learning to derive rules for cue word selection. These rules will be applied by the turn planner in the new CIRCSIM-Tutor.

Sentence Construction. In order to produce better dialogue in the new version of CIRCSIM-Tutor, I have used the same marked-up dialogue to dissect the structure of tutorial sentences. I set out to imitate human tutorial dialogue by assembling sentences out of clauses and segments. My analysis groups similar sentences, finds their common elements, and identifies what knowledge from the tutoring and dialogue context is needed for selecting the different parts.

Writing the Genkit Grammar. For the final surface sentence generation, I wrote grammar rules for the Genkit grammar compiler. It compiles grammar rules to Lisp code which generates sentences. The input to this sentence generator is feature structures which will be built by the turn planner.

1.2 Research Goals

My goal was to produce a human-like sentence generator for CIRCSIM-Tutor v.3. Toward this purpose I have accomplished the following:

- Found an effective style for tutoring language.
- Used an approach that assembles sentences from parts that are sensitive to the tutoring and dialogue context.
- Wrote a Genkit grammar rule for sentence generation based on feature structures.
- Produced variety in the generated text, which is similar to the variety the human tutors use.
- Produced turns where the sentences are not totally isolated, but have some cohesion between them or are combined. The research on discourse markers is part of this goal.

My work on the rhetorical structure of tutoring and the varieties of student responses is being used by others in the construction of the planners for CST v.3. This will result in more human-like tutorial dialogue, with much more variety than is supported in current versions of CST.

In this thesis I discuss these goals and describe my work in more detail.

1.3 Organization of This Thesis

In Chapter 2, I give an overview of the CIRCSIM-Tutor system and focus on earlier work on the text generator of CIRCSIM-Tutor. Chapter 3 presents some discussion of theories related to my work. In particular, I show some text generation systems that deal with issues like ours. Chapter 4 describes the method I developed for analyzing and marking up transcripts. I give some explanation of hierarchical goal structure, arguments for tutoring tactics, and sentence generation. Chapter 5 analyzes the difference in tutoring styles between expert tutors and novice tutors. I concentrate especially on differences that can be counted and tested for statistical significance. Chapter 6 shows how we choose cue words appropriately during dialogue. Also, I show my counting data and machine learning results. Chapter 7 discusses how to extract features and knowledge for surface generation from the marked-up transcripts. Also I show the grammar for the sentence generator. This thesis ends with the presentation of a summary, a discussion of the significance of this thesis, and future work.

CHAPTER II

BACKGROUND

2.1 What is CIRCSIM-Tutor?

2.1.1 History of Project.

Joel Michael and Allen Rovick of Rush Medical College have a long history of using computers to teach the baroreceptor reflex in their physiology course for first-year medical students. The baroreceptor reflex is a negative feedback loop that maintains a steady blood pressure in the human body. It is one of the most difficult topics in the first-year physiology course. Michael and Rovick have pioneered in the use of computer-based instruction to augment their classroom instruction by getting their students to work problems exploring the behavior of the reflex, which they learned about in lectures.

In the beginning Rovick and Michael used MACMAN [Dickinson et al. 1973] which is a mathematical quantitative model of the reflex behavior. Students ran “experiments” with MACMAN, varying the values of physiological parameters and using MACMAN to check the behavior of the reflex system. However, using MACMAN to teach in this manner was not very successful, partly because students were not very good at designing and running experiments and interpreting the results, and partly because a skilled instructor is still needed.

Due to these difficulties HEARTSIM [Rovick & Brenner 1983] was developed. It has two components: a mathematical model (MACMAN) and a teaching module. The predictions table [Rovick & Michael 1992] was introduced as a way for the student to make qualitative predictions. In a qualitative prediction, the student predicts only whether the value of a physiological variable increases, decreases, or stays the same, without discussing actual numerical quantities. The teaching module defines a set of procedures, so the students did not need to define their own experiments. One procedure contains a description of some changes to the body that will affect blood pressure. Students are asked to select a procedure, then make qualitative predictions. Then logical and relationship errors among the student's predictions are checked and corrected. After that, the MACMAN component produces numerical results, plotted and displayed in a table. After the comparison of the student's prediction with the actual result, text feedback is provided to correct the student's misconceptions.

Due to the difficulty of access to the PLATO infrastructure for computer aided instruction, needed for running HEARTSIM, Michael and Rovick [1986] wrote the DOS program CIRCSIM in Basic. During the process of moving to CIRCSIM from HEARTSIM, they decided that the important thing for teaching is the correct qualitative predictions for each procedure, not quantitative outputs from the model. So CIRCSIM has no mathematical model. It has limited, stored data to display the quantitative and qualitative results. CIRCSIM also made big progress in checking for more student errors and providing instructional feedback. Depending on the pattern of student errors it displays one of over 240 paragraphs of canned text. CIRCSIM is still in use today as a required exercise for the students in the first year physiology class.

Still, Rovick and Michael felt that there are many kinds of student errors and misconceptions that are hard to discover and remedy. There is usually an instructor present when the students are using CIRCSIM, also they have experience teaching the baroreceptor reflex to students in small groups and individually. They concluded that natural language dialogue is needed, both for diagnosing and repairing the student's misconceptions. So they proposed to Martha Evens that they build a tutoring system together capable of carrying on a natural language dialogue.

<u>2.1.2 CIRCSIM-TUTOR.</u> CIRCSIM-Tutor (CST) is a dialogue based intelligent tutoring system covering the same domain as CIRCSIM. The system defines a problem, an event that destabilizes the blooe pressure. Students are then asked to predict the direction of change of seven core variables, the most important cardiovascular parameters at each of three different stages. The problem is divided into three chronological stages: DR, RR, and SS. DR is the Direct Response stage, which means immediately after the event that destabilizes the blood pressure and before operation of the baroreceptor reflex. RR is the Reflex Response stage, showing the nervous system's response. SS is the Steady State stage, showing final, steady, values relative to before the procedure started. The tutor analyzes these predictions, finds the errors, and chooses a method to tutor each one. The tutor then embarks on a remedial dialogue. CIRCSIM-Tutor uses natural language as input and output and can handle some syntactic errors such as sentence fragments or misspelled words.

Kim [1989] developed a prototype CIRCSIM-Tutor, which was implemented in Prolog. CST v.2 was implemented in 1992 by Woo [1991] and others in Lisp. Implementation of CST v.3 is in progress now with a different planning view and many improvements in the natural language capabilities.

CST v.2 is composed of the following modules:

Input Understander: understands the student’s input and handles ill-formed as well as well-formed utterances. In addition to the student’s answers to the questions, it can understand some simple student initiatives such as “What is Cardiac Contractility?” It parses the student’s sentence using information about the current topic, and returns a logic form to the planner.

Student Modeller: represents the state of the student’s current knowledge and misconceptions. After analyzing the student’s predictions and answers to questions it updates the system’s model of the student.

Instructional Planner: determines what to do next during a tutoring session. It generates a global lesson plan and a discourse plan.

Screen Manager: is the interface between student and system. It maintains the prediction table containing the student’s predictions and the correct answers. The student’s question or answer is input through the tutoring dialogue window and the tutor’s explanations are displayed. Also, it shows a description of the current problem.

Knowledge Base: was built with a network of frames. Each frame has domain concepts and the causal relationships between parameters. It is used for problem solving and causal explanation.

Text Generator: receives logic forms from the planner and emits each as a separate sentence. With this information it outputs a natural language sentence. The current version of text generator receives information only from the planner and the input understander.

Problem Solver: solves problems presented to the student. CST has two problem solvers, a main problem solver and a subproblem solver. The main problem solver involves generating the correct answers for all the procedures. The subproblem solver involves determining the correct answer to the question under discussion at the moment.

Although an updated version 2 of CST has been used successfully in two trials each involving 50 medical students, Khuwaja [1994], Sanders [1995], and Hume et al. [1996] discovered serious problems, particularly in the discourse generation part of the system and we set out to build a new version.

In this thesis I want to focus on the Text Generator of CST v.3, including problems of discourse goals and strategies, turn-taking tactics, and sentence generation.

2.2 Earlier Work on the CST Text Generator

Zhang [1991] was the first to investigate the process of generating tutorial dialogue for CST. She was the first to note that an entire tutoring session could be viewed as a discourse, so that pedagogical planning should include elements of dialogue analysis. She showed how to plan and generate large fragments of tutorial dialogue, such as explanations and multi-turn tutorial exchanges. She designed a domain knowledge base that solved baroreceptor reflex problems and also provided the knowledge needed for generating explanations. Solving a problem and describing how to solve a problem are completely different activities.

When she analyzed human tutoring sessions she discovered that dialogue schemas are needed to represent the goals and patterns of tutorial discourse. She made three types of schemas: ‘tutoring goal schemas,’ ‘functional schemas,’ and ‘content schemas.’ With these schemas she could generate tutorial discourse patterns, integrating “how to teach” (dialogue management) with “what to teach.”

Chang [1992] wrote the sentence generator currently used in CST v.2. It transforms logic forms from the planner into sentences, where each logic form represents one sentence. Chang's generator has a small data structure for every possible sentential form, which is used for building a Lexical-Functional Grammar F-structure, which is then converted to a sentence using an ATN algorithm.

Chang's generator is more grammatically sophisticated than it needs to be. Since it always produces individual sentences in isolation, and it never varies the syntax of a sentence, it does not need to use its syntactic knowledge. It could be replaced by a simple method which has each possible sentence pre-constructed, with a few variables to be filled in. Furthermore, since the process of building the F-structure and the ATN grammar is somewhat complicated, CST v.2 programmers after Chang have tended to bypass the text generator and emit the sentences they need directly.

Freedman [1996b, pp. 76-78] pointed out the problems of CST v.2 as follows:

- Any given concept can be taught in only one way.
- There is no inter-turn or intra-turn cohesion, so discourse markers cannot be inserted within the sentence and there is some difficulty in putting turns together.

  Example:

  (1) Good, you got the correct answer.
  What is the correct value of SV?

  (2) Correct, so what is the value of SV?

The above example shows some difference between turns (1) and (2). Turn (1) consists of two sentences that were generated separately, as in CST v.2. Turn (2) contains the same two sentences combined. Although they serve the same purpose, with the addition of the discourse marker “so” turn (2) becomes smoother. In this case “so” functions as an indication that we are returning to a previous topic.

- There is no way to change from one tutoring plan to another after the first tutoring plan fails.

To overcome the above problems of CST v.2 Freedman [1996b, pp. 78-79] suggested several kinds of pedagogical and linguistic improvements:

- Using more complex tutorial patterns to provide variety and alternatives: the tutor can ask a ‘pseudo-diagnostic question’ or use a ‘show-contradiction’ pattern.
- Using deeper domain knowledge when a tutoring plan fails: Deeper level domain knowledge is very helpful to the student’s understanding of how to identify determinants. It also helps to anchor the student’s knowledge of physiology to existing knowledge about anatomy and biochemistry.
- Combining patterns: tutorial patterns can be nested and switched.
- Combining concepts inside a turn into a cohesive discourse structure.
- Finding multiple ways to express the same knowledge.

Freedman found that we cannot generate text that carries out these notions with a single hierarchical planner, because turns straddle the boundaries of instructional plans at higher levels.

Like Zhang, Freedman observed that tutorial planning requires the use of schemas, some of which she outlines. I have analyzed our transcripts in detail and created a tutoring goal hierarchy to be used in the text generation modules of CST v.3, using Freedman's schemas as my starting point. This work is described in Chapter 4 below.

2.3 New Modules for CST v.3

Cho et al. [1999] added a curriculum planner module to the new CST. Our new CST will have eighty-three procedures with five content categories, five procedure difficulty levels, and four procedure description levels. The curriculum planner decides which procedures will be suggested for student selection according to the student's prior performance and the student's domain knowledge. This module should give motivation to the student by matching the difficulty level to the student's ability.

The new version of CST will have an instructional discourse planner module to achieve pedagogical goals and to maintain an agenda for tutoring. The discourse planner generates a series of semantic forms. For these modules the Atlas Planning Engine [Freedman, 1999] will be adopted. The Atlas Planning Engine (APE) is developed for mixed-initiative, multimodal dialogue. It is a reactive planner using a hierarchical task network.

APE conducts a question/ answer dialogue with the student via natural language input or GUI action. For efficiency student initiatives are limited to some degree. For the practical needs of a tutoring system, it allows dropping the current dialogue plan or changing to a better plan. In a dialogue-based tutoring system full planning of the dialogue in advance is not possible due to the student's unexpected actions.

CST has a schema-based goal hierarchy structure that is described in Chapter 4. According to the student's unexpected answer the tutor changes or modifies the current tutoring schema to complete the tutor's goal. Zhou et al. [1999b] studied a wide variety of these unexpected responses and how the tutor could respond to them in a manner that improves teaching. For this purpose we need a reactive planner such as APE. Also, our input understander understands some degree of student's initiatives, which we also intend to process.

Another new item is a turn planner module to assemble a single turn from the semantic forms that come from discourse planner. The primary goal of the turn planner is to give cohesiveness and variety to the generated text. The output of the turn planner goes into a new surface sentence generator. These procedures are described in detail in Chapter 6.

CHAPTER III

THEORY AND METHODS

3.1 Representation of Meaning (Logic Form)

In natural language generation systems, semantic representation is an important topic. CST has used logic forms as an interface between the planner and the input understander or the text generator. In particular, the tutorial and turn planners create logic forms that represent sentences to be emitted. The original logic forms in CST were designed by Yoon-Hee Lee [1990] based on the research of Kieras [1985]. Kieras's research is based on the propositional theory of Kintsch.

3.1.1 Propositional Theory.

Kintsch [1974] identifies "word concept", "proposition", and "text base" as the elements of his logic form representation of text. Together they specify a meaning of the text, but not the particular lexical or syntactic form.

- "Word concepts" are the basic units of meaning. Word concepts are joined to make propositions.
- "Propositions" are made by n-tuples of word concepts. The first item in the tuple is a predicate, the rest are arguments. For example, (BAKE MARY CAKE) is a proposition, where BAKE, MARY, and CAKE are each word concepts and BAKE is the predicate. According to syntactic transformation rules and pragmatic rules, the proposition (BAKE MARY CAKE) might have one of following expressions:

Mary is baking a cake.
A cake is being baked by Mary.
The baking of a cake by Mary.
Mary's baking of a cake. [Kintsch 1974, p. 14]

- The "text base" is constructed of ordered lists of propositions. It should have all necessary information for the making of a text. Within a text base coreference is sorted out by indexing.

Kintsch insisted that knowledge is represented by the terms of propositions, and a sentence may composed with the terms of meaning conceptually. Kintsch's propositional theory became a base model for language.

3.1.2 Format of propositional representation. Bovair and Kieras [1985] showed the following format for propositional representations.

```
The fat cat ate the gray mouse
   P1 (EAT  CAT  MOUSE)
   P2 (MOD  CAT  FAT)
   P3 (MOD  MOUSE  GRAY)                    [Bovair & Kieras 1985, p. 316]
```

The example above shows the predicate and argument style. The predicate (EAT or MOD) is shown first, followed by its arguments (CAT, MOUSE, etc.). The MOD predicate shows that the second argument modifies the first argument. These are written in upper case in singular form as word concepts to distinguish them from usual words. An argument to a proposition can be either a word concept or a reference to another proposition.

They chose the simplest way to make a useful representation. So they avoided embedded forms, unnecessary propositions, and unnecessary variants such as "quickly" instead of "quick." Predicate forms can be invented freely whenever they are needed. For recall that is easy to paraphrase they chose partial representations instead of summarized ones, as shown in the following example:

```
Radio galaxies emit radio waves.
    P1 (EMIT  GALAXY  WAVE)
    P2 (MOD  GALAXY  RADIO)
    P3 (MOD  WAVE  RADIO)

not
    P1 (EMIT  RADIO-GALAXY  RADIO-WAVE)
                                              [Bovair & Kieras 1985, p. 319]
```

Tenses are disregarded and auxiliaries are not represented. There is no difference in structure between the passive and active forms as shown in the next example.

```
The radio galaxy is called DA240 by scientists.
    P1 (CALL  SCIENTIST  RADIO-GALAXY  DA240)
                                      [Bovair & Kieras 1985, p. 319]
```

The next example shows an argument that is a reference to another proposition, in this case because the verb "think" requires a propositional argument.

```
Astronomers think that X-ray stars are black holes.
    P1 (THINK  ASTRONOMER  P2)
    P2 (ISA  X-RAY-STAR  BLACK-HOLE)
                                      [Bovair & Kieras, 1985, p. 318]
```

Kieras [1985] said that important content can be derived or identified by thematic processes and it can be distinguished from details or irrelevancies. So, he insists that the propositional structure of the passage content is the most important information source.

3.2 Analysis of Discourse

3.2.1 Analysis of classroom discourse by Sinclair and Coulthard.

Sinclair and Coulthard [1975] studied pedagogical discourse in an elementary school classroom. Two important similarities to the transcripts I wish to analyze are: the discourse is part of a tutoring session, and the tutor generally has control of the conversation. Sinclair and

Coulthard were most interested in the function of an utterance, the types of follow-up utterance, the style of introducing and developing topics, and the distribution of turns in the discourse.

The meaning and function of an utterance is not known from its words and syntax, such as when a declarative sentence is actually used to ask a question. Sinclair [Sinclair and Coulthard 1975, pp. 2-3] earlier proposed to study discourse semantics by examining examples of real dialogue. He suggested that only by examining the context of an utterance, including intentions and presuppositions, can its semantics be known. Later Sinclair went on to study the relationship between the discourse function of an utterance and its structure. Partly this came from the observation that a single conversation can be a coherent text, even though its parts are produced by different participants. Sinclair's approach of analyzing meaning in context was in contrast to efforts to describe the meaning of a sentence starting from the word and syntax. In Chapter 7 I use the same approach. There I analyze real transcripts to find the intention and discourse patterns of the tutor, before examining the insides of the sentence. Through this early research, Sinclair and Coulthard became interested in the following problems:

- How are successive utterances related.
- Who controls the discourse.
- How does he do it.
- How, if at all, do other participants take control.
- How do the rules of speaker and listener pass from one participant to another.
- How are new topics introduced and old ones ended.

To solve the above problems they decided to study a simple type of spoken discourse. They found that desultory conversation was not appropriate. In desultory conversation all participants can propose topics and they can change topics any time. In that situation ambiguity and misunderstanding can happen frequently. So they chose a classroom situation where the teacher has the maximum amount of control. Their sample data was six lesson tapes which were made by eight 10- or 11-year-old children and their teachers.

They analyzed their sample dialogues from both a pedagogical view and a linguistic view. To find out pedagogical goals and linguistic goals their work started from the smallest linguistic unit. Because of its flexibility, they used a “rank scale” model based on the assumption that a unit is composed of one or more units of the rank below. A structure of one rank (except the lowest rank) can be represented by the units of the next rank below. They focused on adjacent conversational turns to find out what constitutes an appropriate reply to a teacher’s question, and how the teacher acknowledges the student’s reply. They found out that classroom discourse consists of an initiation from the teacher, followed by a response from the pupil, then feedback to the pupil’s response. An acknowledgment such as “right”, “good”, “OK”, or “now” has the function of indicating boundaries in the lesson.

The text was segmented according to three separate aspects: once according to non-linguistic pedagogical organization, once according to grammatical structures, and once according to discourse segments. Figure 3.1 shows the hierarchical ranking of the segments within each aspect. The rankings have been lined up to show correspondences, thus one discourse “act” corresponds to a grammatical “clause” and a discourse “transaction” corresponds to a pedagogical “topic”.

In this research Sinclair and Coulthard note they have borrowed the terminology of Halliday, such as ‘structure’, ‘system’, ‘rank’, and ‘level’.

Non-Linguistic Organization	Discourse	Grammar
course		
period	lesson	
topic	transaction	
	exchange	
	move	sentence
	act	clause
		group
		word
		morpheme

Figure 3.1. Levels and Ranks. (After [Sinclair & Coulthard 1975, p. 24])

The composition of the discourse ranks are as follows:

- An ‘Act’ indicates a move at the lowest rank of discourse. Each move has a different function, e. g., a ‘focusing move’ represents a change of plan.
- Five classes of moves are identified: ‘Boundary’, ‘Teaching’, ‘Framing’, and ‘Focusing’ moves are categorized as ‘Boundary’ exchanges. ‘Opening’, ‘Answering’, and ‘Follow-up’ moves belong to ‘Teaching’ exchanges.

- ‘Boundary’ exchanges designate the beginning or ending of a stage of lesson. ‘Teaching’ exchanges show how the lesson progresses.
- A ‘Transaction’ is composed of several exchanges. The major transaction types are ‘informing’, ‘directing’, and ‘eliciting’.
- At the highest unit of discourse, a ‘Lesson’ has a series of transactions. The structure of a lesson is determined by the teacher’s performance style. Due to an unexpected student reaction or student misunderstanding, the ordering of transactions can not be known in advance.

As we have seen above, this research is text-based and uses hierarchical ranks to focus on every turn. Our analysis of our tutoring transcripts is similar in style to Sinclair & Coulthard’s. We have identified a hierarchy of discourse goals. Near the top are the most general such as “tutor one stage of the reflex response”, in the middle are goals such as “tutor this variable by the determinants method” and near the bottom are tasks like informing or eliciting individual pieces of knowledge. Furthermore, like Sinclair and Coulthard, we are interested in how these discourse goals correspond to various syntactic constructs, so we can collect together all the ways that various tutoring goals are expressed.

3.2.2 Use of Halliday’s Theory. According to Halliday [1985], language is used to express three kinds of meaning simultaneously: experiential meaning, the propositional content of an utterance, textual meaning, which represents the contribution of the utterance to the narrative coherence of the conversation, and interpersonal meaning, which represents the attitude of the speaker.

Experiential meanings are expressed by the plain meanings of the sentences in isolation. The textual meanings are expressed by the order of content, by conjunctions, and by boundary markers. The interpersonal meanings are expressed by the intonation contour, by the mood, and by expressions of modality.

Inspired by Halliday's theory, we added some arguments to our analysis of transcripts. In our planner, the content axis is represented by the plan operators themselves in addition to content-based arguments, e.g. ***info***. We add additional arguments when it is desirable to represent the interpersonal and narrative axes. In this way we can considerably enrich the range of concepts which we can express. This ability sets v. 3 of CIRCSIM-Tutor apart from earlier versions as well as from question-answering systems.

The ***attitude*** feature is used to express the tutor's personal stance with respect to the material being uttered. For example, consider the differences between the following sentences:

(1) CO increased.

(2) But remember that CO increases.
(attitude=remind)

(3) CO certainly does increase.
(attitude=support)

The latter might be used, for example, in place of the more common CO increases to reply to a student who has made this assertion along with a number of incorrect assertions.

Similarly, the ***narrative-mode*** argument is used to annotate aspects of the text that relate to the structural coherence of the dialogue. The following examples show some of the distinctions that can be made using this argument.

(4) You predicted that CO increased.
(narrative-mode=reference)

(5) So, CO increases.
(narrative-mode=summary)

3.3 SGML

SGML [Bradley 1996] stands for Standard Generalized Markup Language. Compared with many other markup languages SGML has several strengths, such as:

- System independent: SGML-aware software exists for many computers.
- Non-proprietary: it is an international standard.
- Separates content from format: the text is annotated according to content and function, how the text is formatted is a separate question.
- Control over information content
- Human-readable: intermediate results are understandable and inspectable.

SGML consists of text markings added to plain text files. Data in the file is usually enclosed by start-tags and end-tags. A start-tag is delimited by "<" and ">" characters, an end-tag is delimited by "</" and ">".

Figure 3.2 shows an example taken from our own marked-up transcripts. The SGML tags are in upper case to distinguish them from the actual transcript dialogue. Actually SGML is not case sensitive.

An SGML tag consists of an element and optional attributes. In the first tag in Figure 3.2, the element is T-SHOWS-CONTRADICTION and the attribute is

TYPE=NEURAL. It indicates that the entire fragment of dialogue (until the </T-SHOWS-CONTRADICTION> end-tag) is an attempt by the tutor to persuade the student of a self-contradiction in the student's answers. This fragment is divided into two parts: T-PRESENTS-CONTRADICTION (where the tutor presents the relevant data to the student) followed by T-TUTORS-CONTRADICTION (where the tutor gives the student the opportunity to notice there is a difficulty).

SGML markup follows this kind of context-free pattern, one element is divided into several lower-level elements. For example, a THESIS is divided into FRONT-MATTER followed by several CHAPTERS followed by a BIB. Then FRONT-MATTER is divided into TITLE-PAGE, CONTENTS, ACKNOWLEDGMENTS, and so on. Because we are allowed to create our own content tags, we can define our own new tag T-SHOWS-CONTRADICTION, which is divided into two pieces as shown.

The software that processes SGML files uses an SGML parser. The parser requires a special file describing the allowable tags, called the Document Type Definition, or DTD. The DTD is usually separate from the marked-up file, so you can use it to describe the allowable markings for a whole set of files. Inside the DTD are element definitions and attribute definitions. For example, part of the DTD for our transcripts might look like this:

```
<!ELEMENT T-SHOWS-CONTRADICTION - -
      (T-PRESENTS-CONTRADICTION? T-TUTORS-CONTRADICTION?)>
<!ATTLIST T-SHOWS-CONTRADICTION - -
      TYPE (3-VARS | NEURAL) #IMPLIED>
<!ELEMENT T-INFORMS - - CDATA>
```

Here the element definition for T-SHOWS-CONTRADICTION is showing how to divide it into smaller elements. Its attribute definition shows that there is an attribute called

TYPE, which can have the values 3-VARS or NEURAL. The element definition for T-INFORMS shows that it is a primitive, it contains only actual transcript data. This example is a very simple one; there are many complicated options for the element and attribute definitions.

After the marked-up file has been read by an SGML parser, several kinds of processing are possible. For my own work in writing a CST text generator, I can perform operations such as “extract every T-INFORMS which is inside of a neural T-PRESENTS-CONTRADICTION.”

```
<T-SHOWS-CONTRADICTION TYPE=NEURAL>
     <T-PRESENTS-CONTRADICTION>
          <T-INFORMS INFO=VAR-VALUE ATTI=REFERENCE>
               You predicted that CC would go up.
          </T-INFORMS>
          <T-INFORMS INFO=DR-INFO  ATTI=REMIND>
               But remember that we’re dealing with the
               period before there can be any neural
               changes.
          </T-INFORMS>
     </T-PRESENTS-CONTRADICTION>
     <T-TUTORS-CONTRADICTION>
          <T-ELICITS INFO=REASON>
               How can CC go up if it is under neural
               control?
          </T-ELICITS>
          <S-ANS CATG=INCORRECT>
               (Here the student gave some incorrect
               answer)
          </S-ANS>
          (etc.)
     </T-TUTORS-CONTRADICTION>
</T-SHOWS-CONTRADICTION>
```

Figure 3.2. SGML Marked-Up Transcript

As we use the SGML markup and analyze more transcripts, we add and revise the tags as needed. The DTD ensures that we have some consistency in the markup. Also it catches spelling and markup errors. The software we are using is the LTNSL package [McKelvie et al. 1997] from the University of Edinburgh. It will run on any Unix computer.

