International Journal of Artificial Intelligence in Education (2001), 12, 85-99

Bridging the Gap Between Empirical Data on Open-Ended Tutorial Interactions and Computational Models

John Cook, *Learning Technology Research Institute, University of North London, 166-220 Holloway Road, London N7 8DB, UK.*
j.cook@unl.ac.uk

Abstract. In this paper we present an approach to using empirical data on human teacher-learner interactions to guide the development of a pedagogical agent for supporting musical composition learning. Our approach to bridging the gap between tutorial interaction analysis and computational models, intended for use in learning support systems, is a new one. We support our claim by pointing out that most of the previous work in the area of using human tutors as models has been conducted in domains that are more procedural than the open-ended subject area investigated here. However, the approach described in this paper seems applicable to most, if not all, domains, whether open-ended or not. In the paper we describe how an empirical study of teacher-learner interactions was linked, specifically modulated, to the construction of a pedagogical agent called MetaMuse. Empirically derived state transition networks were used to provide a semi-open, goal-oriented interaction plan. One distinctive feature of MetaMuse is its use of student input, regarding whether their expectations were met, to stimulate the pedagogical agent into action. The paper concludes with a discussion of the utility of our approach.

INTRODUCTION

One approach to building Intelligent Educational Systems is to base systems design on the study of human teacher-learner dialogue and interaction. Systems design based on such an approach requires an appropriate teacher-learner interaction analysis technique that is able to bridge the gap between tutorial interaction analysis and computational models intended for use in learning support systems. Dialogue analysis results may lead to important insights that can guide systems development. For example, the results may be used to suggest useful tutoring strategies that can be used in a particular learning situation. However, we suggest that when it comes to using fine-grained dialogue analysis results to develop computational models for supporting learning, most research has tended to investigate more procedural domains. We claim that existing research has not, until now, explored the link between open-ended domain dialogues and computational models.

In an open-ended domain problems are typically ill-defined. A well formalised, closed, more procedural domain would be a subject area like arithmetic, where there are often clearly specified problem-solving goals, correct answers and clearly defined criteria for success at problem-solving. The educational interactions most usually analysed are mathematics, science or computer science. For example, SHERLOCK (Lesgold, Lajoie, Bunzo and Eggan, 1992) is a coached practice environment for electronics troubleshooting in the navy. The SHERLOCK system makes use of empirical data from dialogue analysis in order to inform the types of hints that it gives to learners. Pilkington and Parker-Jones (1996) have explored the instructional domain of medical students interacting with a simulation of calcium homeostasis. They found that inquiry dialogues may be a significant factor in prompting conceptual change. More recently, the AutoTutor system (Graesser, Wiemer-Hastings, Wiemer-Hastings, Kreuz and the Tutoring Research Group, 1999; Person, Graesser, Kreuz and Pomeroy and the Tutoring Research Group, 2001) has been designed to assist college students when learning the

fundamentals of hardware, operating systems and the Internet in an introductory computer literacy course. AutoTutor uses an analysis of human tutors as the basis for its dialogue moves and discourse patterns in a curriculum script. Each topic in a curriculum script is represented as a structured set of propositions in order to enable symbolic computational modelling (Graesser et al., 1999, p. 40–43). Of course, more procedural domains have their own set of challenging problems, but such concerns are not the focus of our paper.

This paper presents our approach to linking interaction analysis to computational models for open-ended domains. The example domain of musical composition is used to illustrate the approach, although the techniques involved in our approach appear to be appropriate to other open-ended, and indeed procedural, domains. In this paper we argue that the use of human expertise, when *modulated,* rather than *transferred,* to the computational medium, is an appropriate starting point for pedagogical agent design in open-ended domains. By modulated we refer to the ability to pass from one state into another using a logical progression. Specifically, a logical progression in our context involves the use of dialogue analysis and modelling techniques to enable aspects of interaction data to be converted into a computational model that can be used in a learning support system. We are not attempting to transfer human expertise, which would involve attempts to computationally simulate, in a cognitive science sense, the human teacher. Instead, our goal is to modulate interaction data into the design of a computer-based pedagogical agent.

The relationship between agent concepts and dialogue in the interaction analysis approach presented in this paper is based on the idea of high-level dialogue (Kiss, 1986), which we will describe below. Briefly, high-level dialogue refers to linguistic and non-linguistic communication that occurs between rational agents pursuing overlapping sets of goals. Furthermore, the aim of our research is to develop a pedagogical agent that can structure tutorial interactions (Winograd, 1988; Baker and Lund, 1997) in a similar way to the approach observed in human teacher-learner interactions. Thus, not only is our agent design based on high-level dialogue concepts, but the pedagogical agent described below also attempts to engage learners in open-ended dialogue. In this paper we describe a set of pedagogical goals involved in a mentoring style of teaching. Mentoring is an approach to teaching that aims to support learners' creative, metacognitive and critical thinking, these being essential to musical composition and other open-ended domains.

In the next section we elaborate on what we mean by open-ended domains. In subsequent sections we describe how an empirical study of teacher-learner interactions was used to construct the computational model in a pedagogical agent called MetaMuse. The paper concludes with a discussion of the strengths and weaknesses of our approach.

OPEN-ENDED DOMAINS

An open-ended domain as an educational setting is where there is typically a lack of clearly defined goals, no single correct solution to a problem and no immediately obvious criteria for making a judgement about what constitutes a correct solution. Andriessen and Sandberg (1999) have commented on a "a shift in attention towards open domains" (p. 3) and have noted that open problems "such as writing an argumentative essay on the prohibition of smoking appear to have more than one acceptable solution" (p. 9). They go on to suggest that in open domains, the learning goal is not fixed but may change during the course of the negotiation process. Indeed, we have already suggested (Cook, 1998b, p. 80–81) that the need for dialogue is especially relevant in open-ended domains such as musical composition learning. Given the open-endedness in music, of both what problems could be addressed, and the space of possible solutions, any educational intervention must be similarly open. Teaching interventions can not be restricted to a simple correct or incorrect response. Thus, open-ended domains usually require some form of open interaction between tutor and student. This shifts the emphasis away from the assertion of facts and towards interactions that encourage the types of creative, metacognitive and critical thinking that have already been reported in detail in Cook (1998b).

