A Framework for Representing Tutorial Discourse[1]

Beverly Woolf and Tom Murray
Department of Computer and Information Science
University of Massachusetts
Amherst, Massachusetts 01003

Abstract

We set forth general techniques for managing discourse in an intelligent tutor. These techniques are being implemented in a structure that dynamically reasons about the discourse, a student's response, and the tutor's move. The structure is flexible, domain-independent, and designed to be rebuilt – decision points and machine actions are modifiable through a visual editor. We discuss this formal reasoning structure and its application in an intelligent tutor.

I. Tutorial Discourse

We have built a process model for tutorial discourse that provides custom-tailored feedback to students in the form of examples, analogies, and simulations. The processing model views discourse as navigation through a set of possible discourse situations. Transition from one situation to another is dynamically generated so the system is capable of tracking and responding to contingencies in discourse alternatives. Fundamental to our perspective is the view that tutoring conversation is motivated by general rules (principles) of discourse and selection of intervention techniques based on error and misconception analysis.

Effective tutoring requires sophisticated and dynamic reasoning about selection of tutoring strategy, the choice of a path through the curriculum, updating the student model, and assessment of student errors and misconceptions. Conversational actions produced by the tutor and responses from students will change the state of the discourse, and the structure must decide what the tutor should say and how it should interpret and act on subsequent student responses. Hence a desiderata on the design of any discourse manager is that it respond fluidly to the user and that it coordinate its utterances in a more flexible manner than has been required for question/answer or summarization systems.

Networks and production rule formalisms have been used to identify admissible instructional actions in tutoring systems (e.g., [Cerri, 1978; Clancey, 1982]). However, such formalisms are often domain-dependent and restricted to a narrow set of didactic responses. We have extended such formalisms by modifying the basic ATN architecture, adding greater functionality to it, and by developing a domain-independent structure that can be used in a variety of tutors.

We have also incorporated results from cognitive research about effective tutoring into the framework, including principles that drive production of good discourse. We are making the discourse framework modifiable through a visual editor that allows a knowledge engineer to access the machine's response decisions and actions. In the long term, we intend to make the steps of this reasoning process available to human teachers who can then modify the tutor for use in a classroom.

II. Discourse Formalism

In earlier work, we described a discourse manager to facilitate context-dependent interpretations of machine response [Woolf & McDonald, 1984]. In this research, we modify that earlier architecture and add a taxonomy of frequently observed discourse sequences to provide default responses for the tutor. The new structure is based on discourse *schemas*, or collections of discourse activities and tutoring responses, as shown in Figure 1. These schemas are derived from empirical research into tutoring discourse, including studies of teaching and learning [Brown et al., 1986; Littman et al., 1986], misconception research [Clement, 1982; Stevens et al, 1982], felicity laws [vanLehn, 1983], and general rules of discourse structure [Grosz & Sidner, 1985].

In this architecture, machine response is generated by traversal through the formal structure called a *Tutoring ACtion Transition Network (TACTN)*,[2] [McDonald et al., 1986]. The space of possible discourse situations is expressed by arcs, defined as predicate sets, that track the state of the conversation. Nodes provide actions for the tutor. The outer loop of the discourse manager first accesses the situation indicated by the arcs, resolving any conflicts between multiply satisfied predicate sets, and then directs the system's other components (e.g., the underlying domain expert component, the language component, or the

[1]This work was supported in part by the Air Force Systems Command, Rome Air Development Center, Griffiss AFB, New York, 13441 and the Air Force Office of Scientific Research, Bolling AFB, DC 20332 under contract No. F30602-85-C-0008. This contract supports the Northeast Artificial Intelligence Consortium (NAIC). Partial support also from ONR University Research Initiative Contract No. N00014-86-K-0764.