3.4 Views of the Generation Task

3.4.1 Natural Language Generation Architecture.

Ehud Reiter [1994] surveyed some applications-oriented Natural Language Generation (NLG) systems that produce technical English texts. He claimed that although these systems have different theoretical bases they have a similar architecture of modules as follows.

A) Content Determination: This module decides what information should be included in communication and how it should be rhetorically structured. The output is a ‘semantic form’ or a ‘conceptual representation.’ It has two main functions: ‘Deep content determination’ and ‘Rhetorical planning.’ Every system has its own content determination mechanisms, e.g., schemas.

B) Sentence Planning: It converts the semantic form to an ‘abstract linguistic representation’ that shows content words and grammatical relationships. Every system has a different name, such as the ‘lexical chooser’ of FUF [Elhadad, 1993], the ‘sentence planner’ of SPOKESMAN [Meteer, 1989], and the ‘text planner’ of JOYCE [Rambow and Korelsky, 1992].

C) Surface Generation: From the abstract linguistic representation, the surface form of a sentence is generated. In this module grammatical relationships are converted to word order in the sentence. Most systems have the sentence planning and surface generation modules separate.

D) Morphology and Formatting (Realizer): Due to the simplicity of English morphology, most of the systems have a simple realizer. Here, punctuation rules, noun/verb agreement, and reflexive pronouns, for example, are checked and modified as needed.

3.4.2 TEXT: Schema-based System. TEXT [McKeown 1985] was a system to answer questions about objects from a US Navy database describing vehicles and destructive devices. McKeown used schemas as a content determination and paragraph planning mechanism, defining a schema as "a representation of a standard pattern of discourse structure which efficiently encodes the set of communicative techniques that a speaker can use for a particular discourse purpose" [McKeown 1985, p. 20]. McKeown's schemas were based on rhetorical predicates. She studied both classical and modern rhetoricians and linguists, then selected a number of expository texts to analyze, eventually creating schemas for four primary rhetorical goals: identification, constituency, attribution, and comparison.

McKeown's schemas are recursive, in that one element of a schema can eventually become either a single proposition or another schema. The embedded schema can possibly be the same type. For example a several-paragraph-long text might be dedicated to a comparison (say between two types of ships), within that text might be a constituency enumeration (of several aspects of the ships), and within that might be smaller comparisons (of each aspect).

Since the TEXT system's schemas contain numerous optional elements, alternatives, and arbitrary repetitions, and each element may expand into another schema, the eventual tree-structured description of the text to be generated may be quite complex.

The identification schema, for example, contains optional comparisons, specific examples, descriptions of attributes, etc.

Queries to TEXT correspond to three discourse goals: define, describe, and compare. A major task of TEXT is to figure out which schema applies to the discourse goal at hand. For example, a request to define "guided" produced a constituency schema, while a request to define "aircraft carrier" yielded an identification schema.

This choice of rhetorical goal (schema) is determined by both the discourse goal and what relevant information is available in the database. In particular TEXT figures out if there is enough information of the right type to fill various candidate schemas. In the above example, the database had a lot of information about "aircraft carrier," making a long identification paragraph possible. On the other hand "guided" was poorly represented, so a simpler rhetorical goal was picked. McKeown notes a similarity between filling in a schema and planning, where the tree of schemas to be instantiated by picking alternatives and options is analogous to a tree of planning goals which must be satisfied.

Converting a filled schema to a paragraph involves, among other processes, applying focus rules. Focus rules control pronominalization, passivization, and there-insertion. For example, the fact that "ship" is the focus controls the pronominalization in "A ship is a water-going vehicle.... Its surface-going capabilities ...."

McKeown's rhetorical schemas are comparable to the tutorial and discourse planning goals. Circsim-Tutor's tutoring goals are specialized rhetorical patterns, such as "show contradiction to the student." Deciding on what to say and how to say it, in TEXT as well as Circsim-Tutor, are tied together by the process of finding the appropriate rhetorical schema to fill in.

A big difference between TEXT and Circsim-Tutor is that TEXT is not generating dialogue. The user asks a question, TEXT plans and emits a paragraph. Then it is done. In our tutoring, a multi-turn tutoring pattern is planned, and only part of it is emitted. Then the student responds. But the student response cannot be predicted, so it is possible that the dialogue that Circsim-Tutor planned has to be thrown away and the discourse continued according to a new schema. So the planning job is not the same.

3.4.3 EDGE: a Dialogue-based Explanation System. Cawsey [1992] showed an explanation generation model in the EDGE system, which is a dialogue-based explanation system. The discourse planner of EDGE generates interactive explanations of circuit behavior.

Cawsey has transcripts of human experts describing simple circuits to novices. Some were effectively monologues, but many were dialogues where the novice interrupted with questions and the expert sometimes asked questions of the novice. From the monologues she was able to extract the basic organization and style of an explanation. From the dialogue it was possible to analyze the kinds of questions novices ask and the way tutors respond and change their tutoring plans.

The EDGE system was designed to imitate an expert giving an explanation, with interactive dialogue possible. The expert's behavior is composed of four types of utterances: 'initial', 'explanation', 'follow-up questions', and 'response' (to student questions).

The explanation’s progress can be interrupted by the student’s questions, so the EDGE planner allows interleaved explanation. By interleaving planning and execution, this system reacts to the student’s misunderstandings.

The EDGE planner has two sets of rules: ‘Content planning rules’ and ‘Dialogue planning rules’. Overall content of the dialogue is determined by the content planning rules and dialogue structure is controlled by the dialogue planning rules. The content planning rules decide what to say, based on a large number of factors including the user’s knowledge, the current object being explained, and patterns of explanation. The main goal can be decomposed into subgoals according to these influences. For example the goal of HOW-IT-WORKS (LIGHT-UNIT) will cause EDGE to plan an explanation composed of STRUCTURE (LIGHT-UNIT), PROCESS (LIGHT-UNIT), and BEHAVIOR (LIGHT-UNIT). The explanation may ultimately start with the sentence “A light detector unit is a kind of potential divider circuit”, assuming the hearer doesn’t know that already.

Dialogue planning rules (some of which are actually combined with the content planning rules in EDGE) include things like “boundary exchanges”, for example “Right, now I’m going to explain how the light detector unit works”, which integrate the explanation into the dialogue. Also included in dialogue planning is determining what the hearer wants (from the question), deciding when the system should ask a follow-up question, and acknowledging the hearer’s answer.

EDGE (like Circsim-Tutor) has a planner for the generation of dialogue. After interleaving the student’s question, this simple planner returns to its original goals. So in EDGE changing the topic is not possible. CST v.3 will have a variety of teaching methods instead of using only follow-up questions. The dialogue of EDGE is finished when the student is satisfied. In contrast, CST requires a confirmation of student understanding through correct answers. The separation of ‘content planning rules’ and ‘dialogue planning rules’ is similar to our two planners of version 3 of CST, the tutorial planner and the turn planner.

<u>3.4.4 The Program Enhancement Advisor (PEA)</u>. Moore [1995] investigated user feedback in the plan-based interactive explanation generation system, PEA (Program Enhancement Advisor). PEA gives advice to users for improving their Common LISP programs. The important observation was that when human experts give explanations, the hearers usually ask questions to get the explanations revised, extended, or clarified. Explanation among human beings is usually dialogue, not monologue. PEA first asks the users what characteristics of the LISP program they want to improve. The user can pick one or both of readability or maintainability. Then PEA gives recommendations. After that the user can ask questions about those recommendations.

Moore noted the following requirements for interactive explanation systems: naturalness, responsiveness, flexibility, and sensitivity. When the system has knowledge of discourse structure and strategies, coherent natural language explanatory dialogue can be generated. An interactive explanation system needs the ability to accept the user’s feedback, and the ability to respond to follow-up questions. Having a variety of strategies for the same goal can give some flexibility to the system. The explanation should be sensitive to the user’s goals and knowledge.

To fulfill these requirements for explanation systems, Moore focuses on reacting to feedback from the user. Because the users often do not understand the expert's responses and they want to ask a follow-up question, the feedback is especially needed.

After PEA recommends a transformation of the LISP program to the user, users may approve the transformation, reject the transformation, ask a follow-up question, or indicate that the answer is unsatisfactory. The follow-up question can take one of the following forms:

Why?
Why are you trying to achieve <u>goal?</u>
Why are you using <u>method</u> to achieve <u>goal</u>
Why are you doing <u>act?</u>
How do you achieve <u>goal?</u> [Moore 1995, p.179]

Moore argues that the intent (and therefore the system's handling) of follow-up questions depends on discourse context and the user's knowledge. She illustrates this points with the following sample dialogue:

SYSTEM:	What characteristics of the program would you like to enhance?	[1]
USER:	Readability and maintainability.	[2]
	. . .	
SYSTEM:	You should replace (SETQ X 1) with (SETF X 1).	[3]
USER:	Why?	[4]
SYSTEM:	I am trying to enhance the maintainability of the program by applying transformations... SETQ-TO-SETF is a transformation that enhances maintainability.	[5]
USER:	Why?	[6]
	. . .	

[Moore 1995, p. 181]

In the above example, the discourse context indicates that "Why?" in line 4 is a request to justify the system's recommendation. What about the "Why?" on line 6?

Among the possibilities, the user could be asking “Why does SETQ-TO-SETF enhance maintainability” or “Why are we applying transformations as opposed to some other method?” or several other things. In this case PEA must infer the user’s intention based on what it believes the user already knows and various principles that Moore derived by observing real dialogues. The first rules PEA applies are: examine the immediate focus, don’t tell users things they already know, and don’t repeat things you already said.

Users frequently fail to understand the expert’s responses. So they cannot formulate clear follow-up questions. In that case the student asks a vague question such as “Huh?.” As with “Why?” questions, the system checks the previous response produced by the system itself to find failed communicative goals using recovery heuristics.

Moore gives stress to the previous utterances of the system as an important aspect of the context for replies to questions. She insisted that to react to the user’s request the system must have recorded information about its own utterances and have reasoning strategies.

This is a little different from our Circsim-Tutor. In general Circsim-Tutor asks the questions, not the student. Circsim-Tutor maintains and uses information about the state of the student from former student answers, PEA’s model of the student is based only on what PEA has said earlier. However, there have been analyses of our human tutoring dialogue to categorize the student’s follow-up questions and the tutor’s behavior [Sanders et al. 1992; Shah 1997]. It is possible that Moore’s analysis could be quite useful to us in the future.

3.4.5 IDAS: Documentation-Question Project. IDAS (the Intelligent Documentation Advisory System) [Reiter et al. 1995, Reiter & Mellish 1993] was jointly developed by the University of Edinburgh, Racal Instruments Ltd., Racal Research Ltd., and Inference Europe Ltd. to build an advanced on-line documentation system for expert users of automatic test equipment.

IDAS has two parts: an advanced version of a conventional canned-text 'help' package, and a natural language generation system, which computes help messages on-the-fly from the user's query and a knowledge base. Among the motivations for trying to mechanically generate documentation in this manner are:

- Easier maintenance of documentation. For example, if the description of some part changes, it will be changed only once in the knowledge base and will appear in updated form in all generated text.
- More consistency between documentation and design, to the extent that the knowledge base used for documentation can be automatically kept consistent with the CAD database that describes the machine.
- Guaranteed conformance to documentation standards, such as consistent use of terminology, avoidance of various ambiguous grammatical constructs, etc.
- The ability to tailor the text to the context, e.g. the user's level of expertise or the context of the previous questions.

The authors of IDAS recognize a significant problem. Natural language generation systems that rely only on domain specific knowledge plus a lot of planning, called “deep generation”, are computationally much too slow for interactive use. Furthermore, it is very hard to develop the plans that produce text in a principled way. Even how to construct reasonable paragraphs in a restricted domain can be a big research project. On the other hand “shallow” approaches, where much of the text is pre-determined by non-principled means, risk losing some of the advantages in the above list. The authors of IDAS use several intermediate approaches, two to simplify the sentence planning and surface generation tasks, and a rule-based method to simplified the content determination task.

The most radical approach to simplification is the canned text with embedded references method, where the entire sentence is stored except for references to knowledge base objects. An example is “Carefully slide [BOARD] out along its guides.” In this sentence, “[BOARD]” might ultimately become “the DMM” or “the board in slot 6 of the instrument rack.” (This and other IDAS examples are from [Reiter & Mellish 1993]).

Another approach to simplification starts with case frames, which IDAS is using as an intermediate representation of clauses. A case frame conceptually describes one verb plus its arguments. To this, it is possible to add canned text. An example is:

```
REMOVE (actor=User, actee=Board, source=Instrument-Rack,
manner=“gently”)
```

This will became a sentence where the main action is “remove” (although another verb might be used), the arguments are obtained from the knowledge base, and the adverb “gently” is inserted into the sentence without any processing.

IDAS is similar to Circsim-Tutor in that it is generating text that imitates human text. IDAS is imitating technical writers. If a technical writer would use “gently” in a certain situation, then IDAS should use it also, even if there is no easy principled way to compute the fact. Similarly, Circsim-Tutor is imitating our human tutors. If they have effective ways to express themselves, CST must do the same. Also, both IDAS and CST operate interactively. So, they need to save computational time. The lesson here is that text generation in CST v.3 should probably use the same kinds of simplifications that IDAS uses, and stay away from deep generation.

3.4.6 PHRED: A generator for Natural Language Interfaces. Jacobs [1988] introduced a natural language generator, PHRED (PHRasal English Diction), that is designed for use in a variety of domains. It produces natural language output from a conceptual representation. It shares its knowledge base with PHRAN (PHRasal ANalyzer). PHRED and PHRAN are part of the Unix Consultant system that coaches novice users of Unix. The PHRAN analyzer makes a conceptual interpretation of the user’s free text input. The response of Unix Consultant goes to PHRED as a conceptual form to be converted to natural language output.

The knowledge base consists of pattern-concept pairs. The pattern-concept pairs are associations between linguistic structure and conceptual templates. The following example shows a pattern-concept pair associating the use of the verb “remove” with a particular “stage change” concept:

```
Pattern: <agent> <root = remove> <physob>
                  <<word = form> <container>>
Concept: (state-change (object ?rem-object)
                  (state-name location)
                  (from (inside-of (object ?cont)))
                  (to (not (concept (inside-of (object ?cont))))))
Properties: tense = (value 2 tense)
            rem-object = (value 3)
            cont = (value 5)
            forms = (active-s passive-s)
                                            [Jacobs 1988, p. 316]
```

Inside of angle brackets (< >) the pattern has linguistic information and conceptual categories.

This pattern and concept pair contributes to producing the sentence “You should remove the file from the top level directory” or the clause “to remove a file from the top level directory” according to the combination of general linguistic constraints such as surface order as well as the needs of the utterance. These pattern and concept pairs have been used to represent specialized linguistic knowledge without having rigid surface structures. Concepts which are expressed by fixed phrases or partially fixed constructions can be represented as well, by inserting the fixed parts in the pattern.

PHRED produces an utterance in three phases: Fetching, Restriction, and Interpretation.

The input to PHRED is an instantiated concept with some additional constraint information. The concept needs be unified against the concept-halves of all the pattern-concept pairs in the knowledge base. When unification is successful, the corresponding pattern-half describes one possible way to say the sentence. In order to satisfy the needs of a real-time system and counter the slowness of unification, the fetched phase extracts a list of candidate pattern-concept pairs by using a fast hashing algorithm.

The restriction phase unifies the incoming concept with each potential concept-pattern pair, instantiates the pattern, and applies constraints. When this is successful, the result pattern is given to the interpretation phase to produce the sentence. Elements of the pattern may cause PHRED to call itself recursively to construct constituent phrases.

The sharing of the pattern-concept knowledge base between PHRED and PHRAN eliminates redundancy and adds consistency between input and output. Also, it allows Unix Consultant to have an interface that is interchangeable between English and Spanish.

PHRED uses a very flexible phrasal approach for language processing. Jacobs gave attention to having patterns describe linguistic knowledge more and less specifically as needed. They range from general syntax through partially specified constructions to fixed phrases.

In terms of Reiter's consensus architecture, PHRED is concerned mostly with sentence planning and surface generation. It has no content determination. Its input is the conceptual representation of a sentence, the output is the sentence.

3.4.7 Ana: Stock Report Generation System. Kukich [1988] introduced a natural language generation system, called Ana, that generates a stock market report similar to what you may see in the newspaper. A set of half-hourly stock data that came from the Dow Jones News and Information Retrieval System is input to Ana. Then Ana outputs a summary report of the day's activity on Wall Street. The generated output is similar to human output. Ana's reports are evaluated as highly accurate, extensible, and tailorable.

Kukich believes that fluency skills are a “result of the effective identification and integration of some specific knowledge processes” (p. 280). Her research focused on the understanding of the interaction between knowledge processing skills such as semantic skills, lexical skills, syntactic skills, and grammar skills to generate fluent text.

Kukich suggested three fundamental design principles to produce fluent text in this system: the knowledge-engineering principle, the macro-level knowledge-processing principle, and the situation dependent integrated knowledge grammar principle.

The knowledge-engineering principle embodies the idea that specific semantic and linguistic knowledge of the domain of discourse are essential to compose an intelligent, fluent report. Not only domain-specific semantic knowledge and domain-specific linguistic knowledge, but also grammar rules are important. The grammar rules integrate a variety of linguistic knowledge processes. This principle defines the scope of required semantic and linguistic knowledge.

The following example of Ana’s output illustrates why considerable knowledge engineering was required:

> The stock market meandered upward through most of the morning but was pushed downhill late in the day and posted a small loss yesterday.
> [Kukich 1988, p. 299]

This is perfectly good English, but there are many domain-specific ways in which this is different from ordinary English. There is the choice of sentence construction and the choice of verbs to use such as “posted” and of metaphors such as “pushed downhill.” There is the recognition and choice of features in the Dow Jones Industrial Average data to report. That these are all part of the domain-specific sublanguage of stock market reports can be illustrated by turning it into a weather report, viz: “The <u>temperature</u> meandered upward through most of the morning but was pushed downhill late in the day and posted a small loss yesterday.”

The macro-level knowledge-processing principle focuses on “semantic units consisting of whole messages, lexical items consisting of whole phrases, syntactic categories at the clause level” [Kukich p. 289], etc. Kukich follows Becker [1975] in believing that much human language manipulation is performed at the level of fixed phrases. This macro-level knowledge processing is a top level of multi-leveled language processing theory. Processing can be shifted among macro-level, intermediate-level, and low-level in this theory. Especially, the Ana report generation system does not need to have detailed knowledge from the elementary components of the phrases for the generation of fluent text. So, most text generation in Ana is done by assembling phrases into sentences and paragraphs. Macro-level processing picks the phrases, and lower level processing creates coordination, pronominalization, etc.

The integrated-knowledge situation-dependent grammar principle came from the idea that the required grammatical knowledge is extremely situation-dependent. So, the choice of the appropriate grammatical item is a result of a set of integrated rules that came from a variety of types of semantic and linguistic knowledge at any decision point.

The macro-level phrasal approach of Jacobs and Kukich may be helpful in Circsim-Tutor.

<u>3.4.8 Stroke Consultant: Medical Expert System.</u> Collier et al. [1988] show two different text-generation systems for a medical expert system Stroke Consultant. One

system assembles phrases from a data dictionary and parses roughly to confirm that the result is well formed. Another system uses Sager's Linguistic String Parser (LSP) to generate freer text.

The first system was actually used in the Stroke Consultant system due to the advantage of speed and storage. This one produces three kinds of reports: a Current Status Report, a Discharge Report, and a Case Summary. The Current Status Report is used by the physician for resuming an earlier case. A permanent medical record, the Discharge Report is generated for the information about the patient's discharge or death. The Case Summary gives a quick review of the patient's state and physician's recommended treatment. The Current Status Report generator and Discharge Report generator have the same strategy. These generators consist of four major components: the concept chooser, the phrase builder, the grammar engine, and the output formatter. After choosing concepts (what to say and what not to say about the topic), the phrase builder finds proper phrases and puts them together. The grammar engine confirms the well-formedness of the text and corrects simple errors. This grammar engine recasts the phrase from the right hand end to left to build an intermediate form for the phrases. Then from left to right, it builds a sentence. This second pass generates a well-formed sentence from the intermediate representation. This two pass method simplifies the phrase building process. The appropriate output for the screen is emitted by the output formatter. The Case Summary is generated by a simple process mostly in tabular form. So the grammar engine component is not needed.

The second, experimental, report generator uses a different approach. Using Sager’s Linguistic String Parser, Collier et al. parsed actual reports and generated information formats. Those format slots contain information that is needed for generation such as the patient’s identification, admission data, deficit data, past medical history, and test result data. These information formats are transformed to simple sentences. Next they borrow from the Linguistic String Parser project. One basic operation of the LSP parser separates a sentence into many individual propositions. It was possible to apply those transformations in reverse, combining the simple sentences into bigger ones. Finally, parsing with LSP confirms their well-formedness.

With two text-generation systems, Collier shows several tradeoffs such as quality of sentence, required effort, time, and space efficiency. The LSP-based generator generates various sentences and structures. However, the simpler system showed the advantage of time and space efficiency. We may eventually want to use a freer method of text generation, but the second approach is not appropriate for Circsim-Tutor.

3.5 Dialogue Annotation Scheme: DAMSL

Allen and Core [1997] introduced an annotation scheme DAMSL (Dialogue Act Markup in Several Layers), developed for task-oriented dialogues. They claim that the annotation is important for showing the roles of utterances and the relationship between utterances. Ideally, good annotation that is reliable, flexible, and comprehensive, can be applied to other domain projects with other purposes later.

Allen and Core were analyzing spoken dialogues. They analyzed the speaker’s intentions in each turn, and used the intentional structure to break up the dialogue into utterances. An utterance is typically a few words from one speaker serving one goal. Overlapping speech, interleaved speech, and long pauses pose problems in spoken dialogue that were solved by bracketed markup. Utterances from two different speakers that overlap are bracketed and given index numbers, so the overlapping parts can be matched up.

Compared to former speech act theory, DAMSL allows multiple labels to multiple layers of an utterance to annotate the simultaneous performance of several purposes by one utterance.

DAMSL has three layers: Forward Communicative Functions, Backward Communicative Functions, and Utterance Features.

The Forward Communicative Function indicates the effect and interaction of an utterance on subsequent dialogue. It has speech act categories such as statements, directives, and commissives. Commissives commit the speaker to future action, e.g. by promising to do something. (It is useful to recall here that they are marking up collaborative tasks.) Directives direct the hearer to do something, for example by requesting information or by demanding an action. Statements convey information, perhaps to influence the hearer. An individual utterance might have several such functions simultaneously. The Backward Communicative Function shows the relationship between the current utterance and the previous dialogue. It has independent classes named agreement, understanding, answer, and information-relation. Agreement, for example, shows whether the utterer is agreeing or disagreeing with a previous utterance. The

Utterance Features relate to structure and content of the utterance. The system considers classes for the information level, communicative status, and syntactic features.

Core and Allen [1997] conducted a reliability experiment with the DAMSL scheme using a corpus of dialogues of humans solving transportation problems. Informal training of three undergraduates and one graduate student consisted of annotated dialogues with a GUI-based DAMSL annotation tool and compared the results with each other and against a master version. Later they annotated a series of dialogues independently. The reliability of their results was measured by computing the kappa statistic. It showed a little bit lower than the minimum value of acceptable reliability in some experiments. Two examples of reasons are: (1) it is hard to decide whether a response is an 'acceptance' or an 'acknowledgment.' (2) 'Ok' is very ambiguous in meaning. However, in most cases the kappas proved the annotations are fairly reliable. So with many independent layers DAMSL annotation scheme labels simultaneous, various actions reliably.

Some big differences between DAMSL and our own markup of tutorial dialogue in Chapter 4 are:

- Our dialogue is not spoken, it is typed, so overlapping does not occur.
- Our dialogue is not a collaborative task, it is a teacher-driven tutoring task. Many of the DAMSL annotations are oriented around the interactions between peers. Operations like conceding, promising, proposing, and so forth do not occur in our dialogues.

- We were interested in deriving tutorial goals and patterns, with the intention of programming a computer to do them. Therefore we did not annotate dialogue which was too complicated or messy, and we did not try to derive a student goal structure at all.

3.6 Effective Tutoring Patterns

Graesser et al. [1995] analyzed tutorial dialogue patterns and identified the effective learning components. This study identified the boundaries between unskilled tutoring and skilled tutoring and found out how normal unskilled tutors behave. These results are helpful to train tutors and to develop human-like intelligent tutoring systems.

They started with two sets of videotaped tutoring sessions: 44 sessions in the domain of an undergraduate psychology research methods course, 22 sessions in the domain of seventh-grade algebra. The tutors were graduate students and high-school students respectively. The dialogue in the videotapes was transcribed and analyzed.

The most striking conclusion is that even though the tutors might not have been ideal, they were very effective. An ideal tutor would be a domain expert, trained in tutoring techniques, possessing years of tutoring experience. The high school and graduate students who served as tutors were probably not ideal, they are called “normal” in this study. This study partly confirmed the metastudy of Cohen et al. [1982], which showed that tutoring as it is normally practiced raises scores significantly.

One starting point for the analysis of the normal tutoring dialogues was a list of components that are identified in the educational literature as being important to learning:

1. Active student learning
2. Sophisticated pedagogical strategies
3. Anchored learning in specific examples and cases
4. Collaborative problem solving and question answering
5. Deep explanatory reasoning
6. Convergence toward shared meanings
7. Feedback, error diagnosis, and remediation
8. Affect and motivation

[Graesser et al. 1995, p. 497]

In fact, many of these components could not often be identified in the tutoring dialogues they studied. The missing components, such as active student learning, sophisticated pedagogical strategies, and error diagnosis, are the pedagogical strategies of skilled tutors. Instead, it appears that the conversational dialogue patterns of unskilled tutors are effective in any case. From among the above components, they observed that anchored learning in specific examples and cases, collaborative problem solving, question answering, and explanatory reasoning are the most prevalent in normal tutoring.

Comparing classroom teaching to one-to-one tutoring, students were more active and learned more in contexts with frequent questions. One lesson for the intelligent tutoring system is that it should give some chance to the student to take control of the dialogue. Even during sophisticated tutoring methods such as the Socratic method, the tutoring plan must be revised in response to the student’s answers. Also, during tutoring with this strategy students can discover their misconceptions.

Another observation from this study was the use of a five step dialogue frame as a collaborative problem solving conversational pattern. The steps are: tutor asks question, student answers question, tutor gives short feedback on the quality of the student’s answer, tutor and student collaboratively improve the quality of the answer, and tutor assesses the student's understanding of the answer. One warning is that, especially in the assessment step, comprehension-gauging questions such as "Do you understand?" did not yield useful results. Graesser et al. suggested that using follow-up question is more effective.

An important observation from the dialogues was that deep reasoning questions about domain knowledge, asked by both the tutor and student, are prevalent in normal one-to-one tutoring. Student achievement and deep reasoning questions are correlated, good students frequently asked deep reasoning questions. It is not so obvious how this lesson can be applied to CIRCSIM-Tutor, but the effect is so strong that we should not forget it.

3.7 Machine Learning

The goal of machine learning is to have the machine generalize knowledge from examples. The machine learning is cumulative, so knowledge already learned is used to support learning from new experience.