In order concretize what we are talking about, Table 1 gives an example of an open-ended domain interaction. The term interaction analysis is used in this paper, rather than the more common dialogue analysis, because interaction includes linguistic communication, i.e. speech, and non-linguistic forms of communication, e.g. musical actions that have communicative intent or actions like pointing at the computer screen. The example shown in Table 1 is a previously unpublished extract obtained in the study described below (see also Cook, 1998a). The tutor is a highly skilled composer and music tutor, the learner is an undergraduate music student.

Table 1. Examples of open-ended domain interaction.

Utterance Number	Agent	Utterance
1	tutor	I mean, if you were, if you were playing this on a fiddle, // where would you put the, and you were, and you were bowing it, // you were using the legato bows // you were playing several phrases in, in each bow. // How would you bow it? (1.8) Where would you put the phrasement bars?
2	learner	Umm.
3	learner	Uh huh.
4	learner	Uh huh.
5	learner	Uh huh.
6	learner	(4.82) Well there would, yeah, there would, there wou:ld be a segmentation there [POINTS WITH FINGER TO 28]=
7	tutor	=There, yeah.
8	learner	(1.7) That, but, that would be a separate phrase on its own=
9	tutor	=It would be a separate phrase=
10	learner	=Yeah=
11	tutor	=Where does it end?
12	learner	At the end of 28.
13	tutor	At the end of 28.
14	learner	Yes because you go [HE SINGS PART OF THE PHRASE FOR 3.7 SECONDS]=
15	tutor	= Oh right>. So 28 is the end of the phrase, you see that as building up to that do you? Or coming away from that?
16	learner	No, I see it as, as something totally different, not part of the // phraseology.
17	tutor	*Not part, ahh right.*=
18	learner	Do you see what I mean, so that [POINTS TO 28] (phrase stands by) itself. =[
19	tutor	So this, is it, almost, you know like in a Bach's suite having a, like, like, like in a solo suite where you have a, perhaps a figure playing at a different octave, which gives the impression of there's another voice?
20	learner	Yeah, I (5.0) I, I see, I see [USES TWO FINGERS TO POINT AT 28 AND 24] them as, as if they had been put on top, as if they // are something else=
21	tutor	Right.
22	tutor	=So they're something else.
23	tutor	Yeah.
24	learner	Which they are, because they're, they are higher than everything else.
25	tutor	Right.

We need to explain a few points about the transcription method used in our work, which is illustrated in Table 1. The utterances are transcribed exactly as they were spoken. Protocol transcription was based on the approach to Conversational Analysis used by Fox (1993). The // symbol indicates the place at which a speaker's utterance is overlapped by an utterance from another speaker. Utterances that are overlapped more than once have more than one double slash in them, and the utterances that do the overlapping are given in sequential order after the overlapped utterance. We use a [symbol at the beginning of a line to indicate that two

utterances, the ones above and below the symbol, began simultaneously. We use the = to indicate latching, which is where the next speaker begins without the usual beat of silence after the current speaker finishes talking. In this case there is an equal sign at the end of the current speaker's utterance and another at the beginning of the next speaker's utterance. A =[indicates that the utterances above and below simultaneously latch, i.e. agents talk at once. [] brackets with capitalized words enclosed represents non transcribed material, e.g. music or noises that are non-linguistic. > after a word indicates rising, but not terminal rising, spoken intonation, as is often found at the end of each member of a list. Numbers given in parentheses indicate elapsed silence, measured in tenths of a second. Only pauses of (0.7) upwards were recorded. Punctuation is used to suggest intonation; italics indicate stress. A colon after a letter means that the sound represented by that letter is somewhat lengthened; a series of colons means that the sound is increasingly lengthened. Questionable transcriptions are encoded within parentheses, e.g. it is not clear who made the utterance or what is being said.

In the interaction extract shown in Table 1, the numerical values that are not in round parentheses refer to musical phrase values, e.g. "At the end of 28" at utterance 12. An explanation of these numerical musical phrase values will be given below, an understanding of them is not required for the purposes of this section. Table 1 gives a clear example of an open-ended domain interaction between tutor and student. Before the interaction shown in Table 1 took place the learner had created a musical phrase in response to a task set by the tutor. This new phrase has now become the focus of discussion. At utterance 1 in Table 1 the tutor is attempting to get the learner to explain what his creative intention is. There is no one correct solution to the tutorial task. In fact it is not until utterance 17, following an explanation by the learner of his musical intention, that the tutor seems to understand what the learner had in mind. This prompts the tutor to provide an example of a similar musical idea at utterance 19. Much of the interaction shown in Table 1 thus involves the tutor and the learner in a negotiation that takes the form of them coming to a common understanding of what the learner's musical intention or goal is and an exposition of how this new musical intention compares to compositional methods used by other composers.

EMPIRICAL STUDY OF OPEN-ENDED TUTORIAL INTERACTIONS

Two possible approaches to designing computer-based pedagogical agents are theory-intuition to design-evaluate-and-refine, and from human-teacher-student-dialogues and theory to design-evaluate-and-refine. We choose the second approach, because, given that open-ended domains have not yet been adequately formalized, our intuitions can not be sufficiently informed. It is therefore reasonable to begin from what human experts do, to modulate this knowledge to system design and refine the system on the basis of evaluation. Because most of the previous work in the area of using human tutors as models has been conducted in domains that are more procedural than the open-ended subject area investigated here, an empirical study was conducted that investigated the way in which a human teacher supported higher-order, musical thinking in learners. This empirical study has already been reported in detail in an earlier IJAIED article (Cook, 1998b); we will not, therefore, repeat this information here. Instead, we will focus on those aspects of the empirical work that were eventually modulated to the computational model in a pedagogical agent called MetaMuse.

The empirical work involved seven detailed analyses of the corpus collected, i.e. transcriptions of the interaction data. With the benefit of hindsight, we can report that analysis one, five and six provided results that were the most usable in subsequent pedagogical agent construction. Consequently, we will focus on briefly reporting these results below.

