CHAPTER 4

CURRICULUM AND INSTRUCTION IN AUTOMATED TUTORS

Henry M. Half
Chief Scientist
Half Resources, Inc.

People learn many things without benefit of instruction, but we are distinguished as a species by our ability to pass knowledge from the competent to the less competent. To endow machines with this same instructional ability is, to a large extent, to cast the principles of instruction in precise information-processing terms. This chapter assesses the progress that has been made on one important aspect of this task, namely, that of codifying the principles of tutoring.

Intelligent Tutoring Systems

This chapter is concerned with only one genre of instruction, tutoring\(^1\), and with only one design approach to this instruction, that based on artificial intelligence technologies. It is necessary therefore to say a bit more about what constitutes an intelligent tutor from an instructional point of view.

**Tutoring.** Tutors can use many different instructional techniques, but tutorial interactions, however they are conducted, must exhibit three characteristics:

1. A tutor must exercise some control over curriculum; that is, the selection and sequencing of material to be presented.

2. A tutor must be able to respond to students' questions about the subject matter.

3. A tutor must be able to determine when students need help in the course of practicing a skill and what sort of help is needed.

Some tutors, automated and human, have very weak models for one or two of these functions, but the design of any tutorial system must include some approach to each.

**Curriculum and instruction.** By curriculum I mean the selection of and sequencing of material to be presented to students. By instruction, I mean the actual presentation of that material to students.

For teaching methods such as lectures, which are less dynamic than tutoring, both curriculum and instruction can be developed prior to delivery, with as much or as little accountability to principle as the developers feel is needed. Tutorial systems afford no such luxury because a tutor, human or machine, is bound to tailor the selection, sequencing, and methods of delivering instruction to meet the ongoing needs of individual students. Developing curricula and instruction for tutoring therefore is the problem of developing methods for selecting and sequencing material and methods for presenting that material.

The organization of this paper is straightforward. After a brief discussion of some issues central to

---

\(^1\)This is not to say that tutoring per se is the only way that automated tutors can function. Team or group instruction could be and has been (Brown, Burton, & DeKleer, 1982) implemented to considerable advantage with automated tutors.
intelligent tutoring, the major approaches to curriculum and instruction in automated tutors are considered. The chapter concludes with a discussion of major research issues and some tentative guidelines for implementing automated tutors.

Three Central Issues

Throughout the chapter, several major issues or distinctions will recur. They deserve some mention at the outset. The common view of learning and teaching tends to obscure these distinctions, but they are all too evident in the context of intelligent tutoring systems.

The nature of learning. Most approaches to instruction are based on an unspoken "blank slate" assumption. Entering students who cannot perform a particular task or recall a particular fact are viewed as lacking the skill or missing the fact. Although this assumption may hold in a number of situations, there may well be others in which students possess all the wrong skills or all too much knowledge. Since the time of Socrates, scholars have recognized this possibility, but it has certainly not received widespread recognition in current educational practices. There is little in advice to teachers or instructional designers that directs them to the process of weeding out inappropriate knowledge at the same time that they are sowing useful knowledge.

The nature of teaching. The view of learning that dominates current instruction is derived from studies of how individual organisms manage to learn on their own in a variety of environments. The unwritten assumption behind this approach is that instruction should be designed to take best advantage of the mechanisms of individual learning. However, much learning, and indeed the learning that distinguishes us as human, is a cooperative venture that depends crucially on certain conventions (primarily linguistic) for communication among students and teachers. Since communication is a particularly salient aspect of tutoring, we need to understand instruction not only from the point of view of conventional learning theory but also as a process of communication.

The nature of the subject matter. The short history of automated tutoring exhibits a curious split in the choice of instructional objectives, a split that has implications for all aspects of the field including curriculum and instruction. Some tutors, which are called expository tutors, are primarily concerned with factual knowledge and inferential skills. They teach students a body of factual knowledge and the skills needed to draw first-order inferences from that knowledge. They rely on declarative knowledge in the sense discussed by Anderson (this volume). Dialogue is the primary instructional tool used by these tutors. Carbonell's (1970) tutor, for example, engaged students in systematic discussions of South American geography. Collins and Stevens (1982) describe a tutor that uses dialogue to teach certain principles of meteorology.

Other tutors, which are called procedure tutors, teach skills and procedures that have application outside of the tutorial situation. Although memory for facts is important in learning such skills, tutors of this genre are much more concerned with the procedures that operate on memory. As a consequence, procedure tutors function much more like coaches. They present examples to exhibit problem-solving skills, and they pose exercises for purposes of testing and practice.

Curriculum

The problem of curriculum can be broken into two problems, formulating a representation of the material and selecting and sequencing of particular concepts from that representation. In automated tutors, representing knowledge for instruction involves, at least, an adequate expert module of the type discussed by Anderson (this volume). Only one topic need be added to his discussion, and that is for propaedeutics, the knowledge needed for learning but not for proficient performance. A brief
treatment of paedaeutics precedes the major topic of this section; namely, selection and sequencing of material.

**Propaedeutics: Representing Knowledge for Instruction**

The most common strategy among those few who design automated tutors is to adopt an expert model as the representation of material to be taught. The rationale for this strategy is that learning involves progressive acquisition of the cognitive structures that support expert performance. Under many circumstances, this strategy may be the most appropriate, but there also may be cases in which a tutor should use a knowledge representation that is suited to instruction but not to skilled performance. One example of such a representation is NEOMYCIN (Clancey, 1984; Clancey & Letsinger, 1981), described by Anderson (this volume). Another example is Heller and Reif’s (Heller & Reif, 1984; Reif & Heller, 1982) verbal representation of some procedures for solving physics problems (see Table 4.1 for an example).