Russell [1996] classifies machine learning according to four basic representations: attribute-based representation, first-order logic, neural networks, and probabilistic functions. In Chapter 6 of this thesis, an attribute-based representation is used for extracting knowledge from the human transcripts. This representation starts with a set of cases that are described by attributes and discrete values for each attribute. One attribute is picked as a target, then the machine learning algorithm will try to explain the target as a function of the other attributes. For example, in Chapter 6 one case represents one instance of a discourse marker within a tutorial dialogue. The attributes of the case might be which discourse marker occurs and what kind of sentences occur before and after the discourse marker. Many different cases are collected. Then machine learning is used for predicting which discourse marker is used, based on the types of sentences before and after.

Decision tree induction is included in the attribute-based-representation. Decision trees are fully expressive within the class of attribute-based languages. In other words, any rule describing the data can be expressed as a decision tree. Finding a pattern means being able to describe a large number of cases in a small, consistent tree. I used Quinlan's C4.5 algorithm [Quinlan 1993]. First of all this algorithm tries to find which attribute most explains the target. This attribute is put at the root of the tree, and splits the data into subsets according to the values of that attribute. It builds recursively subtrees for every branch off the root, using one fewer attribute for each set of subtrees.

Prunning is needed to complete the tree. The leaves of the decision tree, farthest from the root, are the least explanatory and describe the fewest number of cases. Pruning prevents splitting on attributes that are not clearly relevant. With some standard statistical test for significance it yields smaller trees with higher predictive accuracy.

The output of the decision tree is a set of rules. Given values of the other attributes, it predicts the value of the target. Sometimes misclassification happens due to incorrect data, non-deterministic attributes, or an insufficient set of attributes.

Machine learning has played an important role for developing rules for the discourse planner and the turn planner in the new version of CIRCSIM-Tutor.

3.8 Cue Phrases

Many researchers have studied discourse markers and other conversational cue phrases from a variety of linguistic viewpoints. For example, Halliday and Hasan [1976] analyzed discourse markers and investigated them as part of their comprehensive account of textual cohesion. Schiffrin [1987] analyzed eleven discourse markers, starting from an operational definition that was tied to no particular theory. In Schiffrin's analysis some discourse markers contribute to coherence between different utterances, for example, "so" and "because" often indicate cause and result. In this analysis "oh" is a discourse marker that sometimes marks repair, other discourse markers serve as temporal adverbs. "Well," when marking the answer to a request, can signal that the speaker is not fully complying with the request. In short, discourse markers operate in a variety of ways for a variety of purposes.

In the computational processing of text, discourse markers are useful for many tasks, for example plan recognition, anaphora resolution, and providing coherence in generated text. Some examples are cited in [Moser and Moore 1995]. Di Eugenio et al. [1997] cite evidence that proper selection and placement of discourse markers improves reading and recall. Litman [1994] devised a computational approach to distinguish between structural and sentential uses of cue words in text. The word "incidentally," for example, can have a structural use as a discourse marker indicating that a diversion follows. It can also occur as an ordinary adverb with no special discourse function. Litman used machine learning on marked-up text to make cue phrase classification rules. A noteworthy difference between Litman's rules and most previous studies is that Litman relied only on observable countable textual features.

Here, I introduce some researchers' computational approaches toward analyzing cue phrases. All three studies are similar to Litman's in that they apply machine learning to marked-up text. Moser and Moore [1995] derived rules for selection and placement of cue words. Di Eugenio et al. [1997] investigated occurrence and placement in a similar manner. Nakano and Kato [1998] claimed that the cue phrase controls dialogue and makes it more understandable especially in instruction dialogue. A difference between these three studies and Moser and Moore is that in these studies the text was marked into discourse segments, with the relations between the segments annotated. This was not the kind of directly observable information available to Litman.

<u>3.8.1 Relational Discourse Analysis.</u> Moser and Moore [1995] suggested a coding scheme called Relational Discourse Analysis (RDA) for annotating the relationship between segments of discourse in their study of features that help to predict cue placement and selection for automatic text generation. They describe RDA as a hybrid between Grosz and Sidner's ideas [1986] and Mann and Thompson's Rhetorical Structure Theory [1988]. RDA suggests that segments of discourse consist of a core that expresses the segment's purpose and any number of contributor segments that serve the core. They analyzed these "core:contributor" relations from an intentional perspective and an informational perspective. Figure 3.3 illustrates the RDA analysis of this piece of instructional text:

```
Although  A. you know that part1 is good,
          B. you should eliminate part2
                 before troubleshooting inside part3
This is because  C     1. part2 is moved frequently
          And thus     2. is more susceptible to damage than part3.
                                        [Moser and Moore 1995, p. 132]
```

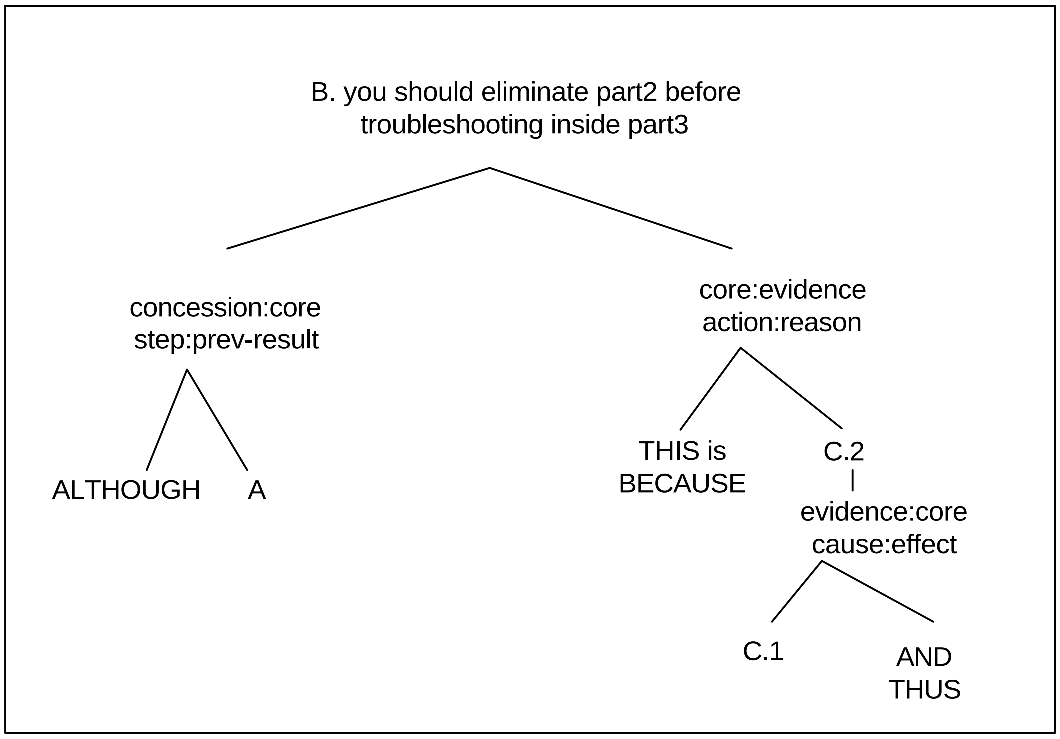

Figure 3.3. Example of RDA Analysis. [Moser and Moore 1995, p. 132]

The core of the segment is (B), “you should eliminate part2.” All else is commentary in one way or another. Segment (A) has a concessive relationship to the core, while segment (C) provides evidence for the truth of the core. Inside of (C) the core is (C.2) with contributor (C.1) providing cause-and-effect evidence.

After coding the corpus with RDA they compared the distribution of cue words “since” and “because” in instructional dialogues. They noticed that the order of core and contributor affect the choice between the two cues. Also, the choice can be affected by other factors. In contrast to earlier studies, which considered discourse to be a linear sequence of segments, they discovered that the structural embeddedness of a segment can prevent a cue from being used within an embedded relation if it also occurs in the outer unembedded relation. Overall, core position had the most important effect on cue placement and selection.

3.8.2 Predictive Features for Cue Occurrence and Cue Placement. Di Eugenio et al. [1997] identified features of cue occurrence and cue placement with an eye toward automatic explanation generation. They applied the C4.5 machine learning algorithm to the same RDA marked-up dialogues that were analyzed by Moser and Moore. The result is a decision tree based on features selected from the annotations on the text.

From the several possible distinct problems of cue usage: occurrence, placement and selection, their research focused on cue occurrence and cue placement. The Di Eugeneo et al. study found that discourse structure, intentional relations, syntactic structure, and segment complexity give the most influence to cue occurrence and cue placement.

They used the following features:

- Segment Structure: shows the global structure of the "core:contributer" relation between segments such as position of contributor relative to the core.
- Core:Contributer relation: captures intentional relation, informational relation, syntactic relation, and adjacency of the two segments.

- Embedding: shows the type of core and contributor and the number of relations in the hierarchy containing both of them.

After several experiments they obtained the best tree with the lowest error rate. This tree showed that the structure of segments is a fundamental feature for determining cue occurrence. Also, informational and syntactic relations are influential as well. However, adjacency did not function as a predictor. But just using individual features it was hard to reliably predict the cue occurrence. Cue placement, on the other hand, is predictable from the syntactic relation between core and contributor.

Although coding core:contributor relations in CIRCSIM-Tutor project dialogues is different with our mark-up as described in Chapter 4, this approach became the first step in my cue selection work using machine learning.

<u>3.8.3 Predictive Features for Cue Phrase Selection.</u> Nakano and Kato [1998] also applied C4.5 to determine factors and precise selection rules for cue phrases. The dialogues they studied involved a teacher instructing a student in a task. Their work focused on sentence-initial cue phrases because cue phrases in this position frequently refer to goals or direct actions, and are therefore important to the student who is trying to learn and execute the task.

They annotated the corpus of instructional dialogue and classified cue phrases into three classes: called the changeover, conjunctive, and ordinal classes. Their annotation focused on discourse segment boundary and the level of embeddedness of the segment, since they wanted to see the relationship between cue phrases and dialogue structure. Their work showed hierarchical relationships between tutorial goals, similar to my annotation in Chapter 4 of this thesis. Also, they annotated discourse purposes and dialogue exchanges such as confirmation-answer, question-answer. This work is similar to our analysis of goal structure.

Using C4.5 they carried out learning experiments focused on the usage of three kinds of cue phrases (changeover, conjunctive, ordinal), starting with ten features that came from discourse structure, task structure and dialogue context. Their best model that uses the fewer number of learning features without sacrificing accuracy has the following six features:

- Embedding: The depth from the top level.
- Place: The number of elder sister segments.
- Discourse Transition: Type of change in attentional state, such as pop or push.
- Subgoal: The number of subgoals of the current goal.
- Pre-exchange: The type of exchange from the preceding segment, such as question-answer.
- Preceding segment: The cue phrase in the preceding segment.

With the above six features the best model showed 70% accuracy and 25% error rate. They concluded that discourse structure influences selection. Also, they insisted that task structure and dialogue context are factors that cannot be ignored in the choice of cue phrases.

CHAPTER IV

ANALYSIS AND MARK UP OF TUTORING TRANSCRIPTS

The CIRCSIM-Tutor project has collected about 50 transcripts of physiology professors tutoring students individually. In these tutoring sessions the professor and student were in separate rooms, communicating only by typing on computer keyboards. These transcripts are our basic data from which we have discovered tutor goals and dialogue moves, ways to express each move, and how students respond.

In this thesis excerpts from transcripts contain reference tags such as “K10:29.” This interpreted as meaning the excerpt came from transcript number K10 turn number 29. Frequently I will specify a range of turns, “K10:29~38.” specifies turns 29 through 38. Transcripts with K-numbers were taught by expert tutors while transcripts with N-numbers were taught by novice tutors, as described in Chapter 5. If I edited the excerpt I mark it with the word “after,” as in “after K10:29~38.”

With the intention of determining plans for the CST v.3 planner, Reva Freedman and I analyzed these transcripts. We have analyzed over 270 turns from the 5000 turns of dialogue in our collected transcripts. We marked up the transcripts in an SGML style so some processing by computer is possible.

In this chapter I show the planning goals and arguments that resulted. I also show how the tutor changes plans according to the student’s unexpected answer. Finally I show some examples of the detailed markup.

4.1 What We are Marking Up

Here is an explanation of how the dialogue in the transcripts is structured. The tutoring sessions concern a particular topic in physiology, a mechanism called the 'baroreceptor reflex', which is part of blood pressure regulation.

- The session starts with the tutor and student introducing themselves in some way, and the student is given instructions.
- The student reads a description of a procedure that disturbs the blood pressure, for example, the patient loses a liter of blood.
- The student has a chart called the 'predictions table.' It is a table containing the names of seven important physiological variables, and three stages in which they vary over time. The student is asked to predict a qualitative change: increase (+), decrease (–), or no change (0).
- The student's first job is to identify which of the variables in the predictions table is affected first, and whether it increases or decreases.
- Some conversation (teaching) between the teacher and the student may ensue until the student gets this first prediction correct.
- The student then predicts the changes in the other variables for the first (DR) stage, writing the prediction in the prediction table (which is in the student's possession) and typing the prediction to the tutor

in the other room. Sometimes there is more conversation at this point.

- (*) After the last DR prediction, the tutor starts to teach, usually until satisfied that the student understands the behavior of the seven variables in the DR stage.
- The student makes predictions for the RR phase for all seven variables. Again, some tutoring may happen while the student is predicting the variables one-by-one and transmitting the predictions to the tutor.
- (*) The tutor teaches the RR phase
- The student makes SS phase predictions.
- (*) The tutor teaches the SS phase.

It is those sections of the conversation marked (*) above, which we are trying to analyze. The reason is that these sections most closely resemble the tutoring regime used by the CIRCSIM-Tutor program, which in general does not attempt to teach until it has a set of predictions to work with. A substantial fraction of the dialogue is concerned with the mechanics of the session and collecting predictions from the student. This has not been included in the transcripts we are to mark up since students using CIRCSIM-Tutor enter predictions in table, instead. We are also skipping any tutoring dialogue that took place while the predictions were being collected.

4.2 Structure of the Goal Hierarchy

I marked up the tutorial goal structure in the transcripts [Kim et al. 1998a, 1998b] to use as a basis for plan-based text generation [Freedman & Evens 1996]. Their analysis produces multiple nested annotations showing both global goals for tutoring and local goals for maintaining a coherent conversation and providing appropriate responses to the student.

CIRCSIM-Tutor v. 3 requires a set of tutorial and conversational goal schemas in order to produce coherent conversations. The analysis in this chapter is an extension of the one introduced by Freedman [1996b]. It is based on approximately 350 instances of global tutoring goals and 50 instances of local goals.

Tutorial goals are expanded in a hierarchy, as shown in Figures 4.1, 4.2 and 4.3. At the highest levels, ***T-tutors-procedure*** and ***T-tutors-stage*** show the tutoring of the particular procedure and the stage of the reflex response. The ***T-tutors-stage*** expands to:

T-introduces-stage	"Let's take a look at your predictions in DR."
T-corrects-variable	"Take the SV first, can you tell me...."
T-concludes-stage	"All of your other DR predictions were correct."

Below the ***T-corrects-variable*** level, two sections of dialogue are generated for each variable that the student did not predict correctly. ***T-introduces-variable*** introduces the variable as a referent in the conversation and ***T-tutors-variable*** does the actual tutoring. Tutoring requires at least three levels of goals below the variable level: the 'method' level, the 'topic' level, and the 'primitive' level. The method level shows how to teach about a variable. The topic level represents each item that must be taught. These content items largely involve domain content. The primitive level shows how this information is communicated to the student.

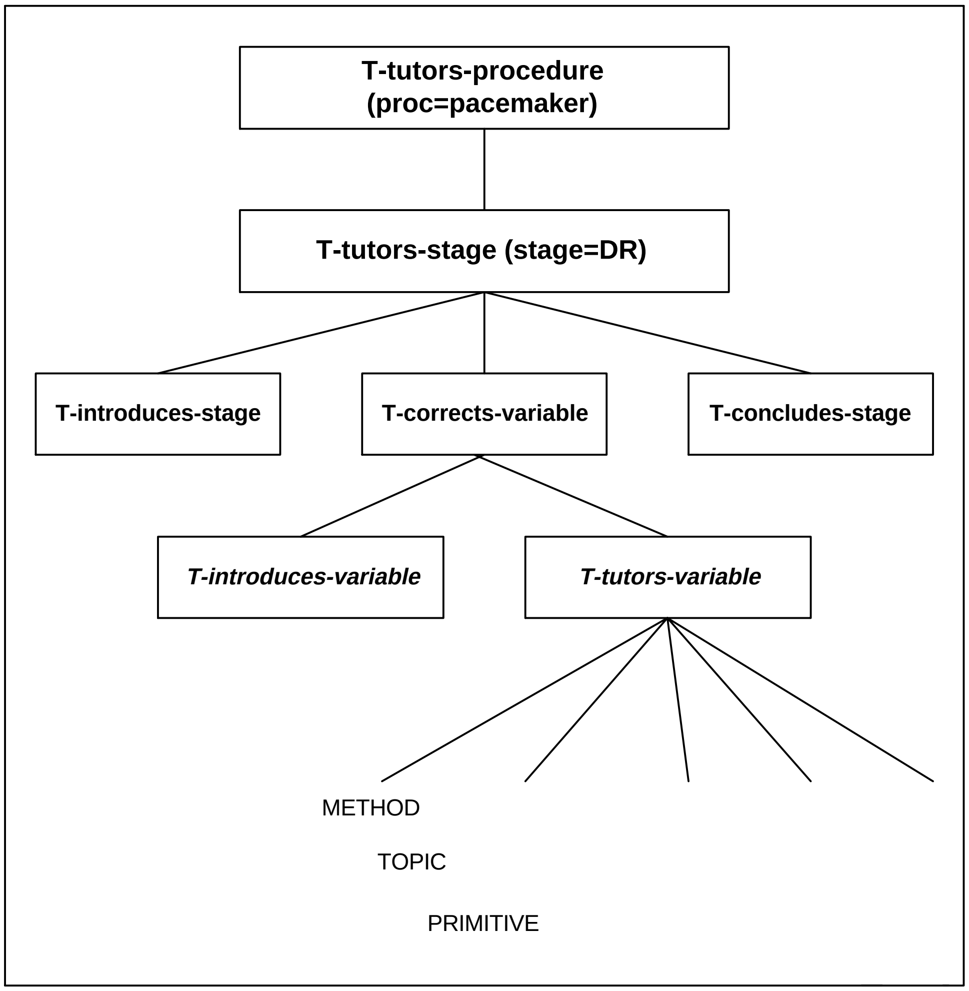

Figure 4.1. Goal Hierarchy I

4.3 The Categories of Methods, Topics, and Primitives

4.3.1 The Method Level. The method level shows how to teach about a variable. It can be used to express various types of deductive reasoning, interactive questioning and exploration of anomalies.

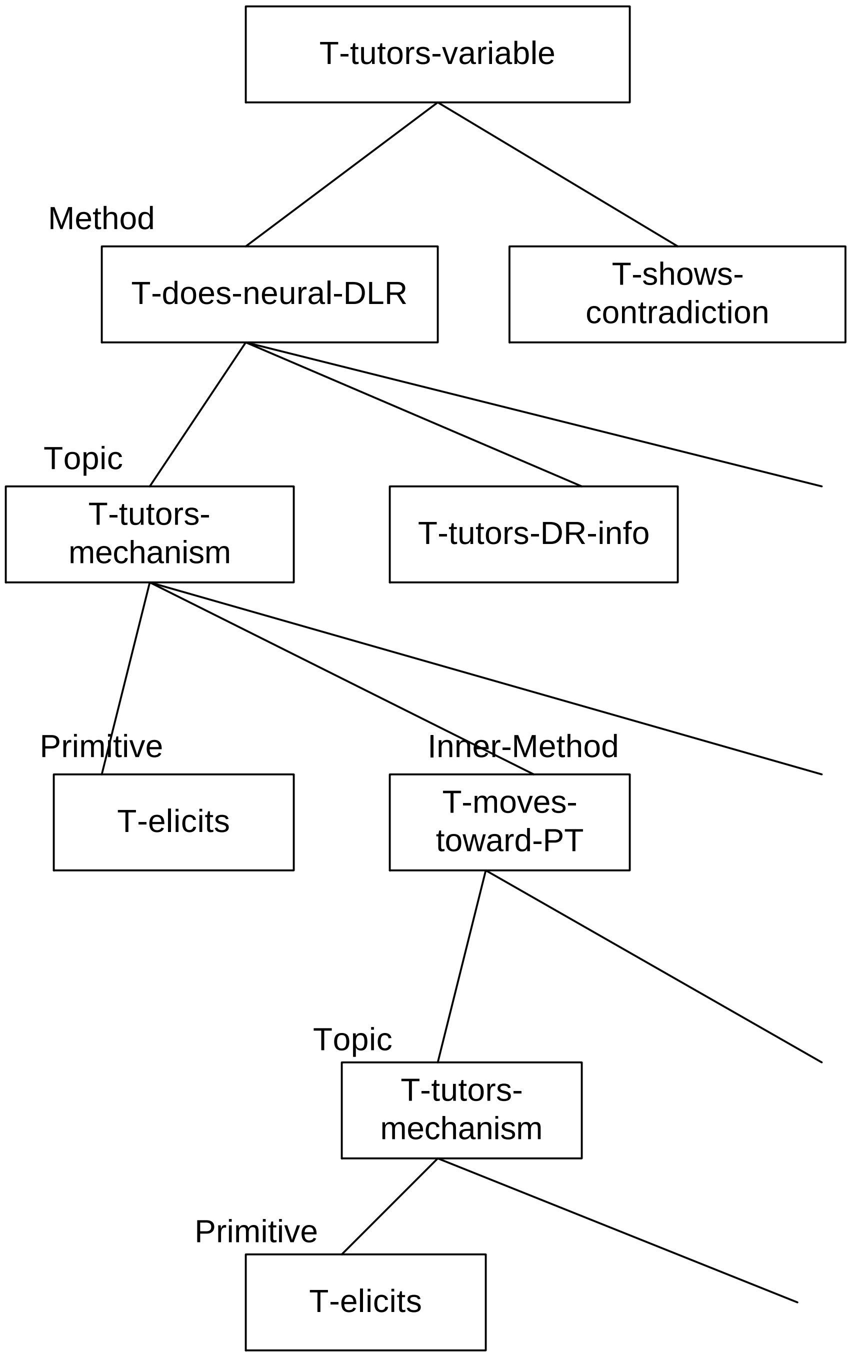

Figure 4.2. Goal Hierarchy II

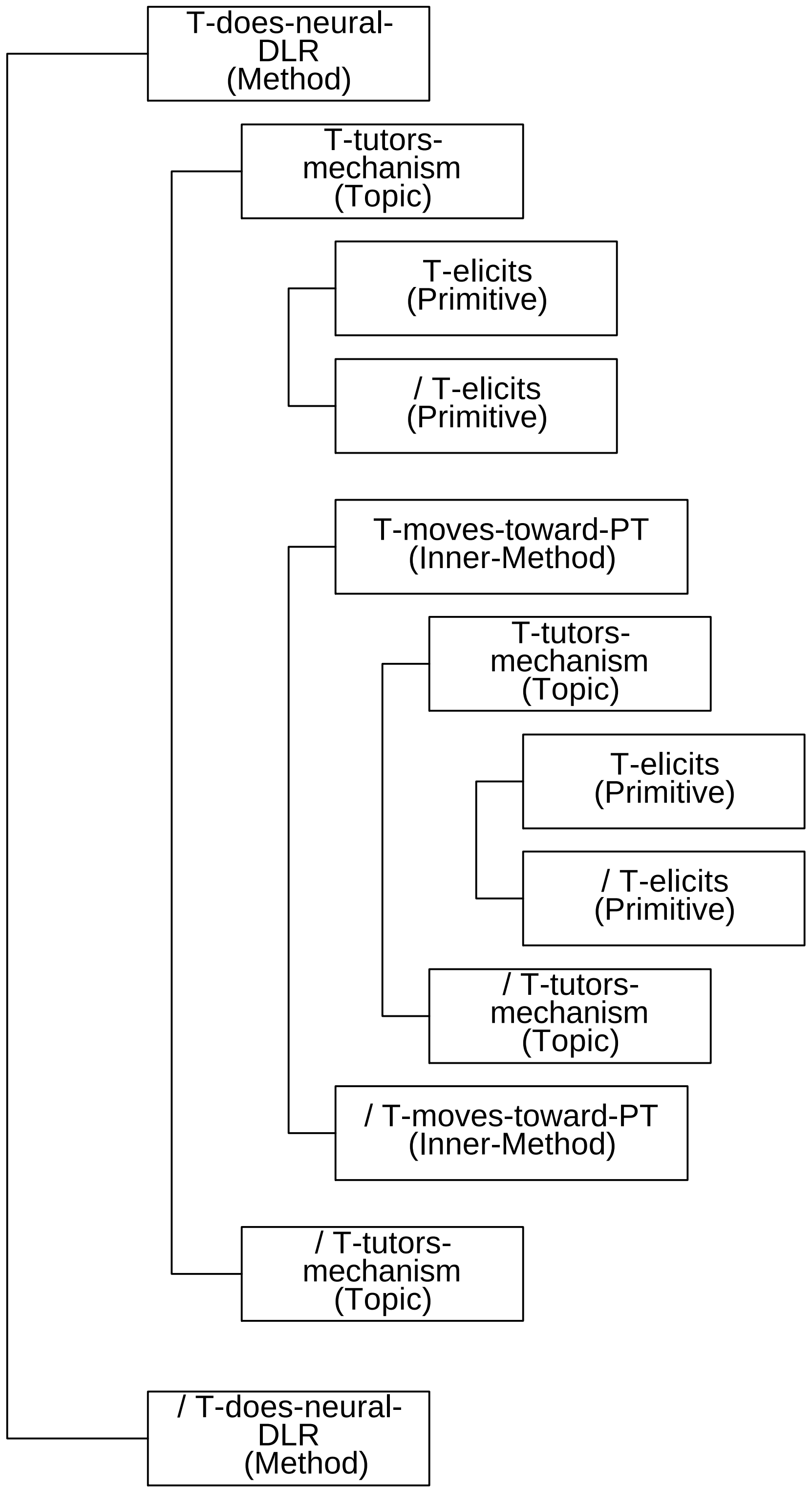
T-does-neural-DLR (Method)
T-tutors-mechanism (Topic)
T-elicits (Primitive)
/ T-elicits (Primitive)
T-moves-toward-PT (Inner-Method)
T-tutors-mechanism (Topic)
T-elicits (Primitive)
/ T-elicits (Primitive)
/ T-tutors-mechanism (Topic)
/ T-moves-toward-PT (Inner-Method)
/ T-tutors-mechanism (Topic)
/ T-does-neural-DLR (Method)

Figure.4.3. Nested Goal Structure

To refine ***t-tutors-variable*** the tutor chooses a method depending on a number of factors, including domain knowledge (e.g., the mechanism of action of a variable), dialogue history (e.g., the student's previous utterance), and the student model (i.e., how well the student is doing).

Here is a collection of method-level tutoring goals we have observed in the transcripts.