[2]Rhymes with ACT-IN

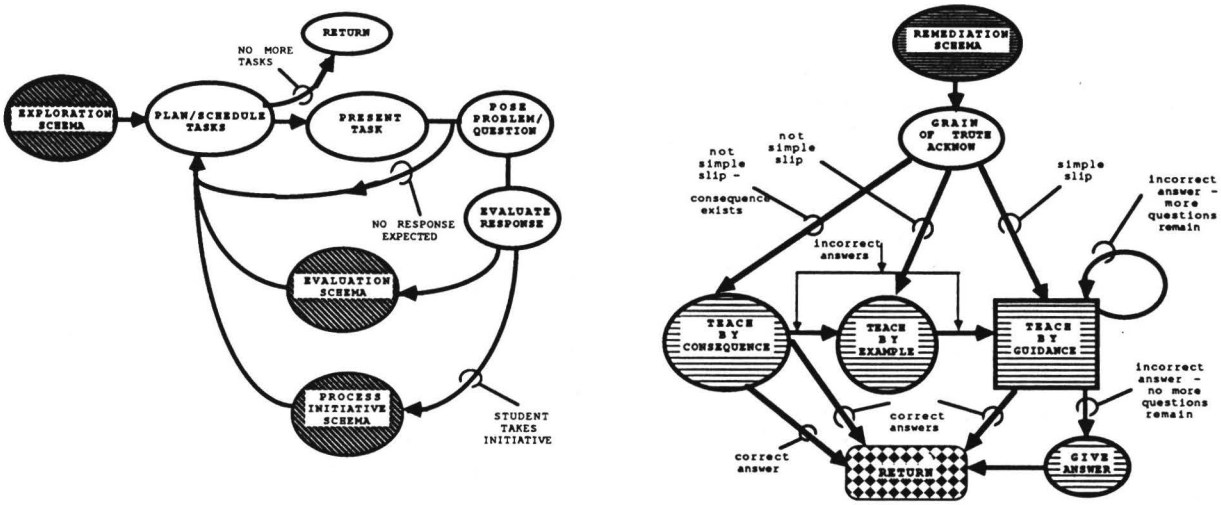

Figure 1: Tutoring Discourse Schemas

student model) to carry out the action indicated by the node.

Discourse planning consists of passage through the arcs and nodes of the schemas, with the number and type of schemas depending on context (see Figure 1). For example, if the student's answer is correct, Evaluation, and Process Correct Answer (not shown) Schemas will be activated. However, if the student's answer is incorrect, up to six schemas will be traversed, including three schemas activated by the Remediation Schema: Teach by Consequence, Teach by Example, and Teach by Guidance. The exact number of schemas depends on the tutor's assessment of student error, i.e., whether it was *a simple slip, not a simple slip,* or *not a simple slip - consequence exists.*

In the next section we provide a brief example of how this discourse planning framework is being implemented in a series of physics tutors. In the following sections we describe the tutoring structure in more detail.

III. Case Example

We are building science tutors as part of the Exploring Systems Earth (ESE) consortium.[3] The science tutors provide interactive simulations whose aim is to encourage students to work with "elements" of physics, such as mass, acceleration, and force. The goal is to aid students in developing problem-solving skills, knowledge about physics concepts, and intuitions to facilitate learning about knowledge and skills.

In these tutors, students explore heuristic simulations, such as the one shown in Figure 2, called the crane boom problem. In this example, students are asked to identify forces and torques on a crane boom and wall such that the boom will remain in equilibrium, i.e., there will be no vertical or horizontal movement. Students must draw appropriate force vectors or to solve the problem through equations.

The Exploration Schema in Figure 1 provides the main control loop for the tutor's interactions about this and other physics problems. It schedules tasks, which might be items from a top-level curriculum list or subgoals generated by earlier tasks. Associated with each task is information about how to present the problem situation, what question(s) to ask, which knowledge is assumed of the student, etc.

If a student had provided the force vectors shown in the center of Figure 2, the solution would be incorrect because of a missing force vector (located at the wall and pointing upwards). The tutor compares the student's response with the correct one and then activates the Remediation Schema in Figure 1. The conversation produced is shown in Figure 3.

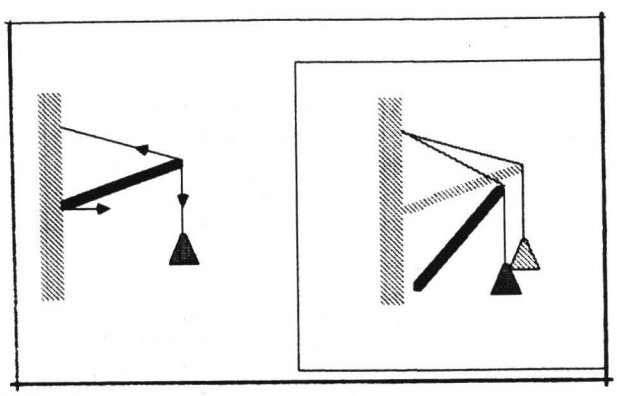

Figure 2: Simulation Presented to the Student

[3]ESE is a group of universities working together to develop intelligent science tutors. The schools include the University of Massachusetts, San Francisco State University, and University of Hawaii.

