The Knowledge Engineering Review, Vol. 10:3, 235–268

CBT II—Case-based computer-aided instruction: survey of principles, applications and issues

TARIQ KHAN and YAU JIM YIP
Information Technology Institute, University of Salford, Salford M5 4WT, UK

Abstract

In this paper we consider the role case-based reasoning has had in development of computer-aided instruction systems. We survey several case-based teaching systems, each of which is representative of a basic pedagogical principle that motivated its development. Firstly, 15 pedagogical principles are presented that were identified from the analysis of case-based teaching systems. We present some background to the principles, and indicate which systems they are incorporated in. Next, the teaching systems themselves are described, with emphasis on how case-based reasoning has been applied. Finally, we discuss some problems to be addressed when case-based planning is applied to lesson planning within a tutoring system.

1 Introduction

Two areas of artificial intelligence that have received much attention recently are intelligent computer-aided instruction (ICAI), or intelligent tutoring systems (ITS), and case-based reasoning (CBR). Both stem from seminal work done in the early 1970s, and after two decades realisable implementations are being produced. Intelligent tutoring has a history of impressive systems that have incorporated different knowledge representation methods: frames (e.g. PROUST—Johnson et al., 1985), rules (e.g. GUIDON—Clancey, 1987), scripts (e.g. WHY—Stevens et al., 1982) and causal models (e.g. QUEST—White et al., 1987). Case-based systems have had an equally impressive history in many different applications: classification (e.g. PROTOS—Bariess et al., 1988), planning (e.g. CHEF—Hammond, 1989), and explanation (e.g. SWALE—Schank, 1986). However, the movement has been towards architectures comprising hybrid knowledge representation and reasoning methods that together can provide functionality not possible by rules, cases or causal models alone (e.g. CASEY—Koton, 1988). Other important concerns in intelligent tutoring are how to ease implementation effort, enable re-useability of software components, and how to realise the more ambitious objective of self-improvement (e.g. automated acquisition of pedagogical, problem-solving and planning knowledge). These objectives of intelligent tutoring are commonly stated virtues of case-based reasoning (e.g. Kolodner, 1993) so it seems apt that developers of intelligent tutoring systems should consider how case-based reasoning can benefit them. This is the subject of the survey.

We present a survey of applications of case-based reasoning to intelligent tutoring. The objective is to provide a representative source of the principles and techniques applied to case-based reasoning in instruction. We have not included a detailed description of the theory of intelligent tutoring or case-based reasoning because there are many good recent publications covering these areas (e.g. Sokolnicki, 1991; Watson et al., 1994). In determining the contents, we were faced with deciding on the balance between ITS and CBR issues to include. We appreciate that some readers will be more interested in ITS issues than in CBR, and vice versa. Where

possible, we have tried to highlight both the pedagogical principles and CBR research issues embraced in the research we describe, but recognise that emphasis is on CBR issues.

Firstly, 15 pedagogical principles that were identified from the analysis of case-based teaching systems are presented. Each of these principles is addressed by at least one teaching system described here. We present some background to the principles, and indicate which systems they are incorporated in. Next, the teaching systems themselves are described with emphasis on how case-based reasoning has been applied. Finally, we present a discussion of some problems to be addressed when case-based planning is to be applied to lesson planning within a tutoring system, which is the objective of our own research.

2 Case-based teaching principles

A reason for developing intelligent tutoring systems other than to realise a software package for instruction is to test out a theory—either psychological or pedagogical. In the systems we describe in the survey there is at least one pedagogical principle on test. Often, it is this principle that has motivated the application of case-based reasoning to the design of intelligent tutoring systems. In other cases, the pedagogical principle has been a secondary concern, but nevertheless an important one. In the following section, we present 15 pedagogical principles that have either explicitly or implicitly contributed to the development of case-based teaching systems. We do not necessarily concur with all them, but think it is useful to give attention to the principles as they provide some measure of the soundness of applying case-based reasoning in the various applications. Figure 1 presents the principles and their inter-relationships. The simple organization apparent in figure 1 imposes a structure that can be used to differentiate the systems that comprise these principles.

2.1 Story-based teaching

Good story telling has been promoted as an element of good teaching (Schank, 1991). A story is a case-study of a real situation presented as a sequence of video scenes. Stories are useful for teaching because information is presented in a more interesting way than the traditional presentation of dry facts as text. Through relating experiences in the form of stories, it is proposed that expert teachers can impart their knowledge to students. A basic assumption here is that an expert's knowledge is represented as a repository of cases, and students can become expert by building their own case-

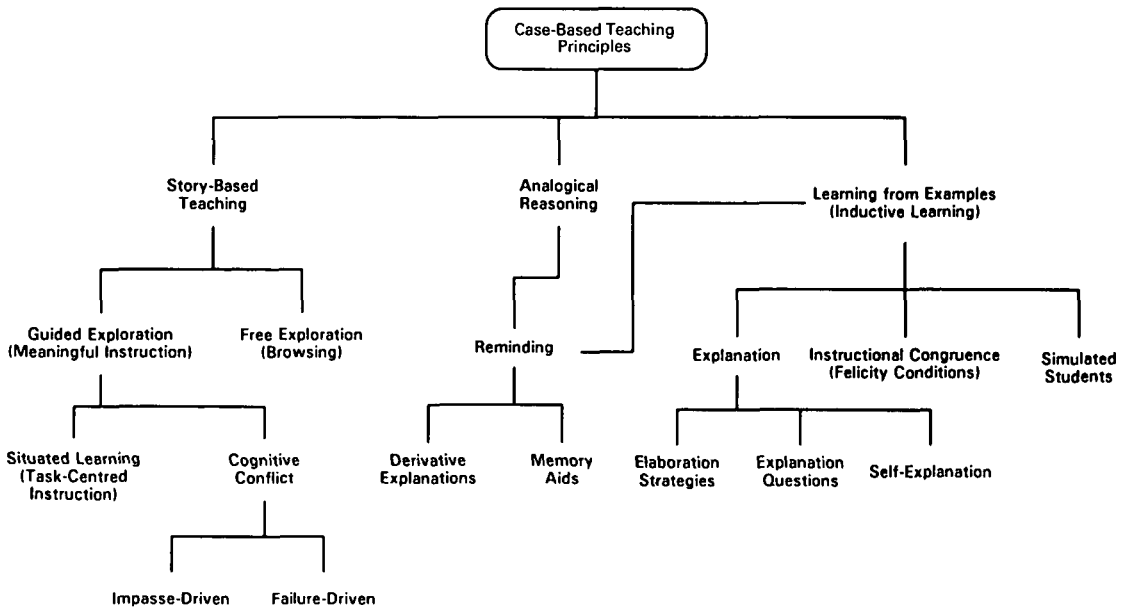

Figure 1 Principles of case-based teaching

base. The primary justification for this assumption is that people apparently use reasoning from cases in some domains, e.g. medicine, law and business (Kolodner, 1993). If this assumption is accepted, a basic principle of story-based teaching can be stated (c.f. Schank et al., 1991):

(Principle 1) story-based teaching *If expertise is best represented in terms of specific experiences then a natural way to facilitate the assimilation of new information by humans is to present that information in the form of cases or STORIES.*

Principle 1 is actualised in the ASK-TOM system (Schank et al., 1991), which is a multi-media browsing system based on a hypertext architecture. There is very little direction imposed on how students explore so ASK-TOM also incorporates the principle of *free exploration*. This principle advocates that students should explore a domain as they see best to develop or refine a model themselves, rather than having a model imposed on them as in traditional tutoring. Intelligent tutoring systems are used to support instruction by presenting information to students based on preconceived ideas of how best to make the presentation. In contrast, *learning environments* (we would classify ASK-TOM as a learning environment) let students effect their own ideas of how to learn, so emphasis behind these systems is self-discovery.

(Principle 2) free exploration/self-directed learning *Students are motivated to refine their extant cognitive model of a domain by self-directed exploration of an environment that allows that model to be tested.*

Just presenting a set of stories in response to student's exploration might not be sufficient to allow them to develop a cognitive model of the domain. Besides domain *knowledge*, students need to learn domain *structure*. Students need to know how one element of domain knowledge relates and influences another, and they need to learn how to recognise situations where some learned knowledge applies. In the MACH-III tutor (Kurland, 1992), for example, a *functional hierarchy* of domain rules is maintained, which among other benefits provides a well-organised body of knowledge that is easier for the student to learn. Rules that define actions in the domain can be presented in the context of the overall cognitive framework the student is to acquire. Similarly, presentation of well chosen cases or stories will facilitate development of a cognitive model of the domain. A third principle of story-based teaching can be stated now:

(Principle 3) meaningful instruction *Stories are best presented in a context enabling the listener to determine where they should be indexed in memory and how they should be connected to other stories.*

The *case-method* is an established technique for teaching in those domains where reasoning is predominantly case-based, e.g. medicine. GUIDON (Clancey, 1987) is a well known tutoring system that applies this method. What case-based reasoning enables is to soften the restriction imposed by the *hard-wiring* of cases with situations in which they are to be used. Case-based indexing allows cases and stories to be retrieved using a notion of *usefulness* that is context sensitive and which functions at multiple levels, e.g. surface or structural similarity, as well as other relations such as cause, function and purpose. As long as the tutor can specify to the student, either explicitly or implicitly, the relation between the retrieved case or story and the present context many different cases can be used in any situation, and many different situations can be satisfied by any one case. It is intended that consequently the student will develop a cognitive model of the domain that recognises the inter-relationships between the stories as they were presented, and therefore, the inter-relationships between knowledge contained within stories.