The first interaction analysis involved a categorization of the study data, a corpus of transcriptions of the mentoring sessions, into pedagogical goals, sub-goals and communicative acts (Gazdar, 1981; Baker, 1994). The pedagogical goals were creative thinking, metacognitive interventions, metacognitive thinking, critical thinking, motivation and the task goal. In our work we have focussed on high-level interaction, which is concerned with the overall goal

structure of interactions. Specifically, the term high-level dialogue (Kiss, 1986) refers to linguistic and non-linguistic communication that occurs between rational agents pursuing overlapping sets of goals in a task domain. In our analysis framework teacher and learner have their own set of goal trees that formalize the goals that they can pursue and the communicative acts that can be used to realize a goal. Analysis one generated frequency counts for each category of goal and communicative act (see Cook, 1998b, p. 66). Some of the sub-goal categories will be described in the discussion below. In the fifth analysis we attempted to pull out a macro, high-level view of interactions of the categorized data produced by analysis one. The sub-goals for each session were analyzed to see if any patterns or stages within a session could be detected. The sixth analysis involved the mapping of various state transition networks to represent the sequence in which goals and sub-goals were pursued in interactions. Analysis six took as its starting point the result from analysis five, i.e. the seven mentoring stages, which are described briefly below.

Analysis one showed that in the open-ended domain of musical composition, creative problem-solving interactions had the underlying sub-goals of *probing, target, clarify* and *give reasons*. Specifically, the results of our empirical work indicated that the two most frequently used teaching interventions related to creative problem-solving were as follows. First was critical *probing,* where focused questions were used by a teacher, questions which were based on observations of a student's musical phrase; these questions had the intention of either helping the learner find an interesting problem to work on, or of prompting the learner to elaborate on their creative intention. Second was *target* monitoring or reflection level; this was open-ended questioning that involved the teacher's attempts to elicit verbal self-explanations from the learner about their own attempts at specifying an interesting problem.

A result from the fifth analysis was evidence for seven mentoring stages, which can be briefly summarize as: open the session, introduce the task, initial use of target monitoring or reflection level, teacher led creative and critical thinking, more target monitoring or reflection level, return to teacher critical probing and end the session. These seven mentoring stages represent an empirically based plan for a mentoring session.

Since a detailed elaboration of these seven mentoring stages could prove an invaluable resource as a computational model in a pedagogical agent, analysis six included an in-depth exploration of each stage. State transition networks were generated for all seven stages of mentoring identified in analysis five. Unlike the sub-goals used to achieve transitions in mentoring stages 1 and 7, i.e. open session and end session, which are relatively simple, the networks for mentoring stages 2 to 6 are more complicated. The seven state transition networks are all empirically derived, i.e. they represent all the sub-goals identified by analysis one, with a few exceptions, which were always noted. That is to say, the sequencing of sub-goals identified in analysis one, for all four sessions, is directly represented in the networks. We will expand on this point below by highlighting the relationship between the interaction data from the study and our state transition networks.

In our use of state transition networks, which is adapted from Winograd and Flores (1986), the course of an interaction can be plotted using circles to represent a possible state of the agent, e.g. a decision point, and lines/arcs to represent sub-goals, Winograd and Flores only represent speech acts. The lines thus indicate sub-goals that can be taken by the teaching agent and the learning agent. Each sub-goal in turn leads to a different state, with its own space of possibilities. Woolf and co-workers (Woolf, Murray, Suthers and Schultz, 1988) have already used Tutoring Action Transition Networks as a control tool for facilitating the specification and modification of prototypical patterns of tutorial behaviour. The tool is intended to be used for eliciting discourse patterns from domain experts whilst building an Intelligent Tutoring System (ITS) for science education. Arcs, i.e. arrows, in Woolf et al.'s approach represent predicates which track the state of the dialogue, e.g. simple slip, incorrect answer. Nodes represent a tutor's action, or an entire network recursively invoked, e.g. teach by consequence or teach by example. Our approach is slightly different to Woolf et al.'s in that we use arcs to represent sub-goals, which may lead to action or communicative acts. The nodes in our approach represent a state at which a decision is made about which transition should be selected next.

Our networks are then embedded in a computational model that aims to structure interactions with a learner (we return to this point in the next section). The network for stage 3 has already been published in Cook (1998b, p. 73) and all seven networks and associated discussions are available in Appendix 8 of Cook (1998a).[1]

Table 2 gives a small example of part of stage 2, i.e. introduce the task, taken from the start of session 3 of our empirical study; it is included here as a small example of how the study interaction data is related to our networks. One could take any dialogue extract from our study corpus and step through the relevant network on a turn-by-turn basis, i.e. our networks describe the high-level dialogue, as sub-goals, contained within the tutorial interactions of the four sessions that were observed in the study. Table 2 shows some initial probing by the teacher into the learner's previous experience.

Table 2. Example interaction at sub-goal level for stage 2 network.

Agent	**Utterance**	**Route through the network (initial state→sub-goal→end-state)**
tutor	Can you emm, I want you to try and recall how you started off that exercise. Did you pick the four notes at random? Or did you choose them?	state 3 → critical probing → state 4
learner	I picked the four notes a random by just by clicking on the keyboard down the left-hand side. [THIS REFERS TO THE SCREEN OF THE SEQUENCER SOFTWARE]	state 4 → critical clarification → state 5
tutor	Right	state 5 → motivation encouragement → state 3
tutor	So there was no, thinking carefully sort of carefully 'oh this is, you know, this is going to be a motif?'	state 3 → critical probing → state 4
learner	No just picked the notes.	state 4 → critical clarification → state 5

The exercise referred to in the first tutor utterance shown in Table 2 is a piece of class work that the tutor had set the learner some weeks earlier. Table 2 also shows that the tutor is trying to find out if the learner has in the past been composing in an intentional manner. At the bottom of Table 2 the learner makes it clear that he had no creative plan behind his choice of musical notes when he performed the earlier class exercise. The first state transition shown for stage 2 in Table 2, i.e. state 3 → 4, can be achieved by the two communicative acts, request and question, which are shown in Table 3. Note that the primary act (Cook, 1998b, p. 13) request moves the agents to state 3.1 and question moves the agents from state 3.1 to state 4.

Table 3. Communicative acts used to achieve critical probing.

