Published in CHI '83: Human Factors in Computer Systems, ACM: New York, December 1983, pp. 230-234.

Human-Computer Discourse in the Design of a PASCAL Tutor

Beverly Woolf
David D. McDonald

COINS Technical Report 83-27
Nov 1983

Department of Computer and Information Science
University of Massachusetts
Amherst, Massachusetts 01003

Abstract

An effective human-computer discourse system requires more than a clever grammar or a rich knowledge base. It needs knowledge about the user and his understanding of the domain in order to produce a relevant and coherent discourse. We describe MENO, a prototype tutor for elementary PASCAL, which uses a set of speech patterns modelled after complex human discourse and a richly annotated knowledge base to produce a flexible interactive system for the user.

1.0 INTRODUCTION

We want our computer systems to produce high quality and natural sounding texts comparable to those a human would write. This requires more than a rich grammar or even an adroit planning component. It requires a domain where the system can select the appropriate content based on an understanding of its audience. We need to understand the epistemology of the domain and to have a realistic way to accumulate a rich model of the user.

In this short paper we discuss how these requirements can be satisfied in the production of a tutoring text in a domain where prior cognitive studies exist. We also describe our on-going experience with human and machine tutoring systems to demonstrate how the depth of knowledge we can formalize about human tutorial protocols has permitted us to explore a level of sophistication in human-machine discourse not previously available.

2.0 A TAXONOMY OF HUMAN DISCOURSE

Research on natural language generation has reached a level of technical competence where the proper goal is to try to match actual human performance. To this end, our work on tutoring has included an examination of the human dialogues that were recorded in earlier Artificial Intelligence (A. I.) work. Trying to make sense of the different patterns of speech patterns we found, we arrived at the following four-part taxonomy (see Figure 1) based on differences in the expert's goals and assumptions about the novice.

The most potent distinction for our purposes is between *guidance discourse* and *reconstruction discourse*. In *guidance discourse* the goal of the discourse is to accomplish a task; the expert directs the listener to largely new information without concern for what she already knows. In *reconstruction discourse* the goal is to reconstruct the listener's faulty or confused knowledge. The speaker clarifies and corrects the listener's knowledge, possibly by pinpointing misconceptions through careful diagnosis and argumentation.

An instance of *guidance discourse* was presented by Grosz [1976] who recorded a dialogue in which an expert guided a novice through the first steps to assemble a water pump. The expert directed and corrected the novice, but largely ignored his prior knowledge. He neither interrogated nor attempted to remodel

GUIDANCE DISCOURSE

FOCUS	*ASSUMPTIONS*	*DISCOURSE PLAN*
Accomplish a task	Minimal experience	Provide instructions and Correct
Explore knowledge	Incomplete knowledge	Provide instructions and interrogate

RECONSTRUCTION DISCOURSE

FOCUS	*ASSUMPTIONS*	*DISCOURSE PLAN*
Rebuild the student's cognitive model	Confused knowledge	Interrogate and correct
Reevaluate the student's assumptions	Wrong knowledge	Interrogate and provide reasoned arguments

Figure 1: Planning Human Discourse .

---

the novice's beliefs about the workings of a water pump.

An example of *reconstruction discourse* was given by Pollack [1982] in a dialogue between a tax expert and a novice tax payer. The expert convinced the novice that she must adjust her ideas about taking a charitable deduction and must accept a solution which she had rejected for the wrong reasons. In this dialogue the expert tested the novice's almost complete knowledge and "remade" its fuzzy or imperfect areas.

Reconstruction discourse is among the most tightly coherent found in speech. This is because of the amount of knowledge that contributes to it. The expert speaker in this kind of dialogue must know about:

1. typical "correct" solutions,

2. errors and misconceptions which may suggest the need for reconstruction,

3. a rich model of the difference between expert and novice models of the knowledge,

4. strategies which can bring an incorrect model around to a more acceptable form.