Principle 3 suggests that some system of constraint is needed to provide guidance to the student's exploration and to control the selection of stories. Traditional intelligent tutoring systems have been criticised for being too directive, while exploration systems too noncommittal, and it is suggested that a mixed architecture is better, *viz. Guided Discovery* (Elsom-Cook, 1990). Students are allowed to explore as they wish in this environment, and the tutor is permitted to be directive provided this direction is in keeping with the student's own goals. In such environments, the system

must deduce the student's goals if it is to ensure its direction is compatible with the student's objectives. This prerequisite indicates the necessity of a student model to infer the student's goals. However, stories are frequently represented as video clips, and the system has no knowledge of the contents of a story. Consequently, the system is handicapped when attempting to model the student's mental state when considering what knowledge is learned from cases. However, guidance is still needed, therefore, the notion of *task-centred instruction* (Bell et al., 1992) or *Goal-based Scenario* (Bell et al., 1993) is introduced to facilitate guided exploration.

2.2 Guided exploration

The objective of guided exploration is to develop systems that have the free learning features of self-guided exploration, and also have an underlying pedagogy. To facilitate free exploration and still maintain pedagogical relevance, story-based teaching takes the following position.

(Principle 4) impasse-driven learning *Stories should be used to illustrate pedagogical points only when the student needs to know information.*

In ASK-TOM some guidance is provided by limiting the scope of exploration the student is permitted through hard-wiring the possible paths. Students can explore as they want, but are limited to avenues provided for by a limited set of questions that are permanently linked to reflect pedagogical relevance. Students browse when they need information to overcome an impasse (e.g. curiosity poses a question they cannot yet answer themselves) by asking appropriate questions of the expert. However, hard-wired constraints are not in the spirit of discovery systems, so alternatives are needed that are more attuned to the free exploration ideology. Two further principles that impart guidance are now considered (c.f. Bell et al., 1992; and Schank, 1991):

(Principle 5) situated learning/task-centred instruction *Skills to be taught should be situated in tasks where knowledge is normally applied.*

(Principle 6) cognitive conflict/failure-driven learning *Students should be motivated to learn by providing for them to fail at a task.*

Principles 4 and 6 are based on the *cognitive conflict* hypothesis (e.g. Ohlsson, 1993), which asserts students are most receptive to information when they want to do something, but do not have the necessary knowledge. The best time for students to be presented with new information is when they want to learn. To capitalise on the cognitive conflict hypothesis, principle 5 advocates the notion of a task. The system should create a situation where students are faced with an interesting challenge and allowed to attempt a solution. They should be allowed to fail, and this failure results in the students being in a motivated state receptive to information. In the sense of inferring students' goals, the presence of a failure indicates to the system that students' goals are now "the attainment of the information that will allow them to complete the task". Stories can be selected from the case-base that are relevant to explicating the desired information. In this way, students not only learn the pedagogical point of the story, but also appreciate the context in which the point was made, e.g. reference to the failure experienced by students, relation to the other principles contained in the story and relation to other stories presented. Research on the Story Archive (Bariess et al., 1991) has as its goal to develop an indexing scheme that relates stories to each other to facilitate instructional goals.

The introduction of a task has the effect of imparting a coherent structure on the knowledge the student is required to have and, therefore, the knowledge that will be taught. Information presented to the student in the context of solving the task will be better *indexed* by the student also in the context of the task. The results of task-centred instruction are intended to be the following (c.f. Bell et al., 1992):

(i) student becomes aware of the context in which information will be helpful;
(ii) student is motivated to understand the content presented;

(iii) student is more likely to remember the content when appropriate in some subsequent real world situation.

Cases in the case-base are required to be indexed with reference to the task so that learning occurs in the context of performing the task. The greatest development effort is in designing a task that addresses all the instructional objectives that must be taught (see below for an example in human resource planning), and in maintaining an indexing scheme that allows relevant stories to be retrieved in the context of the student's attempts at solving the task. Therefore, stories must be indexed according to the features of the task. For example, in Bell et al. (1992), general problem descriptions are defined and proposed for indexing stories. When a student experiences a particular problem, a match is sought between the problem and a problem description to retrieve a story that helps.

An example of a task designed to meet two instructional goals in human resource planning is shown below:

Instruction goals

(1) The learner will be aware of the impact of human resource decisions on the health company.
(2) The learner will be familiar with the life cycle of an employee and the kind of issues that must be dealt with in each phase of that cycle.

Task

You are the manager of the human resources department of the case company. You must make human resource decisions for the next four years, without destroying the company. To make your job easier, you will only make decisions for three employees. The computer will generalise your decisions to the rest of the staff.

Schank (1991) has specified a four step procedure that incorporates the principles discussed above:

(1) Present an interesting situation for a student to try out (SITUATION).
(2) Allow him or her to fail (FAILURE).
(3) Have a story to tell that which relates to what the student did wrong (INDEXING).
(4) Tell the story when consulted (STORY TELLING).

Several instructional systems have been developed using this plan, and we describe two in this survey—DUSTIN and CREANIMATE. We also describe two systems based on the task-centred approach—Sickle Cell Counselor and DECIDER.

2.3 Learning from examples

The distinction between stories and examples is not definite, but we make a distinction based on stories being a sequence of events or dialogue usually represented as video clips that illustrate a pedagogical point. Examples are symbolized typically by schema-based memory structures that contain mainly factual information. Also, contents of a story are not reasoned about by the system, whereas an example can be, for instance, adapted to create a new one.

Hayes-Roth (1977) describes a theory of learning from examples that proposes humans learn general rules abstracted directly form properties of examples. Each test example "causes learning of relevant and irrelevant criteria that are consistent with it". To maximise learning of relevant criteria, critical properties of the knowledge to be learned must be identified, and examples chosen that ensure inclusion of all these properties. To minimise learning of irrelevant criteria, examples presented must minimise inclusion of irrelevant criteria and maximise variability on irrelevant properties within multiple examples. Vanlehn has further considered requirements for maximising learning in the SIERRA system.

SIERRA (Vanlehn, 1987) is an induction system that uses the ordering of lessons to aid induction of procedures. It is based on *inductive learning from examples*, in which procedures are communicated to a student through successive examples, and the student must generalise

procedures from these examples. This style of teaching comprises Vanlehn's *felicity conditions*, which are elements of good induction-based teaching observed in teachers, and guide the sequence of example presentation to maximise learning. Therefore, learning from examples requires a sequence of well chosen examples from which the right concept can be inferred. As with story-based teaching, case-based reasoning is used in example-based learning to provide a well indexed repository of examples from which a tutoring system can retrieve the most appropriate examples in response to the student's needs.

(Principle 7) inductive learning *People learn domain knowledge by inducing generalized rules from given examples. Therefore, teaching can proceed by presenting well chosen examples.*

CABAT (Schult, 1993) is an environment that uses case-based reasoning to maintain an indexing scheme around *structural* and *superficial similarity* features. Its primary objective is to organize examples in a way that allows retrieval during different tutorial contexts. CATO (Ashley *et al.*, 1992) maintains a case base organised around *Dialectical Arguments* from which a small set of cases is retrieved to meet the requirements of an instructional goal. These systems function on the following principle:

(Principle 8) instructional congruence/felicity conditions *A conducive selection of examples will ensure realisation of intended instructional goals and avoid inducing misunderstanding.*

Besides indexing, case-based reasoning has been used to test the principle of learning from examples to determine how student's generalise from specific examples to produce example-independent knowledge, and how such learning can be facilitated.

ELM (Weber, 1993) models the process of knowledge acquisition from examples as a student progresses from novice to expert. It performs cognitive analysis of LISP code provided by students to generate student models. It is based on the following hypothesis:

(Principle 9) reminding/analogical reasoning *Students use reminding of prior solutions to solve new problems, and later generalise to form problem-independent knowledge. Analysis of this can indicate students' understanding.*

Similar objectives to those of ELM are derived from the research of Chi et al. (1989) on the effect of inter-differences between students on how well they learn and generalise knowledge from examples. It is conjectured that how well students learn from worked-out examples depends upon how well they have encoded the declarative knowledge contained within the examples. This, in turn, depends upon how well students can explicate implicit information in examples through constructing an explanation that relates elements of the example to basic principles. Only by forming conclusions and making inferences can students generalise knowledge to beyond example-specific form. The first corollary to this research is tested in AXE (Reimann et al., 1993):

(Principle 10) elaboration strategies *Students can learn to create good self-explanations if they are taught good strategies for elaborating the contents of worked-out examples.*

In AXE, the position is taken that if an expert demonstrates good elaboration strategies to students, they in turn can learn to generate good self-explanations that result in improved understanding. A computer model is developed that can generate explanations of given examples, and a case-based component is used to store these explanations in a problem-solving component. The case-based problem solver is a test of how well various elaboration strategies can elicit information from examples, and how declarative knowledge is encoded in memory.