Agent	**Utterance**	**Acts used to move from state 3 to 4**
tutor	Can you emm, I want you to try and recall how you started off that exercise.	state 3 → request → state 3.1
tutor	Did you pick the four notes at random? Or did you choose them?	state 3.1 → questions → state 4

Above we have argued that in open-ended domains it is reasonable to begin from an understanding of what human experts do, and then to modulate this knowledge to system design. Two important result of our empirical study of human tutoring were seven mentoring stages and the set of state transition networks for the seven mentoring stages. In the next section

[1] Appendix 8 can be obtained from http://www2.unl.ac.uk/~exbzcookj/appendix8

we will present our approach to using these findings from our empirical study in the computational model of a pedagogical agent designed for supporting learning in the open-ended domain of musical composition.

COMPUTATIONAL MODEL

As we point out above, our approach to bridging the gap between tutorial interaction analysis and computational models, intended for use in learning support systems, is a new one. We have supported our claim by demonstrating that most of the previous work in the area of using human tutors as models has been conducted in domains that are more procedural than the open-ended subject area investigated here. In this section we describe the overall design of our pedagogical agent, called MetaMuse, and describe how the empirical data outlined above was modulated to a computational model for structuring open-ended learning interactions. Empirically derived state transition networks were used to provide a semi-open, goal-oriented interaction plan. Following a comparison of our approach to a standard problem-solving system we conclude this section by briefly describing the implementation of MetaMuse.

Overall Design

Following Kiss, Domingue and Hopkins (1991) our pedagogical agent, which is called MetaMuse, is characterized in terms of what it perceives, what it knows, what kind of reasoning it can do, what values and goals it has, and what actions it can take. These five characteristics essentially define the architecture of our artificial agent, i.e. an agent possessing these characteristics will be able to react to the environment and will be capable of acting in an autonomous manner. A value cannot be achieved or abandoned like goals can. Values are persistent. In this paper the agent value represented is pedagogical, concerned to be mentoring well, as defined by the analysis of human tutoring. We acknowledge that an agent can have values without feeling the need to take action. However, because our approach is embedded within the theory of agency and action proposed by Kiss (1986) we retain values as the antecedent of an agent's goals rather than, for example, using desire. A goal is a state of the world which an agent explicitly wants to achieve, preserve, avoid, or destroy. Below we briefly describe the way in which these characteristics were instantiated in the computational model of the pedagogical agent, i.e. how we used empirical data in our agent design.

The sensory inputs to the pedagogical agent, essentially information about the environment, comes from two sources. First, the student makes an input into the pedagogical agent in the form of a list of numbers to transpose a four note musical pattern. Transposition is the process of moving a melodic figure, section or composition up or down in semi-tones. The task that is used in our system is one of asking the learner to generate a musical phrase by transposition of a pre-set four note pattern. Slonimsky (1947) pattern No 1, which is C C# F# G, is given by the system at first. No alteration of the rhythm was allowed, although such a possibility could be discussed. The only input required to play the music is a string of transposition numbers, with a number representing the amount of chromatic transposition in semi-tones. For example, the list 0 1 would generate a short musical phrase of C C# F# G D D# G G#. The second type of sensory input comes into the system through the interface. The interface is used to structure the learner's interactions by presenting menu options, buttons and data input boxes; these are used to present a restricted number of sub-goals and communicative acts and to thus enable interaction.

These two types of input are subject to a variable degree of perceptual processing, e.g. the learner input list is analyzed to see if there is evidence of large interval jumps, which may form musical phrase boundaries. Perceptual processing results in the pedagogical agent building a representation of environmental knowledge, in the example given, it also represents an attempt to infer the learner's musical intention with respect to their phrase. An Interaction History is also built up by a perceptual processor. An Interaction History is a summary of goals and sub-

goals used with a learner and a summary of both types of sensory input. The pedagogical agent has a simple User Model reasoner, a perceptual processor. Our User Model is an abstraction over the Interaction History, it does not include mechanisms for belief maintenance.

What the pedagogical agent 'knows' is very limited. That is to say, the agent's knowledge-base of musical information is small. The focus for the pedagogical agent is the potential that the choice of musical intervals provides for musical material to be sectionalised into phrases. Heuristics about how to teach this topic have been taken from the empirical study described above and augmented by some musical grouping constraints as proposed by Lerdahl (1988). Thus, the pedagogical agent is capable of certain types of symbolic reasoning based on its knowledge. For example, the pedagogical agent knows that 'an interval leap of 23 or more may indicate a phrase boundary'. The agent also knows some good teaching tactics, i.e. sub-goals; for example, a tactic for the sub-goal critical probing is that of getting the learner to imagine that they are playing their phrase on an instrument, as this can help the learner firm up their own perception of the purpose of the interval leap, in metacognitive terms go-above and regulate their knowledge (for an example see utterance 1 in Table 1). This gives a good example of the challenge of mentoring in an open-ended domain, i.e. mentoring involves tutoring with non-absolute rules, where there is no right or wrong answer. Essentially, we have built into our agent a lot of knowledge relating to approaches for structuring interactions in an educationally meaningful manner, which compensates for its small musical knowledge-base.

In order to participate in a mentoring interaction, the agent must be able to generate utterances and actions that it believes can satisfy its values. Thus, the design of the pedagogical agent consists of "defining the computation that leads from sensory inputs to knowledge; and leads from values/goals and knowledge to action" (Kiss et al., 1991, p. 3). The design of the pedagogical agent is modular, after Kiss et al. (1991). A description of the design of each module is beyond the scope of this paper (see Cook, 1998a). In the computational model of our pedagogical agent an action cycle, shown in Figure 1, is used to determine what action the pedagogical agent is to take at each time increment of the current situation.