4.3.1.1 T-does-neural-DLR. If the variable is controlled by the nervous system, the tutor often chooses the question and answer style method ***t-does-neural-DLR***. (DLR stands for directed line of reasoning, a form of Socratic dialogue.)

tu: Can you tell me how TPR is controlled?
st: Autonomic nervous system.
tu: And the predictions that you are making are
for the period before any neural changes take place.
So what about TPR?
st: No change. [after K10:29~38]

4.3.1.2 T-tutors-via-determinants. For non-neural variables the most common schema is ***t-tutors-via-determinants***. With this method the tutor corrects the value of a variable by invoking a relationship with another core variable.

tu: What parameter produces a change in RAP?
st: CO.
tu: Do you remember a relationship between CO and RAP?
st: Inverse.
tu: Right, then what is the value of RAP? [after K22:40~50]

4.3.1.3 T-moves-forward. This method is similar to ***t-tutors-via-determinants*** but it applies when the determinant has already been mentioned in the conversation. Compared with the ***t-does-neural-DLR*** method and the ***t-tutors-via-determinants*** method, this method is less directly based on the domain reasoning used by the tutor to solve the problem.

tu: So, CO inversely determines RAP and you predicted that CO would increase. So what happens to RAP?
st: RAP decreases
tu: And if RAP decreases what will happen to SV? [K36:170~172]]

4.3.1.4 T-shows-contradiction. With this method, the tutor corrects the student's error by pointing out a physiological inconsistency in the student's answers.

tu: So you have now predicted that CO will fall and that HR is down but SV is up. How is this possible? [K44:154]

4.3.1.5 T-explores-anomaly. This method is superficially similar, but it is used in cases where the reported facts only appear inconsistent. Its goal is to ensure that the student really understands the deeper qualitative relationships among the variables.

tu: So, we have HR down, SV up and CO down. How is that possible? [K27:72]

4.3.1.6 T-diagnoses-error. This method is used when the tutor wants to identify a student misconception. Although the computer tutor may not be able to handle this method, we observe in the transcripts that human tutors do it sometimes.

tu: Why do you think that TPR will decrease? [K27:50]

4.3.1.7 T-tutors-via-deeper-concepts. This is used to give more detailed explanations to the student after failing to get a correct answer from the student by using only the seven core variables. This method gives information to the student (or elicits it from the student) in terms of a more detailed physiological model.

tu: The central venous compartment is a compliant structure
that contains a certain volume of blood ... [K14:39]

4.3.2 Inner Method Level. The method level describes a schema for correcting one variable or exploring one anomaly as described before. A method schema is composed of individual statements and questions we call topics, described below in 4.3.3. You can think of each topic as a single unit of knowledge. After each topic has been successfully communicated to or elicited from the student, the method is completed. But what happens when a topic is not successfully communicated? In particular, what happens when the student gives a wrong answer to a question? An action the tutor can take when the student incorrectly answers a question is to correct the one topic the student had a problem with, then proceed with the original method. Figure 4.4 contains such an example, where the student answered “radius of arterioles” instead of “neural.” The schema which corrects one topic in response to an unexpected answer is called an inner method.

Usually an inner method refers to a more detailed physiological model. The **t-moves-toward-PT** inner method shown in Figure 4.4 is an instance of this. This inner method tries to correct the student’s answer by referring to one of the prediction table variables or the baroreceptor reflex, using the student’s more detailed knowledge. These inner dialogues also follow the method / topic / primitive hierarchy and are nested inside the topic that provoked the student’s near-miss response. We especially want to add this

```
<T-does-neural-DLR>
  <T-tutors-mechanism>
    <T-elicits >
        tu:What is the primary mechanism of control of TPR?
    <S-ans, catg=near-miss>
        st:Radius of arterioles.
    </S-ans>
    <T-ack  type=positive>
        tu:Yes.
    </T-ack>
    </T-elicits>

    <T-moves-toward-PT  method-type=inner>
      <T-tutors-mechanism  var=RA>
        <T-elicits  DM="and">
        tu:And what is the primary mechanism by which
                arteriolar radius is controlled?
        <S-ans, catg=correct>
        st:Sympathetics.
        </S-ans>
        </T-elicits>
      </T-tutors-mechanism>
    </T-moves-toward-PT>

  </T-tutors-mechanism>

                      . . .
</T-does-neural-DLR>
                                                  [K12:37~40]
```

Figure 4.4. Inner Method I

feature to our tutoring system because it is a way of tailoring our responses to the student's individual needs.

Figure 4.5 shows another inner-method ***t-moves-to-previous-concepts***.

4.3.3 The Topic Level. The topic level represents each item that must be taught. These content items largely involve domain content. A method typically consists of a series of topic operators. For example, the following topic operators can be used to build the ***t-tutors-via-determinants*** method mentioned above. Each topic operator is illustrated with an example from the transcripts.

T-tutors-determinant

First what parameter determines the value of RAP? [K13:37]

T-tutors-relationship

Do you know how RAP will change if something produces a change in CO? [K14:43]

T-tutors-value

So, what would happen to RAP? [K11:69]

To build the ***t-does-neural-DLR*** form, the tutor may use the following topic operators, followed by ***t-tutors-value***.

T-tutors-definition

Can you tell me what you think that IS means? [K47:56]

T-tutors-mechanism

Can you tell me how TPR is controlled? [K10:56]

```
<T-tutors-determinant  var=RA>
  <T-elicits>
     tu:If I have a single blood vessel, what parameter most strongly
            determines its resistance to flow?
  <S-ans  catg=near-miss>
     st:Diameter.
  </S-ans>
  </T-elicits>

  <T-moves-to-previous-concepts  method-type=inner>
    <T-tutors-determinant  var=RA>
      <T-elicits  DM="and">
     tu:And physiologically, what determines the diameter of the blood
            vessels?
      <S-ans  catg=correct>
     st:The sympathetic tone supplied to its smooth musculature.
      </S-ans>
      <T-ack  type=positive>
     tu:Right.
      </T-ack>
      </T-elicits>
    </T-tutors-determinant>
  </T-moves-to-previous-concepts>

</T-tutors-determinant>
                                                        [K27:52~56]
```

T-tutors-DR-info

And the predictions that you are making are for the period before any neural changes take place. [K10:31]

When the tutor wants to teach by showing a contradiction, the ***t-presents-contradiction*** and ***t-tutors-contradiction*** topics are needed.

T-presents-contradiction

You predicted that it would go up......But remember that we're dealing with the period before there can be any neural changes. [K10:41~43]

T-tutors-contradiction

How can CC go up if it's under neural control?
(And then the tutor expands tutoring by giving the answer.) [K10:43]

Whenever a deeper conceptual model has been introduced, the tutor must eventually return to the core variable that started the discussion. The topic ***t-tutors-PT-entry*** can be used for this purpose:

T-tutors-PT-entry

What parameter in the prediction table reflects the filling of the left ventricle?
[K27:66]

Inside of the ***t-diagnoses-errors*** method the ***t-identifies-problem*** topic is used to diagnose a problem:

T-identifies-problem

Why do you think that TPR will decrease? [K27:50]

T-tutors-compliance-info

The central venous compartment is a compliant structure that contains a certain volume of blood. [K14-39]

The ***t-explores-anomaly*** method is expanded by the ***t-presents-anomaly*** and ***t-tutors-anomaly*** topics.

T-presents-anomaly

case 1: So, in DR HR is up, CO is up, but SV is down. [K25:62]

case 2: So, CO decreases even though SV increases. [K26:76]

case 3: So, we have HR down, SV up and CO down. [K27:72]

T-tutors-anomaly

case 1: How can you explain this?
The decrease due to the lowered heart rate is greater
then the increase due to increased stroke volume. [K26:36~37]

case 2: How is this possible? HR is down more than SV is up. [K27:72~73]

The ***t-tutors-consequence-value*** topic is used to show the value of a variable as a consequence of the value of a determinant. The determinant, its value, and the variable being tutored are usually mentioned within this topic, and the value of the variable is elicited. This topic occurs inside of the ***t-moves-forward*** method.

T-tutors-consequence-value

tu: Next, when RAP increases what effect would that have on SV?
st: SV would increases also.
tu: Sure. [K39: 128~130]

4.3.4 The Primitive Level. The topics share the primitive operators ***t-elicits*** and ***t-informs***. The ***t-elicits*** operator is used when we want the student to participate actively by answering a question. With ***t-informs*** the tutor gives some information to the student.

T-elicits info=var-value

So, what happens to SV? [K14:53]

T-informs info=DR-info attitude=remind

But remember that we are dealing with the period before
there can be any neural changes. [K10:43]

T-informs narrative-mode=summary

So, we have HR down, SV up and CO down. [K27:72]

Figures 4.2 and 4.3 show the hierarchical and nested structure of all three levels: the method, topic, and primitive levels.

4.4 The Arguments

At any level operators can have arguments such as the variable name or the information desired. Other arguments refer to interpersonal aspects of an utterance (attitude) or textual aspects (narrative mode). Arguments are also inherited from higher level, enclosing goals.

Here is a primitive-level goal showing several types of arguments:

```
<T-elicits    info=determinant      attitude=rephrase-question
                        narrative-mode=reference    attempt=2>
```

Information can be any piece of content we are trying to inform or elicit. Attitude shows the tutor's personal intentions and narrative-mode implies textual meanings. The number of attempts shows how many times the tutor has tried tutoring inside of the same method or topic. Next, I show argument groups that I have defined from my reading of the transcripts:

4.4.1 Variable

We have 7 core variables: HR, IS, TPR, CO, MAP, CVP, SV. There are other variables that appear infrequently. Some older transcripts use CC in place of IS and RAP in place of CVP.

(1) var = HR

(2) var = BV (Blood Volume)

4.4.2 Type

(1) method-type = inner: This is used for the inner-method type.

<T-moves-toward-PT method-type=inner>
  <T-tutors-mechanism var=RA>
    <T-elicits DM="and">
And what is the primary mechanism by which arteriolar
radius is controlled?
. . . [K12:39]

(2) type = neural

<T-shows-contradiction type=neural>
tu: But (if) CC is under neural control, how would it be affected
in the DR period?
st: I EDV.
tu: You can't have it both ways.
</T-shows-contradiction> [K11:55~57]

(3) type = 3-vars or type = 2-vars

<T-explores-anomaly type=3-vars>
So, in DR HR is up, CO is up, but SV is down.
</T-explores-anomaly> [K25:62]

(4) type = explain-DR or type = mention-DR

Sometimes the tutor explains the meaning of the DR stage and sometimes just mentions the name of the DR stage as a reminder.

case 1:
<T-informs info=DR-info type=explain-DR atti=remind>
Remember that we're dealing with the short period before you get a reflex response.
</T-informs> [K12:35]

case 2:
<T-informs type=mention-DR DM="so">
So, in the DR .....
</T-informs> [K27:60]

4.4.3 Information

(1) info = var-value

<T-informs info=var-value>
You predicted that it would go up.
</T-informs> [K10:41]

(2) info = reason

<T-elicits info=reason>
What must be true if all three of these predictions are correct?
</T-elicits> [k25:66]

(3) info = HR-vs-SV

<T-informs info=HR-vs-SV atti=speak-to-answer>
It is true that CO=SV x HR.
</T-informs> [K14:49]

(4) info = EDV-anatomy

<T-elicits info=EDV-anatomy >
When you talk about EDV what structure in the heart are you referring to?
</T-elicits> [K22:42]

(5) info = PT-info

<T-informs info=PT-info>
However, neither of these are in the predictions table.
</T-informs> [K22:46]

(6) info = DR-info

<T-informs info=DR-info>
Remember that we’re dealing with the short period before you get a reflex response.
</T-informs> [K12:35]

(7) info = RR-info

<T-informs info=RR-info>
Now we are trying to think about what happens to the system when the baroreceptor reflex is activated.
</T-informs> [K45:102]

(8) info = neural-info

```
<T-informs  info=neural-info>
CC is under neural control
</T-informs>                                                  [K11:57]
```

(9) info = Starling's law

(10) info = determinant-value

(11) info = determinant (of variable)

(12) info = relationship (between 2 variables)

(13) info = definition

<u>4.4.4 Attempt #.</u> Default value is 1.

(1) attempts = 2

```
<T-elicits>
Then what is the value of CC?
<s-ans catg=incorrect>
 . . .
</T-elicits>
 . . .
<T-elicits   attempts=2 >
So what's your prediction about CC?
</T-elicits>                                              [K11:57~61]
```

<u>4.4.5 Discourse Marker</u>

(1) DM = "and"

```
<T-elicits  DM="and">
And during DR what changes in ANS activity occur?
</T-elicits>                                                  [K48:48]
```

(2) DM = " but"

(3) DM = "so"

(4) DM = " now"

(5) DM = " then"

(6) DM = " therefore"

(7) DM = " first"

(8) DM = " well"

(9) DM = " since"

(10) DM = "however"

4.4.6 Attitude

(1) attitude = rephrase-question

tu: Now, what two parameters in the predictions table together
 determine the value of the SV?
st: CO and HR.
tu: No.
 It is true that CO=SV x HR.
<T-elicits atti=rephrase-question attempts=2>
 What I was asking is what determines how much blood is
 ejected from the heart each time it beats (the SV)?
</T-elicits> [K14:47~49]

(2) attitude = bolster-answer

case1:
tu: What are the determinants of SV?
st: Determinants are end-diastolic volume, afterload i. e. MAP,
 and I think to a small degree, heart rate.
tu: Well that's partly correct.
<T-informs attitude=bolster-answer>
 EDV is certainly a determinant.
</T-informs> [K20:34~36]

case 2:
tu: But what determines the volume of blood in the central
 venous compartment?
st: How about CO?
<T-informs attitude=bolster-answer>
tu: Certainly, CO is the determinant I'm looking for here.
</T-informs> [K25:52~54]

(3) attitude = qualify-answer

Qualify-answer is a case when the student's answer was a near-miss or partially-correct and the tutor corrected it.

tu: What are the determinants of SV?
st: Determinants are end-diastolic volume, afterload i. e. MAP,
tu: Well that's partly correct.
EDV is certainly a determinant.
<T-informs attitude=qualify-answer>
tu: Afterload (I. E. aortic pressure is important but only when it
Otherwise MAP has little effect on SV.
</T-informs> [K20:34~36]

(4) attitude = reject-answer

Reject-answer is a case where the student's answer is not correct, and it would be perfectly permissible for the tutor to say so and not address it at all.

tu: what parameter in the predictions table relates to the volume
that will be present in the central venous compartment?
st: Co and SV.
tu: Well CO certainly does
<T-informs attitude=reject-answer>
SV is a determinant of CO.
</T-informs> [K14:41~43]

(5) attitude = give-answer

case 1:
tu: First, what parameter determines the value of RAP?
st: Venous return and peripheral resistance influences return.
tu: Not in the way that you seem to think.
<T-informs attitude=give-answer>
CO is made to vary
</T-informs> [K13:37~39]

case 2:
tu: Do you know how RAP will change if something produces a change in CO?
st: If CO increases then RAP should also increase.
tu: No.
<T-informs attitude=give-answer>
  When a change in CO is the independent variable (the thing changed)
  then RAP changes as the dependent variable in the opposite direction
  (CO and RAP are inversely related under these conditions).
</T-informs> [K14:43~45]

(6) atttitude = repeat-answer

tu: So what does alter RAP?
st: Venous resistance and blood volume
tu: You are correct,
<T-informs attitude=repeat-answer>
  both of these would alter RAP.
</T-informs> [K22:44~46]

(7) attitude = speak-to-answer

Speak-to-answer is the case where the student's answer was not correct, but there was enough of a correct idea in it that the tutor did not simply reject it. Similarly the student might have misapplied a completely correct idea.

tu: Now, what two parameters in the predictions table together
  determine the value of the SV?
st: CO and HR.
tu: No.
<T-informs info=CO-equation attitude=speak-to-answer>
  It is true that CO=SV x HR.
</T-informs> [K14: 47~49]

(8) attitude = give-hint

tu: What parameter DOES determine RAP?
st: Map.
<T-informs info=CVP attitude=give-hint>
RAP is approximately equivalent to central venous pressure.
</T-informs> [K14:31~35]

(9) attitude = rephrase-answer

tu: How is TPR controlled?
st: Sympathetic vasoconstriction.
tu: Right.
<T-informs attitude=rephrase-answer>
TPR is primarily under neural control.
</T-informs> [K11:49~51]

(10) attitude = remind

case 1:
<T-informs attitude=remind>
But remember that we're dealing with the period before there can be any neural changes.
</T-informs> [K10:43]

case 2:
<T-informs attitude=remind>
Remember that we're dealing with the short period before you get a reflex response.
</T-informs> [K12:35]

4.4.7 Narrative-mode

(1) narrative-mode = reference

case 1:
<T-informs narrative-mode=reference>
You predicted that CC would be unchanged and that RAP increased.
</T-informs> [K27:68]

case 2:
<T-informs narrative-mode=reference>
You predicted that TPR would increase.
</T-informs> [K48:44]

(2) narrative-mode = summary

case 1:
<T-informs narrative-mode=summary>
So, we have HR down, SV up and CO down.
</T-informs> [K27:72]

case 2:
<T-informs narrative-mode=summary>
So, in DR HR is up, CO is up, but SV is down.
</T-informs> [K25:62]

4.4.8 Softener

(1) softener = "can you tell me"

<T-elicits softener="can you tell me">
Can you tell me how TPR is controlled?
</T-elicits> [K10:29]

(2) softener = "do you think that"

(3) softener = "do you know"

4.4.9 Stage-modifier

(1) Stage = "during DR"

<T-elicits Stage="during DR">
During DR what changes in ANS activity occur?
</T-elicits> [K48:48]

(2) Stage = "in DR"

(3) Stage = "in the DR"

(4) Stage = "during the DR period"

(5) Stage = "in the DR period"

(6) Stage = "DR"

4.4.10 Context-setting-clause

(1) context-setting-clause = “if CVP is down”

<T-elicits context-setting-clause=“if CVP is down”>
So, if CVP is down what happens to (RAP and) SV?
</T-elicits> [K25:60]

(2) context-setting-clause = “given this info”

<T-elicits context-setting-clause=“given this info”>
So what do you think happens to SV, given this info?
</T-elicits> [K20:36]

(3) context-setting-clause = “that being the case”

<T-elicits context-setting-clause=“that being the case”>
That being the case, what will happen to RAP-DR in this situation?
</T-elicits> [K26:72]

(4) context-setting-clause = “when CO decreases”

(5) context-setting-clause = “if all three of these predictions are correct”

(6) context-setting-clause = “if something causes to CO to change that”

(7) context-setting-clause = “if it is under neural control”

4.4.11 Specify-value

(1) specify-value = “go up”

<T-informs specify-value=“go up” >
You predicted that CO-DR would go up.
</T-informs> [K25:56]

(2) specify-value = “decrease”

<T-elicits specify-value=“decrease”>
Why do you think that TPR will decrease?
<T-elicits> [K27:50]

(3) specify-value = “increase”

(4) specify-value = “no change”

(5) specify-value = “go down”

(6) specify-value = “i”

(7) specify-value = “ up”

(8) specify-value = “down”

(9) specify-value = “unchanged”

4.5 Tutor’s Response to Student

The transcripts show several kinds of student answers as follows:

- correct
- clearly incorrect
- near miss answer, which is pedagogically useful but not the desired answer
- don’t know answer, where the student said something equivalent to “I don’t know”
- partially correct answer, meaning some part of the answer is correct and the rest is incorrect

Later, Zhou [1999a] added following kinds of student answers:

- grain of truth, which is an incorrect answer but it shows a partially correct understanding of the problem
- misconception, meaning the answer has common confusion or piece of false knowledge
- mixed answer, shows a combination of other answer categories.

Following are several kinds of tutor acknowledgments.

- positive
- negative
- partially correct

The following example is the usual case of a positive tutor's response to a correct student answer.

```
<T-elicits  DM="so">
tu: So what would happen to RAP?
    <S-ans  catg=correct>
st: D, SV D.
    </S-ans>
    <T-ack  type=positive>
tu: Great.
    </T-ack>
</T-elicits>
                                            [K11:69~71]
```

In the case of a partially-correct answer we marked one more argument ***detail.*** ***Detail*** enumerates the categories of each part of the student's answer, in order. In the following example , "end-diastolic volume" is near-miss, "MAP" is correct, and "heart rate" is incorrect.

```
<T-elicits>
tu: What are the determinants of SV?
    <S-ans  catg=partially-correct  detail="near-miss correct incorrect">
st: Determinants are end-diastolic volume, afterload i. e. MAP, and I think
    to a small degree, heart rate.
    </S-ans>
    <T-ack  type=partially-correct>
tu: Well that's partly correct.
    </T-ack>
</T-elicits>
                                            [K20-tu-34~36]
```

Sometimes the student gives an answer without confidence like the next example. In these cases we marked one more argument type=hedge.

```
<T-elicits>
tu: What other variable is under neural control-primarily?
    <S-ans  catg=correct    type=hedge>
st: CC?
    </S-ans>
    <T-ack  type=positive>
tu: Yes.
    </T-ack>
</T-elicits>
                                             [K10-tu-39-41]
```

Depending upon the category of student answer, the tutor may continue with the current strategy or choose a new one. If the student's answer is correct, the tutor moves to the next goal, sometimes giving an acknowledgment such as "good" or "right". In response to a student answer that is clearly incorrect, the tutor may change to a new method, which sometimes will build on the student's answer. Other possibilities are to ask the question again in a different way, or to give the student the answer so that tutoring can continue. In any of these cases, the tutor may address the student's wrong answer before continuing with the hierarchical expansion of the tutoring plan. This feature is a way of tailoring our responses to the student's needs.

A common motivation for changing to a new tutoring method is to refer to a more detailed physiological model. This is often triggered by the student's use of a term coming from a deeper model, one type of near-miss answer. (The deeper model can also be introduced by the tutor, in cases where it has not been possible to explain a concept using only the core variables.)

"Don't know" answers are treated in a similar fashion to incorrect answers, but the tutor has less information available to fashion a specific response.

In response to a partially correct answer, the tutor usually responds to the correct part, then the erroneous part.

Figure 4.6 shows the tutor changing his tutorial strategy in response to the student's clearly incorrect answer. This is an attempt to teach the student about the value of RAP (right atrial pressure) using the relationship between the determinants. The student gives an incorrect answer about the relationship between CO (cardiac output) and RAP. In this example, the tutor replaces the current tutoring style with one that uses concepts from the deep level of the concept map, which will attempt to teach the same information in a different way. The new method goes step-by-step from CO to CBV (central blood volume), CBV to CVP (central venous pressure), and CVP to RAP. Finally, the tutor obtains a correct answer from the student and tutoring ends.

tu:	How does CO affect RAP?
st:	An increase in CO causes an increase in CVP.
tu:	Increased CO decreases CBV.
	If CBV decreases what happens to CVP?
st:	It decreases.
tu:	So what would happen to RAP?
st:	It decreases.
tu:	Right.

[after K11:65-71]

Figure 4.6. Response to an "Incorrect" Student Answer

Figure 4.7 shows an example of the tutor responding to a near miss. The student mentions CVP, which is not a core variable. (Later, sometime after that tutoring session, CVP became a core variable.) The tutor responds by attempting to lead the student from CVP, which is a step on the right path to the desired answer CO, by using determinants from the deep concept map.

tu:	What parameter determines RAP?	
st:	CVP.	
tu:	What determines CVP?	
st:	Blood volume [CBV].	
tu:	What determines CBV?	
st:	CO.	
tu:	How would RAP change?	
	. . .	[after K25:48-54]

Figure 4.7. Response to a “Near miss” Student Answer

Figure 4.8 shows an example of a response to a “don’t know” answer. In the near miss example of Figure 4.7, the student gave the tutor something to build on, but this is not the case with the “don’t know” answer in Figure 4.8. Therefore the tutor must teach the knowledge from scratch, as in Figure 4.8. So the tutor gives some hint about RAP and CVP. Also, the tutor informs the student about determinants of CVP from the deep level of the concept map. With that information the student gives the right answer for CO. Finally, with that right answer the tutor can correct the value of RAP.

In all three examples the student’s answer triggers a change in the tutor’s plan, thus personalizing the response in accordance with the student’s knowledge.

tu: Do you know what determines the value of RAP?
st: I don’t know.
tu: RAP is essentially the same as CVP.
CVP is determined by the compliance of the system and the volume of blood in the central venous compartment.
What determines that volume?
st: CO.
tu: So what would happen to RAP?
st: It decreases.
[after K26:64-73]

Figure 4.8. Response to a “Don’t know” Student Answer

4.6 The Marked-up Examples

Figure 4.9, shows the whole picture of a markup for the correction of variable CC in the DR stage of the pacemaker procedure according to our goal hierarchy rules and levels. The ***t-tutors-stage*** goal is expanded using the ***t-introduces-stage, t-corrects-variable,*** and ***t-concludes-stage*** subgoals. First of all, the tutor elicits the mechanism with the method ***t-does-neural-DLR*** and the topic ***t-tutors-mechanism***. After getting a correct answer from the student, the tutor changes his topic to ***t-tutors-DR-info***. Within this topic the tutor explains about the DR period. Then the tutor changes the topic to ***t-tutors-value,*** usually the final topic, to obtain the final correct answer for the value of the tutored variable. The category of student answer is marked but it is not included inside of the tutor’s goal hierarchy. The tutor’s acknowledgment is treated the same.

Figure 4.10 shows the markup of a response to a “near-miss” student answer. It is a part of a whole markup for variable RAP. In this example, the tutor asked for the

```
<T-tutors-procedure  proc=proc-pacemaker>
  <T-tutors-stage  stage=DR>
    <T-introduces-stage>
      <T-informs>
        tu: Pretty good job!
      </T-informs>
    </T-introduces-variable>
    <T-corrects-variable  var=CC>
      <T-tutors-variable>
        <T-does-neural-DLR>
          <T-tutors-mechanism>
            <T-elicits>
              tu: What input to the heart causes contractility to change?
              <S-ans  catg=correct>
              st: Sympathetic stimulation.
              </S-ans>
              <T-ack  type=positive>
              tu: Right.
              </T-ack>
            </T-elicits>
          </T-tutors-mechanism>
          <T-tutors-DR-info>
            <T-informs  type=explain-DR>
              tu: The DR occurs during the period of time before any reflex response to
                             the perturbation of the system takes place.
            </T-informs>
          </T-tutors-DR-info>
          <T-tutors-value>
            <T-elicits  DM="so"  Stage="during the DR period">
              tu: So, predict what change will occur to CC during the DR period.
              <S-ans  catg=correct>
              st: None.
              </S-ans>
            </T-elicits>
          </T-tutors-value>
        </T-does-neural-DLR>
      </T-tutors-variable>
    </T-corrects-variable>
    <T-concludes-stage>
      <T-informs>
        tu: All of your other DR predictions were correct, so please read page 6 so we can
        go on.
      </T-informs>
    </T-concludes-stage>
  </T-tutors-stage>
</T-tutors-procedure>                                        [K16:33~47]
```

Figure 4.9. Mark-up of Correction of Variable CC

```
<T-tutors-via-determinants  var=RAP>
  <T-tutors-determinant>
    <T-elicits>
      tu:What parameter determines RAP?*
      <S-ans  catg=near-miss>
      st:CVP.
      </S-ans>
    </T-elicits>

    <T-moves-toward-PT  method-type=inner>
      <T-tutors-determinant  var=CVP>
        <T-elicits>
              tu:What determines CVP?
              <S-ans  catg=near-miss>
              st:Blood volume [CBV].
              </S-ans>
        </T-elicits>
      </T-tutors-determinant>
    </T-moves-toward-PT>

    <T-moves-toward-PT>
      <T-tutors-determinant  var=CBV>
        <T-elicits>
              tu:What determines CBV?
              <S-ans  catg=correct>
              st:CO.
              </S-ans>
        </T-elicits>
      </T-tutors-determinant>
    </T-moves-toward-PT>
  </T-tutors-determinant>

  <T-tutors-value>
    <T-elicits>
      tu: How would RAP change?
      <S-ans  catg=correct>
      st: Decrease.
      </S-ans>
      <T-ack  type=positive>
      tu: Correct.
      </T-ack>
    </T-elicits>
  </T-tutors-value>
</T-tutors-via-determinant>
                                                  [after K25:48~60]
```

Figure 4.10. Mark-up of Response to a “near miss” Student Answer

determinant of RAP. So it is marked up with the method of ***t-tutors-via-determinant***, the topic is ***t-tutors-determinant***, and the primitive is ***t-elicits***. Due to the near-miss answer (CVP), tutoring is expanded with the deep-concept method. After that, the ***t-moves-toward-PT*** inner-method is used to get the tutoring back to a variable that is in the prediction table. After one more near-miss answer, the ***t-moves-toward-PT*** inner-method is used again. After the student gives the right answer for CO, the inner-method is ended. The tutoring of value at the end is included at the original method level.