situation: - incorrect answer
action: [grain of truth acknowledgement]

T: The beam does experience forces in each place you have shown and in the directions you have shown.

situation: - not simple slip and consequence exists
action: [teach by consequence]

T: Here is what the beam would look like if it experienced the two forces you indicated (beam pulls away from the wall, right side of Figure 2).
T: Is this OK or would you like to reconsider your answer?
S: It's OK. *situation: - incorrect answer*
action: [teach by example]

T: Consider a situation in which a roller skate wheel is attached to the left end of the beam. What would happen to the beam then?
S: It would fall because it would roll down the wall.

situation: - correct answer
action: [compare answers]

T: Compare the two situations–with and without the roller. Do you want to reconsider either of your answers in light of this comparison?
S: No. *situation: - incorrect answer*
action: [teach by guidance]

T: You are missing a force somewhere on the beam. Do you want another hint?

Figure 3: Conversation Between Tutor and Student

IV. Representational Power

TACTN structures have the look of conventional transition graph of "nodes" and "arcs" as one would find in ATNs [Woods, 1970], yet they are defined in terms of "actions" and "situations" [McDonald et al., 1986] as one would find in a production system. The motivation to modify the ATN architecture was based on the observation that ATNs were designed for natural language parsing and are non-deterministic. [4]

This type of uncertainty has no counterpart in discourse generation, which requires a **planning** rather than a **parsing** formalism [McDonald et al., 1986]. Any action in a TACTN can be taken deterministically. For example, each of the transitions, Teach by Consequence, Teach by Example, or Teach by Guidance, might be taken locally without the need to wait for a global interpretation. Discourse generation requires choosing between actions, not between interpretations, placing it in the realm of a planner, not a parser. Since we were not using all the capacity provided by ATNs, we decided to modify the architecture.

Elements from both production systems and network formalisms have been built into TACTNs. Situations (arcs) are associated with actions (nodes), in the manner of a production system, and discourse history is encoded in arcs and nodes, in the manner of a network system. Every situation/arc implicitly includes as one of its constitutent predicates the action(s)/node(s) from which it came. Thus, if an arc originated in a particular action, there is a tacit predicate included in the arc's situation to the effect that the action must have just been taken. The single notational framework has the flexibility of a production system and the record-keeping and sequencing ability of a network system, by virtue of its context.

A pure production system discourse manager cannot retain and use a large amount of context to handle complex shifts in dynamic conversation. For example, GUIDON [Clancey, 1982] was a tutor based on a set of situation action rules that were driven by chaining backwards from goals. It provided flexibility to respond to dynamically changing discourse circumstances. However, there was no provision for sequencing situation action chains except by using ad hoc state variables. In TACTNs, the act of chaining to arcs from actions provides just such a sequencing mechanism.

A. Situations Defined by Arcs

Arcs in the discourse structure are defined by a set of predicates that track the student and discourse from the perspective of the system. Arcs correspond to discourse situations. For example, in Figure 1 the arc *simple slip* is a compound predicate that is true under two conditions: (1) the current topic is factual and the student has had medium success with it in the past, or (2) the topic is conceptual and the student has had high success with it. In a sense, situations are abstractions over the state of the system or student knowledge, expressing generalized conditions such as *student takes initiative* or *student is confused*.

The definition of arcs as aggregations of predicates in a boolean formula provides a powerful tool for knowledge engineers. Modifying predicates, and therefore arcs, allows fine-grain changes on individual predicates to impact greatly on the system's reasoning ability and to result in consequential changes to the tutor's discourse activity.

Within this structure, several arcs might define nearly equivalent situations as in the case when two arcs share one or more predicates. At such time, a conflict between arcs will be resolved through global or local (associated with specific nodes) conflict resolution stategies.[5] One solution is to order the arcs in the set according to their specificity and to execute the first triggered arc. In this way, the most

[4] Nodes in the original ATN represented accepted definitions for incoming tokens and arcs represented tests made on those incoming tokens. Non-determinism was motivated by uncertainty or the need to wait for an accumulated global interpretation before the system could be confident about the local interpretation of each token being scanned.

[5] Conflict resolution in this sense is analogous to what happens in a production system when the left-hand sides of more than one rule are satisfied.

specific subsuming situation will be preferred over other situations with which the arc shares predicates. However, we prefer to evaluate all the arcs, since incomparable situations (i.e. situations whose sets of predicates are disjoint) are likely.