The second corollary to the research on self-explanations is tested and implemented in Dynamic Knowledge Base (DKB) (Fernandez-Valmayor et al., 1992):

(Principle 11) self-explanation *Students learn by constructing explanations that help them understand content, so students' understanding can be tested by analysing their explanations.*

DKB seeks to apply case-based reasoning to mimic the process of encoding declarative knowledge. Its aim is to generate a model that represents the understanding that would result from the

explanation given to it. Through natural language processing based on *Conceptual Dependency Theory* (Schank, 1975), which postulates that the meaning of an utterance can be derived by interpreting it in the context of stored memory structures, DKB interprets the explanations provided by a student. From these, it generates a student model that represents the student's level of understanding.

CELIA (Kolodner, 1993) is a learner modelling system that represents the memory and reasoning capabilities of novice troubleshooters. Besides representing episodic knowledge of troubleshooting experiences as cases, CELIA also represents a causal model of the domain and knowledge of the troubleshooting process. CELIA learns by predicting what the teacher will do and explaining to itself any contradictions with what actually is done. It recognises that novices have representations of domain knowledge that are insufficient to interpret examples, and provides help for the self-explanation phase through an *apprenticeship* mode. The principle of *Simulated Students* is also "implicitly" incorporated in CELIA.

(Principle 12 Simulated Students *A cognitive model that learns from the instruction presented to students will indicate the likely knowledge state of those students.*

A simulated student (Vanlehn et al., 1994) is a machine learning model that responds to instruction (usually as examples), and updates its knowledge of the domain as a result. Sierra, mentioned above, is an example of a simulated student that learns by induction. Simulated students can be used in tutoring systems to indicate to the student model the student's knowledge state. Whereas a student model uses the interaction between a human student and the tutor to develop a model of the student's understanding, a simulated student learns in collaboration with a human student, and is subjected to the same instructional experiences. The hypothesis is that if the mechanisms by which the simulated student learns are the same as the human student's, then the simulated student will veridically depict what the human student has learned from the same instructional experiences. Although no case-based teaching systems have been found in the literature that specifically purport to being simulated students, we present CELIA as essentially having the same purpose and with potential for this role in tutoring.

2.4 Explanation questions

Schank (1986) forwards the notion of *explanation questions* as a means of indexing a memory of stories to answer specific questions. The basis for explanation questions lies in the following assertion. To understand something, people formulate questions that if answered will constitute the explanation they are looking for. The question they formulate is called an *explanation question* (EQ). It has been assumed that there is a finite set of EQs for a domain, and the task is to retrieve one from this set. Since EQs are finite and enumerated, it is possible to provide answers to these EQs and store them in memory. Therefore, the process of formulating a question (i.e., selecting an EQ) will automatically provide the answer. EQs are essentially indices to memory. The greatest reasoning is involved in converting a specific question about a domain into one of the general EQs, from which an answer can be obtained.

(Principle 13) explanation questions *Students learn from the answers given to a set of probing questions.*

Applying explanation questions is a process of matching features of the specific question to features of standard EQs. Explanation questions serve as a medium for indexing and retrieval of stories, which help to explain something about the situation. The pedagogical principle here is that, to understand something, students must ask enough of the right questions to elicit sufficient information for them to make the necessary generalisations, allowing them to understand the situation and transfer this knowledge to problem-solving. Since a student is not knowledgeable enough to ask the right questions, dialogue is initiated by the system and involves the system "asking" as many explanation questions it needs to allow the student to appreciate the pedagogical

goal. The system poses the right questions and provides answers via video clips, from which it is intended the student can internalise the information in a meaningful and structured model of the domain. The immediacy of presentation of appropriate video clips with EQs helps to highlight the implication of information contained in the clip to the didactic intention associated with the question that was posed. We describe CREANIMATE, which makes use of explanation questions.

A different approach to using prior explanations than information questions is implemented in work done on the SHERLOCK learning environment (Rosenblum et al., 1993). Whereas the emphasis with explanation questions is to answer a question by retrieving a stored response, case-based reasoning is applied in SHERLOCK to retrieve previous dialogue between the system and the student in order to set the context for the new explanation. This work is based on the following observation:

(Principle 14) derivative explanations *Teachers incorporate their prior explanations in the derivation of new explanations.*

The new explanation is not strictly new, and it is not an adaptation of an old explanation; it is a derivative of prior dialogue. The history of interaction between teacher and student contributes to defining and placing in context new dialogue, rather than being incorporated verbatim in any new explanation. We describe the architecture developed so far for retrieving relevant prior explanations from a dialogue history.

Like most applications of case-based reasoning described here, there is some recognition of the affinity with human reasoning processes. Intuitively, psychological plausibility might be a desirable feature in a tutoring system, though whether or not the degree of relevance to human reasoning has any tangible effect on how students learn is debateable. However, there are elements that benefit from closer affinity to human reasoning, e.g., causal explanations are favoured by people (Clancey, 1987). Case-based reasoning forwards the argument that if people reason by cases, then they should be taught by cases, therefore knowledge should be represented in case form. This is the justification for story-based teaching discussed before. There might also be benefit from a system that seeks to model all its reasoning on human cognitive processes. In this survey, we present systems that perform one of the following: student modelling; natural language processing; presentation of instruction; and problem-solving. The applications case-based reasoning has been put to and the techniques used are each different, but a concerted effort could conceivably produce a tutoring system entirely based on a case-based approach. One research work (Bumbaca, 1988) described here seeks to realise this end by analysing the knowledge level requirements of an *ideal* tutoring system. It then shows how a memory structure approach based on scripts can provide the symbol level representation needed to provide psychological relevance to the knowledge level requirements. We describe an example from this work of an interaction between a student and the system in which the system presents material to the student during instruction by applying conceptual dependency theory.

2.5 Memory aids

All the systems mentioned so far have the objective of being incorporated within an autonomous computer tutor, which should make all decisions about students' instruction. An alternate view to this is where responsibility for decision making is shared between the computer and the user; Kolodner (1991 a) has called this *decision aiding*. The principle of decision aiding is that humans are inferior to computers when it comes to a test of memory, but are superior at creative tasks like applying analogical reasoning to derive new solutions from past ones, therefore it is advantageous to combine the strong features of each. Systems of this kind contain large repositories of cases that serve as external memories for humans. These systems present cases to serve as remindings for humans, who then perform case-based reasoning to adapt, critique or evaluate solutions. Most of the decision making is done by the human, while the computer is an assistant.

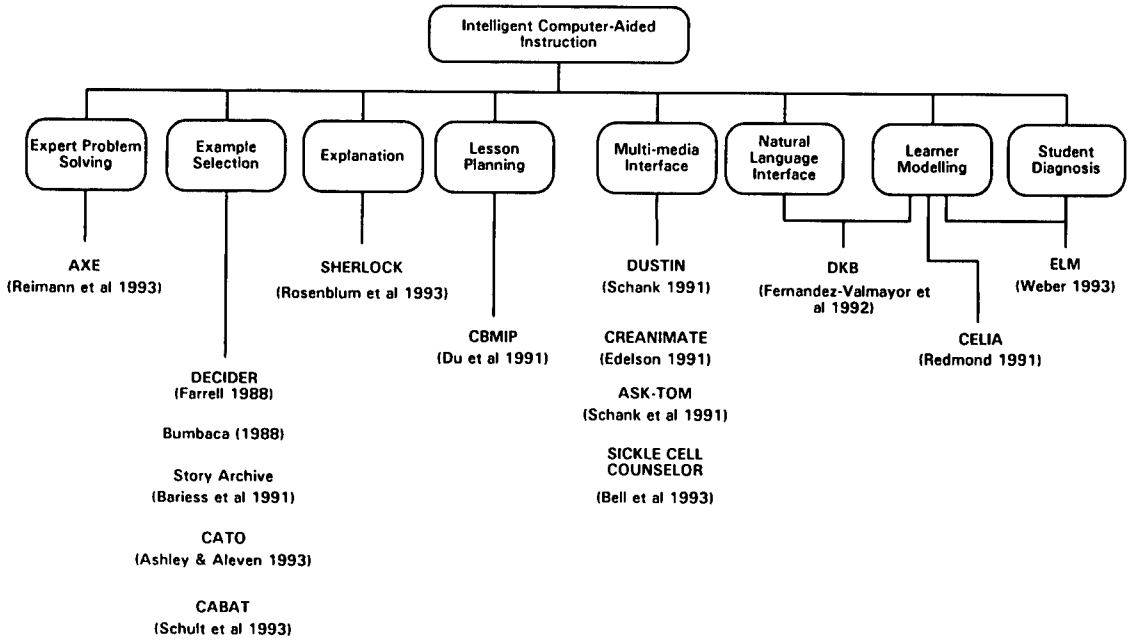

Figure 2 Case-based teaching systems organised by the tutoring function served

(Principle 15) memory aid *Humans function better when assisted by an external memory that aids them with analogical reasoning.*

The Science Education Advisor (SCI-ED) (Kolodner, 1991 b, 1993) is a decision aiding system for assisting teachers with the creation, critiquing and evaluation of plans for lessons in science. It is in the vain of the story-based browsing systems described before, as it also provides stories to serve as guidelines on issues of concern for teachers. Another decision aid for lesson planning called CBMIP (Du et al., 1991) is described in this survey. This system plans lessons in mathematics and outputs the results to a teacher. We describe this system because it is closest to our own work in case-based lesson planning.