CHAPTER V

NOVICE VS. EXPERT TUTORS

In this chapter I count and analyze differences in behavior between expert and inexpert tutors. The goal of finding these differences is to highlight the better tutoring behaviors and language styles. I believe we can apply the better styles to the machine tutor in CST v.3.

So far in this thesis I have analyzed keyboard-to-keyboard transcripts in which the tutors are Joel Michael and Allen Rovick, professors of physiology. These are the "expert" tutors. The Circsim-Tutor project has also collected thirty-one one-hour and two-hour long transcripts where medical students are tutored by second year medical students. We call them "novice" tutors. The problems in the sessions tutored by the novices were the same as those used by the experts and the student populations were similar.

Some of the results obtained by comparing the experts and novice tutors have been reported before. Glass [1999] tabulated whether the tutor or the student was the first to give the correct value for an incorrectly predicted variable. His results showed that the expert tutors persuaded the student to give the answer more often than the novices did. Also in [Glass et al., 1999], we showed that expert tutors engage in a more active tutoring style.

Michael and Rovick are not only experienced tutors. They have also been shown to be more effective tutors, experts, in a comparison of pre- and post-test results. There were four novice tutors in the sample I studied. They were all second-year medical students who were not professional tutors and had not tutored the baroreceptor reflex before, but had taken the same physiology class the previous year. Previous experiments showed that using totally untrained tutors was unsatisfactory, so the professors trained two of the novices in the problem domain, and two of them in tutoring techniques. The domain-trained group learned the baroreceptor reflex largely from written materials. When they worked problems, they submitted written answers and obtained written feedback, so they could not learn any tutoring tactics. The other group studied published papers on tutoring to learn salient features, then practiced tutoring and being tutored. They stayed away from the baroreceptor reflex, instead learning how to teach another unrelated physiological negative feedback mechanism. We labeled the four novice tutors with the pseudonyms “White,” “Green,” “Brown,” and “Gray.”

The fourteen usable transcripts from sessions in which these four novices served as tutors are numbered N17–N27 and N29–N31. The expert transcripts were drawn from K1–K51. For some of the comparisons in this chapter it was necessary to control the comparison for the same issues in the same contexts. Therefore I sometimes selected transcripts or segments of transcripts that were alike in three common features:

- procedure the student was solving (pacemaker malfunction, hemorrhage)
- stage within that procedure (DR, RR, SS)
- physiological variable being tutored (SV, TPR, etc.)

Martha Evens and Jaime Viehweg, a computer science student, aided me in counting the dialogue acts. The variables being controlled for in each experiment are given in the discussion of that experiment.

For the following experiments I first marked up the transcripts in SGML as shown in Figure 5.1. Novice tutors did not seem to show the same goal hierarchies as the experts did, their behavior seemed less organized. So in marking-up I focused on only primitive dialogue acts: **elicit** and **inform**. In the discussion that follows I sometimes refer to questions, by which I mean speech acts eliciting information from the student, whether or not they take the syntactic form of a question. In addition to the dialogue acts previously described in the expert transcripts, I added a new one **t-asks-confirmation**. Tutor questions such as “do you understand,” “right?,” and “do you have any questions?” are all classified into the **t-asks-confirmation** dialogue act category.

```
<T-elicits info=det>
Ok, can you tell me what determines the CVP?.
</T-elicits>
<S-ans>
Contractility of veins
</S-ans>
<T-informs info=compliance-info>
The veins are vessels of compliance, so they really have no
        contractility.
</T-informs>
.........
<T-asks-confirmation>
Does this make sense?
</T-asks-confirmation>
                                                  [N25:48~52]
.....
```

Figure 5.1. Excerpt of a Marked-up Novice Tutoring Transcript

5.1 Experiment 1: Elicit vs. Inform

The most distinct style difference between the expert and novice tutors is the frequency of the primitive dialogue acts **inform** and **elicit**. I counted elicit and inform acts in ten transcripts of sessions tutored by Joel Michael and the fourteen novice transcripts described above. These transcripts all involve the same problem of a pacemaker malfunction. I took only the DR stage here for the comparison because this is the best understood and most comprehensively marked-up part of the corpus. Also, not all transcripts included the later stages, RR and SS. The counts of inform and elicit are shown in Tables 5.1 through 5.3. Compared to the expert, the novices inform relatively more often and elicit relatively less often. Since novice tutor Gray informed distinctly more than the other three novice tutors, I show counts both with and without Gray.

Table 5.1. Elicit vs. Inform Choices of Dr. Michael

transcript	elicit	inform
K13	9	10
K14	11	12
K16	6	6
K22	7	2
K25	9	15
K26	13	10
K27	10	9
K48	18	13
K49	16	12
K51	9	6
Total	**108**	**95**

The counts of inform vs. elicit dialogue acts for the novices and the expert can be summarized in 2×2 contingency tables. Including novice tutor Gray (Table 5.4) $\chi^2 = 16.8$ with one degree of freedom, meaning the hypothesis that novices and experts are the same is rejected at the $p < 0.001$ level. Even excluding the inform-happy Gray (Table 5.5) $\chi^2 = 9.36$, meaning the same hypothesis is rejected at the $p < 0.01$ level.

Table 5.2. Elicit vs. Inform Choices of Four Novices (with Gray)

transcript	elicit	inform
N17	26	28
N18	17	13
N19	2	9
N20	8	21
N21	2	7
N22	16	19
N23	13	24
N24	7	24
N25	8	11
N26	1	9
N27	21	19
N29	12	26
N30	5	33
N31	23	41
Total	**161**	**284**

Table 5.3. Elicit vs. Inform Choices of Three Novices (without Gray)

transcript	elicit	inform
N17	26	28
N18	17	13
N20	8	21
N22	16	19
N23	13	24
N24	7	24
N25	8	11
N27	21	19
N29	12	26
N31	23	41
Total	**151**	**226**

Table 5.4. Elicit vs. Inform Choices (with Gray)

	expert	novice
elicit	108 (53%)	161 (36%)
inform	95 (47%)	284 (64%)

Table 5.5. Elicit vs. Inform Choices (without Gray)

	expert	novice
elicit	108 (53%)	151 (40%)
inform	95 (47%)	226 (60%)

These results show that the novice tutors and the expert tutor definitely differ in using elicit vs. inform dialogue acts. The expert tutors are asking questions relatively more often than the novices are 53% vs. 40% of the time and telling the students much less 47% vs. 60% of the time.

This result seems to agree with the observations of Graesser et al. [1995] in their detailed study of transcripts of inexperienced tutors. As I pointed out in Chapter 3, in that study Graesser et al. listed "active student learning" as one of eight salient components that have been identified as contributing to effective tutoring. Their inexperienced tutors did not much encourage active student learning. They also note that their inexperienced tutors largely ignored various sophisticated tutoring strategies, including ignoring the Socratic style which intensively elicits information from the student. Thus expert tutors might well be expected to elicit more and inform less than the inexpert ones, as we observe.

Lepper et al. [1993] focused on the affective and motivational aspects of the tutoring task, e.g. is the student attentive, how much control does the student have, is the student curious. In short, when the tutor diagnoses a cognitive problem with the student, the tutor's response should address these considerations in addition to the cognitive one. Their experienced tutors to almost never directly say what an error was or explain how to correct it-tactics that directly address the cognitive problem. Instead the tutors might ask a leading question, request an explanation, or indirectly prompt reconsideration. In fact, they note, addressing the cognitive issues and the affective/motivational issues are often at odds. Seen in this light, the novice tutors are paying relatively more attention to the students' knowledge defects by informing the student relatively more often, while the experts are addressing the students attention and motivation by eliciting relatively more often.

5.2 Experiment 2: Language Style Issues

The novice tutors' transcripts showed that they frequently checked confirmation of the student's understanding, with questions such as "Do you understand?," "OK?," "Is this clear for you?," "...right?," and "Does that make sense?" I marked them all as **t-asks-confirmation** dialogue acts. In Tables 5.6 and 5.7 which follow they are tabulated under the heading DYU (meaning "do you understand".) Also, we could frequently see a style of question represented by examples such as "Do you have any questions?" and "Is there anything you want to discuss?" These are tabulated under the rubric DYQ ("do you have questions"). We can see that novice tutors used DYU questions during all stages of the problem. They averaged 2.0 DYUs per stage tutored for the 32 stages within the 14 sessions tabulated, and 0.7 DYQs per stage. The experts, by contrast, averaged a considerably fewer 0.3 DYUs per stage and 0.3 DYQs per stage.

One reason for showing so much detail in Tables 5.6 and 5.7 is to illustrate that the experts prefer to use DYQ at the ends of procedures, after teaching the SS stage. However, the novices use DYQ from the beginning of tutoring.

According to Graesser et al. [1995] and Graesser [1993] questions of this variety, which are called comprehension-gauging questions, do nothing to enhance student learning, and anyhow do not yield reliable answers. In fact they observed a positive correlation between higher student achievement and “no” answers to “do you understand” questions. We might therefore expect that experienced tutors use them less often, as we indeed observe. The fact that the experts in fact ask “do you have any questions” with some frequency at the end of teaching a procedure might mean that this question is motivated by considerations other than gauging comprehension. In the terms of Lepper et al., it could be enhancing the student’s level of control.

Another issue of language style involves the use of the phrase “primary variable.” In the beginning of collecting predictions in the DR stage of the problem, the tutors want to the student to start with the first prediction-table variable which is affected. This is called the primary variable. Knowing the primary variable is important for understanding the causal effects of the initial perturbation but knowing the term “primary variable” does not enhance the student’s knowledge of physiology. However we noticed that the expert tutors rarely used the term “primary variable,” preferring instead to explain the concept when needed, whereas most novice tutors use the term. Therefore, as a measure of the difference in style, I counted the usage of “primary variable.” As we can see in Table 5.6, novice tutors use the phrase in thirteen transcripts out of fourteen. The experts, in Table

Table 5.6. Counts of Language Phenomena per Transcript for Novices

transcript	DYU			DYQ			primary	emphatic
	DR	RR	SS	DR	RR	SS		
N17	6	0	N/A	0	0	N/A	Yes	6
N18	3	1	N/A	0	0	N/A	Yes	6
N19	2	1	1	0	1	1	Yes	5
N20	4	1	N/A	2	2	N/A	Yes	6
N21	0	0	0	0	0	0	Yes	4
N22	7	0	N/A	3	2	N/A	Yes	6
N23	6	0	N/A	0	0	N/A	Yes	5
N24	3	N/A	N/A	3	N/A	N/A	No	9
N25	5	2	N/A	0	0	N/A	Yes	5
N26	1	0	0	0	0	1	Yes	3
N27	2	3	0	1	0	0	Yes	5
N29	5	1	N/A	0	0	N/A	Yes	4
N30	1	1	1	1	0	1	Yes	3
N31	4	2	N/A	4	0	N/A	Yes	9
total	49	12	2	14	5	4		76
average	49/14	12/13	2/5	14/14	5/13	4/6		76/14

5.7, transcripts uttered "primary variable" in only one transcript out of twenty-two. Here is a pair of contrasting examples:

Expert tutor:	Let's start by telling what parameter you want to predict first.	[K26:18]
Novice tutor:	Can you please tell me the primary variable that will change in this problem?	[N18:22]

Another interesting countable phenomenon is instances of emphatic positive acknowledgment. The ordinary, non-emphatic positive acknowledgments in the transcripts are "OK," "Yes," "Right," or "Correct." Frequently we could see more stressed positive acknowledgments such as "Excellent," "Good," "Great," "Exactly," "Perfect," "Very good," "Precisely," "Absolutely," "Super," or "You did a pretty good job!" We tabulated these under the rubric emphatic positive acknowledgments. When the tutor said several such acknowledgments sequentially within a turn we counted them as one. I count these positive acknowledgments through the whole transcripts, included in Tables 5.6 and 5.7. The counting shows that the novices use seventy-six cases in fourteen transcripts, averaging 5.4 cases/transcript. The expert tutors use eighty-three cases in twenty-two transcripts, averaging 3.8 cases/transcript. We can see the novice tutors used more stressed acknowledgments.

<u>5.3 Experiment 3: Usage of Concept Noun Phrases</u>

In this experiment my goal was to discover how many concepts the tutor utilizes during teaching. As a concrete measure of the number of concepts, I counted the number of noun phrases that refer to physiological and anatomical notions in the domain. In the following example the underlined words are concept noun phrases:

Table 5.7. Counts of Language Phenomena per Transcript for Experts

transcript	DYU			DYQ			primary	emphatic
	DR	RR	SS	DR	RR	SS		
K10	4	N/A	N/A	0	N/A	N/A	No	5
K11	2	0	N/A	0	0	N/A	No	2
K12	1	0	N/A	0	0	N/A	No	6
K13	0	0	N/A	2	0	N/A	No	2
K14	0	0	N/A	0	0	N/A	No	4
K15	0	0	0	0	0	0	No	4
K16	0	0	0	0	0	0	No	3
K17	0	0	1	0	0	1	No	4
K18	0	0	0	0	0	1	No	5
K19	0	0	0	0	0	1	No	6
K20	1	2	0	0	0	1	No	5
K21	0	0	0	0	0	1	No	5
K22	0	0	0	0	0	1	No	2
K23	0	0	0	0	0	0	No	5
K24	0	0	0	0	0	0	No	4
K25	0	0	0	0	0	1	No	1
K26	1	0	0	0	0	1	No	1
K27	0	0	2	0	0	1	No	2
K47	1	2	2	0	0	0	No	6
K48	0	0	0	0	0	0	No	8
K49	0	0	0	0	0	0	No	3
K51	0	0	0	0	0	0	Yes	1
total	10	4	5	2	0	9		83
average	10/22	4/21	5/17	2/22	0	9/17		83/22

tu: CO has an effect on another cv variable. Do you know which one this is?
st: SV
tu: No, SV is one of the determinants of CO, along with HR. But CO does not directly affect SV. CO is inversely related to CVP. This is makes sense because an increase in CO means that blood is going to be removed from the central veins that drain into the heart at an accelerated rate. If an increase in CO causes the central venous volume to decrease then the CVP will decrease as well.

[N19:36~38]

In this excerpt the parameter CO was introduced by the tutor, so it was counted as a concept noun. The second parameter SV was broached by the student and the tutor responded with an elaboration on SV, so it, too, was counted as a concept noun. Had the tutor ignored it, we would not have counted it. Also we can easily see concept noun phrases such as “central venous volume” and “central veins,” which are physiological and anatomical terms. Notice that a concept is counted only once, even if it is invoked several times.

Although this measure may not capture all concepts, it does allow relative comparisons. I compared novices versus experts in two ways. One comparison was to count the concept noun phrases for tutoring an entire DR stage. Second, by counting the segment of text for the correction of one variable at a time, I can compute concepts invoked for individual variables corrected. Since teaching different variables involves appealing to different concepts in different ways, I did this experiment for two variables separately.

My judgment from reading transcripts is that the behavior of SV and CVP in the DR stage is difficult for the student to understand. These variables need more and longer instances of tutoring. So in this experiment they became the target variables for counting number of concepts used in the teaching of individual variables. In some of our older transcripts Right Arterial Pressure was used instead of Central Venous Pressure (CVP); the physiology and the tutorial arguments are the same.

As we can see in Table 5.8, I counted in fourteen novice tutor transcripts and ten expert ones. The overall result shows the novice tutors used more concept noun phrases, in both the total number of concepts used and in the number of concepts used normalized to the number of prediction errors that needed remediation.

Table 5.8. Counts of Concept Noun Phrases in DR

N-transcript	Prediction errors	Concept NPs	K-transcript	Prediction errors	Concept NPs
N17	4	28	K13	2	12
N18	3	17	K14	2	9
N19	2	12	K16	1	7
N20	2	15	K22	1	5
N21	2	13	K25	4	12
N22	3	20	K26	2	14
N23	2	22	K27	2	9
N24	1	19	K48	3	14
N25	2	13	K49	1	26
N26	1	13	K51	3	11
N27	3	13	Total	21	119
N29	2	23			
N30	4	19			
N31	2	31			
Total	33	258			
Prediction errs per DR stage		33/14 = 2.4			21/10 = 2.1
Concepts used per DR stage		258/14 = 18.4			119/10 = 11.9
Concepts used per prediction err		258/33 = 7.8			119/21 = 5.7

Since correcting all the prediction errors is the organizing principle of the tutorial dialogues, normalizing the number of concepts to the number of errors adjusts for the case where the tutor has more topics to cover.

Table 5.9 and 5.10 show counts of concept noun phrases used during the teaching of SV and CVP respectively. The novices used twice as many concept noun phrases as the experts to teach the variable SV. This big difference came from the teaching pattern. The expert tutors taught the variable SV by the method of “move-forward” from the determinants, appealing to the causal relationships. This way is simple and easy to understand. Here is a simple example, the underlined words are concept noun phrases:

tu: If CVP decreases what will happen to SV?
st: Decreases.
tu: Right. [K51:54~56]

In contrast, the novice tutors started with a question for diagnosing the student's misconceptions and then explained many details, as following example illustrates:

tu: Next, you predicted that there would be a decreased SV. what is the process behind that?
st: If the heart rate increases, the diastolic time.........
tu: That is good logic, However, what are the parameters which affect stroke volume?
st: Volume of blood in ventricle, amount of Calcium available intracellularly
tu: Good. By volume of blood in the ventricle, we are referring to preoload volume. The amount of Ca available is determined by the stimulation of beta receptors on the cell surface, which in turn increase cAMP through G proteins, and then intracellular mechanisms allow an influx of Ca, either from EC or from internal stores. This Ca determines the inotropic state (IS). There is one other variable which affects the stroke volume, and that is the afterload, which is the volume on the other side of the heart, which impedes the SV, represented here as MAP. What I am getting at is that a change in HR does NOT directly affect the stroke volume, because the change in HR is not directly linked to SV. [N31:54~60]

However, the result of counting concept noun phrases for teaching CVP was different than with SV. The expert and novice seem to be using the same physiological and anatomical arguments, So although their tutoring style was different in that the experts are eliciting more and informing less, the number of concept noun phrases is similar, as is shown in Table 5.10.

Table 5.9. Counts of Concept Noun Phrases for SV in DR

N-transcript	Concept Noun Phrases	K-transcript	Concept Noun Phrases
N24	9	K12	1
N25	8	K14	5
N26	9	K20	7
N29	13	K26	5
N31	18	K27	8
total	57	total	26
average	57/5 = 11.4	average	26/5 = 5.2

Table 5.10. Counts of Concept Noun Phrases for CVP in DR

N-transcript	Concept Noun Phrases	K-transcript	Concept Noun Phrases
N17	5	K11	6
N18	8	K14	7
N19	8	K22	5
N20	6	K25	7
N25	6	K26	6
N30	6		
total	39	total	31
average	39/6 = 6.5	average	31/5 = 6.2

5.4 Experiment 4: Style of Question

As we have seen in experiment 1 the experts use more ‘elicit’ primitive dialogue acts than the novices do. I analyzed the content of the elicit acts, placing them into four categories:

- Curriculum oriented - Deep Question: E (Curri, Deep)
- Curriculum oriented - Simple Question: E (Curri, Simple)
- Student initiative oriented - Deep Question: E (Stu, Deep)
- Student initiative oriented - Simple Question: E (Stu, Simple)

Here, “curriculum oriented” questions came not from student misconceptions that were revealed in the immediate dialogue, but from the tutor’s plans. The question uses domain knowledge or serves a tutoring tactic, including asking for a determinant, asking about the relationship between two variables, asking for the control mechanism, etc. However, some tutor questions are oriented toward addressing the student's questions or incorrect answers to a previous question. These tutor questions are classified as "student-initiative oriented." Some questions are designed to elicit a simple answer such as a variable name, a direction of change, or simply saying "yes" without deep explanation. These are categorized in the category of "simple questions." The opposite style needs some explanation or reasoning for answering, for example a "why" question, so I called them "deep questions." To illustrate, here is an expert tutor:

```
tu: What input to the heart determines contractility?     <E (Curri, Simple)>
st: Sympathetic
tu: Right, so, why didn't contractility change in DR?     <E (Curri, Deep)>
st: Because the change in heart rate did not ...          [K26:54-58]
```

In this example the tutor asked the mechanism of control of contractility according to his tutoring tactic. This is a curriculum oriented simple question. Then, following the student's correct answer, the tutor asked about the direct response stage concept using a "why" question. It is a curriculum oriented deep question.

Now let us look at a novice tutor:

```
tu: What will change first?                               <E (Curri, Simple)>
st: SV
tu: Please explain why the SV will change first.          <E (Stu, Deep)>
st: If you increase the heart rate, you will have a decreased diastolic time, thus
    you will have less blood entering the heart and a decreased SV.
tu: OK, that is good, but then what is the first variable to change in the way you
    have described it?                                    <E (Stu, Simple)>
                                                                  [N31:22-26]
```

In the example above, the tutor asked about the primary variable. The student did not answer correctly, so the tutor asked the reasoning for the answer. So I marked this as a student initiative oriented deep question. After the student's reasoning the tutor asked about the primary variable again, working from the student's reasoning. So it is categorized as a student initiative oriented simple question.

I counted 101 questions from the expert tutors and 151 questions from the novices and then categorized them into the four groups. Tables 5.11 and 5.12 show the data.

With the data I made a 2×4 contingency table, Table 5.13, to find whether the difference in question style is significant. The value of χ^2 is 12.2 with three degrees of freedom. It means the experts and novices are different at a significance level between 0.01 and 0.001. So we found again their styles are quite different.

Table 5.11. Question Styles of Expert Tutors

transcript	curri/ini	deep	simple	total
K13	Curri	1	6	7
	Stu-ini	1	1	2
K14	Curri	0	10	10
	Stu-ini	1	0	1
K16	Curri	0	4	4
	Stu-ini	1	1	2
K25	Curri	2	4	6
	Stu-ini	0	3	3
K26	Curri	6	7	13
	Stu-ini	0	0	0
K27	Curri	1	4	5
	Stu-ini	2	3	5
K48	Curri	2	10	12
	Stu-ini	5	1	6
K49	Curri	3	9	12
	Stu-ini	3	1	4
K51	Curri	3	5	8
	Stu-ini	0	1	1
total		31	70	101

Table 5.12. Question Styles of Novice Tutors

transcript	curri/ini	deep	simple	total
N17	Curri	3	14	17
	Stu-ini	4	5	9
N18	Curri7	0	11	11
	Stu-ini	2	4	6
N20	Curri	1	4	5
	Stu-ini	1	2	3
N22	Curri	1	7	8
	Stu-ini	4	4	8
N23	Curri	1	3	4
	Stu-ini	1	8	9
N24	Curri	5	1	6
	Stu-ini	1	0	1
N25	Curri	0	4	4
	Stu-ini	0	4	4
N27	Curri	1	11	12
	Stu-ini	5	4	9
N29	Curri	3	5	8
	Stu-ini	1	3	4
N31	Curri	1	11	12
	Stu-ini	3	8	11
total		38	113	151

Table 5.13. Expert vs. Novice Question Styles

	expert	novice
deep curri	18	16
deep stu-ini	13	22
simple curr	59	71
simple stu-ini	11	42

Tables 5.14 and 5.15 summarize the data in 5.11 and 5.12. Comparing expert to novice tutors, we can see:

1. Experts ask curriculum oriented questions 77/100 = 76% of the time, compared to 87/151 = 58% for novices.
2. Experts ask deep questions 31/101 = 31% of the time, compared to 38/151 = 25% of the time for novices.

This calculation shows that the expert strongly preferred curriculum oriented questions, and somewhat preferred deep questions, compared to novices. It means the experts keep to the tutoring plan and make students think more deeply. Compared to the experts, the novice tutoring is more often guided by the student answers. This provides some objective support for my observation, reported in [Glass et al., 1999], that it was difficult to annotate the novice tutoring transcripts with the goals of the expert tutors. The novices, it seems, do not stick to their tutoring plans and are wordier than the experts.

Table 5.14. Question Styles of Experts

	deep	simple	total
curri	18	59	77
stu-ini	13	11	24
total	31	70	101

Table 5.15. Question Styles of Novices

	deep	simple	total
curri	16	71	87
stu-ini	22	42	64
total	38	113	151

5.5 Conclusion of Comparisons

On the basis of these experiments I conclude that our experts use a more effective tutoring style than the novices. The experts used a more questioning style to promote the student's active learning and to give more challenge to the students. The novices more often sought confirmation of the student's knowledge by asking "Do you understand?," "Does that make sense?" or "Do you have any question?." However, other authors observe that these questions do not enhance student's learning and do not yield useful diagnostic information, so we are not surprised that our experts engage in them less frequently. And the novices used more concept noun phrases in doing their tutoring. This action made the tutoring session more complicated and wordy. The last comparison of the style of question showed us that the experts preferred curriculum oriented and deep questions. Through more sophisticated questions the tutor could discover the student's misconceptions and could enhance their active learning.

CHAPTER VI

SELECTING CUE WORDS

The addition of cue words such as discourse markers and acknowledgments in tutorial language can make the difference between stilted and natural sounding dialogue. In this chapter I describe some simple rules for the selection of discourse markers and for the acknowledgment of the correctness of a student's answer. These rules were derived for use in the turn planner of CST by applying the machine learning algorithm C4.5.

Di Eugenio et al. [1997] used the same technique to derive rules and decision trees for the inclusion or omission of cue words. Moser and Moore [1995], studying the same dialogue earlier, discovered rules for the choice between "since" and "because," also rules for discourse marker occurrence based on the embeddedness of the discourse segments. They divided the instructional dialogue into discourse segments, then coded various relationships between them according to Relational Discourse Analysis. Then they derived rules for discourse marker usage based on the relationships of the segments being marked. These results show that segment structure is an important factor for determining the occurrence, placement, and selection of cue words.