**TABLE 4.1**

*Procedure for Generating a Theoretical Problem Description in Mechanics*  
*(taken from Heller and Reif, 1984)*

<table>
<thead>
<tr>
<th>Relevant times and systems: At each relevant time (previously identified in the basic description of the problem) identify those systems relevant in the problem because information about them is wanted or because they interact with such systems directly or indirectly.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description of relevant systems:</strong> At each relevant time, describe in the following way each relevant system (if simple enough to be considered a single particle), introducing convenient symbols and expressing simply related quantities in terms of the same symbol.</td>
</tr>
<tr>
<td><strong>Description of motion:</strong> Draw a “motion diagram” indicating available information about the position, velocity, and acceleration of the system.</td>
</tr>
<tr>
<td><strong>Description of forces:</strong> Draw a “force diagram” indicating available information about all external forces on the system. Identify these forces as follows:</td>
</tr>
</tbody>
</table>

**Short-range forces:** Identify each object which touches the given system and thus interacts with it by short-range interaction. For each such interaction, indicate on the diagram the corresponding force and all available information about it.

**Long-range forces:** Identify all objects interacting with the given system by long-range interactions. (Ordinarily this is just the earth interacting with it by gravitational interaction.) For each such interaction, indicate on the diagram the corresponding force and all available information about it.

**Checks of description:** Check that the descriptions of motion and interaction are qualitatively consistent with known motion principles (e.g., that the acceleration of each particle has the same direction as the total force on it, as required by Newton’s motion principle $ma = F$).

These intermediate or paedaeutic representations serve to support performance while more efficient procedures are acquired through practice. Propaedeutic representations have two characteristics. First, they make explicit the functional basis of the procedures used in exercising the skill. Second, they are manageable with the limited cognitive resources available to students. Thus they serve (a) to relate theory to practice; (b) to justify, explain, and test possible problem solutions; (c) as a stepping stone to more efficient problem-solving strategies; and (d) as strategies for management of working memory during intermediate stages of learning.

75
Anderson, Boyle, Corbett, and Lewis (1986) have recently provided some insight into the use of these intermediate representations in instruction. They suggest that declarative knowledge is encoded in special schemata called PUPS\(^2\) structures which indicate, among other things, the form and function of the declarative knowledge that they encode. These schemata are interpreted in the course of working exercises, and the trace of that interpretation is the procedural knowledge underlying the skill to be learned. Although a declarative representation plays no role in the exercise of an established skill, it is crucial to the acquisition of that skill.

Selection and Sequencing

The differences between expository tutors and procedure tutors are evident in the problems associated with selecting and sequencing material. For expository tutors, the problems are those of maintaining focus and coherence and of covering the subject matter in an order that supports later retrieval of the concepts being taught. Procedure tutors have the additional problem of properly ordering the subskills of the target skill and selecting exercises and examples to reflect that order.

**Topic selection in expository tutors.** Curricula in expository tutors must deal with two sources of constraints. One set of constraints arises from the subject matter. Topics must be selected to maintain coherence and to convey the structure of the material being taught. A second set of constraints comes from the tutoring context. Selection of some topic or fact for discussion must reflect the student's reaction to previous tutoring events.

The methods used to construct curricula that reflect the structure of the material have been the subject of much research both in the context of automated tutors and in the larger educational community itself. Work at Bolt Beranek and Newman, starting with SCHOLAR (Carbonell, 1970) and continuing with research by Collins, Stevens, and others (Collins & Stevens, 1982; Collins, Warnock, & Passafiume, 1975; Stevens & Collins, 1977, 1980) has systematically examined how both human and automated tutors plan curricula. Influential work of the same sort can be found in other educational literature (Ausubel, 1968; Reigeluth & Stein, 1983).

The general conclusion of this work is that curricula should conform to an approach called web teaching by Norman (1973). Two principles guide the selection of materials in web teaching:

1. Relatedness—give priority to concepts that are closely related to existing knowledge, and
2. Generality—discuss generalities before specifics.

Web teaching can be justified by reference to a complementary notion called web learning. According to this notion, students develop cognitive structures that reflect the curriculum. The structure provided by web teaching is a framework of general concepts that is anchored in existing knowledge and that serves to support more detailed knowledge.

Web teaching and related approaches provide a static framework for curricula. They do not address the powerful mechanisms that tutors can use to formulate and reformulate curricula within the dynamic context of the tutoring situation. They do not tell us, for example, whether the curriculum should be redirected as the result of some unanticipated question from the student.

Recently, Woolf and McConnel (1985) have developed a sophisticated methodology for studying dynamic formulation and reformulation of curricula. This methodology, implemented in a program

---

\(^2\)PUPS stands for the Penultimate Production System, a rule-based system that Anderson uses to formulate his cognitive theory.
called Meno-tutor, has two distinct mechanisms for directing the tutorial dialogue. One mechanism implements planning mechanisms like web theory for maintaining coherence and focus in the dialogue. These mechanisms are represented in an ATN\textsuperscript{3} grammar, called a Discourse Management Network (DMN).

Meno-tutor has a second curricular mechanism that allows it to respond to a student's particular situation. This mechanism is a set of meta-rules that examine the overall context of instruction for conditions that dictate a change from the normal path of instruction represented in the DMN. The meta-rules consist of conditions on the overall state of the DMN and actions that can effect transitions not allowed by DMN's syntax. For example, when the tutor finishes the discussion of one topic, a meta-rule assesses the tutor's overall knowledge of the student's competence, and, if it turns out that the tutor knows little about the student, the meta-rule will drive the tutor to a strategy calling for exploration of student knowledge.

Exercise and example selection in procedure tutors. Procedural skills are nearly always taught by exercise and example. In these cases the major curricular issue is that of choosing the correct sequence of exercises and examples. Ideally, the choice of exercises and examples should be dictated by a model of learning, but, as Anderson (this volume) has pointed out, there is no theory of learning that is precise and powerful enough to support an interactive tutoring system. Research on the selection and sequencing of exercises has suggested several standards.

1. Manageability. Every exercise should be solvable and every example should be comprehensible to students who have completed previous parts of the curriculum.

Recommendations for meeting the manageability criterion are well known by researchers concerned with instructional systems design (ISD). Gagne and Briggs (1979) recommend analyzing the skills to be taught into a prerequisite hierarchy of instructional objectives. The highest level of the hierarchy consists of primary objectives. Each descendant of each objective in the hierarchy consists of that objective's immediate prerequisites, called enabling objectives. Gagne and Briggs recommend a curriculum that devotes a single lesson to each instructional objective, that imposes a mastery criterion on the learning of each lesson, and that presents the lesson for each objective after the lessons for its enabling objectives.