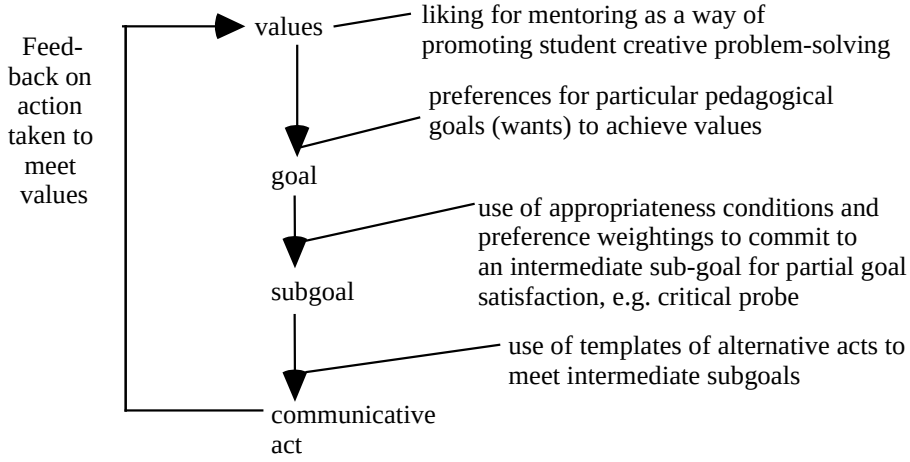

Figure 1. Pedagogical agent action cycle.

Our action cycle is similar to Blandford's (1994) with the important difference that we are attempting to use empirically derived networks, whereas Blandford does not adopt this approach. When it is the pedagogical agent's turn, it generates a list of sub-goals *(wants)* that are available to it at the current node in a network, there is a requirement in our action cycle for the networks, which are used for the planning of mentoring interactions, to be empirically derived from interaction analysis. A want is an attitude towards some proposition or world state, e.g. a goal. An agent may want to achieve all of its goals. Wants refer to a list of actions

an agent might be willing to be committed to, commitment refers to the chosen way of achieving an action.

Empirically derived state transition networks

Empirically derived state transition networks were used to provide a semi-open, goal-oriented interaction plan. The empirical source was the study described above. State transition diagrams are used instead of, for example, production rules because we were attempting to stick as closely as possible to the empirical results which gave transition possibilities but not rules. However, if more than one exit existed for a node then a preference mechanism was used which contained a set of conditions that governed the circumstances under which a particular transition should be considered for use (see Cook, 1998a).

The second, third and fourth stages of mentoring, described above, were selected to explore the implementation problems involved in adapting the state transition networks to a computational model. These state transition networks were used as the basis for the structuring of interactions. The nodes represent a state at which a decision is made, by either the learner or teacher, about which transition should be selected next, assuming a choice is available. The networks are used by the pedagogical agent to provide means-ends beliefs about which sub-goals satisfy a particular mentoring stage or state node. Often, more than one exit is possible from a state node. For example, in Figure 2 at node 26 there are three possible learner (LA) exit transitions and one teacher (TA) exit transition. We chose to structure interactions by offering these options to a learner as a menu, the options would vary from one node to the next. Assuming it is the learner's turn then at node 26, which is part of mentoring stage 4 and which is shown in Figure 2, our system will place the three possible exits in a menu (we return to this point below). Note that 'target M or Ref' in Figure 2 means target monitoring or reflection level.

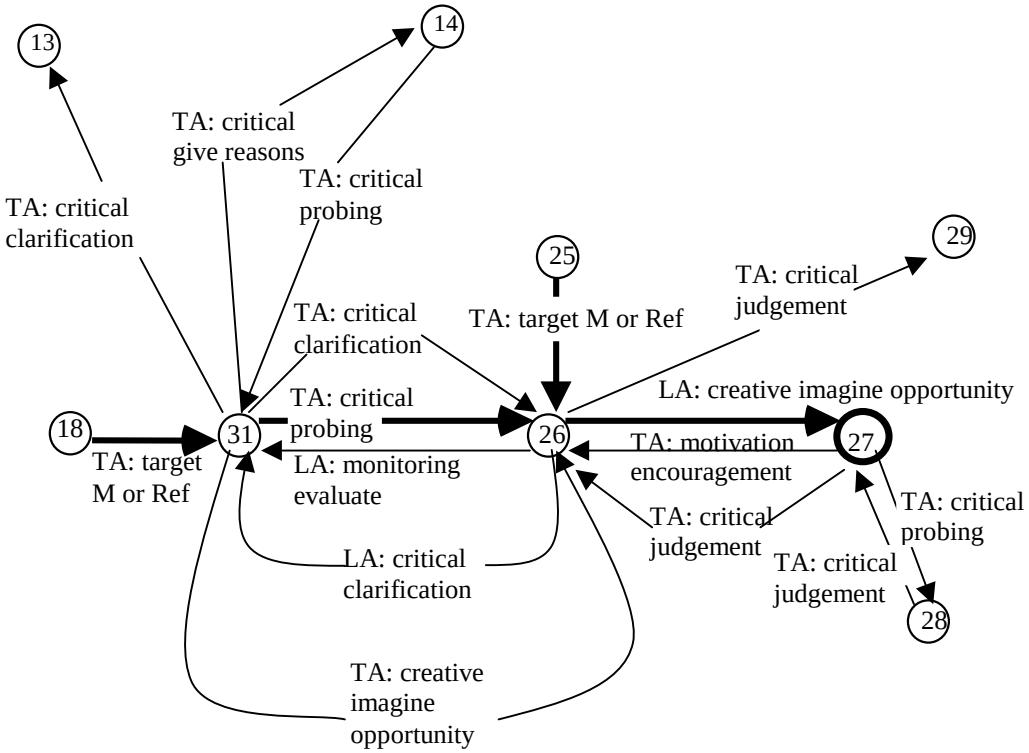

Figure 2. Extract from stage 4 network for mentor led creative and critical thinking.

A sample analysis is given in Table 4 in order to illustrate the way in which the implemented agent MetaMuse uses the state transition networks to structure interactions with a learner.

Table 4.† Example of MetaMuse-Learner interactions

Agent turn-utterance	**Sub-goal (turn)**	**CA (utterance)**	**Other (actions)**	**State transition and commentary**
TA1: Good, you seem to be getting the hang of this.	motivation encourage-ment	assertion		state 10 → state 12 (from a network not given in this paper, we then get a programmed jump to state 31)
What do you see the function of the very large leap, (12 -20) , as? . Select an option from the Respond menu.	critical probing	question dialogue management		state 31 → state 26
LA2:	critical clarification		selects 'Answer the question'. from Respond menu	state 26 → state 31
TA3:		dialogue management	Puts up an Interaction Box .	
LA4:	<above LA2 sub-goal still active>		selects button 'Make a statement about your phrase'.	
TA5: Assert your statement below'		dialogue management	Puts up a data entry box with prompt 'Assert your statement below'.	
LA6: 1. the pattern could not continue upwards and 2. functionally to return to the starting point	<above LA2 sub-goal still active>	assertion assertion	Enters utterance, returns it, then selects button 'Move on'.	