Nakano and Kato [1999], in their analysis of Japanese instructional dialogue, also used a machine learning algorithm to derive rules for discourse markers. They divided their text into segments and coded instructional goals for each segment. Coding instructional goals for each segment was in contrast to the earlier studies, which coded discourse relationships. They showed that they could do a better job of selecting discourse markers by using the instructional goal structure. For my purposes this is a nice result, because CIRCSIM-Tutor transcripts are coded according to tutoring goals, not discourse relations.

Figure 6.1 shows how a turn can be produced in CST v.3 using the discourse planner, turn planner, and sentence generator. The discourse planner decides on goal structures. Then a buffer holds the primitive discourse goals until the primitive “elicits” is received. With these goals the turn planner builds feature structures. The feature structures have all information needed for generation. The choice of discourse markers is decided here in the turn planner. The generator generates the sentences for a turn from the information in the feature structures.

In the CST dialogue, topic boundaries and turn boundaries do not coincide. Some topics are spread among three turns, or one turn can encompass parts of three topics (Figure 6.2). However, within a turn, topic boundaries frequently coincide with a discourse marker. These discourse markers are important for coherent and naturalistic dialogues in CST. Grosz and Sidner [1986] said that discourse markers flag changes in both attentional and intentional state. Also, Mann and Thompson [1988] identified how discourse markers mark rhetorical relations between segments in their Rhetorical Structure Theory.

Another interesting phenomenon is the acknowledgment to the student of the correctness of the student’s answer. Our human tutors do not always emit such acknowledgments and there is some variability in their form. This chapter describes some experiments that resulted in simple rules as to when and how to emit them.

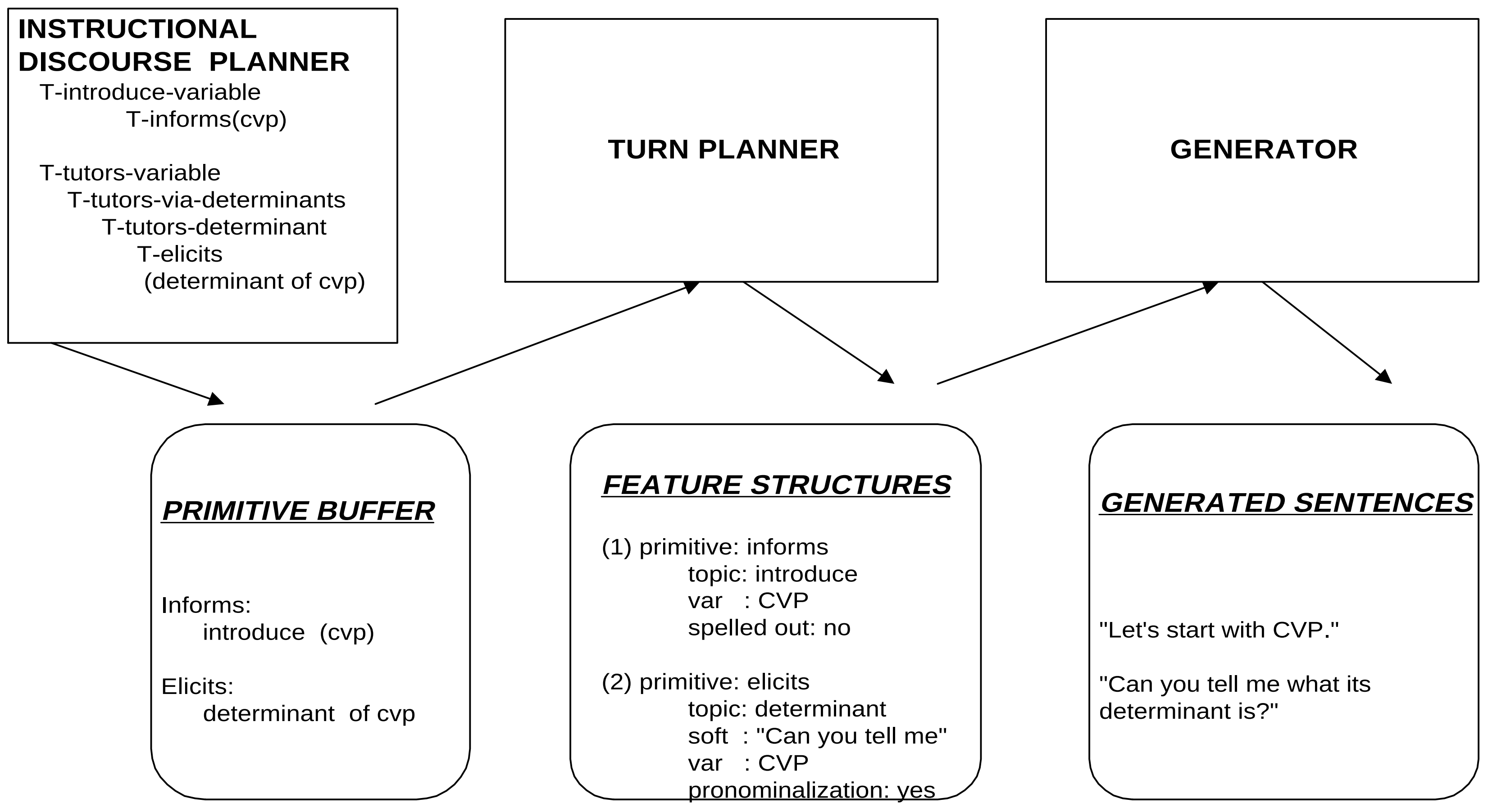

Figure 6.1. Planning One Turn in CST v.3

	Turn	Goal from Tutoring Schema
1.Tu:	Now let's look at your prediction for TPR.	Introduce variable (introduce topic)
	Can you tell me how it is controlled?	Elicit topic 1 (first topic)
2. St:	Parasympathetics	
3. Tu:	Correct! TPR is neurally controlled.	
	And the reflex hasn't started to operate yet	Inform topic 2 (middle topic)
	So what is the value of TPR?	Elicit topic 3 (final topic)
4. St:	Unchanged	
5. Tu:	Great! What other variables are neurally controlled?	Introduce next variable (introduce topic)

Figure 6.2. Turns and Goals

6.1 Choosing Discourse Markers

6.1.1 Experiment 1. This experiment explored how discourse markers are chosen in the tutoring dialogue. In this experiment I considered the discourse markers 'and', 'so', and 'now', which are the most frequently used markers in our transcripts. We looked especially at the question of how they relate to the student answer categories of 'correct' or 'near-miss', discussed in Chapter 4. The following example showing a discourse marker after an acknowledgment of a correct answer:

tu: Can you tell me how TPR is controlled?
st: Autonomic nervous system
tu: Yes, <u>and</u> the predictions that you are making are for the period before any neural changes take place.

[K10-tu-29 ~ 31]

Here as a response to the student's correct answer, the tutor gives an acknowledgment, emits a discourse marker, and provides some information corresponding to the primitive 'inform' all in one turn. We can see the use of the discourse marker 'and' here.

I used a sentence skeleton model to find rules:

1. tu: Question
2. st: Answer {correct, near-miss, or N/A}
3. tu: {Optional Discourse Marker}, Primitive dialogue act {elicit or inform}

I assumed the following four attributes affect the usage of discourse markers:

- category of student's answer

  (correct, near-miss or N/A if the proceeding sentence was the tutor's)
- discourse marker (and, now, so)
- type of primitive dialogue act following the discourse marker (inform or elicit).
- topic position of the sentence following the discourse marker

  (introduce, first, middle, final)

As we discussed before, inside of one method level goal, one or more topics can be nested. I counted the position of the topic within the tutoring goal hierarchy, first, second, etc., because I assumed the dialogue goal structure is an important determinant of the usage of discourse marker. I separated the topic of **t-introduce-variable** as an 'introduce topic', instead of counting it as another sequential topic.

Here is an example of how I marked up the above excerpt:

tu: Can you tell me how TPR is controlled? **< first topic>**
st: Autonomic nervous system **<correct answer>**
tu: Yes, and the predictions that you are making are
for the period before any neural changes take place.
<inform> <DM: and>

With this mark-up I filled out the following feature slots for sixty passages in our discourse:

• category of student answer	=	correct
• discourse marker	=	and
• type of primitive	=	inform
• topic position	=	first

I applied the C4.5 machine learning algorithm to a set of sixty data passages like the above and got the following rules, extracted from a decision tree:

- If the target sentence is part of the <introduce> topic use the discourse marker '**now**'
- If the target sentence is part of the <middle> topic use the discourse marker '**and**'
- If the target sentence is part of the <final> topic use the discourse marker '**so**'
- If the target sentence is part of the <first> topic

  and if the primitive is <inform> use the discourse marker '**so**'

  else {primitive is elicit} use discourse marker '**and**'
- default class is '**so**'.

This rule misclassified 8 of the 60 cases, for an error rate of 13.3%. It shows the order of topic and the type of primitive dialogue act can influence the choice of discourse marker. However, the student answer category did not contribute to the result.

This result matches well with Schiffrin's [1987] theory. She mentioned that 'so' is realized semantically on a fact-based, knowledge-based, or action-based discourse plane. In our transcripts, most of the examples of 'so' are used in the final topic of **t-tutors-value**. The earlier topics such as **t-tutors-determinant** and **t-tutors-relationship** provide background information that warrant the result presented in the final topic. The other place where 'so' is used, in the first topic, is realized as the result of an earlier discussion of another issue. It is used especially often in the **t-present-anomaly** topic that is used to show the inconsistent appearance of reported facts. So, '**so**' is used in a fact-based plane as a 'result' 'caused' by the previous discussion as in the following example.

tu: {Tutoring value of HR, CO and SV}
...
tu: So, in DR HR is up, CO is up, but SV is down.
How is this possible?
st: That HR increases outstrip SV decreases in this case.
tu: Exactly!
[K25-tu-62 ~ 68]

Schiffrin also said that '**now**' is used to give attention to an upcoming idea unit. In our experiment, four of five cases of '**now**' are used in the **t-introduce-variable** topic.

Quirk et al. [1972] said that both '**and**' and '**now**' can be used as transitional discourse markers. Five of the eight cases of '**and**' are found also in the first topic in this experiment.

6.2 Analysis of Acknowledgment

Green and Carberry [1999] said that 13% of responses to ‘yes-no’ questions were indirect answers in spoken English as in the next example. So she insists that a robust dialogue system should have the ability to interpret indirect answers.

Q: Actually you will probably get a car won’t you as soon as you get there”
R: [No]
I can’t drive.
[Green and Carberry, 1999, p. 389]

Although her theory focuses on ‘yes-no’ questions, I applied her theory to our transcripts for acknowledgment generation. I analyzed ninety-one cases of question-answer dialogue acts and found out that 71% of the cases gave an appropriate acknowledgment and 29% cases omitted an acknowledgment. This means that 29% of the cases gave an answer with implicit acknowledgment to the student in the following ‘elicit’ or ‘inform’. So, our new version of CST should have the ability to make implicit acknowledgments as a kind of indirect answer.

Also, Green and Carberry suggested that the reasons for indirect answers are accuracy, efficiency, and politeness. In our transcripts I found some similar effects as shown in the next example:

tu: If CC is under neural control and we are taking about the period before any change in neural activity then CC?
st: But it is also under intrinsic... **<incorrect answer>**
tu: You are confusing Starling Law’s.... **<Give explanation of Starling’s Law>**
[K11-tu-57 ~ 59]

In the above example the tutor provides an extra explanation for an unexpected student answer to correct his/her misconception without a negative acknowledgment. Here we can see more politeness and efficiency compared to the case of replying with only a negative acknowledgment ‘No!’. This extra explanation can reduce student’s follow-up questions such as ‘Then what is a correct answer?’.

Spitkovsky and Evens [1993] showed a study of negative acknowledgment with our expert tutoring transcripts. They made ten categories of negative acknowledgment and ranked them by degree of severity. However, they pointed out the uncertainty of severity ranking. So in this chapter I focus only on whether the negative acknowledgment is expressed or omitted without subcategorizing.

Working with the same dialogues, Brandle [1998] presented a study of acknowledgments using joint actions. He proposed a model for the generation of acknowledgments according to an assessment of how good the student is together with the student’s statement. I wanted to give focus to both the tutor’s tutoring goal strategies and the student model for the generation of acknowledgment. So I tried to find connections between the acknowledgment, the student answer category, and the topic of the tutor’s follow-up sentence.

6.2.1 Experiment 2. I wanted to discover the relationship between an acknowledgment and the type of primitive dialogue act in the following sentence. The following Table 6.1 shows a 2x4 contingency table of acknowledgments vs. the following primitive act. The value of Π^2 is 18.85 with three degree of freedom and $p < 0.001$ significance. This result means that the usage of primitive acts is quite different according to the type of acknowledgment. Here, ‘positive acknowledgment’ means the tutor’s positive response to the student’s correct answer. The ‘not-positive acknowledgment’ means all kinds of responses to other categories of student answer, such as ‘clearly incorrect’, ‘near-miss’, ‘don’t-know’, and ‘partially correct’.

Table 6.1. Acknowledgment vs. Following Primitive Act

Acknowledgment	inform	elicit
positive acknowledgment	24	18
not-positive acknowledgment	22	1
omitted positive acknowledgment	1	6
omitted not-positive acknowledgment	13	6

From the above table, the four most common patterns are follows:

pattern1:	st:	{not correct answer}
	tu:	{not-positive ack}
		{inform}
pattern2:	st:	{correct answer}
	tu:	{omitted positive ack}
		{elicit}
pattern3:	st:	{not correct answer}
	tu:	{omitted not-positive ack}
		{inform}
pattern4:	st:	{correct answer}
	tu:	{positive ack}
		{inform}

Table 6.1 shows that our tutors give acknowledgments most of the time, sixty-five cases out of ninety-one cases. After a student’s not-correct answer the tutor provided some information (perhaps a hint) instead of eliciting directly, regardless of whether there

is an acknowledgment. After a correct student answer the tutor either omitted a positive acknowledgment and elicited the next question or gave a positive acknowledgment and provided some related information. These results match well with the study of Hume et al. [1996]. They analyzed our human transcripts and discussed two kinds of hints: CI hints and PT hints. The CI hint conveys information explicitly and PT hint points to information less directly in order to correct a student's misconceptions. The PT hints, they observed, provide an implied negative acknowledgment. So we can assume the tutor's question about the next topic without an acknowledgment of the preceding student answer functions as an implicit positive acknowledgment.

6.2.2 Experiment 3. After examining the results of experiment 2, I felt some curiosity about the relationship between a change of topic in a follow-up sentence and whether there was an explicit acknowledgment in a turn. I checked the content of the topic of the follow up sentence to see whether there is a transition to a new topic. I examined ninety-one cases with C4.5 to get some rules for change of topic. Three kinds of attributes, type of primitive act (elicit or informs), acknowledgment type, and topic transition (same or different topic) were used. The following is the result of decision tree.

- If acknowledgment type is not-positive the next primitive has the **same** topic as before
- If acknowledgment type is omitted positive the next topic is **different** than before
- If acknowledgment type is omitted not-positive next topic is the **same** as before
- If acknowledgment type is positive and primitive is elicit the next topic is **different**

  otherwise {if primitive is inform} the next topic is the **same** as before
- default class is **same**.

The result shows an error rate of 11%. The data show that when the student answer was not fully correct, then the tutor discussed the same topic 95% of the time with or without acknowledgment. In the case of a correct answer, without positive acknowledgment, the tutor changed to a new topic. And with the positive acknowledgment the tutor asks about a different issue or rephrases same issue. The following examples illustrate these cases:

tu: Do you know how RAP will change if something produces a change in CO?
st: If CO increases then RAP should also increases.
<incorrect answer>
tu: No, **<no-positive ack>**
when CO is the independent variable then RAP changes as the dependent variable in the opposite direction.
<inform-same topic>
[K14-tu-43 ~ 45]

In the example above, after the student's incorrect answer, the tutor gives a 'not-positive acknowledgment' and information about the relationship between the variables CO and RAP. This information is the same topic as the tutor's former turn.

tu: What would happen to CVP when CBV goes down?
st: Decreases. **<correct answer>**
tu: Yes, **<positive ack>**
And what is the relationship between CVP and arterial pressure?
<elicit-different topic>
[K10-tu-57 ~ 59]

The example above shows that the student gives the correct answer. The tutor responds with a positive acknowledgment 'Yes', and elicits the different topic of the relationship between CVP and arterial pressure. Note the discourse marker ‘and’, as predicted by experiment 1.

6.2.3 Experiment 4: During the analysis of transcripts the difference between an ordinary positive acknowledgment and an emphatic positive acknowledgment was noticed. Both of them are positive acknowledgments. In this experiment I further categorized ‘OK’, ‘right’, ‘yes’ and ‘correct’ as ordinary positive acknowledgments, while ‘good’, ‘certainly’, ‘absolutely’, ‘great’, ‘super’, ‘very good’, ‘right again’, etc. are classified as emphatic positive acknowledgments as I described in Chapter 5. I tabulated 88 cases of positive acknowledgments to find out when they were used. Frequently, I found an emphatic positive acknowledgment in the last topic of variable value in a session of teaching one variable. So I separated the cases into two groups: one is a response to the student’s correct answer in the middle of tutoring a variable, and the other is response to the student’s correct answer in the final topic.

The following Examples show a case with an ordinary positive acknowledgment and one with an emphatic positive acknowledgment.

tu: And what is the primary mechanism by which arteriolar
radius is controlled? <**middle topic**>
st: Sympathetics
tu: Yes. **<positive ack>**
[K12-tu-39 ~ 41]

tu: So what’s your prediction of CC in the DR? <**final topic**>
st: no change
tu: Good. **<emphatic positive ack>**
[K10-tu-53 ~ 55]

Table 6.2 is a 2x2 contingency table of these cases. It has value $\Pi^2 = 5.096$ with one degree of freedom and $p < 0.05$. It means that the tutor gives more emphatic positive acknowledgment to a correct answer in the final topic than in a middle topic.

Table 6.2. Positive Acknowledgment Type vs. Order of Topic

order of topic	ordinary positive-acknowledgment	emphatic positive-acknowledgment
correct answer in the middle tutoring	32	11
correct answer of final topic	23	22

6.3 Acknowledgment and Primitive Act Types with Discourse Markers

6.3.1 Experiment 5. We also found a relationship among the existence of an acknowledgment, the primitive act style, and the existence of a discourse marker. I counted one hundred thirty eight sample sentences of tutor turns that directly followed a student answer, for example:

tu1: What input to the heart causes contractility to change?
st1: Sympathetic stimulation
tu2: Right. Does sympathetic stimulation change during the DR phase?
[K16-tu-39 ~ 41]

In the above conversation I collected the turn T2 as a sample case. It has an acknowledgment and an elicit primitive act but no discourse marker. I divided the sample sentences into eight groups:

(1) {Ack, DM, elicit}

(2) {Ack, DM, inform}

(3) {Ack, No-DM, elicit}

(4) {Ack, No-DM, inform}

(5) {No-Ack, DM, elicit}

(6) {No-Ack, DM, inform}

(7) {No-Ack, No-DM, elicit}

(8) {No-Ack, No-DM, inform}

Using these eight categories I constructed Table 6.3. Table 6.3 shows the 2x4 contingency table of acknowledgment vs. discourse marker and primitive act. The value of Π^2 is 7.23 with three degrees of freedom and $p < 0.05$. Although this probability is not so significant, during this analysis I observed the following interesting point.

Table 6.3. Acknowledgment vs. Primitive Act with Discourse marker

Primitive Act with or without DM	Existed Acknowledgment	Omitted Acknowledgment
DM + elicits	17	14
DM + informs	21	10
No-DM + elicits	11	19
No-DM + informs	29	17

When the student gave an expected answer the tutor frequently gave a positive acknowledgment and a discourse marker as in example below.

tu: What parameter here reflects filling of the lv?
st: RAP
tu: Right.
So RAP and C determines SV.
[K27-tu-66~68]

However, in the case of an unexpected answer that meant not exactly what the tutor wanted to hear from the student including incorrect answers, the tutor gave some information or elicited some questions without an acknowledgment and discourse marker as in example below.

tu: And in what direction will it (TPR) change?
st: TPR will decreases.
tu: Remember, this drug is an agonist; that means that it acts just like the transmitter that is normally released.
What is the affect of activating the alpha adrenergic receptors on blood vessels?
[K36-tu-20~22]

To investigate this phenomenon I made the following Table 6.4.

Table 6.4. Answer Type vs. Discourse marker and Acknowledgment

	Expected answer	Unexpected answer
Acknowledgment + Discourse Marker	27	11
No-Acknowledgment + No-Discourse Marker	3	33

Analysis of Table 6.4 shows a distinctly different turn structure given a different type of student answer. The value of Π^2 is 30.17 with one degree of freedom and $p <$ 0.001 level in this table. So I concluded that the content of the student answer can influence the structure of the tutor's sentence.

6.4 Proposed Dialogue Model

Although all the above experiments used relatively small data sets, this style of approach is a big first step for making rules for both the discourse planner and turn planner because the results show some statistical significance and the data comes from our expert tutors whom our CST v.3 wants to mimic.

Applying the above rules I propose to generate the following tutorial dialogue.

(1) tu: Now let's think about TPR. **<introduce topic> <DM: Now>**

(2) By what mechanism will it change? **<first topic> <primitive: elicit>**

(3) st: Autonomic nervous system. **<student's expected correct answer>**

(4) tu: Yes, **<ordinary-positive ack>**

(5) So, TPR is primarily under neural control. **<primitive: inform> <same topic>**

(6) And the predictions that you are making are for the period before any changes take place. **<second topic> <DM: And>**

(7) So, what is the value of TPR? **<final topic> <DM: So>**

(8) st: No change **<student's correct answer>**

(9) tu: Good! **<emphatic-positive ack>**

In the first turn the tutor introduces the variable by using the discourse marker 'now'. Then, as the first topic, the tutor asks for the mechanism. With the student's correct answer in (3), the tutor gives an ordinary positive acknowledgment in (4) and rephrases the student's answer without changing the topic, using the discourse marker 'so' and the primitive act 'inform' (5). From sentences (3), (4), and (5) we can see the relationship {correct answer - give positive ack - inform same topic}. Also sentences (4) and (5) show the relation {expected answer - give positive ack - inform with discourse marker}. As a second topic the tutor provides DR stage information, using the discourse marker 'and' in sentence (6). Then for the final topic the tutor asks the value of the variable, using the discourse marker ‘so’ in (7). With the student’s correct final answer the tutor gives an emphatic positive acknowledgment of ‘Good’ in (9).

CHAPTER VII
SURFACE GENERATION

Eventually I generate sentences. The transcript markup described in Chapter 4 provides the information that is needed.

I started by identifying groups of sentences serving the same tutorial goal and extracted them from the transcripts. Then I analyzed each group to determine a set of content elements that could be used for building the sentences in the group. Then I identified the pieces of knowledge in the planning environment that could be used to determine the values of these elements. With this information, I sketched potential surface structures that the CIRCSIM-Tutor sentence generator could produce. Then I wrote a grammar for the Genkit sentence generator package which assembles the sentences from the elements identified above. Finally, I wrote some software to post-process the sentences resulting from Genkit in order to take care of issues such as capitalization and word morphology.

The result of finding the fixed and variable elements in a set of potential surface structures is the description of the information needed to construct the sentence, in all its variations. It is able to incorporate discourse markers as well as the interpersonal and textual meanings from Halliday's theory of coherence that we described in Chapter 3.

Since the purpose of this exercise was not simply descriptive, but destined to actually generate sentences, I omitted some sentences that do not clearly fit well into the machine tutor or were not part of the dialogue we want CST v.3 to engage in. Some sentences were rejected on the advice of Joe Michael and Allen. Rovick as not meeting their standards for what the tutor should say. Other sentences are not likely to be part of CIRCSIM-Tutor's repertoire, for example "Can you tell me what IS means?" is excluded because the student's probable answer to this question would be hard for the machine to process. Also excluded were many sentences related to the tutor replying to student initiatives, a case that is mostly not handled by CST, although we hope to do so in the future.

7.1 Data Analysis

7.1.1 Overall Analysis of Sentence Constituents. From our annotated transcripts I extracted groups of sentences where the tutor was expressing the same meaning. The sentences in Figure 7.1 constitute such a collection. In all these examples the tutor is eliciting the mechanism of control of a neurally controlled variable.

(1) **How is** TPR **controlled**?
(2) **How is** TPR **determined**?
(3) **What is the primary mechanism of control of** TPR?
(4) Can you tell me **how** TPR **is controlled**?
(5) Do you know **what determines the value of** TPR?
(6) AND **what is the primary mechanism by which** arteriolar radius **is controlled**?

Figure 7.1. Examples of **T-elicits info=mechanism**

For purposes of generation, I need to understand the variation among the sentences in each set. Although different sentences can never have identical meanings, for text generation purposes I only need to represent differences that the tutoring system needs to make. In other words, if two sentences serve the same purpose for our tutor, then they can be considered as different ways of expressing the same thought and can be generated from the same logical form. If two sentences can be used in the same slot, I prefer to consider them as equivalent rather than make arbitrary distinctions.

After collecting a set of related sentences, I was able to decompose them into independent elements. In generating the sentences, different sources of information will be used for deciding the values of these elements. The sentences in Figure 7.1 have been printed to show how they are built from these common elements:

- Main predicate. In this example the main predicate could be called “elicit mechanism of control.” The main predicate is printed in sans serif bold type.
- Name of variable. The name of the variable can be either abbreviated, as in “TPR,” or spelled-out, as “arteriolar radius.”.
- Softener (optional). We are using the term “softener” to describe expressions like “can you tell me” and “do you know,” which are underlined in the figures. It is interesting to note that when one of these expressions is realized as a sentence-initial clause, the surface subject of the sentence is different from the deep subject of the predicate. Glass [1997] points out the same phenomenon in the student’s side of the dialogue.
- Discourse marker (optional). Discourse markers are printed in small caps, e.g., AND in sentence (6). In our dialogues, most of the discourse markers are sentence-initial.

In decomposing the sentences this way, the elements are not always contiguous segments of text. For example, the main predicate “how is X determined?” has a spot in its middle to place the variable name.

Figure 7.2 shows a similar decomposition of sentences serving the purpose of eliciting the value of the variable. In addition to the previous elements, a few new elements are present:

- Stage modifier (optional). This specifies one of the three stages we have divided the physiological response into: DR, RR, and SS. Stage modifiers are printed in sans-serif italics, e.g., *in DR*.
- Context-setting expression (optional). We are using the term "context-setting expression" to describe expressions like "if CC is under neural control" or "that being the case," which set the context for the main clause. Note that some context-setting expressions are constant while others contain slots for additional variable information. Context-setting expressions are printed in serif italics type.
- Pointing expression (optional). Sentence (17) also contains the expression "as you predicted," which points to something which happened earlier in the dialogue. As Figure 7.3 shows, most pointing expressions in our dialogues precede the main clause and follow the optional discourse marker. Pointing expressions are printed in underlined serif italic type.