ISD also makes some recommendations concerning the fine-grained structure of curricula within lessons. These recommendations rely on a taxonomy for the cognitive features of instructional objectives and rules that construct lessons based on the classification of each objective in the taxonomy. For example, component display theory (Merrill, 1983) provides two recommendations for the selection of examples and exercises in classification learning. The divergence principle calls for broadly representative sampling of instances, and the matching principle calls for presentation of both positive and negative instances of the concept, procedure, or principle being taught.

In summary, manageability can be achieved by isolating each objective to be taught, by providing enough material to allow students to master each objective, and by teaching prerequisites first.

2. Structural transparency. The sequence of exercises and examples should reflect the structure of the procedure being taught and should thereby help the student induce the target procedure.

\textsuperscript{3}Augmented Transition Network (ATN) grammars are general and powerful mechanisms for representing procedures. Their principal use is for natural-language understanding, but they can serve, as in Meno-Tutor, to represent complex procedures for other tasks. See Winston (1984, pp. 304-309) for a technical discussion of these grammars.
This principle proposes that curriculum is a form of communication with the student in that the sequence of exercises and examples tells the student something about the subject matter. Theories of this kind of communication must therefore have two components. They must specify how to derive a sequence of exercises and examples from the structure and content of the procedure being taught, and they must explain how a student can interpret the sequence in order to learn something about the procedure.

To date, only two efforts have addressed both components of the structural transparency issue. Smith, Walker, and Spool (1982) proposed certain structuring principles for an existing course in symbolic logic that consisted largely of exercises and examples of proof problems. Smith et al. also constructed a learning model that used these principles to induce the strategies supporting skilled problem solving in the course. Smith et al.'s learning model is schema driven. It matches each unit of the course (e.g., a sequence of examples) to a template that specified an induction principle. The induction principle can be used to infer some problem-solving strategy from the unit. Smith et al. argued that the templates constitute communicative conventions shared by instructional designer and student for the purpose of conveying procedural knowledge through curriculum structure.

A similar but more thorough line of work can be found in VanLehn (1983, 1985, in press). His concern was with curricula in which students induce a procedure solely from the exercises and examples presented to them. Most of his work focused on curricula for multicolumn subtraction problems and student performance in those curricula. In a theory of these curricula called step theory, he proposed that learning is possible in such cases only if certain conventions, called felicity conditions, govern the construction of the curricula. The felicity condition that relates to selection and sequencing requires that the curriculum be divided into discrete lessons, each of which adds a single decision point or step in the procedure to be learned (hence the name step theory). The examples and exercises in each lesson can use only the step to be learned or steps previously addressed in the curriculum. Although VanLehn did not present a particular learning model in his theory, he did demonstrate that no learning procedure could possibly induce the correct procedure unless the curriculum conforms to step theory and the procedure takes advantage of this fact.

3. Individualization. Exercises and examples should be chosen to fit the pattern of skills and weaknesses that characterize the student at the time the exercise or example is chosen.

The approaches to manageability and structural transparency previously described are static in that they do not take advantage of a tutor's ability to dynamically formulate a curriculum to conform to the ongoing instructional context and, in particular, to the student's changing state of mastery. Each exercise or example should be chosen so that it is (a) manageable with skills already possessed by the individual student, and (b) easily related to skills already possessed by the individual student.

BIP-II (Wescourt, Beard, & Gould, 1977) is the only example of a procedural tutor that addresses these desiderata. BIP-II teaches the BASIC programming language by offering students exercises that can be solved in a powerful but nonintelligent programming environment. Of interest here are BIP-II's methods for selecting each exercise.

The three components of BIP-II are illustrated in Figure 4.1. A semantic skills network represents some 93 skills needed for competent BASIC programming and the salient pedagogical relations among them. A student profile maintains an assessment of the student's mastery of each skill in terms of five states of learning. This profile is updated after every exercise, based on student performance. An exercise library contains a large number of exercises and the skills required for each.

---

4I hesitate to call this a student model in the sense in which VanLehn (this volume) uses the term because it is not an information-processing account of how student solve problems in the course.
Figure 4.1. Components of BIP-II.

These components allow BIP-II to dynamically address manageability and structural transparency in its selection of exercises. That is, its selection algorithm chooses exercises that have some optimal combination of learned and unlearned skills and contain unlearned skills that are conceptually related to learned skills.

Selection and sequencing criteria. What lessons do the foregoing examples and suggestions have for curricula in automated tutoring systems? The primary one is that one should look more to the overall goals of curriculum construction than to principles for design in particular situations. Curricula for tutoring situations serve several functions:
1. A curriculum should divide the material to be learned into manageable units. These units should address at most a small number of instructional goals and should present material that will allow students to master them.

2. A curriculum should sequence the material in a way that conveys its structure to students.

3. A curriculum should ensure that the instructional goals presented in each unit are achievable.

4. Tutors should have mechanisms for evaluating the student reaction to instruction on a moment-to-moment basis and for reformulating the curriculum.

**Instruction**

This section concerns the instructional methods that an automated tutor might use to deliver a curriculum. These methods must cover initial presentation of the material, ways of responding to students' questions, and the conditions and content of tutorial intervention.

**Presentation Methods**

The methods used to present material depend on the subject matter and the instructional objectives. Expository tutoring uses dialogue as the chief method of conveying material. Tutors oriented towards procedural skills use examples and coached exercises to develop those skills.

**Dialogue.** The issues involved in formulating dialogue for expository tutors are similar to those involved in formulating curricula for these tutors. In particular, dialogues need to be planned to address the instructional objectives at issue, and dialogues must be sensitive to the evolving tutorial context.

Collins and Stevens (1982) and Collins et al. (1975) have derived some general guidelines for conducting tutorial dialogues once the instructional objective of the dialogue has been established. They treat three types of objectives: the teaching of facts and concepts, the teaching of rules and functional relations, and the teaching of skills for deriving these rules. Note that this classification corresponds to that used in recommendations from ISD (Gagne & Briggs, 1979; Merrill, 1983).