2.6 Summary

Case-based teaching systems have been developed on pedagogical principles, of which some are based in theoretical results, with others on observations of student behaviour and teaching practice. In many systems, the principles have been the driving force behind using case-based reasoning techniques, for example, to test out theories of knowledge acquisition and inductive learning. Other systems have utilised case-based reasoning as one suitable method of implementing a pedagogical principle, for example, free exploration using stories and example retrieval in case-method teaching. In summary, we can distinguish case-based teaching principles into three classes, as depicted by figure 1: those relating to inductive learning; those relating to analogical reasoning; and those relating to exploration. These classes are not mutually exclusive, since some principles have relevance to more than one class. Instructional congruence, for example, is mainly concerned with inductive learning, but there is also some amount of analogical reasoning, since generalisation will involve a certain amount of reasoning from previously presented examples. Impasse-driven learning will involve aspects of self-explanation, and free exploration will involve reminding to similar paths. In the following section, we describe the systems mentioned previously in more detail, and further discuss the principles they comprise.

3 Case-based teaching systems

Figure 2 shows the systems we discuss classified by the main function they serve in ICAI. As with the principles underlying these systems, they do not belong exclusively in just one class. Systems

are classified under "multi-media interface" because they use video to present information, but they also serve functions of "explanation" and "example selection". Similarly, systems classified under "example selection" also serve "explanation" functions. The systems are presented in the order in which the main principle each comprises was discussed in section 2.

3.1 Story Archive—indexing

In support of story-based tutoring, a major research effort at Northwestern University is under way to create a large repository of video stories that adduce a diverse range of human experience. The Story Archive (Bariess et al., 1991) has as its objective to provide case-based tutoring systems with a rich source of video stories that serve pedagogical goals. Case indexing and retrieval are central concerns to enable this. The pedagogical goal is selected by the student or by a tutor, and the Story Archive is charged with selecting and presenting stories that demonstrate a specified *thematic* point in relation to the pedagogical goal of consideration. The instructional premise is the same as ASK-TOM and other story-based tutors, namely the presentation of the right story at the right time will allow the student to draw the necessary information in a structured and meaningful way that further allows problem-solving.

A case or story in the archive is indexed according to one or more of a finite set of *thematic* points. An example of a thematic point is *Plans for achieving a goal may succeed as a result of agents who are exceptionally skilled.* The set of thematic points is sufficiently general so that it relates to the entire archive and every story can be characterised from this set. The set is formed into a classification hierarchy, which allows cases to be placed in the category that best captures the lesson or point of the story. Some stories are placed in more than one category when several thematic points are relevant.

Besides thematic indices, a case is also indexed according to the domain concepts the story relates, i.e. the subject of the story such as politics or physics. Importantly, thematic indices take precedence over domain concepts because the underlying lesson of a story is more important for instruction than any surface similarity. The function of domain concepts is relegated to discriminating several thematically relevant stories that may be retrieved. Therefore, during retrieval a case is selected when a request is made that specifies a thematic point to be made. The Story Archive searches for matches in the thematic hierarchy and chooses from among suitable stories by assessing the degree of domain relevance. Sometimes stories from very different domains may be used together for reinforcement provided they have the same lesson to tell—same thematic point.

Stories often have more than one pedagogical point to make, so a way is needed to focus the student on the desired point. This is achieved in the Story Archive by a technique called *bridging*, which inserts text prior to the presentation of the story to change the student's expectations to what point is to be made. Since stories are retrieved when a link exists between thematic points, a bridge helps to relate the student's present context to the subsequent story by revealing the analogy between the present theme and the instantiation of the theme in the story through comparing role-fillers. By comparing different themes different pedagogical points can be focused on.

3.2 ASK-TOM—free exploration

ASK-TOM (Schank et al., 1991) was developed to investigate the usefulness of video over purely verbal or textual forms of presentation, especially in the context of communicating with a student. The ASK systems are a range of hypermedia browsing systems with the goal of providing information to novices in the form of facts and guidance. In terms of tutoring systems, ASK-TOM is an exploratory system in which the student must discover knowledge through self-guidance. Communication is designed to be a dialogue between a student and a domain expert. The premise is that the student can learn in a more meaningful way by having a conversation with an expert. The student asks questions of the expert (TOM), who replies by presenting video clips that in some way provide the information relevant to the student's questions. The video case library is organised in a

hypertext-like network that presents follow-up clips to answer follow-up questions from the student. The pedagogical principle behind this is that students ask questions when they are most receptive to information, and TOM provides the domain knowledge in the form of suitably related video cases. The intended result of this is that the student is able to internalise the information contained in the video cases in the context of the relevance of the case to the question and its relation to other presented cases. In this way, the student develops a representation of the *structure* of the domain. This domain model encompasses an associative network of information assimilated from the video cases that includes objects and relationships in the domain plus problem-solving knowledge. Together, this should allow transfer of knowledge to enable the student to solve problems.

The domain of ASK-TOM is Trust Banking. The domain is represented by a conceptual map of the most important entities in trust banking designated *Big Picture Models*. These nodes are the major players in Trust Banking (e.g. the trust department of a bank, the individual consultant) and are related by links that represent their inter-relationships. The set of big-picture models serves to specify the initial areas of interest the student can investigate further. More specific big-pictures can be displayed if the student activates one of the top nodes or a link between nodes. At the next level in the network are *Themes* that serve to organise the individual video clips according to the relevance of the "lesson" contained in the clip to problem-solving. A theme (see description of Story Archive) relates to a lesson relevant to the big-picture model it is contained in, and is displayed when the student *focuses* on a big-picture model. Each theme contains clips that are related by *low-level links*. When a clip is chosen by the student from the theme screen, a clip screen is displayed that contains the chosen clip accompanied by eight other clips. These additional clips provide the facility for follow-up questions, and each is labelled by one question related to information in the central clip.

The eight low-level links are constant throughout the network and are: *Background—historical information about the situation described by the clip; Results—describes consequences of the situation; Examples—more detailed examples of the situation; Context—places the clip's lesson into context; Warnings—highlights negative aspects of the situation in a particular context; Opportunities—highlights positive aspects of the situation in a particular context; Alternatives—presents alternative situations in a similar context that may arise; Indicators—links a clip to features that facilitate detection of the situation within the contexts it might arise*. Control is with students who browse the network in a way that provides them with information on questions they want answered.

The links between stories are fixed so that the order of retrieval depends upon where the student explores, and there is no partial matching. Additionally, because the system has no knowledge of the contents of a story, there is no facility for case adaptation. However, work on Bridging allows text to be inserted between the situation and the story to alter the context and the student's expectations of the story.

Contrary to knowledge intensive approaches to tutoring, in which domain knowledge is modelled to allow expert performance for, among other things, to enable deductive diagnostic student modelling, a story-based approach has no explicit knowledge of the contents of a story other than the specified and fixed relationships indicated by the network. There is limited knowledge provided by the class of clip, which indicates the purpose or use that each clip has, for example background, warning or any of the eight classes. Because the system has no knowledge of case content, it is limited in its ability to reason about domain knowledge. In particular, traditional approaches to deductive diagnosis would find it difficult to determine a model of the student's knowledge state. This is not significant in the version of ASK-TOM described here, because there is no pedagogical component that could take advantage of information provided by the student model. However, in its present form ASK-TOM is purely a discovery system, and research has suggested that some pedagogical constraints are needed to improve learning capability from such systems (Elsom-Cook, 1990). New versions intend to incorporate a planning component that would require some indication of what the student knows.

The instructional principle behind having a planner or task manager is that students are more receptive to information when they want to know something. And we can arrange for students to want to know by providing an opportunity where they are presented a situation and fail. The task manager arranges for an appropriate situation to be presented. Once the student has failed, the system can provide appropriate advice via video clips. Two systems that apply the task-centred approach are described next.

3.3 DUSTIN—task guided exploration

DUSTIN (Schank, 1991) is a multi-media environment designed to teach English as a second language. Its approach is to present simulations in the form of video clips of situations that the student is likely to encounter while staying at an educational centre in St. Charles, Illinois, USA. The emphasis is on preparing foreign students for attending language classes in the USA, and freeing them from the problems associated with entering a foreign country and finding their way around.

DUSTIN follows Schank's four step plan outlined in section 1 for each "lesson". First the student is presented a SITUATION and given a goal to achieve, which is relevant to the student's real needs while staying at the educational centre. If the task is to check in at a hotel, DUSTIN retrieves a video clip that presents a receptionist speaking in English. The student must respond also in English by typing a response. This situation serves as a kind of pretest that assesses the student's competence in the particular situation. The student's response to the video clip dictates whether further instruction is required for the situation, or whether the student can proceed to the next situation. In DUSTIN there is no student model, and instead students' goals are induced directly from the assumption that having failed to perform a task presented students will want to learn how to perform that task, and this becomes their instructional objective.

If students do not respond correctly, their response is considered FAILURE. Therefore, DUSTIN seeks to retrieve cases of the correct response and presents those video clips as examples. A case is INDEXED by the situation it relates to. An example constitutes a STORY that conveys information the student wants, i.e. the correct response in English. Following this, the student is again presented the situation and asked to respond. Should failure occur again, DUSTIN breaks down the situation into smaller parts that are taught separately. The student is again tested in the same situation, and if failure occurs again a new situation is presented.

The learning strategy used in DUSTIN is intended to help students structure knowledge in the context of a realistic situation so they get used to using their learned knowledge in situations it is useful.