† TA = PEDAGOGICAL AGENT (METAMUSE); LA = LEARNING AGENT;
CA = COMMUNICATIVE ACT

The extracts shown in Table 4 are taken from a real example generated by a participant in an evaluation of MetaMuse (see Cook, 2000a; Cook, 2000b). In Table 4 an explicit link is made between how the networks from our empirical work were used to structure interactions in the implemented pedagogical agent. This in turn provides detail of the fine-grained means by which we have bridged the gap between our analysis of open-ended tutorial interactions and our computational model. All utterances by the pedagogical agent (TA) in Table 4 appear as they are output by MetaMuse, i.e. the text has not been edited. Note that the network nodes referred to in Table 4 are given in Figure 2, unless otherwise stated.

Prior to the example shown in Table 4, the learner (actually an experienced teacher-composer) had entered a transposition list of '0 8 16 12 -20 0 2 0 1 0 8 16' and returned it. Following a request by MetaMuse, the learner had made the following prediction about their

phrase: “moving pattern up through octave and above, then returning to base only to begin again”. Following playback of the phrase, the learner makes it clear that he is happy with the way the phrase sounded when it was played back by answering “Yes” to the question from MetaMuse “Was it what you expected?”. MetaMuse then invites the learner to say why they think it worked. The learner gave the reason “reasonably accurate planning”. The interactions shown in Table 4 then took place.

The interaction shown in Table 4 shows us how the learner-MetaMuse interactions progresses through a series of transition states, which are indicated in the fifth column. Prior to the critical probing sub-goal at TA1 in Table 4, MetaMuse will have used a perceptual processor to analyze the learner’s phrase and decided that it uses a large leap. So at TA1 we get an agent intervention aimed at trying to see if the learner is composing in an intentional manner. At LA2 the learner provides critical clarification in response to the pedagogical agent’s probing and makes it clear that he is in fact composing in an intentional manner. The interaction then went on to include more critical probing by the pedagogical agent and critical clarification by the learner. The interactions in Table 4 gives a good example of what we are calling an open-ended domain dialogue. The dialogue between teacher and learner was about probing what the learner’s intentions were and about clarifying the concepts that underlie a learner’s creative plans. Thus, the networks derived form our own empirical study were used to give learners options for choice. Furthermore, the networks give the pedagogical agent a principled position from which to mentor; principled in that the networks represent a descriptive model of what one teacher actually did with four students.

Comparison with more standard problem-solving system

The design of our pedagogical agent was partially based on the results, described above, obtained from the analysis of empirical data. One link with the study described above is that like the human teacher, our pedagogical agent has a preference for the adoption of the two sub-goals critical probing and target monitoring and reflection levels. The implicit intention behind the adoption of these sub-goals being to get the learners to (i) make a prediction about their creative intentions before hearing their short musical phrase played back to them, and (ii) explain the musical outcome when matched to their prediction. For example, let us assume that the teacher (TA) has decided to take one of the transition, shown in Figure 2, between states 31 and 26, i.e. 31 → TA: critical probing → 26. At node 26 it is now the learner’s (LA’s) turn. The learner has three possible exit transitions, which are 26 → LA: creative imagine opportunity → 27 or 26 → LA: monitoring evaluate → 31 or 26 → LA: critical clarification → 31. If the learner has already evaluated their phrase at an earlier point in a session, then at 31 → 26 the teacher will have decided to make a critical probing intervention. Such an intervention would be along the lines of “What do you see the function of the very large leap”, which is shown in Table 4 at utterance TA1. In this example, at TA1 the agent is attempting to direct the learner’s subsequent response down the transition 26 → LA: critical clarification → 31, i.e. to get the learner to explain their creative intention.

In our computational model we are essentially using the student input, regarding whether their expectations were met, to stimulate the pedagogical system into action. One important difference between MetaMuse and other systems is that the state descriptors, current and preferred, depend on the user’s stated expectation, e.g. whether the notes sounded as they expected. MetaMuse does not rely on its own assessment of the user’s expected state, it deals with the learner’s own assessment of their expected state. MetaMuse may try to move the user to explain if their expectation was met, and if it was met to say why it worked, and so on. There is more than one possible answer to the musical problem task, the emphasis in our system is to get the learner to detect or explain the difference between the current and expected state. This approach is different to a more standard problem-solving system. In order to clarify this difference below we now briefly describe a more standard problem-solving system.

Andes is an ITS for introductory college physics (Gertner and VanLehn, 2000). In this system the tutor and the student collaborate to solve problems. When a student gets stuck or makes an error, the Andes system detects this and provides hints aimed at bringing the student back onto the correct solution path. In Andes a student is not constrained to performing actions in a particular order, i.e. the problem-solving path of the student may not follow a set pattern. The Andes student model copes with this by combining information about the current state of the problem-solving process with a long term assessment of the student's knowledge of physics using probabilistic reasoning techniques. Hints are generated by selecting a node in a predefined solution graph, these nodes form the basis for the topic of the hint. Thus, the system makes an assessment of where it thinks a learner is in terms of a predefined solution graph and then selects a node that the system thinks the learner is trying to get to, i.e. the system arrives at an assessment of the learner's expected state.

The approach used in Andes and similar systems works quite well for procedural domains like physics where it is possible to generate model answers in the form of solution graphs. However, this technique is not always appropriate for more open-ended domains. In our system the user detects the difference between their current state and expected state after hearing musical notes played back. There is no predefined solution graph. The system is trying to encourage the student to make this comparison between current state and expected state with such interventions as "Was it what you intended?" "What is the purpose of the large interval leap?" However, we do have predefined teaching networks that form the basis of how the agent would prefer to intervene in order to encourage the learner to make these comparisons between different states.