Table 4.2 summarizes Collins and Stevens' guidelines for dialogues addressing each objective. Teaching of facts and concepts is accomplished by asking for or explaining the material. The decision to ask or tell is made on the basis of the importance of the material and the student's knowledge thereof. Teaching of rules in tutorial sessions usually involves inducing the student to consider the relevant data and to formulate the rule. This can be done by presenting case data that makes the rule clear or by entrapment strategies that enable the student to eliminate incorrect versions of the rule. Skills for deriving rules are taught as procedures. These procedures are broken down into their components (e.g., listing factors, generating cases to specification), and exercises and examples are provided that address each subskill.

The dialogue plans suggested by Collins and Stevens are interactive in the sense that particular tutorial utterances are conditioned by the student's responses, but these dialogues do conform to rigid plans that cannot be reformulated in the middle of an interaction. By contrast, Woolf and McDonald's (1985) Memo-tutor, which has been described previously, offers the same dynamic flexibility at the instructional level as it does at the curricular level. Memo-tutor's DMN has some 27 instructional (as opposed to curricular) states, each representing a different method of presenting tutorial materials. The DMN, for example, makes a distinction between feedback used to dismiss a topic (a simple "no" or "well...") and that used to maintain the topic at the center of attention.
TABLE 4.2
Tutorial Dialogue Strategies for Different Instructional Objectives

<table>
<thead>
<tr>
<th>Instructional Objective</th>
<th>Strategies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teach facts and concepts</td>
<td>Elicit fact or concept</td>
</tr>
<tr>
<td>Explain fact or concept</td>
<td>Teach rules and relations</td>
</tr>
<tr>
<td></td>
<td>Case selection strategies</td>
</tr>
<tr>
<td></td>
<td>Entrapment</td>
</tr>
<tr>
<td>Teach induction skills</td>
<td>Exercises and examples</td>
</tr>
<tr>
<td></td>
<td>oriented to subskills</td>
</tr>
</tbody>
</table>

More to the point, meta-rules allow for dynamic reformulation of the tutorial at the instructional level as well as the curricular level. The method of presentation can therefore be determined by default assumptions in the DMN, or, if circumstances dictate, by needs that arise in the particular instructional context. Normal circumstances might, for example, dictate active correction of a student error, but Meno-tutor possesses a rule that allows it to give a less emphatic correction if it decides that the student is confused at that point in the dialogue.

**Instructional modeling.** Instructional modeling, the use of worked examples or guided practice, is a prime vehicle for introducing students to procedures that they must learn. Essential to the success of modeling in intelligent tutoring systems is the formulation and presentation of procedures for working the examples. These procedures must be based on the representations (including propaedeutic representations) that students need to acquire the target skills, and they must be presented to the student in a manner that shows how each step applies to the case being modeled.

SOPHIE II (Brown, Burton, & de Kleer, 1982) is one early example of a training system that faced these issues. It demonstrated procedures for troubleshooting arbitrary faults in a simple electronic device. The significance of Brown et al.'s work is in the discipline they used to formulate and present SOPHIE-II's troubleshooting procedure. In particular, they restricted SOPHIE-II to general (device-independent) procedures that were cognitively faithful to human troubleshooters, and they gave SOPHIE the facility to verbally account for its troubleshooting decisions as it demonstrated these procedures.

Language is not the only vehicle that can be used to explain procedures during instructional modeling. Hutchins and his colleagues (Hutchins & McCandless, 1982; Hutchins, McCandless, Woodworth, & Dutton, 1984) developed a system (MANBOARD) to aid in the training of relative-motion problems in naval surface operations. This system is able to demonstrate procedures and illustrate these demonstrations with displays (like the one in Figure 4.2) of ships in both relative and geographic coordinates. These displays make clear the geometric basis of the procedures being taught.

VanLehn (1983) found another important application of visual explanation, not in a tutoring situation but in his examination of multicolour subtraction procedures. He found that the indications of crossing out and borrowing in worked examples (see Figure 4.3) were crucial to learning in that they made explicit the intermediate steps of procedures. Without these indications, students are, in principle, unable to induce the procedure from the examples given them in typical curricula. The import of this show-work felicity condition is obvious for instructional modeling; a tutor should provide whatever description is necessary to ensure that the student can grasp the intermediate mental steps of the procedure.
Have computer generate a new set of values

Figure 4.2. Display of Hutchins et al.'s Training System for Relative Motion Problems (Hutchins, McCandless, Woodworth, & Dutton, 1984). The left panel provides a relative-motion plot, and the right panel provides the corresponding geographic plot.

Answering Questions

Responding to questions is an essential function of human tutors, and one might expect to find the same function in automated tutors. In fact, however, question answering has not been the focus of many of the automated tutors that have been developed. The major stumbling block to effective question answering, as Anderson mentions in Chapter 2, is the difficulty of natural language comprehension and generation. One attempt to work around the problem can be found in SCHOLAR's (Carbonell, 1970) use of a template-matching strategy to deal with students' questions.

SOPHIE (Brown et al., 1982) was a more sophisticated attempt to deal with both epistemological and linguistic aspects of answering students' questions. SOPHIE I (Brown, Burton, & Bell, 1974) answered the questions of students learning to troubleshoot the device also used in SOPHIE II. Many of these questions, such as hypotheticals, might have required considerable search, but SOPHIE I determined the answers by systematically running a mathematical model of the device under various conditions. Because no reasoning was involved in these runs, SOPHIE I had no way to explain and justify the methods used in the search and could therefore not produce the reasoning needed to answer such questions. This problem and later observations of both novice and expert troubleshooters led Brown et al. to the conclusion that causal explanations of device function were necessary for understanding that device. Subsequent investigations associated with and subsequent to SOPHIE III have provided some
deep insights in the field of qualitative mental models (de Kleer & Brown, 1983). These investigations are treated more fully in Chapter 2.