Figure 7.3 contains a selection of sentences that inform the value of the variable to the student. One new piece of information is required:

- Value of variable. It is qualitative: increase, decrease, or no change. Since the value is often expressed by the main verb, e.g. "decrease," I have not marked the value typographically, but instead include it within the main predicate sentence element.

(7) So **what about** TPR?
(8) So **what's your prediction of** CC *in the DR*?
(9) Now **what do you say about** TPR?
(10) But *if CC is under neural control*, **how would** it **be affected** *in the DR period*?
(11) So **what's your prediction about** CC?
(12) So **what would happen to** RAP?
(13) *That being the case*, **how would** RAP **change** *in DR*?
(14) So, *in the DR* **will there be any change in** TPR?
(15) And *if RAP increases* **what would happen to** RAP-*DR*?
(16) *That being the case*, **what will happen to** RAP-*DR* in this situation?
(17) *If cardiac output decreased* (*DR*) *as you predicted*, **what would happen to** RAP?

Figure 7.2. Examples of **T-elicits info=value**

Occasionally we need to add information to further qualify an element. For example, note that (20) simply refers to the fact that some variables are predicted to change, without mentioning the value of the change. The other examples in Figure 7.3 give a specific value for the prediction. When we need to differentiate between these two cases, we add an argument to the machine's description of the sentence to note whether the value is actually expressed or not in the final sentence.

(18) *You predicted that* CC **would go up**.
(19) But *remember that you said that* MAP **decreases** *in DR*.
(20) Well, *you made predictions about* **how** RAP and CC **would change**.
(21) *You predicted that* CO *in DR* **would go up**.

Figure 7.3. Examples of **T-informs info=value**

Figure 7.4 shows a group of sentences serving the goal of ***t-informs info=dr-info***. All of these sentences convey information about the definition of the Direct Response stage. This is one of the important concepts for correcting misunderstandings in the DR stage. There is one new sentence element:

- Remind (optional). This is expressed as "remember that" in sentences (22) and (25) in Figure 7.4.

Remind is a narrative mode, as described in Chapter 4.

(22) BUT *remember that* **we're dealing with the period before there can be any neural changes.**
(23) **We're talking about what happens before there are any neural changes.**
(24) **We're talking about the period before any change in neural activity.**
(25) *Remember that* **we're dealing with the short period before you get a reflex response.**
(26) AND **we're dealing with the period before any change in nervous activity occurs.**

Figure 7.4. Examples of **T-informs info=DR-info**

Generalizing from all these examples in Figures 7.1 through 7.4, we can see that our sentences are constructed from four kinds of elements:

- Main predicate and required arguments. In the examples above, the primary example of a required argument is the variable name.
- Optional arguments. Time qualifiers (e.g. *in DR*) are an example of an optional argument. If location qualifiers were used, they would also fall in this category.
- Interpersonal modifiers. Softening expressions are an example of this category.
- Narrative modifiers. Discourse markers, pointing expressions, context-setting expressions, and reminding expressions are included in this category.

For each element, we must determine which features in the planning environment are needed to determine its value and, if it is optional, whether it should be included at all. The tutorial history, the domain model, and the student model are among the data structures that can be consulted. However, I am most interested in features that come from the current planning agenda, which includes the current planning goal, the goals above it in the current hierarchy, and the arguments of these goals.

For example, consider the sentences in Figure 7.2 again. In these sentences, the main predicate can be determined from the current goal, i.e. **t-elicits**. The content to be elicited is carried down from the **info** argument of the parent goal. All of these sentences are derived from topic goals that contain either an explicit or implicit **info** argument.

According to these initial results, much of the tutoring language is stylized enough that sentences can be assembled in a simple manner. The analysis described in this chapter, which shows the relationship between tutorial goals and surface structure, can be used to determine the output of the sentence generator.

<u>7.1.2 Tabulated Sentence Decomposition</u>. The next step was to produce tables for each group of sentences, showing specimen sentences along with the different values of the different features for each sentence. These tables are useful for detailed analysis of the components to be used in sentence generation, where each sentence will be described by a collection of features. I found that for this purpose most of the hierarchical goal structure was not important, only the primitive and topic level operators are needed. So I did not include the method level operators in these tables. However the arguments carried along

with the goal structure, for example, the variable being tutored, are quite useful and thus assigned them as separate features.

From the human tutoring transcripts I picked ninety-one sentences that might be useful for machine generation and I made twenty-one tables. Each table mostly described a pair consisting of a primitive and a topic operator. For example, if the primitive operator is **elicit** and the topic operator is **t-tutors-value**, I made a table named “elicits-value.” All information for the tables came from the marked-up transcripts. Tables 7.1 and 7.2 are illustrative examples.

As we see in Table 7.1, cases (27) and (28) are different only in the attitude feature. In sentence (28) the attitude feature is unmarked, but in (27) the ‘rephrase-answer’ attitude inserts the adverb “primarily.” Table 7.1 sentences (28) and (29) have identical features and feature values except for the name of the variable, so the sentence generator can emit either sentence provided it has a slot in the sentence for placing the variable name. Although Table 7.1 case (30) comes from a different topic I included it here because it is expressing the same information as the other sentences. In case (30) we notice the presence of the discourse marker “but.” The “but” discourse marker is present because this sentence is serving a **t-presents-contradiction** topic. It is conveying the same information as the other three sentences, which are serving **t-tutors-mechanism**, with the addition of the contrastive discourse marker. Combining all these cases together allows us to add some generality to the eventual grammar. We will use the same bundle of features to describe all these sentences, with the discourse marker being an optional feature.

Table 7.1. Feature Descriptions of <T-informs-mechanism> Sentences

Primitive	Topic	Attitude	Var	Designate info	Discourse Marker	Case Sentence
T-informs	T-tutors-mechanism	rephrase-answer	TPR	neural-info		(27) TPR is primarily under neural control.
T-informs	T-tutors-mechanism		CC	neural-info		(28) CC is under neural control.
T-informs	T-tutors-mechanism		IS	neural-info		(29) IS is only affected by changes in sympathetic stimulation of the heart muscle.
T-informs	T-presents-contradiction		CC	neural-info	But	(30) But CC is under neural control.

Table 7.2 shows sentences for eliciting the determinants of a variable. Table 7.2 sentences (33) and (34) are typical short requests, differing only in the name of the variable. However this simple notion can be expressed in more complex ways. One reason is that when the tuto rs re-ask a question they often phrase it differently. I added the feature attempt number, showing how many times the same question has been asked inside of tutoring the same topic. The attempt number attribute governs the phrase “what I was asking is” in Table 7.2 case (32). Case (36) contains an example of the method type feature, which is used for marking that the context is an inner method correcting a topic that the student got wrong in the course of tutoring a normal method. In most of the cases, the beginning sentence of an inner method starts with a discourse marker “and.”

Sentence (32) has the “rephrase question” attitude, which results in replacing the variable name “SV” by a short definition. The result, “what determines how much blood is ejected from the heart each time it beats?” is far more complex than the simple “what are the determinants of SV?” in Table 7.2 case (33). Notice that replacing the name “SV” with its definition also changes the main verb of the sentence from “are” to “determines.” The whole sentence structure has changed.

7.1.3 Sketch of Potential Surface Structure. Comparing the different sets of examples, we notice many regularities in the surface syntax. A large percent of the sentences can be generated from the following:

- Discourse particle (optional). Source: discourse marker
- Front clause (optional). Source: pointing expression, context-setting expression, or softener

Table 7.2. Feature Descriptions of <T-elicits-determinant> Sentences

Primitive	Topic	Var	Softener	Attitude	DM	Attempt number	Method-type	Case Sentence
T-elicits	T-tutors-determinant	SV			now			(31) Now what two parameters in the prediction table together determine the value of the SV?
T-elicits	T-tutors-determinant	SV		rephrase-question		2		(32) What I was asking is what determines how much blood is ejected from the heart each time it beats ?
T-elicits	T-tutors-determinant	SV						(33) What are the determinants of SV?
T-elicits	T-tutors-determinant	RAP						(34) What parameter determines the value of RAP?
T-elicits	T-tutors-determinant	RAP	Can you tell me			2		(35) Can you now tell me what determines RAP
T-elicits	T-tutors-determinant	CVP			and		inner	(36) And what determines CVP?

- Main clause. Source: main predicate with required arguments in slots
- Additional arguments (optional). Source: stage modifiers and other potential optional arguments

We can imagine a sentence generator that basically just pastes together these kinds of pieces. Each input element can produce a different part of the sentence. Another approach is to build sentences from words according to English grammar. That approach would allow more complex rules for forming sentence components from the original meaning elements. The previous CST sentence generator [Chang, 1992] tried this method. The resulting grammar is complicated and inflexible.

In fact, I adopted something of a hybrid approach. In many cases, the sentences are pasted together as above. However, there are a number of instances where it is necessary to select on a combination of features, the “rephrase question” feature of sentence (32) being a good example. Furthermore, even when sentences are simply pasted together from pieces, often selecting the correct variant of the main predicate is constrained by auxiliary features. Contrast sentences (1) and (4) from Figure 7.1, which I repeat here:

(37)How is TPR controlled?

(38)Can you tell me how TPR is controlled?

The addition of the softener adds a main verb to the surface structure of the sentence. The result is that the verb “is” becomes inverted in (37) and not inverted in (38). Cases such as this one mean that even though the sentence generator is usually simply pasting parts of sentences together, it is required to check combinations of features and do something more sophisticated when picking some of the elements.

7.2 Development of the Grammar for Genkit

7.2.1 About the Genkit Grammar. Genkit [Tomita & Nyberg, 1988] is a generation package developed in 1988 at the Center for Machine Translation at Carnegie Mellon University. It compiles grammar rules into executable Lisp code. This code will be at the heart of Circsim-Tutor's sentence generator, converting the output emitted by the turn planner into sentences.

Genkit uses a subset of a unification grammar; it is simplified somewhat by not supporting full unification. The sentences to be generated are described by feature structures, in a manner similar to the input to the FUF generator [Elhadad, 1993]. The turn planner will put together a package of features, and pass this to the generator.

A feature structure is a set of (label, value) pairs. The label is a name, and the value can be either a simple value (a string, a number, etc.) or another, embedded, feature structure, as illustrated in Example 7.1.

Example 7.1:

```
((primitive  elicit)
 (var         ((var-name    TPR)
               (spelled     +)))
```

This feature structure has two features, labeled primitive and var. The primitive feature has the simple value elicit. The var feature has an embedded feature structure as its value.

Each Genkit grammar rule has two parts. The first part is a context-free phrase structure rule. While this phrase structure rule is being interpreted, every non-terminal symbol has an associated feature structure. The second part of a Genkit rule is a list of pseudo-equations relating these associated feature structures. A typical rule is illustrated here in Example 7.2.

Example 7.2:

```
;; Phrase structure rule: <start> --> <ask>
(<start> = => (<ask>)
  ;; Feature structure constraint equations
  (((X0 primitive) =c elicit)
   (X1 = X0)))
```

In the phrase structure rule, the left hand side is a non-terminal symbol and the right hand side is a non-terminal, a terminal, or a "%" which serves as a wild card symbol. In the feature structure constraint equations, the different feature structures are labeled X0, X1, X2, etc. Here, X0 refers to the feature structure associated with the <start> non-terminal in the left hand side, and X1 refers to the feature structure for <ask>, the non-terminal in the right hand side. This grammar formalism is nearly identical to the PATR-II formalism [Shieber, 1986].

Genkit grammar rules are interpreted according to the following procedure. The input to the generator is a feature structure, which becomes associated with the starting non-terminal. The generator tries to expand the non-terminal, trying the phrase structure rules one at a time in succession. Each time it tries a rule, it applies the feature structure constraints. In Example 7.2, the constraint equation:

```
((X0 primitive) =c elicit)
```

verifies that the primitive feature in feature structure X0 has the value elicit. If the input to the generator were the feature structure shown in Example 7.1, this equation would be satisfied. If the equation cannot be satisfied by the input feature structure, the generator proceeds to try another rule. The constraint equation `X1 = X0` says that the feature structure associated with first right hand side constituent, <ask>, will be identical to the feature structure associated with the left hand side. The generator makes this happen, providing a feature structure to associate with <ask>. Now the generator operates recursively, finding a rule for expanding the non-terminal <ask>. The process stops, of course, when there are no more non-terminals to expand or there are no applicable grammar rules.

Terminal symbols on the right hand side are simply Lisp atoms not encased in angle brackets. Because these are simple Lisp atoms, the list of terminal symbols that is the result of the generation process does not carry lower case / upper case information. This list of symbols is therefore post-processed to insert typographic case and other niceties.

There are numerous extensions to the grammar formalism to make it more expressive or efficient. For example, it is possible to insert Lisp function calls into the constraint equations. Various other extensions to the constraint equations are quite procedural in flavor. A wild card extension can be used to extract values from the feature structure and insert them into the right hand side of the phrase structure production. This enables the turn planner to place values in the feature structure strings which will be echoed in the output sentence. For example, the value of the DM (discourse marker) feature can be the discourse marker itself.

The Genkit grammar uses two arrows in its phrase structure rules: "==>" and "->." The former arrow produces a space between right hand side constituents, as between words. The latter arrow produces no spaces between them, as between the components of a word.

7.2.2 Producing Elicit Mechanism Sentences. Sentences that elicit the mechanism of control, such as those in Figure 7.1, serve as a fairly typical example of how our Genkit grammar for the sentence generator works. Figure 7.5 shows some of the rules for picking a predicate for asking the mechanism of control. The grammar begins with the <start> non-terminal. In this section I will show some of the capabilities of the grammar I wrote by giving some examples.

The grammar rules in Figure 7.5 pick non-terminal <EM1>, <EM2>, etc., based on the values of the various features. Each <EM> turn into a different main predicate for asking the mechanism of control. Only <EM4> is shown in Figure 7.5 for illustration. I also show that the optional discourse marker is inserted before the main predicate, controlled by the DM feature. The various main predicates are as follows:

- EM1 <softwhat> is the primary mechanism of control of <var> ?
- EM2 How is <var> controlled?
- EM3 <softwhat> is the primary mechanism by which <var> is controlled?
- EM4 <softwhat> is the mechanism that will cause <var> to <speci-val>?

Choosing EM4 is controlled by whether there is a value specified, for example "increase." This form of the sentence is the only one that requires a value. I have no principled method to pick among the other three possible main predicates. Therefore I created a feature called PICK which takes integral values. The PICK feature is set by the turn planner. In the absence of other governing features the grammar picks a variant of the main predicate according to the PICK value. When the PICK feature is absent, a grammar rule chooses a default main predicate. So if there is some reason to prefer a particular version of a sentence, the turn planner can pick it.

The non-terminal <softwhat> deserves mention. It derives the word “what” plus an optional softener. The four options implemented so far are:

- “What”
- “What do you think”
- “Can you tell me what”
- “Do you know what”

If a softener is present, picking among these is governed by the SOFTWPICK feature in the same manner as the PICK feature. Main predicate EM2, “How is <var> controlled?,” comes in two variants to handle the verb inversion problem caused by introducing a softener, as described in section 7.1.3. One reason for creating a terminal which covers both “what” plus softener is that the order between the two elements is not fixed. In the analysis of section 7.1, it appears that the softener occurs in a fixed location at the beginning of the sentence. However in the sentence “what do you think is the mechanism …” the softener is not sentence-initial, it occurs immediately after “what.”

The <DM> non-terminal takes the discourse marker specified by the turn planner in the value of the DM feature. If there is no discourse marker, the symbol /NO-DM is inserted in its place. This dummy terminal is removed in post-processing, at the same time typographical case is adjusted. The /NO-DM symbol serves only a debugging purpose. It is very useful to know when the discourse marker rules did not run, which inserts nothing in the sentence, and when they did run and concluded there was no discourse marker, which inserts /NO-DM. Discourse marker grammar rules are as follows:

```
;; If no DM feature is present, insert dummy marker
(<DM> ==> (</NO-DM>)
  ((x0 DM) = *UNDEFINED*)))

;; Otherwise value of DM feature is Discourse Marker
(<DM> ==> (%)
  (((x1 value) <= (x0 DM))))
```

In this way, if the input feature structure contains (DM BUT), then "but" will be added to the output sentence.

The <var> non-terminal produces a variable name. It is controlled by several features, for example it can be spelled out or abbreviated, or it can be reduced to a pronoun. Variables that are known to the grammar are represented in the feature structure by atoms, e.g. TPR. The following feature structure will produce "Total Peripheral Resistance" in the output stream:

```
(var  (var-name    tpr)
      (spelled-out +))
```

If the variable name is unknown to the grammar, it is placed in the output stream untouched. This enables the turn planner to easily generate sentences regarding any variable, regardless of whether the grammar has been updated. It is also more robust than giving an error message, or refusing to produce a sentence, or leaving the variable name blank in the output. Here is a possible input feature structure:

```
(var  (var-name    |Left Ventricular Filling|))
```

The above will produce the following in the output stream:

```
/LIT Left Ventricular Filling /ENDLIT
```

The /LIT and /ENDLIT symbols are flags intended for the post-processing program. They indicate that the text between them is to be taken literally. For example, it will not have its typographical case adjusted.

The <speci-val> non-terminal in the rule for <EM4> in Figure 7.5 is controlled by the speci-val feature, expressing a qualitative value. The value must be up, down, or unchanged. A PICK feature allows one to choose between “go up” / “go down” and “increase” / “decrease” expressions of the changed value. In the rule for <EM4> in Figure 7.5 the constraint equation:

```
((x10 tense) <= 'inf)
```

forces the feature structure associated with <speci-val> into the infinitive for this sentence. For most sentences, features tense and modal control the generation expressions such as “will go up” and “should decrease.” However, in the <EM4> sentence the tense must be infinitive, so this tense is forced in by the grammar rule regardless of whether the user specified a tense.

```
;; All ELICITS sentences come from <ask>
(<start> ==> (<ask>)
  (((X0 primitive) =c elicits)
   (X1 = X0)))

;; <ask-mech> derives all ask mechanism sentences
;; <DM> derives optional discourse marker
(<ask> ==> (<DM> <ask-mech>)
  (((X0 topic) =c mechanism)
   (X1 = X0)
   (X2 = X0)))

;; EM4 is needed when a value is specified
(<ask-mech> ==> (<em4>)
  (((x0 speci-val) = *DEFINED*)
   (x1 = x0)))

;; Otherwise pick among the other <ask-mech>
predicates
(<ask-mech> ==> (<em1>)
  (((x0 PICK) = 1)
   (x1 = x0)))

(<ask-mech> ==> (<em2>)
  (((x0 PICK) = 2)
   (x1 = x0)))

(<em4> ==> (<softwhat> is the mechanism that will
             cause <var> to <speci-val> <qmark>)
  (x8  = x0)
  (x10 = (x0 speci-val))
  ((x10 tense) = inf)
  (x1  = x0)))
```

Figure 7.5. Some Rules for Generation of Elicit Mechanism Sentences

We can now see some sample input feature structure and output sentences from the elicit mechanism set. The simplest possible sentence has no discourse markers, no softeners, and no specified value. The feature structure which is input is:

```
((primitive  elicits)
 (topic      mechanism)
 (var        (var-name tpr))
 (pick       2))
```

This will generate “How is TPR controlled?” A more complicated feature structure for generating an elicit-mechanism sentence might have a specified value, a discourse marker, a softener, and a spelled-out variable name:

```
((primitive  elicits)
 (topic      mechanism)
 (var        (var-name hr)
             (spelled-out +))
 (speci-val  (value -)
             (specivpick 2))
 (dm         so)
 (softwpick  2))
```

Finally that will produce “So can you tell me what is the mechanism that will cause Heart Rate to decrease?”

7.2.3 Generalized Grammar for Stage Modifier. As an illustration of a case where I needed to write grammar rules to implement a construction in a way that is somewhat more like traditional grammar, I show here the rules for constructing the stage modifier. The stage modifier is composed of an optional preposition and a noun phrase, for example “in DR”, “during the DR period”, and “during the direct response period.” The first grammar rule is the usual decomposition of a prepositional phrase:

```
(<stage-pp> ==> (<stage-prep> <stage-NP>)
  ((x1 = (x0 stage-modifier))
   (x2 = (x0 stage-modifier))))
```

The default preposition is “in.” If the duration feature is specified, the preposition becomes “during:”

```
(<stage-prep> ==> (in))
(<stage-prep> ==> (during)
 (((x0 duration) =c +)))
```

For formatting the noun phrase, there is a style feature. If the style feature value is plain or omitted, the output is only the stage name. If the feature has a value of period the output will be “the <stage-name> period:”

```
(<stage-NP> ==> (<stg>)
 ((*or*     (((x0 style) = *UNDEFINED*))
      (((x0 style) =c plain)))
 (x1 = x0)))

(<stage-NP> ==> (the <stg> period)
 (((x0 style) =c period)
 (x2 = x0)))
```

The stage name can be abbreviated or spelled out, as controlled by the stage-name and spelled-out features:

```
 (<stg> ==> (steady state)
 (((x0 stage-name) =c SS)
 ((x0 spelled-out) =c +)))
```

And as a default case, if the value of stage-name is not recognized it is inserted into the output stream as a wild card.

```
(<stg> ==> (%)
 (((x1 value) <= (x0 stage-name))))
```

A complete feature structure for generating “During the Steady State period” would be:

```
(stage-modifier  (stage-name ss)
                 (duration +)
                 (style period)
                 (spelled-out +))
```

The grammar for stage modifiers is illustrated in diagram form in Figure 7.6.

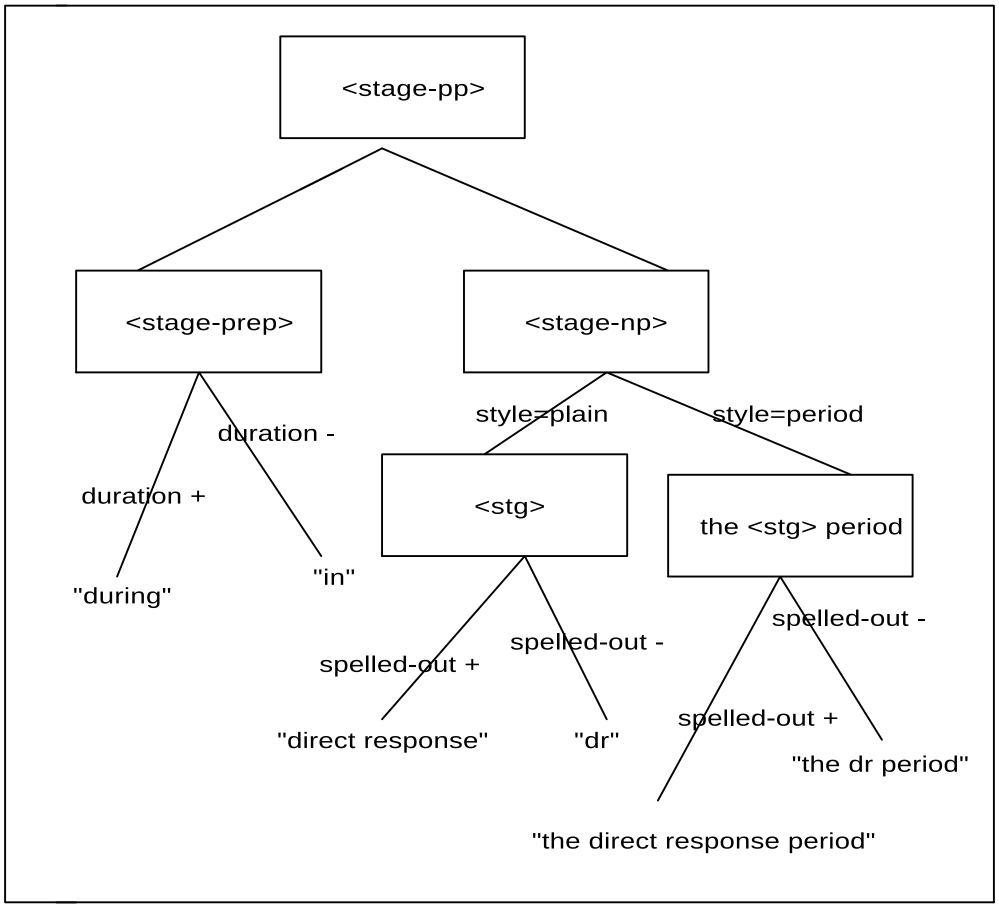

Figure 7.6. Generation of the Stage Modifier

7.2.4 Producing Inform Value Sentences. Figure 7.7 shows the sentence generation grammar for the operator **T-informs info=value**. This covers simple sentences such as “HR goes up,” in addition to the more complicated sentences in Figure 7.3. Figure 7.8 shows some example input feature structures and generated sentences, using the grammar rule from Figure 7.7 and earlier in this chapter.

This grammar more closely follows the pattern of pasting pieces of the sentence together. One thing to note is that this grammar handles the conjunction of two variables in some of its forms.

```
 (<start>  ==> (<tell>)
  (((x0 primitive) =c informs)
   (x1 = x0)))

(<tell> ==> (<DM> <tell-value> <stage-pp> <pmark>)
  ((*or*
      (((x0 topic) =c tutors-value))
      (((x0 topic) =c presents-contradiction)))
   (x2 = x0)
   (x1 = x0)
   (x3 = x0)
   (x4 = x0)))

(<tell-value> ==> (you predicted that <iv1>)
  (((x0 narrative-mode) =c reference)
   (x4 = x0)))

(<tell-value> ==> (<iv2> <context-set>)
  (((x0 context-setting) = *DEFINED*)
   ((x0 attitude) =c bolster-answer)
   (x1 = x0)
   (x2 = x0)))

(<tell-value> ==> (<iv1>)
  ((x1 = x0)))

(<iv1> ==> (<var> <speci-val>)
  ((x1 = x0)
   (x2 = (x0 speci-val))))

(<iv2> ==> (<var> <speci-val> and <var> <speci-val>)
  ((x1 = (x0 var1))
   (x2 = (x0 speci-val1))
   (x4 = (x0 var2))
   (x5 = (x0 speci-val2))))))
```

Figure 7.7. Grammar Rules for **T-informs Info=value**

```
((primitive informs)
 (topic tutors-value)
 (var TPR)
 (speci-val ((val 0)
             (modal would)))
 (stage-modifier ((stage-name DR))))
```

"TPR would not change in DR."

```
((primitive informs)
 (topic tutors-value)
 (var CC)
 (narrative-mode reference)
 (speci-val ((val +)
             (tense future)
             (specivPICK 1))))
```

You predicted that CC will go up.

```
((primitive informs)
 (topic tutors-value)
 (stage-modifier (stage-name dr))
 (context-setting "When CO decreases")
 (var1 ((var ((var-name RAP)))))
 (var2 ((var ((var-name SV )))))
 (attitude bolster-answer)
 (speci-val1 ((val +)
              (tense  p3s)
              (specivPICK 2)))
 (speci-val2 ((val +)
              (tense  p3s)
              (specivPICK 2)))))
```

"When CO decreases, RAP increases and SV increases. in DR"

Figure 7.8. Inform Value Feature Structures Input and Generated Sentences

7.3 Conclusions

I started with a simple idea: grouping together sentences expressing similar meaning, identifying common elements, and producing a grammar that will paste together sentences in the simplest possible way. Ultimately the task proved to be not so simple as originally envisioned. Some care was taken in the selection of sentences, to restrict this exercise to sentences that might reasonably be emitted by CIRCSIM-Tutor, but even so there was quite a variety. When examined closely, pasting sentences together required a certain amount of grammatical knowledge and more grammatical rules than originally planned. Nevertheless, the resulting grammar is quite small compared to one that would describe all the sentences in terms of individual words and parts of speech. I feel it is also reasonably flexible. It was easy to build-in special cases for special needs, without disturbing the whole grammar.