Implementation

MetaMuse cannot understand natural language. The system is not able to engage in dialogue about the free text inputs by the learner. As a result the system sometimes lets the user input some thoughts on what they are doing, stores these explanations and reflections in the Interaction History, this is then abstracted into the User Model. MetaMuse then moves on to the next part of the session. Interactions with MetaMuse thus typically take the form of structured questioning by the system. However, within certain limitations, the system is able to comment on a musical phrase input by the learner.

The implementation was carried out by coding the pedagogical agent in Macintosh Common Lisp version 2.0.1 on a Centris 660av — this is the version of Lisp that Symbolic Composer (Morgan and Tolonen, 1995) for the Centris uses. Code can be loaded under Symbolic Composer, which handles Midi files and contains routines required for transposition of musical patterns. Thus, MetaMuse runs under Symbolic Composer. An example of the interface for the pedagogical agent is shown in Figure 3. The MetaMuse interface structures the interactions between the pedagogical agent modules and the learner by providing a series of menu options and buttons which are required to support mentoring. When a learner selects a menu option or clicks on a button, further text windows or dialogue boxes for input are displayed. The construction of utterances by the learner is thus achieved by the use of menus, buttons and dialogue boxes.

Figure 3 is a screen-shot of the interaction shown in Table 4 at utterance LA2.

In the current implementation of MetaMuse, if it is the learner's turn, then depending on the current state node reached in a network, the Respond menu will display all exit transitions, as defined by the relevant network, for the learner for that node. The contents of the Respond pull-down menu shown in Figure 3 are the learner exits for node 26, which is shown in Figure 2. The three options are: (i) Explain your creative idea, which would take transition 26 → LA: creative imagine → 29 opportunity in Figure 2, (ii) Answer the question, which would take transition 26 → LA: critical clarification → 31 in Figure 2 and (iii) Assert your evaluation of the phrase, which would take transition 26 → LA: monitoring evaluate → 31 in Figure 2.

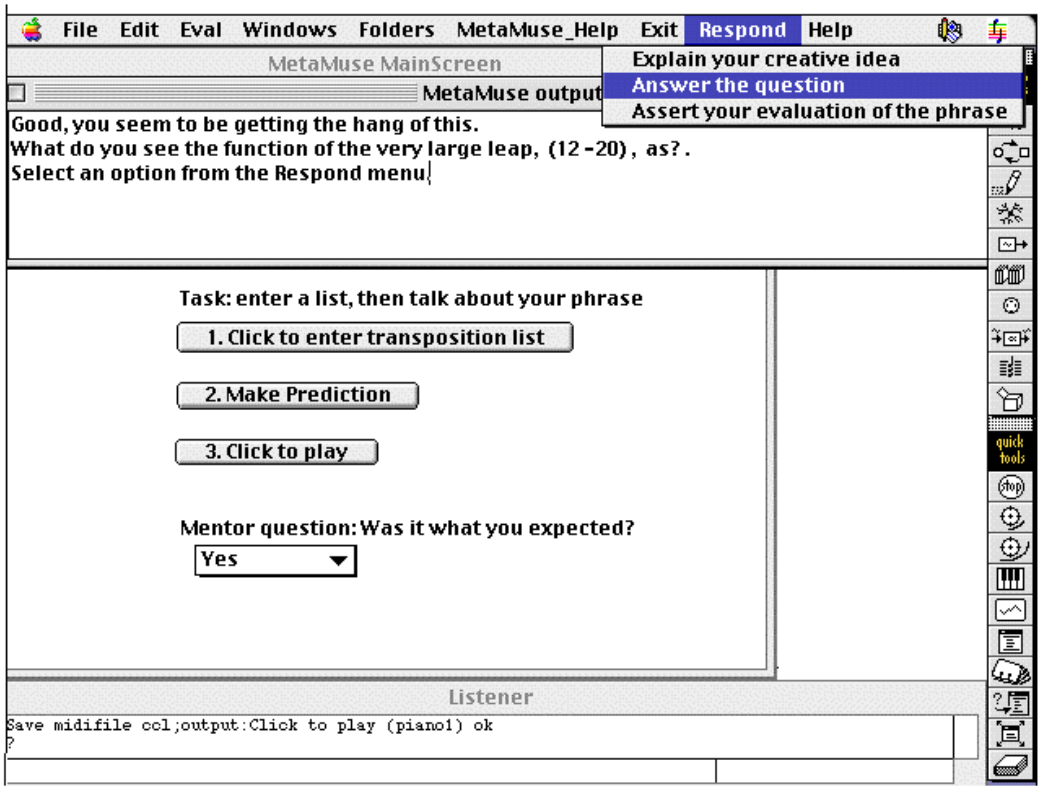

Figure 3. Example of MetaMuse interface

The Respond menu is the main focus of decision making by the learner. The idea that the contents of a menu will change from one turn to the next is not what a user might usually expect. However, users did have this fact pointed out to them at the start of a session in an evaluation of MetaMuse (Cook, 2000a; Cook, 2000b). The use of the Respond menu in this way did not appear to be a usability issue in the evaluation. This is probably because the pedagogical agent is often able to use its decision mechanisms and user model, described Cook (1998a), to give a recommendation about what option it thinks the learner should select.

We conclude this section by claiming that the use of human expertise, when converted to the computational medium, is an appropriate starting point for pedagogical agent design in open-ended domains. We have supported this position by describing how an empirical study of teacher-learner interactions was modulated to the construction of a pedagogical agent called MetaMuse.

CONCLUSIONS

This paper has described how empirical data, from an analysis of open-ended tutorial interactions, has been successfully used to help develop the computational model for a computer-based pedagogical agent. The distinguishing feature of our approach is the use of descriptive dialogue models derived from empirical data, i.e. state transition networks. These models are used as the starting point for interaction planning by a pedagogical agent that can support musical creativity through structured interactions.