Trading Hundreds First

There are 304 birds at the Lincoln Zoo. 126 birds are from North America. How many birds are from other places?

304 - 126 = ___

Need more ones? Yes. But no tens to trade. Need more tens.

<table>
<thead>
<tr>
<th>Trade 1 hundred for 10 tens.</th>
<th>Trade 1 ten for 10 ones.</th>
<th>Subtract the ones.</th>
<th>Subtract the tens.</th>
<th>Subtract the hundreds.</th>
</tr>
</thead>
<tbody>
<tr>
<td>210</td>
<td>2014</td>
<td>9</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>304</td>
<td>304</td>
<td>-126</td>
<td>-126</td>
<td>-126</td>
</tr>
<tr>
<td>-126</td>
<td>-126</td>
<td>-126</td>
<td>-126</td>
<td>178</td>
</tr>
</tbody>
</table>

304 - 126 = 178 178 birds are from other places.

Subtract.

1. 401
2. 205
3. 300
4. 102
5. 406

- 182
- 77
- 151
- 4
- 28

6. 700
7. 608
8. 503
9. 900
10. 802

- 513
- 39
- 304
- 28
- 9

11. 806
12. 500
13. 407
14. 904
15. 600

- 747
- 439
- 8
- 676
- 69

16. 606
17. 306
18. 204
19. 600
20. 508

- 56
- 197
- 7
- 29
- 429

21. 402 - 16
22. 700 - 8
23. 900 - 101


Tutorial Intervention

One of the prime benefits of tutoring is the opportunity that a tutor has to break into a student's ongoing learning activities with whatever intervention is needed to speed the course of instruction. Tutorial intervention is needed to maintain control of the tutorial situation, to protect the student from
inappropriate or incorrect learning, and to keep the student from exploring paths that are not instructionally useful\(^5\). Automating the process of tutorial intervention involves devising rules for deciding when (or when not) to intervene, and formulating the content of the intervention.

**Conditions for intervention.** There are two major approaches to decisions about tutorial intervention. Model tracing calls for intervention whenever the student strays from a known solution path. Issue-based tutoring calls for tutorial intervention only when it can make a positive identification of a particular occasion for intervention.

Both Anderson and VanLehn in this volume have explained the essentials of model tracing. A tutor using this technique maintains a model of the student’s cognitive processing as the student works through an instructional unit. This model reflects the cognitive processes of a competent performer in the instructional setting. As the student progresses, the model traces that behavior, attempting to match it to one of the paths that could be taken by the ideal student. When the matching process fails, the tutor intervenes with advice that will return the student to a successful path.

Whereas model tracing suggests intervention whenever the tutor cannot positively identify the student’s response, issue-based tutoring suggests intervention only when the tutor can make some sense of the student’s response. Issue-based tutoring has certain advantages over model tracing. For one thing, it need not restrict its intervention to remedial instruction. Identifiably good performance may be occasion for intervention along with identifiably bad performance. In addition, issue-based tutors can be more informative in the content of their intervention since they can speak to the issue that caused the intervention. Issue-based tutors can also function with less than perfect expert modules. Model-tracing tutors will intervene even when the student finds a better approach than the expert module, but issue-based tutors will remain silent in these circumstances.

These and other benefits of issue-based tutors are well-illustrated in the tutor called WEST developed by Burton and Brown (1982) and described by Anderson (this volume). WEST offers advice to the player of a computerized arithmetic game. It characterizes the game in terms of a number of issues or strategies that may be of use to a player on certain moves, and it tutors them in these issues by reminding the student of them on carefully chosen occasions throughout the game. The primary criteria for these occasions are the student’s failure to use the issue when appropriate and evidence that the student’s knowledge of the issue is weak.

Systems like WEST offer the opportunity to try a variety of different principles for deciding on intervention. These principles can address cognitive concerns. For example, WEST never intervenes on the first few moves so that students can concentrate on the mechanics of the game. Other principles are motivational. WEST, for example, does not offer advice if a player is doomed to lose no matter what, and it congratulates players on exceptionally good moves.

There is no reason why model-tracing and issue-based techniques cannot exist in the same tutor. Anderson’s tutors (Anderson, Boyle, & Reiser, 1985; Anderson, Boyle, & Yost, 1985) incorporate some aspects of issue-based tutoring within a model-tracing framework. In particular, they rely on a bug catalog (discussed in VanLehn, this volume), a set of inappropriate or incorrect rules that are commonly observed at intermediate stages of learning. When the model trace matches one of these buggy rules, the tutor can direct its advice to the bug.

Gentner (1979) and Gentner and Norman (1977) used another combined approach in Coach, a tutor that monitors students learning a very simple programming language called FLOW. Coach was

\(^{5}\)What constitutes an instructionally useful path depends on instructional objectives. If these objectives include teaching error-recovery skills, then allowing the student to make errors is an important part of instruction.
designed to monitor their every keypress in real time\(^6\) and to intervene under particular circumstances.

Coach's tutoring methods are based on a schema model that encodes the structure of the course as well as the structure of the language FLOW. The model is hierarchical in nature with high-level schemata representing, say, chapters in the manual or entire exercises, and low-level schemata representing individual keypresses. Coach implements student modeling in both a top-down (model tracing) and bottom-up (issue-based) fashion. It also has context-independent buggy schemata (e.g., a schema for improperly ordered steps), an activation-driven mechanism for dealing with unfulfilled expectations, and mechanism for separating bugs from slips on the basis of past performance. Coach is an outstanding example of the leverage that sophisticated student and expert modules can contribute to tutoring.

**The content of intervention.** When a tutor decides to intervene it must also formulate the content of the intervention. There is no uniform approach to the content of intervention among the few computer coaches in the literature. The most obvious technique, directly correcting the problem that caused the intervention, is not used in any of them, and with good reason. Simply informing a student of the low-level actions needed to recover from a bad situation would waste the opportunity for the tutor to teach students about the situation. Thus, some tutors, such as those of Anderson (Anderson, Boyle, & Reiser, 1985; Anderson, Boyle, & Yost, 1985) and WEST, provide advice at the next higher level of abstraction, requiring students to apply this advice to their own concrete situation. Coach attempts to locate the particular schema where the problem arose and offer advice addressing that schema.