Working with this much knowledge can positively bias the selection of content. So much data is brought to bear by the conditions above that a clear indication of the next topic and the method of proceeding is available. We used this complexity of data as a basis for our text generation, as explained below.

3.0 COMPLEX SPEECH ACTS

Our analysis of protocols of human discourse revealed that the speaker's intentions while engaged in *reconstruction discourse* determined, to some extent, the speech patterns used. One of the striking commonalities in the human dialogues we studied was the presence of the same patterns of speech across sessions and across speakers. These patterns, along with "packages" of phrases, were used as templates to handle specific types of audiences under specific conditions.

For instance, the "grain of truth" pattern illustrated in Figure 2 (excerpts from [Stevens et al, 1982]) was used primarily when the student's answer was *incorrect*, yet there was an indication that the student had a partial *correct* understanding of the problem. The tutor's response in both instances was to pick out the "grain of truth" in the student's answer, acknowledge it, and then propose the the correct answer.[1] The tutor's response was "packaged" in a marked

[1] Though the "no" of strategy 2 is missing from the second discourse we believe it is because the student in that discourse indicated uncertainty about the answer.

DISCOURSE 1:

Tutor: What happens to the temperature of moist air when it rises?

Student: It gets warm.

STRATEGY
1: Pose a question
[Note: wrong answer]

Tutor: No, warm air [does] rise.
But, as it rises it cools off.

STRATEGY
2: Indicate wrong answer
3: Reinforce "grain of truth"
4: Indicate contrast
5: State correct answer

Tutor: Do you think the mountains in Washington and Oregon have anything to do with cooling the moist air?

Student: Yes.

STRATEGY
6: Ask a leading question

DISCOURSE 2:

Tutor: Can you tell me what causes air to rise over Ireland?

Student: The wind?

STRATEGY
1: Pose a question
[Note: wrong answer]

Tutor: The wind moves over the air,
But, that is not what causes it to rise.

STRATEGY
3: Reinforce "grain of truth"
4: Indicate contrast

Tutor: Do you think the cold air mass that is usually sitting over Ireland might be important?

Student: If it just sits there it is.

STRATEGY
6: Ask a leading question

Figure 2: Common Patterns in Human Discourse.

syntactic pattern: the reinforcement was kept short and always appeared in the same position in what was effectively a template.[1]

These discourse segments illustrate the use of a single template response to handle topics of varying complexity. In the first discourse, the tutor deemed it possible and appropriate to state the correct answer: "[warm air] rises [as] it cools off". However the tutor in the second segment possibly decided that the topic, "wind over water", was too complex or too important to provide the answer and move on. Instead, the tutor chose to pursue the next logical causal factor by asking a leading question about the effect of a large cold body on rainfall. This question manifests a "hidden agenda" to explore a commonly held misconception that large land or air bodies "cause" rainfall.

4.0 PLANNING STRATEGY

Our analysis of these and other segments reveals complex planning stategies at work. We propose that discourse planning takes place in three levels: *pedagogical*, *strategic* and *tactical planning*.

At the *pedgogical planning* level a choice is made about the the style of tutoring, like Socratic, coach, or web-tutoring This choice is made before the tutoring session begins and typically is not reconsidered unless the student is performing poorly.

At the *strategic planning* level a choice is made among highly schematic tutoring scripts like "question the student", "describe the concept" or "choose a new topic". The two scripts at work in both examples above were "reinforce the grain of truth" "ask a leading question". A refinement to the strategy of choice was made in the second dialogue, in that the tutor used "ask a leading question" without first providing the correct answer.

At the *tactical planning* level the speech-pattern is selected and the conceptual content fit into it. At this level also, domain specific data is accessed for the first time and the student response is interpreted and matched with

[1] There *are* variations in the two discourses: strategy 5 which supplies the answer in the first discourse has been put off in the second. Instead a "leading question" (strategy 6) is presented to give the student a clue as to how to deduce the answer for herself. We assert that these two passages reflect the same speech patterns because we believe that their similarity in form and tempo overrides their "tactical" variation in content.

domain data.[1] The human tutor does not seem to pursue a predefined list of topics. In fact, the same error from different students may elicit a variety of responses depending on student history and potential misconceptions. The tutor will also use considerations about the richness of the domain and the student's knowledge to determine which topics to pursue.