3.4 CREANIMATE—aesopic teaching

CREANIMATE (Edelson 1991, 1992; Schank, 1991) presents video clips of animals exhibiting various behaviours such as flying. From these it is intended the student will appreciate which features of an animal enable it to perform a particular behaviour. Guidance during exploration of the video library is provided by the system giving students tasks to perform. This allows them to view video clips that relate to the task. Instructional goals are specified by students themselves, since a function of CREANIMATE is to help students design new mythical animals, for example a flying cow.

An animal proposed by the student forms the SITUATION. A FAILURE occurs when a feature added to an animal by the student does not achieve the behaviour the student intended. When this happens, CREANIMATE retrieves a case that illustrates some point about the student's attempt. A case is INDEXED in two ways: (1) according to the specific animals that appear in a case using features that achieve their behaviour and (2) according to the general principles a case illustrates. A case is selected when a specific link exists between the video clip and

the student's animal. A STORY in CREANIMATE is the running of a video clip of an animal applying a particular feature to exhibit a relevant behaviour.

The success of CREANIMATE lies in its ability to retrieve relevant stories, which requires a richly indexed video case library. Research emphasis has been on indexing and retrieval strategies employing a dialogue library of explanation questions (EQ) and a story library, both sharing a common indexing scheme.

Explanation questions are used to index memory to continue a dialogue with students, and to select appropriate stories. EQs are chosen that are important in some respect to understanding how animals behave and survive in wild. Collectively, these questions form the EQ set for this domain—animal morphology. An example EQ is "*How does this animal use a particular feature to help it survive?*" Answers to this questions are provided through indexing to video clips that serve as examples, illustrating different animals using different features to survive. Other EQs index their own video clips. Case-based reasoning is used to retrieve and instantiate suitable EQs from a dialogue library by using features from the index for stories.

A story is indexed according to which EQs it can serve. The choice of indices is based on what is considered important to know in this domain. Each clip is considered and indexed according to the lessons it contains (see below for an example of a story index). One important lesson is the influence of physical features on an animals function, so an index is kept that pairs physical features with animal functions. For example: feature: "long, muscular legs" is paired with function: "run fast". Another pair is created between function and behaviour: e.g. function: "run fast" with behaviour: "pursue prey". An index for a story about a cheetah running after prey is shown below:

```
(defframe story cheetah-pursuing prey)
  :title "Cheetah Pursuing Prey"
  :Video (list (list 1822 1910 *african-video* "A")) ; ; video address
  :indices ([index   :animals [a cheetah]
                     :point ([plan :function [function move-fast]
                                   :behaviour [behaviour hunt]])
                     :plans ([plan :function [function run-fast]
                                   :behaviour [behaviour pursue-prey]])
                     :features nil
                     :behaviours ([behaviour pursue-prey])
                     :functions ([function run-fast])
                     :goals nil
                     :feafuns nil]))
```

An index is represented as a frame with slots: **Goals**: basic survival goals shared by all animals, e.g. *nourishment*; **Behaviours**: activities that animals use to achieve goals, e.g. *forage*; **Features**: physical features or qualities, e.g. *eyes*; **Functions**: primitive acts that features enable an animal to perform in order to pursue behaviours, e.g. *seeing*: **Feafuns**: a relationship linking an animal with a feature and the function it enables for that animal, e.g. *eyes to see*; **Plans**: a relationship linking an animal with an action it possesses and the behaviour that the animal uses the action to achieve, e.g. *seeing to look for food*: **Animals**: represented in terms of the features, actions and behaviours they possess.

When students suggest mythical animals, which they do by suggesting changes to real animals, CREANIMATE searches for EQs that relate, for example, to the features changed:

Student's suggestion:	"a tortoise with long legs"
CREANIMATE response	"Why would it be useful for a tortoise to have long legs?"

To support this EQ, CREANIMATE will search for and present video clips that have a feature/function pair that includes "long legs". "Long legs" is an abstraction (in an abstraction hierarchy) of "long, muscular legs" so clips that contain feature/function pairs with "long, muscular legs" can be

presented too as follow up lessons. Additionally, follow-up EQs might focus on the function/behaviour relationship of "long legs", and clips with the appropriate function/pair value can be presented.

As with other story-based systems like ASK-TOM, CREANIMATE has no knowledge of the contents of a video clip. The greatest amount of reasoning is in selecting an EQ from which a video clip is retrieved. However, as with CBR in general, a retrieved case might not be applicable directly. With explanation patterns (Schank, 1986) a retrieved explanation might need tweaking, in CBR a case could need adaptation, but in story-based learning it is not possible to adapt the story, i.e. the video clip. Therefore, the adaptation must be targeted at the "question" and not the answer. Either the context can be changed via bridging, as in Story Archive to change students' expectations of the video clip and help them focus on the intended thematic point of the clip, or the system must rely on the student's own abilities to place in context a video clip that does not obviously associate. This latter constraint puts even more reliance on having an extensive video library from which to choose clips.

3.5 SICKLE CELL COUNSELOR—goal-based scenario guided exploration (Bell et al., 1993)

The student is tasked with advising couples on the risk of their children inheriting Sickle Cell disease. The pedagogical principle here is that, to structure exploration so that the student comes away having explored the desired target skills and knowledge, these skills need to be developed within a single context, i.e. the task of counselling. This approach to instruction has been called *Goal-Based Scenario*. By giving the student a specific role, two requirements of story-based learning are fulfilled: the student has a reason for eliciting information from the system and is receptive; and guidance during exploration is provided, since the student will explore with relevance to the task. The student is given the opportunity to ask pre-specified questions of experts, perform lab tests and offer advice. In a reply to a question, a video clip is presented in which an expert supplies the answer. Three experts are available with each offering answers in different aspects of counselling. Similar to ASK-TOM, follow-up questions are available for each initial question. The major research issue here is in organising a video library in support of individual pedagogical goals so that all necessary skills are learned in a structured way. Goal-Based Scenario is the framework that is intended to facilitate this.

3.6 DECIDER—facilitated self-learning questioning assumptions (Farrell, 1988)

The pedagogical principle of DECIDER and of the *Self-Education AID* methodology is to allow students to formulate their own instructional goals so as to avoid falsely attributing failure to a student's solution if the student has pursued goals other than that assumed by the tutor. Feedback is improved if the tutor takes the stance that students' attempts at a solution are "reasonable" in light of their own goals, and so the task of the tutor is to identify those goals, infer any erroneous assumptions made by students from their solution, and then to highlight what is wrong by presenting counter-examples. In this system, examples are video clips of historical foreign policy decisions.

The lessons conveyed by cases include highlighting the possible invalidity of an assumption by presenting cases where the assumption did not hold true, and presenting one explanation for the assumption not holding true in this case. Video clips are presented in the context of erroneous assumptions, that is, at a time when the student is receptive to information, and impart a structure by being introduced at appropriate times during the student's attempts at creating a plan.

DECIDER sets students a task that requires them to create a plan in foreign policy decision making. Causal analysis is used to predict what POLICY (a political model that relates plans to goals or conditions) the student will apply. From this, DECIDER can predict what the student's goals will be, since it assumes they will be those specified in the prototype POLICY model. These predicted goals are matched to the actual goals specified by the student, which will either confirm or

rebut the predicted POLICY. In the latter case a new POLICY is sought that has a condition that matches the student's specified goal.

DECIDER tries to make aware to students the implicit assumptions in the model that matches their goals by trying to develop a plausible explanation of why the assumption does not hold true in this case. It supports this explanation by retrieving a case where the explanation has been valid, and displaying dramatic video sequences and story-like text. Consequently, cases are indexed according to the POLICY models they relate to.

Although there is no strong student model that has knowledge of the contents of the cases or of the correctness of solutions, DECIDER does benefit from weak student modelling. Guidance is provided according to the inferred goals of the student, and the inference is provided by a detailed causal model. The results of student modelling directly influence the choice of cases presented. However, research issues in DECIDER are emphasised towards causal inference of student goals over application of case-based reasoning. Nevertheless, DECIDER was an early illustration of the role of CBR in instruction, and is interesting in its facility for guiding student exploration supported by cases.

3.7 CATO—dialectical examples for argumentation (Ashley et al., 1992)

CATO is an exercise in reconfiguring an existing case-based reasoning system, HYPO (Ashley, 1991), to allow domain knowledge to be made more explicit to facilitate tutoring. The research emphasis has been on the generation of examples in support of legal argument. CATO is intended to teach students the skills of analysing legal cases where no judgement has been made, and to construct legal arguments in favour of the plaintiff or defendant. Instruction is conducted by a human tutor setting an instructional goal, from which CATO retrieves a small set of cases called an *argument context*. An argument context embodies a particular lesson that focuses on certain argumentation *issues*. CATO's function is to generate a set of cases that together bring up the required issue, and force the student to deal with it.

A legal argument basically entails identifying past cases that can serve as precedents for the present case. All relevant factors in the precedent case must be highlighted to demonstrate their relevance to the present case. These factors are used for representing cases, as is the outcome. During retrieval similar cases that can serve as precedents are identified by the number of factors they share and differ on with the present case, and if the outcome was the one argued for. *Trade Secrets Misappropriation* is the specific legal domain considered in CATO, and factors for this include: *Security-measures*—whether security measures were taken, and *Agreed-Not-To-Disclose*—whether non-disclosure agreements were signed.

Six issues or instructional goals have been included in CATO (Aleven et al., 1992):

(issue 1) Identifying factors that strengthen/weaken each side's position
(issue 2) Citing a case as a precedent in support of own position
(issue 3) Distinguishing an opponent's cited case to refute its significance
(issue 4) Citing a counterexample to respond to opponent's precedent
(issue 5) Preferring the more on point precedent (most relevant factors)
(issue 6) Preferring the nontrumped point for which there is no better counterexample.