Perhaps the most sophisticated approach to formulating the content of tutorial advice is described by Goldstein (1982). He suggests that as students acquire skill, they can be characterized in terms of increasingly sophisticated information-processing models. Tutorial advice should be responsive not only to the student's particular difficulty but also to the student's level of sophistication in the task. A neophyte making an error should receive suggestions of a relatively coarse nature. A more sophisticated student making the same error should receive advice of a more detailed nature.

**Research and Practice in Automated Tutors**

From the foregoing description of where research in automated tutors has been over the past 15 or 20 years, we can look forward in several directions. This section begins with some suggestions for the direction of research in the field. It does not present a laundry list of potential research projects, but instead concentrates on three fundamental research issues. Brief suggestions for the kinds of projects that might illuminate those issues are included. Also mentioned in this section are some lines of research that, while interesting, are not appropriate for investigation at this time. Finally, for those interested in immediate applications, a brief discussion of currently feasible implementations is presented.

**Research Issues**

A broad view of current research in intelligent tutoring systems and in education in general reveals a few crucial issues that deserve serious consideration in any planned research and development effort.

1. An important concern in both research and development is the scope of the efforts; that is, the range and combination of different situations which those efforts address. Researchers in intelligent tutoring systems should look to ISD as a field that is particularly concerned with the

---

\(^6\)It is important to understand, however, that Coach was never implemented in real time. Instead, it was evaluated in terms of its ability to deal with replays of tutored students' protocols.
broad range of instructional applications.

2. Equally important from a scientific point of view is the necessity of being specific about mechanisms. It is not sufficient to simply build automated tutors that work. An effort must be made to characterize the principles of learning and instruction that account for the effectiveness of these tutors.

3. In addition, attention should be paid to the structure of the discipline. A major aim of the research discussed here is the codification of instructional principles. Future researchers need to seriously question the extent to which these principles can be codified independently of the material that they teach and to what extent they are an integral part of that material.

**Automated tutors and instructional design.** One of the major tasks facing researchers in automated tutors is that of relating their work to other research in training and education. Other instructional research has not been discussed prior to this section, because the relationship of research on automated tutors to other instructional research is an important issue in its own right and because the discussion would have been difficult to understand without the context set by the foregoing description of research in automated tutoring.

Most instructional research is tangentially relevant, if it is relevant at all to automated tutoring systems, either because it addresses other forms of instruction or is simply not sufficiently oriented to design to be of direct help. But one branch of instructional research, namely ISD, seeks to provide methods that can be used to design instructional systems. Moreover, the ISD community is in general agreement about the methodology that should be used to design instructional systems.

ISD is a mixed blessing for automated tutors. On the one hand, it offers the kind of systematic decomposition of the instructional problem and the comprehensive coverage of instructional applications that is sorely needed in the intelligent tutoring field at this point. On the other hand, ISD strives for a level of specificity that is appropriate for instructional designers but nowhere near appropriate for computer tutors. In addition, because it has not been particularly concerned with tutoring methods, it makes no recommendations for the kind of student-tutor interaction that makes these methods so effective. To see the research implications of these statements it is necessary to take a more detailed look at each one.

Starting with the benefits of an ISD view of automated tutors, note that ISD proposes a decomposition of the design process that is consistent with the one discussed in this volume and in this chapter in particular. ISD makes a distinction between analysis of instructional needs (the subject of Anderson's chapter) and development of curriculum and instruction (the subject of this chapter). Also, ISD distinguishes between curriculum and instruction. Reigeluth and Merrill (1978) refer to these aspects of instruction as macro- and micro-strategies, respectively. ISD also holds that decisions about curriculum and instruction can be based on a cognitive classification of the instructional objectives. Specific recommendations in this regard can be found in curriculum in Reigeluth and Stein (1983) and for instruction in Merrill (1983). In the automated tutoring literature, many of these recommendations (e.g., teach procedures with exercises and examples) are implicit and far from complete.

A second potential benefit of ISD is the fact that it aims for a comprehensive treatment of instructional design. Even a casual reader of the literature in automated tutoring would have to be struck by the narrow, piecemeal nature of the offerings. The chances of finding an intelligent tutor that meets the needs of a randomly chosen application are quite small indeed. By contrast, ISD offers a top-down approach that covers a large area of the instructional waterfront. This means, for one thing, that researchers or designers need not tailor their application to ISD methodology; rather, the methodology will tailor itself to the application. In addition, ISD can deal with complex combinations of different kinds of instructional objectives and find the corresponding combination of instructional methods. Most skills require a combination of declarative and procedural knowledge. Whereas most automated tutors are

86
specialized to teach one or the other. ISD offers as part of the design process methods for teaching both where they are needed.

However, ISD is not without features that make it difficult to apply to intelligent tutoring systems. One of the most evident of these problems is the fact that ISD is meant to be used by intelligent designers, and it takes full advantage of their powers of intellect. Although the prescriptions of ISD are precise enough to be understood by people (and are often seen as annoying in their precision), they come nowhere near the specificity necessary for formalization and programming on a computer. Designers can fill in many of the details in, say, Merrill's divergence principle, discussed previously; but the task of writing a single computer program that could apply that principle to concepts as diverse as well-formed Russian sentences and identifier names in Pascal is well beyond the state of the art.

The state of the art in intelligent tutoring systems is quite different. Well-specified solutions exist, but only for a small number of problems. Of course, it is possible to create, by hand, additional solutions by writing programs that apply ISD principles in particular cases; but this strategy will not significantly advance the task of formalizing those principles themselves. If ISD has any power that is independent of the intellect of its users, then expressing its principles in formal mechanistic terms is a most appropriate venture.