5.0 A TUTOR FOR ELEMENTARY PASCAL

We have adapted the planning and explanation heuristics described above for use in a machine tutor for novice PASCAL programmers. Our research has focused on *reconstruction discourse* because (1) we came to the work having already identified a set of two dozen misconceptions commonly held by novice programmers [Soloway et al. 1982], and (2) we felt that we could better motivate the textual details of a *reconstruction discourse* (and also better debug faulty generation rules) because of the conceptual richness provided by the misconceptions in conjunction with the annotations on the knowledge base.

The MENO system [Woolf, Ph.D. in preparation] is a prototype tutor nearing completion at UMASS. It receives data about run-time semantic errors from the Bug Finder [Soloway et al, 1981], and interrogates the student, proposing relevant misconceptions and suggesting missing knowledge. The Bug Finder has been in use for 4 semesters on classes of several hundred students at UMASS.

As an example of what MENO does, consider the program in Figure 3, and the dialogue that it engenders in Figure 4. (The program was submitted by a novice programmer. The dialogue was generated by the Tutor based on a simulated student model and responses fed into the system.)

Given the program of Figure 3 and what the student has gotten right, we can be fairly certain that the student understands the basics of loop programming. Still we must be certain of just where tutoring can begin, and that the tutor and the student share a common vocabulary. The first two interactions (lines 1 and 2) question the student about general looping concepts.

---

[1] In the machine dialogue, presented later in this paper, we restrict the user's input to multiple choice since we don't have enough common-sense knowledge about the student's replies to handle proper language understanding.

```
1   PROGRAM LESSON1(INPUT, OUTPUT);
2     VAR
3       SUM,GRADES,STUDENTS:INTEGER;
4       MEDIAN:REAL;
5     BEGIN
6       SUM:=0;
7       STUDENTS:=0;
8       READ(GRADES);
9       WHILE GRADES>0 DO
10        BEGIN
11        SUM:=SUM + GRADES;
12        STUDENTS:=STUDENTS+1;
13        GRADES:=GRADES+1;
                [ should be READ(GRADES) ]
14        END;
15      MEDIAN:=SUM/STUDENTS;
16      WRITELN
17       ('THE MEDIAN GRADE IS', MEDIAN:8:3)
18     END.
```

Figure 3: An actual student program with the READ-loop bug.

In line 3 the tutor begins to analyze deeper programming concepts. Psychological studies have identified several misconceptions which might underlie the "READ-loop bug". It seems to stem from either of two misconceptions: (1) that a single call to READ will access all the values of a variable at once, or (2) that updating the control variable (line 13) will do a tacit READ through each value of the variable each time through the loop. The questions in lines 3 and 4 discriminate between these two misconceptions and in this case indicate, as is often the case, that the student had both misconceptions.

The tutor's response in line 5 is an example of the "grain of truth" speech pattern. It confirms and reinforces the student's correct linkage between the ADD 1 operation and bumping the control variable, while pointing out that it is inappropriate in this particular case.

1. **Machine:** Do you know that GRADES in line 9,
   is a control variable for the WHILE loop?

   Student: YES

2. **Machine:** Does that mean that GRADES should be
   a) Initiated before the loop?
   b) Added into the value SUM within the loop?
   c) Modified within the loop?
   d) Both a) and c)

   Student: D

3. **Machine:** You seem to understand the basic
   concepts around control variables. Lets
   talk about the READ command. What do you think
   READ (GRADES) in line 8 does with the value of
   GRADES?
   a) Reads all values of GRADES at one time
   b) Reads in a single value of GRADES
   c) Assigns several new input values to GRADES

   Student: A

4. **Machine:** Do you think that GRADES := GRADES + 1
   updates GRADES to the next of those values?