The approach described in this paper seems applicable to most, if not all, domains, whether open-ended or not. An implicit claim of our work is that teaching can be usefully modelled as a general process, and that these models can be applied to a specific content area. The following question now arises as a consequence of this claim. Does the current system create some

senseless dialogue or pose some inappropriate questions? For example, sub-optimal dialogue may arise because the pedagogical agent is insensitive to actual domain content or aspects of the interactions. The simple answer to this question in the context of MetaMuse is yes, sometimes. Although an evaluation of MetaMuse (Cook, 2000a; Cook 2000b) drew a generally favorable response from teachers and potential students, there were occasions when evaluation subjects complained that the system had not paid attention to an earlier comment they had made. Essentially, any sensitivity to a learner's utterance is more or less hard-wired into the system by the programmer, i.e. the author of this paper. Although our system does make use of a decision mechanism that tries to make responses that are in part based on earlier interactions, the system can not understand the content of a learner's textual inputs. However, within certain limitations MetaMuse is able to understand musical inputs from a learner and can respond in a pedagogically meaningful manner. Future work will attempt to overcome the problem of insensitivity to earlier dialogue by making use of the Latent Semantic Analysis techniques that have already been used with some success in AutoTutor (Graesser et al., 1999). However, it remains unclear as to whether Latent Semantic Analysis can deal with more open-ended tutorial dialogues like those that have been described in this paper.

Acknowledgements

Most of the work described in this paper was conducted as part of the author's Ph.D. work (Cook, 1998a) under the supervision of Michael Baker. Some of the code for MetaMuse is an adaptation of Ann Blandford's WOMBAT. Thanks for sharing and explaining the code Ann. Thanks also to Tom Boyle, Michael Baker, the anonymous reviewers and the editor for making useful comments on drafts of this paper.

References

Andriessen, J. and Sandberg, J. (1999). Where is Education Heading and How About AI? *International Journal of Artificial Intelligence in Education*, **10**, 130–150.

Baker, M. J. (1994). A Model for Negotiation in Teaching-Learning Dialogues. *Journal of Artificial Intelligence in Education*, **5(2)**, 199–254.

Baker, M. J. and Lund, K. (1997). Promoting Reflective Interactions in a CSCL Environment. *Journal of Computer Assisted Learning*, **13**, 175–193.

Blandford, A. E. (1994). Teaching Through Collaborative Problem Solving. *Journal of Artificial Intelligence in Education*, **5(1)**, 51–84.

Cook, J. (2000a). Cooperative Problem-Seeking Dialogues in Learning. In G. Gauthier, C. Frasson and K. VanLehn (Eds.) *Intelligent Tutoring Systems: 5th International Conference, ITS 2000 Montréal, Canada, June 2000 Proceedings*, p. 615–624. Berlin Heidelberg New York: Springer-Verlag.

Cook, J. (2000b). Evaluation of a Support Tool for Musical Problem-Seeking. *Proceedings of ED-Media 2000 — World Conference on Educational Multimedia, Hypermedia & Telecommunications*, p. 203–208, held in Montréal, Canada. AACE.

Cook, J. (1998a). *Knowledge Mentoring as a Framework for Designing Computer-Based Agents for Supporting Musical Composition Learning*, Unpublished Ph.D. thesis, Computing Department, The Open University, UK. (Contact author for a PDF copy.)

Cook, J. (1998b). Mentoring, Metacognition and Music: Interaction Analyses and Implications for Intelligent Learning Environments. *International Journal of Artificial Intelligence in Education*, **9**, 45–87.

Fox, B. A. (1993). *The Human Tutorial Dialogue Project: Issues in the Design of Instructional Systems.* Hillside, NJ: Lawrence Erlbaum.

Gazdar, G. (1981). Speech act assignment. In A. Joshi, B. Webber and I. Sag (Eds.), *Elements of Discourse Understanding*, p. 64–83. Cambridge: Cambridge University Press.

Gertner, A. S. and VanLehn, K. (2000). Andes: A Coached Problem Solving Environment for Physics. In G. Gauthier, C. Frasson and K. VanLehn (Eds.) *Intelligent Tutoring Systems:*

5th International Conference, ITS 2000 Montréal, Canada, June 2000 Proceedings, p. 133–142. Berlin Heidelberg New York: Springer-Verlag.

Graesser, A., Wiemer-Hastings, K., Wiemer-Hastings, P., Kreuz, R., & the Tutoring Research Group (1999). AutoTutor: A simulation of a Human Tutor. *Journal of Cognitive Systems Research,* **1**, 35–51.

Kiss, G. (1986). *High-level Dialogue in Man-Machine Interaction.* HCRL Technical Report No. 44, Knowledge Media Institute, The Open University, UK.

Kiss, G., Domingue, J. and Hopkins, C. (1991). *A Multi-Agent System for Telephone Network Management.* HCRL Technical Report No. 72, Knowledge Media Institute, The Open University, UK.

Lerdahl, F. (1988). Cognitive Constraints on Compositional Systems. In J. Sloboda, Ed. *Generative Processes in Music: The Psychology of Performance, Improvisation and Composition.* New York, NY: Oxford University Press.

Lesgold, A., Lajoie, S. P., Bunzo, M. and Eggan, G. (1992). SHERLOCK: A Coached Practice Environment for an Electronics Troubleshooting Job. In J. H. Larkin and R. W. Chabay (Eds.) *Computer-Assisted Instruction and Intelligent Tutoring Systems,* p. 201–238. Hillsdale, NJ: Lawrence Erlbaum.

Morgan, N. and Tolonen, P. (1995). *Symbolic Composer Professional — a software application for Apple Macintosh computers.* Amsterdam: Tonality Systems.

Person, N. K., Graesser, A. C., Kreuz, R. J., Pomeroy, V. and the Tutoring Research Group. (2001). Simulating human tutor dialogue moves in AutoTutor. *International Journal of Artificial Intelligence in Education,* 12(1), this issue.

Pilkington, R. and Parker-Jones, C. (1996). Interacting with Computer-Based Simulation: The role of Dialogue. *Computers and Education,* **27(1)**, 1–14.

Slonimsky, N. (1947). *Thesaurus of Scales and Melodic Patterns.* Collier MacMillan.

Winograd, T. (1988). A Language/Action Perspective on the Design of Cooperative Work. *Human-Computer Interaction,* **3**, 3–30.

Winograd, T. and Flores, F. (1986). *Understanding Computers and Cognition: A New Foundation for Design.* Norwood, NJ: Ablex.

Woolf, B., Murray, T., Suthers, D. and Schultz, K. (1988). Knowledge Primitives for Tutoring Systems. In *Proceedings of the International Conference on Intelligent Tutoring Systems (ITS–88),* p. 491–498, held in Montréal, Canada.