Another feature of ISD that limits its current applicability to automated tutors is its lack of emphasis on tutorial situations. Tutoring, after all, is an expensive and uncommon instructional method, and for this reason alone may have failed to capture the attention of the ISD community. Woolf and McDonald's (1985) research suggests a parallel that helps make the shortcomings of ISD apparent in tutorial situations. Recall that they proposed two levels of tutorial interaction. One, governed by the DMN, corresponds to the kinds of instructional plans that can be developed using ISD. The other, governed by meta-rules, allows for global evaluation of the instructional context and dynamic modification of the instructional plan. The principles for effecting the former, planned level of interaction are consistent with the principles of ISD and in fact are given an extensive treatment in Gagne and Briggs (1979). However, I see no way that the second more global level of interaction can be accommodated under ISD as it currently stands. Extending ISD to allow for ongoing global evaluation of the instructional context would make it more applicable to automated tutoring and would be a significant advance in ISD itself.

Research suggestions for instructional design. As a first step towards a design approach to automated tutors, laboratories for the systematic manipulation of alternative tutoring methods are needed. Meno-tutor and WEST are good examples of these laboratories because they provide a tutorial shell that can host a variety of instructional methods. Design knowledge can also come from observation. Of interest in this regard are Wizard-of-Oz systems\(^7\), semiautomated tutors in which a human tutor (like the Wizard of Oz) replaces some or all of the instructional functions of an automated tutor (like the machine that the Wizard used to project a wizandly presence to visitors). Studies of these systems might range from systematic observations of tutors' case-selection strategies to development of a sophisticated tutor's assistant, designed to support real tutoring activities as well as collect data on tutors' behaviors.

Theories of learning and instruction. Many of the problems that afflict ISD and other approaches to instruction occur because they lack a foundation in a precise theory of learning. That is, there are no models of the mechanisms that govern a student's interpretation of particular instructional presentations. An obvious approach to these problems is to discover laws of learning which will specify these mechanisms, and in fact much work over the past century has been devoted to the discovery of such laws. A question posed earlier by Anderson (this volume) again arises. Why are there no automated tutors that can work with a model of a learning student?

\(^7\) I would like to thank Jim Miller for suggesting this concept and the term Wizard of Oz.
The answer lies in the complexity of the instructional enterprise, a complexity manifest on several levels. The first level is that of cognition. Laws of learning apply not to overt stimuli and responses but rather to internal symbolic representations of the type described by Anderson and VanLehn (this volume). A second level of complexity stems from the communicative nature of the instructional enterprise. Laws of learning are incomplete descriptions of what goes on in instructional situations. Needed is a joint theory of how instruction is formulated by the tutor and how it is interpreted by the student; neither aspect makes sense without the other. Even greater complexity is introduced by the possibility that students do not already know the instructional conventions when they come to the tutorial, but rather must learn them during the course of instruction. Do children come to second grade fully prepared to take advantage of VanLehn’s felicity conditions, and if not, what laws govern their learning about these conditions? Does a tutor based on laws of instruction have to arrange to teach those laws to students? There is also the possibility that the principles that govern teaching and learning are not immutable but rather are selected, modified, or generated through negotiation between tutor and student. A tutor who fails in using one form of communication may change the rules in hopes that another form will succeed.

The abbreviated argument presented here takes us from a simple stimulus-response theory of learning to a complex theory of instruction that makes little reference to basic laws of learning. I do not mean to suggest that simple laws have no use in instructional design. Indeed, Schneider (1985) has gotten considerable instructional mileage from a few simple stimulus-response principles. I do, however, want to make clear that there is much to the tutoring enterprise that does not follow from simple laws of learning and that demands a theory of instruction in its own right. A research program in automated tutoring must have a special concern for the particular nature of instruction as a cooperative enterprise involving instructional designer, teacher, and student.

Suggested research on learning and instruction. In summary, the field of automated tutoring needs an account of the mechanism whereby automated tutors achieve (or fail to achieve) their effectiveness. Such an account may rest on fundamental laws of learning or it may appeal to complex theories of communication between tutor and student. Research on this question is therefore needed at several levels. Theories of human learning and machine models of those theories (notably Anderson, 1983) have provided and will continue to provide singular benefits to the field of automated tutoring. Observations of natural tutorial interactions, and particularly of procedure tutoring are also needed. In addition new theoretical stances need to be applied to research and development in tutoring. Mehri’s (1979) analysis of communicative mechanisms in a classroom might well be extended to tutorial situations in a way that supports the development of automated tutors.

Modularity: The independence of instructional and domain knowledge. One of the most important working hypotheses in research on automated tutors is that diagnostic and instructional methods can be formulated in a domain-independent fashion and that, conversely, the domain knowledge (i.e., the expert module) can be formulated without reference to particular instructional methods. This hypothesis, which I call the modularity hypothesis, suggests that diagnostic and instructional modules can be used across a broad range of domains. It also suggests the less common converse, namely, the use of several different instructional methods for the same material; see Crawford and Hollan (1983) for an example of this kind of experiment.

Because it lies at the foundation of the work on automated tutors, the modularity hypothesis deserves serious examination in its own right. Parts of the foregoing discussion call this hypothesis into question. For one thing, it is known that different diagnostic and instructional methods apply to different kinds of instructional objectives. In view of this, rules of correspondence of the sort detailed in Merrill (1983) might be used to preserve modularity. These rules allow for the systematic tailoring of diagnostic and instructional modules to different kinds of domains. Conceivably, there could be a tutor maker that would use these rules of correspondence to generate an automated tutor for a particular application.
A more serious retreat from modularity might be needed in the light of the previous discussion of propaedeutic representations. Recall that these representations are models of the subject matter that are needed for instruction but not for skilled performance. Since these representations are derived from a combination of first principles about the domain and the cognitive capacities of students, there is little hope of generating them from any expert model. Propaedeutic representations are therefore a form of instructional knowledge that is specific to a particular domain.

**Research on modularity.** A number of research approaches could illuminate our understanding of the modularity problem. Studies on tutoring shells or tutor generators are certainly appropriate. Such studies should develop the rules that govern the design of automated tutors and attempt to implement these rules in programs that generate or configure automated tutors for particular applications. Also needed are broader studies of propaedeutic representations. NEOMYCIN and Reif and Heller's work (Heller & Reif, 1984; Reif & Heller, 1982) are the most systematic efforts in this area to date. Needed are more examples, and particularly needed are instructional studies that examine how these representations function in learning. The development of techniques that tutors could use to tailor their materials to particular specifications could also illuminate the structure of the material to be taught. Domain-independent tutors may start with domain-independent methods for generating instructional materials from the expert module.