CHAPTER VIII

CONCLUSIONS

8.1 Summary

To generate more interactive and more natural dialogue in Circsim-Tutor I have developed a new approach to the analysis of human tutoring behavior including strategy, tactics, and the use of language from the CIRCSIM-Tutor project tutoring transcripts.

To write the goals for the planning engines (what to say, when to say it, and how to say it) in the new CIRCSIM-Tutor, I analyzed and marked up transcripts of human tutoring sessions. Over 270 turns, from the pedagogical portions of the sessions, have been analyzed in depth. Many more have been partially analyzed for specific purposes. This analysis produced multiple nested annotations showing both global goals for tutoring and local goals for maintaining a coherent conversation, as well as a categorization of the student responses.

Among many attempts by other researchers to analyze our human tutoring transcripts, this is the first to produce a result where the annotations can be accessed by machine utilities. It is also the most detailed. Besides the original purpose of discovering tutorial planning goals, it has provided the data for our machine learning research and become the basis for my analysis of sentence realization.

In the comparison of expert tutors and novice tutors I concluded that our experts promote the student's active learning with more questions. The experts also detect student misconceptions with more sophisticated questions. In contrast to experts, novice tutors frequently asked the student “Do you understand?” to confirm student’s knowledge and their teaching style is more complicated and wordy.

In the research on cue words, discourse markers are important for marking topic boundaries for cohesion in naturalistic dialogues. Also, I found that there is a relationship between cue words and primitive dialogue acts.

I have further analyzed the construction of sentences using the knowledge of the tutoring and discourse goals in the marked-up transcripts. To achieve this, I collected together instances of similar tutor sentences as a group. I also collected all the sentences that were uttered in identical situations. With these results I identified values of features for each sentence. Using the Genkit package I produced a generator that generates sentences from the output of the turn planner.

8.2 Significance

There are not very many intelligent tutoring systems that use natural language for both input and output. CIRCSIM-Tutor is a break from the tradition of communicating with students using only completely-canned text, and a step toward more natural communication.

My research has great significance for the new version of Circsim-Tutor.

1. Producing the first machine-usable markup, an input to several endeavors including the machine learning research
2. Finding detailed tutoring patterns from transcripts
3. Finding an effective tutoring style from the comparison of expert and novice tutors

4. Making those tutoring patterns rigorous enough that they can serve as planner goals
5. Making turn's more coherent with machine derived rules for cue words.
6. Analyzing sentences in detail, so that we can generate much better ones than CST v.2 does
7. Making it possible to generate a variety of sentences for the same tutorial goal.

Additionally, this work will influence the development of the machine tutoring field.

8.3 Future Work

There is some future work needed to complete the new version of the CST sentence generator.

1. Complete markup for all stages: My work focused on the DR stage mostly and some of the RR stage. We need more SGML-style markup for RR and SS stages. Also, we need more markup of novice tutoring sessions.
2. Evaluation of the results of markup: With the calculation of a statistical value of reliability we can ensure the accuracy of our analysis. For this purpose we need to train several persons and we should to compare their individual results to get some index of reliability.
3. Further machine learning: To enhance the fluency of the dialogue. We can apply machine learning to other sentence elements such as softeners or stage-modifiers.
4. Generalization and expansion of the Genkit grammar.

BIBLIOGRAPHY

Allen, James and Mark Core. 1997. Draft of DAMSL: Dialog Act Markup in Several Layers. Available from the University of Rochester Department of Computer Science at ftp://ftp.cs.rochester.edu/pub/packages/dialog-annotation/manual.ps.gz.

Appelt, Douglas E. 1985. *Planning English Sentences*. Cambridge: Cambridge University Press.

Becker, Joseph D. 1975. "The Phrasal Lexicon," *Proceedings of the Conference on Theoretical Issues in Natural Language Processing*, Cambridge MA.

Bovair, Susan, and David Kieras. 1985. "A Guide to Propositional Analysis for Research on Technical Prose." In B. K. Britton and J. B. Black, eds., *Understanding Expository Text*, Hillsdale, NJ: Lawrence Erlbaum, pp. 315–362.

Bradley, Neil. 1996. *The Concise SGML Companion*. Reading, MA: Addison-Wesley.

Brandle, Stefan. 1998. *Using Joint Actions to Explain Acknowledgments in Tutorial Discourse: Application to Intelligent Tutoring Systems.* Ph.D. thesis, Department of Computer Science, Illinois Institute of Technology.

Cawsey, Alison. 1992. *Explanation and Interaction: The Computer Generation of Explanatory Dialogues*. Cambridge, MA: MIT Press.

Chang, Ru-Charn. 1992. *Surface Level Generation of Tutorial Dialogue Using a Specially Developed Lexical Functional Grammar and Lexicon*. Ph.D. thesis, Department of Computer Science, Illinois Institute of Technology.

Charniak, Eugene and Drew McDermott. 1985. *Introduction to Artificial Intelligence*. Reading, MA: Addison-Wesley.

Cho, Byung-In, Joel A. Michael, Allen A. Rovick, and Martha W. Evens. 1999. "A Curriculum Planning Model for an Intelligent Tutoring System," *Proceedings of the Twelfth International Florida AI Research Society Conference (FLAIRS-99), Orlando, FL*, Menlo Park: AAAI Press, pp. 197–201.

Cohen, P., J. Kulik and C. Kulik. 1982. "Educational Outcomes of Tutoring: A Meta-Analysis of Findings," *American Educational Research Journal*, vol. 19, pp. 237–248.

Collier, John Takao, Martha W. Evens, Daniel Hier, and Ping-Yang Li. 1988. "Generating Case Reports for a Medical Expert System," *International Journal of Expert Systems*, vol. 1, no. 4, pp.307–328.

Core, Mark and James Allen. 1997. Coding Dialogues with the DAMSL Annotation Scheme, presented at AAAI Fall 1997 Symposium on Communicative Action in Humans and Machines, Available from University of Rochester Department of Computer Science at ftp://ftp.cs.rochester.edu/pub/papers/ai/97.Core-Allen.AAAI.ps.gz. Working notes from the symposium, including this paper plus an update, are available at http://www.cs.umd.edu/~traum/CA/.

Di Eugenio, Barbara, Johanna D. Moore, and Massimo Paolucci. 1997. "Learning Features that Predict Cue Usage," *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL '97)*, also published as *Proceedings of the 8th Conference of the European Chapter of the Association for Computational Linguistics (EACL '97), Madrid*, New Brunswick: NJ: Association for Computational Linguistics, pp. 80–87.

Dickinson, C. J., C. H. Goldsmith and David L. Sackett. 1973. "MACMAN: A Digital Computer Model for Teaching Some Basic Principles of Hemodynamics," *Journal of Clinical Computing*, vol. 2, no. 4, pp. 42–50.

Eggins, Suzanne. 1994. *An Introduction to Systemic Functional Linguistics*. London: Pinter.

Elhadad, Michael. 1993. *FUF: The Universal Unifier, User Manual, Version 5.2.* Available from Department of Computer Science, Ben Gurion University of the Negev. Available from the FUF web page: http://www.cs.bgu.ac.il/surge/

Evens, Martha W., John Spitkovsky, Patrick Boyle, Joel A. Michael and Allen A. Rovick. 1993. "Synthesizing Tutorial Dialogues," *Proceedings of the 15th Annual Conference of the Cognitive Science Society*, Boulder, 1993. Hillsdale, NJ: Lawrence Erlbaum.

Fox, Barbara. 1993. *The Human Tutorial Dialogue Project*. Hillsdale, NJ: Lawrence Erlbaum.

Freedman, Reva. 1995. "Using Pedagogical Knowledge to Structure Text Generation in an Intelligent Tutoring System," *Proceedings of the 1995 Midwest Artificial Intelligence and Cognitive Science Society Conference*, Carbondale, pp. 48–52.

Freedman, Reva. 1996a. "Using a Text Planner to Model the Behavior of Human Tutors in an ITS." In Michael Gasser, ed., *Online Proceedings of the 1996 Midwest Artificial Intelligence and Cognitive Science Conference*, Bloomington, IN.

Freedman, Reva. 1996b. "Using Tutoring Patterns to Generate More Cohesive Text in an Intelligent Tutoring System." In Daniel C. Edelson and Eric A. Domeshek, eds., *Proceedings of International Conference on the Learning Sciences, 1996*, Evanston, IL, pp. 75–82.

Freedman, Reva. 1996c. *Interaction of Discourse Planning, Instructional Planning and Dialogue Management in an Interactive Tutoring System*. Ph.D. thesis, Dept. of EECS, Northwestern University.

Freedman, Reva. 1999. “Atlas: A Plan Manager for Mixed-Initiative, Multimodal Dialogue,” AAAI ’99 Workshop on Mixed-Initiative Intelligence, Orlando.

Freedman, Reva and Martha W. Evens. 1996. “Generating and Revising Hierarchical Multi-turn Text Plans in an ITS.” In Claude Frasson, Gilles Gauthier and Alan Lesgold, eds., *Intelligent Tutoring Systems: Third International Conference (ITS ‘96), Proceedings*, Montreal. (Springer-Verlag Lecture Notes in Computer Science, no. 1086.) Berlin: Springer-Verlag, pp. 632–640.

Freedman, Reva, Yujian Zhou, Jung Hee Kim, Michael Glass, and Martha W. Evens. 1998a. “SGML-Based Markup as a Step toward Improving Knowledge Acquisition for Text Generation,” *AAAI 1998 Spring Symposium: Applying Machine Learning to Discourse Processing*, pp. 114-117.

Freedman, Reva, Yujian Zhou, Michael Glass, Jung Hee Kim, and Martha W. Evens. 1998b. “Using Rule Induction to Assist in Rule Construction for a Natural-Language Based Intelligent Tutoring System,” *Proceedings of the Twentieth Annual Conference of the Cognitive Science Society*, Madison. Hillsdale, NJ: Lawrence Erlbaum, pp. 362-367.

Freedman, Reva, Stefan Brandle, Michael Glass, Jung Hee Kim, Yujian Zhou, and Martha W. Evens. 1998c. “Content Planning as the Basis for an Intelligent Tutoring System” description of a system demonstration, *Proceedings of the Ninth International Workshop on Natural Language Generation (INLG-9), Niagara-on-the-Lake, Ontario*, New Brunswick, NJ: Association for Computational Linguistics, pp. 280–283.

Glass, Michael. 1997. “Some Phenomena Handled by the Circsim-Tutor Version 3 Input Understander,” *Proceedings of the Tenth Florida Artificial Intelligent Research Symposium*, Daytona Beach, FL, pp. 21–25.

Glass, Michael. 1999. *Broadening Input Understanding in an Intelligent Tutoring System.* Ph.D. thesis, Department of Computer Science, Illinois Institute of Technology.

Glass, Michael, Jung Hee Kim, Martha W. Evens, Joel A. Michael, and Allen A. Rovick. 1999. “Novice vs. Expert Tutors: A Comparison of Style,” *Proceedings of the Tenth Midwest Artificial Intelligence and Cognitive Science Conference (MAICS-99), Bloomington, IN,* Menlo Park: AAAI Press, pp. 43–49.

Graesser, Arthur C. 1993. *Questioning Mechanisms During Tutoring, Conversation, and Human-Computer Interaction,* Technical Report R&T 4422576 of the Cognitive Science Program, Office of Naval Research.

Graesser, Arthur C, Natalie K. Person, Joseph P. Magliano. 1995. “Collaborative Dialogue Patterns in Naturalistic One-to-One Tutoring,” *Applied Cognitive Psychology*, vol. 9, pp. 495–522.

Green, Nancy and Sandra Carberry. 1999. “Interpreting and Generating Indirect Answers,” *Computational Linguistics*, vol. 25, no. 3, pp. 389–435.

Grosz, Barbara J. and Candace L. Sidner. 1986. “Attention, Intention, and the Structure of Discourse,” *Computational Linguistics*, vol. 12, no. 3, pp. 175–204.

Halliday, M. A. K. 1985. *An Introduction to Functional Grammar*. London: Edward Arnold.

Halliday, M. A. K. and Ruqaiya Hasan. 1976. *Cohesion in English*. London: Longman.

Hume, Gregory D. 1995. *Using Student Modelling to Determine When and How to Hint in an Intelligent Tutoring System*. Ph.D. thesis, Department of Computer Science, Illinois Institute of Technology.

Hume, Gregory D., Joel A. Michael, Allen A. Rovick, and Martha W. Evens. 1993. “The Use of Hints as a Tutorial Tactic,” *Proceedings of the 15th Annual Conference of the Cognitive Science Society*, Boulder. Hillsdale, NJ: Lawrence Erlbaum, pp. 563–568.

Hume, Gregory D., Joel A. Michael, Allen A. Rovick, and Martha W. Evens. 1995. “Controlling Active Learning: How Tutors Decide When to Generate Hints,” *Proceedings of the 8th Florida Artificial Intelligence Research Symposium*, Pensacola, pp. 157–161.

Hume, Gregory D., Joel A. Michael, Allen A. Rovick, and Martha W. Evens. 1996. “Hinting as a Tactic in One-on-One Tutoring,” *Journal of the Learning Sciences*, vol. 5 no. 1, pp. 32–47.

Jacobs, Paul S. 1988. “PHRED: A Generator for Natural Language Interfaces.” In David D. McDonald and Leonard Bolc, eds., *Natural Language Generation Systems*, New York, Springer-Verlag, pp. 312–351.

Khuwaja, Ramzan A. 1994. *A Model of Tutoring: Facilitating Knowledge Integration using Multiple Models of the Domain*. Ph.D. thesis, Department of Computer Science, Illinois Institute of Technology.

Khuwaja, Ramzan A., Allen A. Rovick, Joel A. Michael and Martha W. Evens. 1995. “A Tale of Three Protocols: The Implications for Intelligent Tutoring Systems.” In E. A. Yfantis, ed., *Intelligent Systems: Third Golden West International Conference*, Las Vegas, 1994 (Theory and Decision Library, Series D: System Theory, Knowledge Engineering, and Problem Solving, v. 15), Dordrecht: Kluwer Academic, pp. 109–118.

Kieras, David E. 1985. “Thematic Processes in the Comprehension of Technical Prose.” In B. K. Britton & J. B. Black, eds., *Understanding Expository Text*, Hillsdale, NJ: Lawrence Erlbaum, pp. 89–108.

Kim, Jung Hee, Reva Freedman, and Martha W. Evens. 1998a. “Responding to Unexpected Student Utterances in CIRCSIM-Tutor v. 3: Analysis of Transcripts,” *Proceedings of the Eleventh Florida Artificial Intelligence Research Symposium (FLAIRS ’98),* Sanibel Island, FL, pp. 153–157.

Kim, Jung Hee, Reva Freedman, and Martha W. Evens. 1998b. “Relationship between Tutorial Goals and Sentence Structure in a Corpus of Tutoring Transcripts,” *Proceedings of Nineth Midwest AI and Cognitive Science Conference*, Dayton, OH, Menlo Park, CA: AAAI Press, 1998, pp. 124–131.

Kim, Jung Hee, Michael Glass, Reva Freedman, and Martha W. Evens. 2000. “Learning Use of Discourse Markers in Tutorial Dialogue for an Intelligent Tutoring System,” *Proceedings of Twenty second Annual Meetings of the Cognitive Science Society*, Philadelphia, PA, to appear.

Kim, Nakhoon. *Circsim-Tutor: An Intelligent Tutoring System for Circulatory Physiology*. 1989. Ph.D. thesis, Department of Computer Science, Illinois Institute of Technology.

Kinstsch, Walter. 1974. *The Representation of Meaning in Memory*. Hillsdale, New Jersey: Lawrence Erlbaum.

Kukich, Karen. 1988. “Fluency in Natural Language Reports.” In David D. McDonald and Leonard Bolc, eds., *Natural Language Generation Systems*. New York, Springer-Verlag, pp. 280–311.

Lee, Yoon-Hee. 1990. *Handling Ill-Formed Natural Language Input for an Intelligent Tutoring System*. Ph.D. thesis, Department of Computer Science, Illinois Institute of Technology.

Lepper, Mark R., Maria Woolverton, Donna L. Mumme and Jean-Luc Gurtner. 1993. “Motivational Techniques of Expert Human Tutors: Lessons for the Design of Computer Based Tutors.” In Susanne. P. Lajoie and Sharon. J. Derry, eds., *Computers as Cognitive Tools,* Hillsdale, NJ: Lawrence Erlbaum, pp. 75–105.

Litman, Diane J. 1994. “Classifying Cue Phrases in Text and Speech Using Machine Learning,” *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)* Seattle, WA, AAAI Press, pp. 806–813.

Mann, William C. and Sandra A. Thompson. 1988. “Rhetorical Structure Theory: Towards a Functional Theory of Text Organization,” *Text*, vol. 8, no. 3, pp. 243–281.

McKelvie, David, Henry Thompson, Richard Tobin, Chris Brew, and Andrei Mikheev. 1997. *The Normalized SGML Library LT NSL version 1.5*, Language Technology Group Human Communication Research Centre, University of Edinburgh. Information available at `http://www.ltg.ed.ac.uk/`.

McKeown, Kathleen R. 1985. *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge: Cambridge University Press.

Meteer, Marie. 1989. “The SPOKESMAN Natural Language Generation System.” Report 7090, BBN Systems and Technologies, Cambridge, MA.

Michael, Joel A. and Allen A. Rovick. 1986. “CIRCSIM: An IBM PC Computer Teaching Exercise on Blood Pressure Regulation.” *XXX Congress of the International Union of Physiological Sciences*, Vancouver Canada.

Moore, Johanna D. 1995. *Participating in Explanatory Dialogues: Interpreting and Responding to Questions in Context*, Cambridge, MA: MIT Press.

Moser, Megan and Johanna D. Moore. 1995. “Investigating Cue Selection and Placement in Tutorial Discourse,” *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL ’95)*, Cambridge, MA, New Brunswick: NJ: Association for Computational Linguistics, pp. 130–135.

Nakano, Yukiko I. and Tsuneaki Kato. 1998 “Cue Phrase Selection in Instruction Dialogue Using Machine Learning,” *Discourse Relations and Discourse Markers: Proceedings of the Workshop at Coling-ACL ’98, Montreal*, New Brunswick, NJ: Association for Computational Linguistics, pp. 100–106.

Nyberg, Eric H. III, Rita McCardell, Donna Gates and Sergei Nirenburg. 1991. “Target Text Generation.” In Goodman, Kenneth and Sergei Nirenburg, eds., *The KBMT Project: A Case Study in Knowledge-Based Machine Translation*, San Mateo, CA: Morgan Kaufmann, pp. 231–261.

Paris, Cécile L. 1993. *User Modelling in Text Generation*. London: Pinter. An earlier version was published as the author’s Ph.D. thesis at Columbia University in 1987.

Quinlan, J. Ross. 1986 “Induction of Decision Trees,” *Machine Learning*, vol. 1, pp. 81–106.

Quinlan, J. Ross. 1993. *C4.5: Programs for Machine Learning*, San Mateo, CA: Morgan Kaufmann Publishers.

Quirk, Randolph, Sidney Greenbaum, Geoffrey Leech, and Jan Svartvik. 1972. *A Grammar of Contemporary English*, London: Longman.

Rambow, Owen and Tanya Korelsky. 1992. “Applied text generation,” *Proceedings of the Third Conference on Applied Natural Language Processing (ANLP-1992)*, pp. 40–47.

Reiter, Ehud. 1994. “Has a Consensus NL Generation Architecture Appeared, and is it Psycholinguistically Plausible?,” *Proceedings of the Seventh International Workshop on Natural Language Generation*, Kennebunkport, ME, pp. 163–170. Available from the cmplg archive as paper 9411032.

Reiter, Ehud. 1995. “NLG vs. Templates,” *Proceedings of the Fifth European Workshop on Natural Language Generation*, Leiden. Available from the cmplg archive as paper 9504013.

Reiter, Ehud. 1996. “Building Natural-Language Generation Systems.” In Alison Cawsey, ed., *Proceedings of the AI and Patient Education Workshop*, Glasgow, GIST Technical Report G95.3, Department of Computing Science, University of Glasgow.

Reiter, Ehud and Chris Mellish. 1993. “Optimizing the Cost and Benefits of Natural Language Generation,” *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-1993)*, vol. 2, pp. 1164–1169.

Reiter, Ehud and Robert Dale. 1997. “Building Applied Natural Language Generation Systems.” *Journal of Natural Language Engineering*, vol. 3 no. 1, pp. 57–87.

Reiter, Ehud, Chris Mellish, and John Levine. 1995. “Automatic Generation of Technical Documentation,” *Applied Artificial Intelligence*, vol. 9, pp. 259–287. Available from the cmplg archive as paper cmp-lg/9411031.

Robin, Jacques. 1994. *Revision-Based Generation of Natural Language Summaries Providing Historical Background: Corpus-Based Analysis, Design, Implementation and Evaluation*. Ph.D. thesis, Department of Computer Science, Columbia University. Technical Report CUCS-034-94.

Rovick, Allen A. and Joel A. Michael. 1992. “The Predictions Table: A Tool for Assessing Students’ Knowledge,” *American Journal of Physiology*, vol. 263 no. 6, part 3, pp. S33–S36. Also available as *Advances in Physiology Education*, vol. 8 no. 1, pp. S33–S36.

Rovick, Allen A. and Lisa Brenner. 1983. “HEARTSIM: A Cardiovascular Simulation with Didactic Feedback,” *The Physiologist*, vol. 26 no. 4, pp. 236–239.

Russell, Stuart. 1996. “Machine Learning.” In Margaret A. Boden, ed., *Artificial Intelligence*, San Diego, CA: Academic Press, pp. 89–133.

Sanders, Gregory A. 1995. *Generation of Explanations and Multi-Turn Discourse Structures in Tutorial Dialogue, Based on Transcript Analysis*. Ph.D. thesis, Department of Computer Science, Illinois Institute of Technology.

Sanders, Gregory A., Martha W. Evens, Allen A. Rovick, and Joel A. Michael. 1992. "An Analysis of How Students Take the Initiative in Keyboard-to-Keyboard Tutorial Dialogues in a Fixed Domain," *Proceedings of the Fourteenth Annual Meeting of the Cognitive Science Society*, Bloomington, IN. Hillsdale, NJ: Lawrence Erlbaum, pp. 1086–1091.

Schiffrin, Deborah. 1987. *Discourse Markers*. Cambridge: Cambridge University Press.

Seu, Jai Hyun. 1992. *The Development of an Input Understander for an Intelligent Tutoring System based on a Sublanguage Study*. Ph.D. thesis, Department of Computer Science, Illinois Institute of Technology.

Seu, Jai, Ru-Charn Chang, Jun Li, Martha W. Evens, Joel A. Michael and Allen A. Rovick. 1991. "Language Differences in Face-to-Face and Keyboard-to-Keyboard Tutoring Sessions," *Proceedings of the 13th Annual Conference of the Cognitive Science Society*, Chicago. Hillsdale, NJ: Lawrence Erlbaum, pp. 576-580.

Shah, Farhana. 1997. *Recognizing and Responding to Student Plans in an Intelligent Tutoring System: Circsim-Tutor*. Ph.D. thesis, Department of Computer Science, Illinois Institute of Technology.

Shieber, Stuart M. 1986. *An Introduction to Unification-Based Approaches to Grammar*. Stanford, CA: Center for the Study of Language and Information.

Sinclair, John M. and Richard M. Coulthard. 1975. *Towards an Analysis of Discourse: The English Used by Teachers and Pupils.* London: Oxford University Press.

Sleeman, Derek H. and John S. Brown, eds. 1982. *Intelligent Tutoring Systems.* New York: Academic Press.

Spitkovsky, John A. and Martha W. Evens. 1993. "Negative Acknowledgments in Natural Language Tutoring Systems," Fifth Midwest AI and Cognitive Science Conference (MAICS-93), Chesterton IN, pp. 41–45.

Tomita, Masaru and Eric H. Nyberg 3rd. 1988. *Generation Kit and Transformation Kit Version 3.2 User's Manual*, Available from Language Technology Institute at Carnegie Mellon University.

Traum, David R. and James F. Allen. 1994. "Discourse Obligations in Dialogue Processing," *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, Las Cruces, NM, pp. 1-8.

Wenger, Etienne. 1987. *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge*. Los Altos, CA: Morgan Kaufmann.

Woo, Chong W. 1991. *Instructional Planning in an Intelligent Tutoring System: Combining Global Lesson Plans with Local Discourse Control*. Ph.D. thesis, Department of Computer Science, Illinois Institute of Technology.

Yang, Feng-Jen, Jung Hee Kim, Michael Glass and Martha Evens. 2000a. "Turn Planning in CIRCSIM-Tutor," *Proceedings of the Thirteenth Florida Artificial Intelligence Research Symposium (FLAIRS '2000), Orlando, FL*, to appear.

Yang, Feng-Jen, Jung Hee Kim, Michael Glass and Martha Evens. 2000b. "Lexical Usage in the Tutoring Schema of CIRCSIM-Tutor: Analysis of Variable References and Discourse Markers," *Proceedings of Human Interaction with Complex Systems(HICS) 2000*, Urbana-Champaign, IL, to appear.

Yang, Qiang. 1997. *Intelligent Planning*, Berlin: Springer.

Young, R. Michael. 1994. *A Developer's Guide to the LONGBOW Discourse Planning System*. Technical Report 94–4, Intelligent Systems Program, University of Pittsburgh. Available from project web page at `http://www.isp.pitt.edu/~young/longbow/`

Zhang, Yuemei. 1991. *Knowledge-Based Discourse Generation for an Intelligent Tutoring System*. Ph.D. thesis, Department of Computer Science, Illinois Institute of Technology.

Zhou, Yujian and Martha Evens. 1999c. "A Practical Student Model in an Intelligent Tutoring System," *Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence,* Chicago, IL, pp. 13-18.

Zhou, Yujian, Reva Freedman, Michael Glass, Joel A. Michael, Allen A. Rovick, and Martha W. Evens. 1999a. "Delivering Hints in a Dialogue-Based Intelligent Tutoring System," *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99), Orlando, FL*, Menlo Park: AAAI Press, pp. 128–134.

Zhou, Yujian, Reva Freedman, Michael Glass, Joel A. Michael, Allen A. Rovick, and Martha W. Evens. 1999b. "What Should the Tutor Do When the Student Cannot Answer a Question?," *Proceedings of the Twelfth International Florida AI Research Society Conference (FLAIRS-99), Orlando, FL*, Menlo Park: AAAI Press, pp. 187–191.

Zhou, Yujian, Reva Freedman, Michael Glass, Joel A. Michael, Allen A. Rovick, and Martha W. Evens. 1999c. "Delivering Hints in a Dialogue-Based Intelligent Tutoring System," *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99), Orlando, FL,* Menlo Park: AAAI Press, pp. 128–134.