When an issue is input CATO attempts to retrieve a set of up to four or five cases to form an argument context. The cases are organised in an acyclic directed graph called a *claim lattice*. Each node represents at least one case, and starting from the root node, which represents the current situation, further along the graph the cases become less "on point", i.e. have fewer common factors. Each factor in the case is assigned either being supportive of the plaintiff or defendant's position.

Instruction proceeds initially with the human instructor asking the student to study these cases and consider the relative position of each party based on the factors (issue 1). Then the student is asked to make a legal argument for one side to win. This requires the student to choose between the

relevant cases, and cite one or other of the cases in the argument context as best precedent (issues 2, 5 and 6). Other issues can be introduced by asking the student to argue against the cited precedent (issues 3 and 4). The pedagogical principle is that students get training in performing tasks they will actually have to do for real.

To facilitate instruction, five types of argument contexts are available and can be used for different pedagogical aims. These are standard configurations of cases that make a point effectively, and are called *dialectical examples* (Ashley et al., 1992). Their use in legal reasoning is supporting claims made about the importance of particular factors in justifying a cited case as a valid precedent; they also have pedagogical usefulness and essentially constitute retrieval strategies for instructional goals.

The five types of dialectical examples are:

(1) *Representative examples* (*Vanilla* in which at most two factors are included in a case and both favour the winner; *single conflict* in which only one factor is consistent with the outcome while all others conflict; *packed* in which one party has majority of factors in support): these cases are used to introduce the factors in the domain to the student simply, and such that the effect of factors on case outcomes is emphasised. These cases serve Issue 1.
(2) *Conflict resolution examples*: resolve situations where both sides present sets of factors and contend that theirs is more important. A case is sought in such situations that contains both sets of factors and has been resolved in favour of a particular side. These cases are useful for teaching students about conflict resolution and serve Issues 2, 3, 4 and 5.
(3) *Refutational comparisons (counterexamples)*: refute an assertion of the support given by a set of factors to an opponents position. Counterexamples are sought that include those factors, but were judged against the opponent's position. Trumping occurs when the counterexample not only refutes the opponent's case but also supports your own. These cases are useful for teaching students about trumping and serve Issues 3, 4, 5 and 6.
(4) *Ceteris paribus comparisons*: highlight the importance of a single factor to the outcome of cases that otherwise are identical. Pairs of cases are presented in which all but one factor are the same, and the outcome differed for the presence of that factor. These cases serve Issue 1.
(5) *Coherence example*: a group of cases that, taken together, counter an opponent's strengths, though individually they contribute minimally. These cases serve Issues 3 and 5.

Dialectical examples are rhetorical skills that need to be learned by students to strengthen their legal arguments. Consequently, these argument contexts have been designed into the system so that an instructor can specify what instructional context is needed, and CATO will generate the appropriate set of examples. The student is thereby given an opportunity to practice these skills.

There is no student model or instructional planner in CATO, so pedagogical reasoning is limited to generating argument contexts in response to required issues, which are input by a human instructor. Also, there is no feedback or guidance provided by the system, and so we would classify CATO in its present form as a memory-aid to be used by a human instructor rather than a tutoring system or self-guided exploration. (See Ashley et al., 1991, for a conceptual description of a proposed ITS that includes these components.)

Systems discussed now are concerned with applying the memory organisation theory underlying the case-based methodology to facilitate development of instructional systems, and provide better communication between the system and the student. This includes explanation of the system's reasoning, and natural language processing in support of student modelling.

3.8 DKB—Dynamic Knowledge Base—conceptual dependency + memory organization

The application of a dynamic memory based on the *Memory Organization Package* (MOP) (Reisbeck et al., 1989) to the task of student modelling is considered in this system. At the stage of development as reported in Fernandez-Valmayor et al. (1992), the system was an experimental effort and some way from being operational. The objective thus far has been to test the knowledge

representation mechanisms and knowledge acquisition processes that would be needed eventually to allow student modelling. The focus of research with the Dynamic Knowledge Base (DKB) is in allowing a tutoring system to improve its understanding of the domain via interactions with an expert teacher, and to create a contemporaneous representation of the understanding of students via interaction with them. This approach to student modelling is of the type that seek to simulate the learning process. Of central concern is modelling how people (students and experts) incorporate and organise textual information into their memories; for example, while trying to understand a textual explanation of a problem.

The pedagogical issue in this work derives from the research of Chi et al. (1989), which suggests students learn by constructing explanations that help them understand the worked-out examples they are presented. A good student will be able to generalise from a few examples to produce example dependent knowledge. A good explanation must contain certain necessary elements such as reference to fundamental laws and principles for the domain (Hempel, 1942). Therefore, it is implied that we can test a student's understanding by considering these self-explanations to see if they contain certain elements; Chi et al. specify four such elements:

(1) Conditions of application of action.
(2) Consequences of actions.
(3) Relationship of actions to goals.
(4) Relationship of goals and actions to natural laws.

A good explanation suggests the student has understood the example and is likely to do well. An outcome of this analysis is that the system can develop a representation of what it believes the student knows. In essence, this is student modelling and has implications for subsequent pedagogical decision making.

The most natural way for students to present their explanations to the system is through natural language, therefore the system must perform some natural language processing. The approach taken here is to apply Conceptual Dependency Theory (Schank, 1975) with Memory Organization Packages (MOP) to structure a dynamic memory able to understand and learn utterances.

3.8.1 MOPs

MOPs are frame-like structures containing generalised "norms" characterising their members. MOPs are closely related to scripts (Schank et al., 1977), but differ by maintaining individual instances of generalised events in a network structure, whereas scripts maintain only generalised episodes. MOPs and their derivative E-MOPS (Kolodner, 1983, 1993) are a popular representation in case-based systems (CHEF—Hammond, 1989; CYRUS—Kolodner, 1983).

A MOP does not specify necessary and sufficient conditions for class membership. Instead it loosely defines a "covering" description in terms of norm features, which is used to focus search during classification and retrieval. MOP norms are generalised from examples they contain by identifying similarity of features. Cases are indexed explicitly by features that differentiate them from the generalised description. When two or more cases share features not specified as norms, a sub-MOP is created to include those cases. Inheritance from parent MOPs and similarity between "seed" cases is used to produce generalised MOP norms, and discriminating features are used to index each case under the sub-MOP. As new cases are added, generalised norms are checked for consistency and modified if needed. The process of creating new MOPs results in a *Discrimination Network*—a tree structure in which nodes are MOPs and lower nodes are specialisations of MOPs or specific instances. The links between MOPs serve to direct search via the indices used to discriminate among MOPs, sub-MOPs and instances.

3.8.2 Conceptual Dependency Theory (CD) and MOPs

CD theory (Schank, 1975) is concerned with representing the meaning of sentences, and is based on two axioms: (i) *for any two sentences that are identical in meaning, regardless of language, there should be only one representation*; (ii) *any information in a sentence that is implicit must be made*

explicit in the representation of the meaning of that sentence. CD theory applies a limited set of primitive concepts to represent natural language statements, which can be *active* conceptualizations of the form "*Actor Action Object Direction (Instrument)*", or *stative* conceptualizations of the form "*Object (is in) State (with value)*". Some of these primitives are: Picture Producers (PP)—actors and objects; Action (ACT)—e.g. ATRANS (transfer of abstract entity, e.g. control and possession), MTRANS (transfer of mental information, e.g. learn and tell), PTRANS (transfer of physical entity, e.g. go and put).

In DKB a basic sentence is represented by a MOP containing slot-filler pairs, which are CD structures. More complicated sentences are represented by a network of MOPs where links represent the relationships between actions and concepts (e.g. causal and temporal). Words in a sentence need to be mapped onto the primitive CD structures, e.g. verbs onto a unique combination of primitive actions such as ATRANS, PTRANS, MTRANS, etc., plus states. In DKB this is done manually for the present.

Example
A CD representation for the sentence "Galileo tells Andrea to drop a small stone" is:

ACTION Type MTRANS
Vf TELLS
Actor GALILEO
Object ACTION Type GRASP
Vf DROP
Mode NEGATIVE
Actor ANDREA
Object PP is-a STONE
Size SMALL
From GALILEO
To ANDREA

The DKB system maintains a root memory of MOPs from which an abstraction hierarchy of more specialised versions of the MOPs are indexed. The abstraction hierarchy terminates at specific instances. When information is presented to the system, it seeks to find a MOP that shares CD structures with the new text. If one is found, the new text is indexed with the retrieved MOP, otherwise a new root MOP is created. Subsequent information will be indexed under this new root MOP if it is compatible with the CD structures contained within. That is, the new text must relate in some way with the text used to create the root MOP. In this way, a knowledge base is created that reflects the information the system was presented, which could be interpreted as an indication of the student's current understanding. There is a long way to go before this kind of student modelling could be applied in an intelligent tutoring system, but results so far provide an interesting and challenging agenda for further research, such as automating the parsing of text to CD structures, and support for the principle that self-explanations *can* yield information about a student's understanding of examples. At the moment DKB is concerned with representation issues, but in future any development will have to consider ways of communicating the contents of DKB to the other components in an ICAI system, particularly techniques for making pedagogical decisions based on information in the student model.