   Student: Yes

5. **Machine:** There is your misconception.
   Adding one to a variable *is* a common way
   to step a variable through its range of values.
   But it cannot apply here because at line 13,
   the variable GRADES has only one value.
   The READ instruction in line 9 has read only
   a single value of GRADES.

   You should replace line 13 with a second call
   to READ(GRADES).

Figure 4: Interaction with the user about the program of Figure 3.

6.0 SYSTEM COMPONENTS AND CURRENT STATUS

The MENO system contains three primary knowedge sources: (1) a model of PASCAL looping constructs, (2) assumptions about incorrect yet commonly held misconceptions and (3) a dynamic model of the user. The system is driven by a three-level planning hierarchy, similar to that described in section 4 and one which elaborates on Clancy's dialogue management rules [1979]. The actual text is produced by MUMBLE, a text generator which attends to details of low-level coherency such as lexical choice.

The knowledge bases are developed from cognitive studies of novice programmers [Bonar, 1982] and are encoded in Kl-One semantic networks. They use programming plans to represent expert PASCAL knowledge and variations on these plans to represent incorrect student knowledge. Annotations on this knowledge guide the planning and production of discourse. The annotations indicate the importance of topics and their relation to other topics. This system of annotations represents a first pass on implementing Goldstein's Genetic Graph [Goldstein, 1979] which proposed that domain knowledge be structured for pedagogical presentation to the student.

MENO is Woolf's Doctoral thesis, with completion expected by the end of this year. The system now contains the knowledge and strategies described above for the "READ bug" and we plan to introduce more domain knowledge to enable MENO to react flexibly to several other looping situations.

6.1 Conclusions

Our observations about human discourse have suggested primitives for tutoring and for structuring human-computer discourse. By incorporating these principles into a domain where we understand both the substantive knowledge of the domain and the cognitive structures to be taught, the tutorial system begins to be flexible in its interactions with the student.

6.2 References

Bonar, J., "Natural Problem Solving Strategies and Programming Language Constructs", in *Proceedings of the Fourth Annual Conference of the Cognitive Science Society*, 1982.

Clancy, W., "Transfer of Rule-based Expertise through Tutorial dialogue", Ph.D. dissertation, Department of Computer Science, Stanford University, 1979.

Collins, A., Warnock, E., and Passafiume, J., "Analysis and Synthesis of Tutorial Dialogues", *Psychology of Learning and Motivation*, Vol 9, Academic Press, 1975.

Goldstein, I.P., "The Genetic Graph: A Representation for the Evolution of Procedural Knowledge", *International Journal of Man-Machine Studies*, 1,11, pp 51-78, 1979.

Grosz, B., "Discourse Analysis", in D.E. Walker (ed.), *Speech Understanding Research*, Menlo Park, CA., Stanford Research Institute, 1976.

McDonald, D., "Description Directed Control: Its implications for natural language generation", in *International Journal of Computers and Mathematics*, Vol. 9, No. 1, 1983.

Pollack, M., Hirschberg, J., Webber, B., "Participation in the Reasoning Process of Expert Systems", in *Proceedings of the American Association of Artificial Intelligence*, 1982.

Soloway, E., Ehrlich K., Bonar J., Greenspan, J., "What Do Novices Know About Programming?", in *Directions in Human-Computer Interactions*, edited by Shneiderman, B and Badre A., Ablex Publishing Company, 1982.

Soloway, E., Woolf, B., Rubin, E. and Barth P., "Meno-II: An Intelligent Tutoring System for Novice Programmers", *Proceedings of International Joint Conference in Artificial Intelligence*, Vancouver, British Columbia, 1981.

Stevens, A., Collins A., and Goldin, S., "Misconceptions in Student's Understanding", in Sleeman, D. & Brown, J.S. ed, *Intelligent Tutoring Systems*, 1982, Academic Press.

Woolf, B., "Tutorial Discourse Based on Context Driven Text Generation", Ph.D. Dissertation, in preparation, Computer and Information Sciences, University of Massachusetts, Amherst.