**Research Pitfalls**

Research in intelligent tutoring systems is somewhat like a mine field; so it is fitting to point out a few of the issues that researchers do not know how to approach but that could easily sink a development effort.

**Tutors that must learn the material.** The common working assumption for intelligent tutoring systems is that the expert model is fully competent in what it is trying to teach or at least possesses much more competence than the student does. Two common situations for which this assumption holds are illustrated in Figure 4.4, panels a and b. Panel a illustrates a blank-slate situation in which the student knows little or nothing about the domain, and the tutor knows just about all that there is to know. Panel b illustrates a situation appropriate for Socratic teaching. The student has little useful knowledge but a good deal of misconceptual knowledge. The tutor, as in Panel a, is a master of the subject matter.

Panels c and d of Figure 4.4 illustrate two situations that may often occur in real tutoring situations. Panel c illustrates a peer tutoring situation in which the tutor has only a small advantage over the student. Communication between tutor and student is dramatically altered in this situation because the tutor must effectively convey his or her own shortcomings to the student. Also, both tutor and student in these situations are often involved in a cooperative learning enterprise in which each grows in competence. Meeting either one of these demands is well beyond the state of the art at this time, and the combination is even further from the grasp of current methods.

Panel d of Figure 4.4 presents an even more difficult case, one in which the student is actually more competent than the tutor. Automated tutors that can function well in this kind of situation have a tremendous advantage over those limited to the situations illustrated in Panels a and b because they can be of use even with a less than complete expert module.

**Tutors that must learn to teach.** The underlying goal of most research in intelligent tutoring systems is the successful representation of teaching knowledge and its implementation in a machine. However, at least two efforts in the field have inquired into the possibility that automated tutors could, themselves, learn to teach. Both of these efforts missed the mark in my opinion. One tutor (Kimball, 1982) improved its technique through successive refinement not of teaching strategies but of its student model. The other (O'Shea, 1982) used a complex generate-and-test procedure to try out various commonsensical notions about teaching techniques. Little is known about good teaching and less about
how it is learned. Our own ignorance aside, it is doubtful that any intelligent system, human or machine, could learn to teach on the basis of experience alone. Hence, automated tutors that can really improve their technique on the basis of interactions with students are probably not going to appear in the foreseeable future.

Figure 4.4. Possible Configurations of Student and Tutor Knowledge. (a—Blank-slate model; b—Socratic model; c—inexpert tutor, inexpert student; d—inexpert tutor, expert student.)
Building Automated Tutors with Today's Technology

Finding one's way to a feasible application of automated tutors is a difficult job at best. Nonetheless, I offer the following guidelines for deciding when and how to implement automated tutors. The reader should be aware that the shelf life of lists such as these is vanishingly small.

Choosing an application

1. Work with a domain that can be formalized. Choose an application that can be formalized, one for which, in particular where it is feasible to build an expert module, a propaedeutic representation, or both. Formal problem-solving situations such as troubleshooting or programming are highly suitable. In domains such as tactical planning, which have a more subjective content, select subtasks that can be formalized. Domains such as literary criticism or foreign policy analysis are not within the reach of today's automated tutors.

2. Stay away from natural language. Anderson (this volume) has pointed out that natural language understanding is the Achilles' heel of many potential tutors and of expository tutors in particular. If an application calls for an expository tutor, look for techniques such as those described in Crawford and Hollan (1983) that do not require natural language understanding.

Instructional design considerations

3. Use known principles of sound instruction. Although a good many of the principles of ISD are difficult to automate, many can be used in the design of an automated tutor. At the least, tutors can be designed to conform to the curricular constraints that make for manageability, coherence, and structural transparency. In addition, the show-work principle from step theory deserves serious consideration in any procedure tutor.

4. Use both model-tracing and issue-based tutoring. Both of these instructional techniques are known to work in selected cases. They can be combined in the same system and they will compensate for each other's failures. Hence the design of an automated tutor, starting with the student and expert modules, should provide for both of these techniques.

A general design consideration

5. Design for modularity and robustness. Implementing automated tutors is a risky business. They should therefore be designed to function even if one or more of the parts is ineffective or inoperable. With respect to curriculum and instruction, for example, the tutor should be designed to function with a fixed default curriculum, and it should provide a useful instructional environment even if the tutor is completely silent. The Wizard-of-Oz systems previously mentioned, which use human tutors instead of machine tutors, may also be a possibility in some cases.

Conclusion

What then is the current state of the task of codifying the principles of effective tutoring? There are a number of instructional guidelines (e.g., step theory) that can support the design of automated tutors, and there are some technological tools (e.g., model tracing) that can be used to build effective automated tutors for certain applications.
The existence of these guidelines and the tools for implementing them represent real progress in the field of intelligent tutoring systems. However, the major issues associated with curriculum and instruction in intelligent tutoring systems are still unresolved. The design principles needed to specify the range of automated tutoring applications and the structure of that range do not exist. Precise mechanistic theories that can account for the effectiveness of particular instructional techniques have not been formulated. Clear notions of what constitutes an instructional principle and what constitutes an instructionally useful aspect of some particular domain are also not available.

The very fact that these issues are recognized is a sign of real progress. Fifteen years ago, when the field was in its infancy, there was little to say about the representation of knowledge for teaching purposes and even less to say about the instructional process. Until very recently, a representation of expert knowledge was deemed sufficient for teaching purposes, and theories of learning in uninstructed situations were deemed sufficient for describing instructed learning. Awareness of these issues and the technology for exploring them will make the next few years of research in intelligent tutoring systems at least as exciting and profitable as the past 15 years.
REFERENCES


DISCUSSION

Curriculum and Instruction in Automated Tutors

The discussion on Curriculum and Instruction in Automated Tutors is deferred to page 125. There M. David Merrill discusses both Chapter 4 and Chapter 5.