For example, suppose that students ask many questions after giving a wrong answer, and suppose that they also ask several seemingly random questions. One tutoring convention says that answering students' questions should take priority; another says that random questions should be discouraged. In such a case a non-conventional resolution mechanism must be used to resolve the conflict.

B. Actions Defined by Nodes

Nodes correspond to actions available to the system; they define alternative conversations and tutoring strategies. Nodes differ in the actions they perform and in the manner they present tasks to students. Shifting actions to the nodes, as we did for TACTNs, instead of leaving them on the arcs, as was done for ATNs, facilitates the notation of expanding abstract actions into more concrete substeps. Abstract actions are nodes that are to be expanded and refined one or more times before taking on a form that can be executed. For example, the node Evaluation Schema is an abstract node whose expansion led to a second abstract node called Remediation Schema. On the other hand, the node Present Task represents an immediately executable action. Action expansion is an activity of the discourse manager. This notion of abstract planning borrows principally from Sacerdoti [1974] and Stefik [1981].

V. Future Work

The techniques we are working on allow a machine tutor to remain flexible while cooperatively engaged in conversation. The goal is to continually adjust the discourse to real-time changes in either the knowledge base or the user by admitting to the possibility of multiple discourse paths arising asynchronously depending on current context. These techniques are being applied toward improving a tutor's abilty to reason about discourse and to select appropriate remediation activities. As part of this work, we are building a visual editor that will allow a knowledge engineer to use screen figures, similar to those in Figure 1 to reconfigure the machine's response.

The goal is to reduce the excessive time needed for building intelligent tutors by providing structures that can be easily refined and rebuilt as new systems are tested. TACTNs are designed to allow a wider circle of authors, e.g., psychologists, teachers, curriculum developers, etc. to participate in the implementation of new intelligent tutors.

VI. Acknowledgement

We thank Klaus Schultz of the University of Massachusetts for his identification of key misconceptions about the crane boom problem.

VII. References

Brown, D., Clement, J., and Murray, T., Tutoring Specifications for a Computer Program Which Uses Analogies to Teach Mechanics, *Cognitive Processes Research Group Working Paper*, Department of Physics, University of Massachusetts, April 1986.

Cerri, S.A., and Breuker, J., A Rather Intelligent Language Teacher, in Hart (Ed.),*Studies in Language Learning*, University of Illinois, Urbana, Ill., Vol. 3, # 1, pp. 182-192, 1981.

Clancey, W., Tutoring rules for Guiding a Case Method Dialogue, in D. Sleeman and J. S. Brown (Eds.), *Intelligent Tutoring Systems*, Academic Press, Cambridge, MA, 1982.

Clement, J., Student's Preconceptions in Introductory Mechanics, *American Journal of Physics*, 50 (1), January 1982.

Grosz, B., and Sidner, C., The Structures of Discourse Structure, *Proceedings of the National Association of Artificial Intelligence*, 1985.

Littman, D., Pinto, J., and Soloway, E., An Analysis of Tutorial Reasoning about Programming Bugs, in *Proceedings of the National Association of Artificial Intelligence*, August 1986.

McDonald, D., Brooks, J.A., Woolf, B., and Werner, P., Transition Networks for Discourse Management , *COINS Technical Report* # 86-34, 1986.

Sacerdoti, E., Planning in a Hierarchy of Abstraction Spaces, in *Artificial Intelligence*, Vol. 5:2, pp 115-135, 1974.

Stefik, M., Planning with Constraints , in *Artificial Intelligence*, Vol. 16, pp. 111-140, 1981.

Stevens, A., Collins, A., and Goldin, S., Diagnosing Student's Misconceptions in Causal Models, in D. Sleeman and J.S. Brown (Eds.), *Intelligent Tutoring Systems*, Academic Press, Cambridge, MA, 1982.

van Lehn, K., Felicity Conditions for Human Skill Acquisition: Validating an AI Theory, *Report Number CSL-21*, Palo Alto, CA, Xerox Palo Alto Research Center, 1983.

Woods, W., Transition Network Grammars for Natural Language Analysis, *Communications of the ACM*, Vol 13:10, pp. 591-606, 1970.

Woolf, B., and McDonald, D., Context-dependent Transitions in Tutoring Discourse, *Proceedings of the National Association of Artificial Intelligence*, Austin, TX, 1984.