3.9 Conceptual dependency and psychological relevance in ICAI

The design of ICAI systems from a knowledge level analysis (Newell, 1982) is considered in Bumbaca (1988), who presents a design of a knowledge representation scheme that has psychological relevance to students' and tutors' memory structures. It has been suggested (e.g. Wenger, 1987) that psychological relevance has some bearing on how effective a teaching system is in coping with students. For instance, in furnishing explanations of the reasoning performed by the tutor, a more articulate knowledge representation than compiled production rules is needed (Clancey, 1987) to allow expression in causal terms. Case-based reasoning is forwarded as a psychological model of

human cognition (e.g. Kolodner, 1992; Abelson, 1981), so the memory structures associated with case-based reasoning (scripts and MOPs) have been adopted in Bumbaca's work with the intention of being applied to every component of an ICAI system (tutoring module, student module, expert module and natural language interface) as a common knowledge representation framework.

The pedagogical research issues involved in Bumbaca's work are that, to develop effective ICAI systems, we must perform a knowledge level analysis to determine what a good system is to do. This is done by considering what a good human tutor does. Following this, we seek to develop a symbol level representation that has psychological relevance to the knowledge level requirements of the system and approximates the memory structures and processes of a good human tutor. The CBR research issue is in being used to provide the symbol level representation, e.g. applying a memory-based approach to the task of natural language understanding (conceptual dependency theory and scripts) in a tutoring environment.

In addition to the conceptual structures described before, some important terms are defined here: Primitive Action **MBUILD**—the construction of new information from old, e.g. decide, conclude, imagine and consider; Memory components **CP**—conscious processor where something is thought of, and **LTM**—long-term memory where things are stored. Important conceptualizations in instruction are then, for instance, remember—**MTRANS** from **LTM** to **CP**, learn—**MTRANS**-ing of information to **LTM**.

These ideas have not been implemented in a working system, but a conceptual description has been presented. An example of a script for concept teaching is discussed below with some elements of the script presented.

> **PROPS**: C—concept to be taught Pc—preconditions to be true before concept can be taught Q—questions R—responses Xs—experts solution to problem posed to student Ss—student's solution Ex—literal explanation of the concept **ROLES**: S—Student T—Tutor
>
> The tutor first determines if the student already knows the concept, and other factors such as the student's intelligence, motivation and tutorial style. This would be a scene in the script. It does this through a mental transfer **MTRANS** by the tutor **T** from the student's **LTM** to the tutors **CP**. The information it has retrieved from the student model allows the tutor to build **MBUILD** in its **CP** an explanation of the concept **C** at the appropriate level as indicated by the student model. The tutor then transfers this textual explanation physically **ATRANS** from the **CP** to the video monitor. Once the student has asked questions **Q** and the tutor responded **R** the tutor can test the student's knowledge and then place the concept **C** in the student's **LTM**, i.e. update the student model.

A far more detailed coverage of the use of scripts, which includes conflict resolution teach script, stereotype student script, misconception and correct concept scripts, tutorial style and student script, expert script, plus student and tutor themes, goals and plans, is presented in Bumbaca (1988), but their description is beyond the scope of this paper. If psychological relevance is a significant issue in tutoring effectiveness, then it is desirable that knowledge representation considerations focus on providing psychological plausibility. The use of scripts as described in Bumbaca (1988) suggests one way this might be accomplished.

In keeping with the desire for psychological plausibility, and along similar lines to the Dynamic Knowledge Base, Weber (Weber et al., 1988; Weber, 1991, 1993) presents ELM—a student modelling system that applies case-based reasoning to maintain a cognitively relevant learner model for an ITS. ELM differs with DKB in that it attempts to build a model of the student's understanding by cognitive analysis of LISP program code (e.g. inferring plans and algorithms from code) rather than by interpreting natural language. Consequently, although the objective is the same, the techniques used in ELM are very much different than in DKB. Knowledge in DKB is built on the principle that students' articulated explanations of a worked-out example can give insight to their understanding; however, in ELM the precept is that a student builds memory structures partly by reading text and partly by solving problems. Therefore, the students' own

behaviour while solving a problem should be the basis for interpreting their level of understanding. To this end, diagnosis in ELM is performed while a student is attempting to solve a given problem.

3.10 ELM—episodic memory for student diagnosis

The motivation behind ELM (Episodic Learner Model) was to build a LISP tutor to model the processes of knowledge acquisition. Weber et al. (1988) advocate the supposition that students use reminding of prior solutions while solving new problems in LISP. And later, when students become more expert, they generalise specific cases and use the generalisations to solve problems. ELM is an attempt at modelling this process of acquiring and generalising domain knowledge via cases, and applying it to solve problems.

The domain of LISP programming was analysed and the results were interpreted to suggest the existence of three distinct levels of programming knowledge:

(1) *Syntactic*—consists mainly of syntactic and semantic knowledge of LISP.
(2) *Algorithmic*—consists of steps of computation, may be in natural language.
(3) *Planning*—comprising the intentions of the programmer, i.e. specifications.

It is suggested that novices concentrate mainly on the syntactic level while experts concentrate on more abstract levels. This is a phenomenon witnessed in many other domains (e.g. Chi et al., 1981). Therefore, the pedagogical objective of ELM-LISP tutor is to effect a transfer of knowledge representation from the student's to the expert's (i.e. teach the student to consider abstract elements of a problem), and additionally, to represent this change in the learner model. A rule-based approach is used to perform the cognitive analyses in the first place, and CBR is chosen to provide the capabilities needed for a dynamic episodic learner model.

One of the functions of CBR in ELM is to facilitate explanations of why a student's solution is buggy by enabling the tutor to retrieve prior examples of similar problems and solutions. ELM diagnoses non-syntactic bugs in a student's code by trying to recreate the same code. It uses case-based reasoning to retrieve solutions that match the same subgoals and plans from the episodic learner model. This is possible because in addition to representing knowledge structures that indicate a student's "version" of the domain knowledge, ELM also represents the problems solved by the student as episodes. The retrieved cases serve as reminders for the student to explain how an error is similar to one committed before by this student, and how it can be solved. Additionally, a retrieved case that has a similar solution to the one needed for the current problem can be used by the tutorial component to aid the student in solving the problem.

3.10.1 Error diagnosis

A student's solution for a problem is analysed in the context of a given task. The specification of the task given to the student is first described in relation to the knowledge base in terms of the goals the task contains (general and LISP-specific programming concepts, plans and schemata). Each goal has associated with it stereotypical programming methods for its implementation called *programming plans*. Plans and rules specify different ways of achieving the same programming goal. There are correct rules, rules that are correct but give a suboptimal solution, and buggy rules. Together these rules constitute an expert knowledge base of the right way to solve problems and a bug library of common errors. The rules are applied to the standard plan to instantiate a solution, which is then compared to a part of the student's code. The result of this analysis is a derivation tree that is built from the concepts and rules matched to the student's solution, and which represents an *explanation* of the student's solution.

Explanations of plans and subplans are stored as distributed instances of the concepts in the task. Each subplan has an episodic frame, and together the set of episodic frames constitutes an episode. When more student solutions are represented in the episodic learner model, it is possible to form generalisations that represent classes of plans and date types, which aid further diagnosis. For instance, if a new solution's plans and subplans match the plans stored with an episodic frame,

and the code exactly matches the code stored with the episodic frame, then the explanation can be retrieved and applied as an analogy to the current problem by reconstructing it from the episodic frames that relate to the same episode. Additionally, if only a partial match is possible, an explanation is first generated by a cognitive analysis of the task and the case-base is searched for episodic frames that match the ones generated for the new explanation. These episodic frames can be used to suggest the best rules to apply in the present case to find analogies for the missing parts of the partial solution (from the student's perspective).

In this mode, case-based reasoning is used not as the main component of diagnosis, but as a support tool to improve the effectiveness with which rule-based cognitive analysis is conducted. Moreover, the process of generalisation forms hierarchies, which represent students' knowledge acquisition as they attempt more problems. Thus, the set of episodic frames and the generalisation hierarchies constitute an episodic learner model that is particular to a student and the problems he or she has encountered.

3.11 CELIA—simulated student

Several pedagogical principles are incorporated in CELIA (Redmond, 1991; Kolodner, 1993), which is a learner model that tries to improve its ability to retrieve relevant past cases to make predictions about the actions of an expert. CELIA's objective is to eventually match the problem-solving ability of an expert by, for example, "training" the student to recognise the predictive significance of particular features, rather than relying solely on surface similarities. Therefore, the process of improving predictive ability represents the progress of a student from novice trouble-shooter to expert. Since CELIA operates in an *apprenticeship* mode, in which it is exposed to examples of expert problem-solving behaviour, it incorporates the principle of *learning from examples*. The student is required to make predictions about the expert's actions, which is a form of *task-centred instruction*. A prediction might be wrong, and when this happens the student has the goal of acquiring the knowledge needed to make the right prediction in future, therefore *failure-driven learning* is a part of the pedagogy. If a wrong prediction is made, the student is also required to explain the expert's behaviour in terms of the goals the expert might be pursuing, therefore *self-explanation* is also involved.

Unlike DKB, which leaves the process of generating an explanation entirely to the student, CELIA takes the position that in the early stages of learning a student is in a poor capacity to generate explanations. This is so because novices have limited domain knowledge to apply to interpreting the actions of an expert. The apprenticeship mode is intended to aid the process of explanation by focusing the student's attention on the step that caused failure. Consequently, the blame assignment problem is simplified, and students are better able to explain why their predictions were wrong and why the expert's solution is right. (CELIA assumes that the expert's solutions are always right.) In DKB, the self-explanation effect is achieved by students alone, and is used to measure their level of understanding; in CELIA, it is achieved with help provided through the apprenticeship mode and is enforced as a part of the learning process.

There are three classes of knowledge that CELIA improves. Causal knowledge of the domain—automobile troubleshooting. CELIA starts with a deficient model that includes misconceptions and, through interaction with an expert troubleshooter, it modifies its model. Through apprentice-ship, whereby it observes an expert solve problems, CELIA also improves it knowledge of the troubleshooting process itself, which is the second class of knowledge. The third class is episodic knowledge, represented as past cases of problems and the actions taken to solve them. Tests on CELIA indicated that early in the learning process the greatest improvement in problem-solving ability came from improvements in the case base. After exposure to limited examples, students not only added cases to memory, but also refined the way in which cases were indexed to make retrieval much more useful. Episodic knowledge is held in a case base in CELIA, and this is discussed further below.

The case-base contains *distributed cases* (Redmond, 1990) made up of small portions of a case, called *snippets*. Snippets are indexed according to primitive subgoals encountered during the problem-solving process, and are linked to other snippets that together form a case. The purpose of representing cases as snippets is to enable the reasoner to retrieve relevant portions of a case as well as entire cases. Snippets also function in a way analogous to *scenes* in scripts. Scenes correspond to sub-actions in a sequence of actions that together form a script. Reasoning is performed by matching input to a scene stored within a script, and then using the context the scene is within the whole script to reason about possible forthcoming actions, and to provide background information. Similarly, a snippet enables the reasoner to make predictions about possible subsequent actions by interpreting the snippet in the context of the whole case.

Instruction proceeds as follows: a student observes an expert perform troubleshooting actions on an example problem called the *correct context*. The actions performed such as diagnostic tests and their results are available to the student, who then attempts to retrieve a case from memory that matches the problem description, tests performed, test results, plus other features of the troubleshooting process. Based on the retrieved case, which is called the *incorrect context*, the student makes a prediction about the likely cause of the problem. This is called the *incorrect* prediction (if it turns out to be wrong). The expert's prediction is called the *correct* prediction. The student then retrieves a case that would have provided the correct prediction if it had been indexed appropriately for this situation, and this is called the *correct context*. The goal of the student is now to adapt the way in which the correct context is indexed in memory, so that in future it will be retrieved in a similar situation to the current context. This is the explanation phase in which the student must assign credit to the features in the current context that make the correct context appropriate, and blame to features that make the incorrect context so.

A similarity-based comparison of features is performed to mark portions of a case as being appropriate or otherwise to the correct prediction. The comparison is done on snippets rather than whole cases, and consists of a five step procedure that compares the correct snippet to the incorrect snippet:

(1) Features that are the same in both snippets are eliminated, i.e. non-discriminating features.
(2) Features in the current context that match the correct context more than the incorrect are selected.
(3) Each feature can have multiple values so those distinguishing values of the selected features in the current context are identified.
(4) A causal explanation linking feature values to the actions taken by the expert is created to justify their significance.
(5) Those values for which significance is established are saved as *indices* to enable the correct snippet to be retrieved in future similar situations, and as *censors* to avoid retrieving the incorrect snippet.

The five steps result in CELIA re-indexing its memory to make retrieval more useful and accurate. In essence, CELIA has learned to make better predictions by observing an expert solve problems and comparing its own solutions with the experts. Feedback from the encounter and self-explanation of failure produce the learning experience.

As described CELIA models how a novice with limited knowledge can learn from interactions with an expert. In this sense, CELIA could be considered a *simulated student* (Vanlehn et al., 1994), and conceivably could be applied within a tutoring system to provide information to the tutor on how students are learning and what their present state of knowledge is likely to be. However, CELIA is limited in its versatility for handling differences in how students learn, but this might be overcome by changing the initial knowledge it is provided, e.g. different case base and causal model, which could be provided by a student model. In any case, CELIA shares a common inadequacy of simulated students by being restricted to the strategy of "learning from examples" and, thus far, applicable only in well structured domains.

3.12 AXE—Active Example Elaborator—skill acquisition (Reimann et al., 1993)

Active Example Elaborator (AXE) is a first step towards developing a model of skill acquisition from examples, which eventually should be examinable by students to teach them how to explain worked-out examples. (See Schult et al., 1993, for a description of *an integrated case-based teaching environment.*) The pedagogical background for this work is the *self-explanation* research of Chi and others (e.g. Chi et al., 1989). Like the Dynamic Knowledge Base (DKB) described earlier, one objective of AXE is to test this research in the form of a computer model. However, whereas DKB had the more specific objective of developing a student model for purposes of diagnosis, AXE is concerned with teaching the strategy of good self-explanation. The instructional principle is that students can learn to create good self-explanations of worked-out examples, and thereby learn the principles presented in the examples by watching an expert (AXE) demonstrate good elaboration strategies. In this respect AXE is similar to CELIA, since it too has an apprenticeship mode; the difference lies in what is taught through apprenticeship. CELIA teaches domain knowledge as a result of self-explanation of an expert's performance, and AXE teaches the process of generating self-explanations through watching an expert.

There are two main components of the AXE model:

(1) An example understander that elaborates a worked-out example, i.e. produces a self-explanation that explicates the information/principles contained in the example.
(2) A problem solver that produces a solution to a given problem description. The problem solver depends greatly upon the understander's ability to elicit knowledge from examples.

Knowledge for the example understander is represented in two forms: taxonomic and partonomic information as frames; and procedural knowledge as rules. There are four elaboration strategies: (1) By applying backward chaining, the example understander can *fill in* implicit information in the example; (2) and (3) Information in frames is used to generalise objects and relations in the example: for instance, the example *Holds ceiling.c string.a* is elaborated into *a* FIXED OBJECT *is connected to a* MOVABLE OBJECT by applying the abstractions *ceiling.c*:-FIXED OBJECT; *string.a*:MOVABLE OBJECT; and *Holds*:is-connected-to; (4) AXE also elaborates goals by applying specified goals (represented declaratively) when it recognises the conditions of the goal have been satisfied in the problem.

Case-based reasoning is used to store examples and their elaborations in an episodic memory which is later used for problem-solving. An example is represented as a sequence of states, goals and actions connected by temporal links. The problem solver applies analogical reasoning to retrieve partial solutions to a new problem. At the first stage of retrieval a match is attempted between the given problem and a case directly. If this fails, the abstraction information in the elaborations is applied to determine similarity at a more general level for objects and relations in the problem. However, if still no match is found, the problem solver reverts to its domain knowledge.

CBR here is used in a standard function to add new cases to memory in order to solve more problems than the domain model can alone. The functionality of the episodic memory to solve problems depends upon the cases it contains. This depends upon the understander's ability to elicit knowledge from worked-out examples using any combination of elaboration strategies. Therefore, there is a direct link between the type of elaboration strategy used and the case-based reasoner's problem-solving abilities. This can be translated to human learning, and it is intended to show students the merits of different elaboration strategies (for self-explanation) on how well they understand and learn.

3.13 CABAT—external-memory assistant for students (Schult, 1993)

Case-Based Tutor (CABAT) is a component of a discovery environment in the domain of elastic impacts, where the domain knowledge to be imparted to the student comprises formulas related to collisions between disks. CABAT's method also may be applied to other similar domains that are

formula-based, and where learning from examples is the tutoring strategy. Experiments designed by a student are analysed algebraically by a CONSTRAINT GENERATOR and stored in a discrimination net created by a GRAPHER component, from which they are later retrieved by a CASE SELECTOR as reminding to aid further problem-solving. Case-based reasoning here is used as an aid to the student's memory, and the central issue is in indexing examples by *structural* and *superficial* features. Case-based retrieval must ensure that the most appropriate examples are retrieved so as to avoid inducing misunderstanding in the student. An appropriate example will be used by the student to reason by similarity to solve the present problem, and will help the student to index the problem in relation to the retrieved case and background knowledge. In this implementation, CBR input to tutoring is to organise examples in a way that allows retrieval of meaningful cases in different tutorial contexts.

The CONSTRAINT GENERATOR takes a given experiment and the formula that represents it, and derives simpler versions of the formula by specifying constraints that eliminate the need to consider certain parts of the formula. These simpler formulas represent simpler situations of the experiment and constitute cases, e.g. velocities may be assumed to be zero, or fractions may be eliminated by giving convenient values to variables. Heuristics are then applied to the set of constraints to identify ones with good discrimination ability, and a discrimination net is formed by the GRAPHER. The simplified formulas are a first level judge of structural similarity, and additionally, superficial similarities are used as a second level judge of similarity by the CASE SELECTOR. Priority in retrieval is given to cases that match at the *fine* type, i.e. have structural and superficial similarity. When a case is presented to the student, the type of similarity is specified so the student can make use of the similar part of the solution, if not the whole case. The use of reminding also serves to provide guidance during the exploration process by helping the student to transfer from the prior situation to the current one, and to make generalisations over the two cases.

3.14 Explanation generation

SHERLOCK (Lajoie et al., 1992) is a practice simulation environment in which avionics technicians are trained to troubleshoot electronic navigation equipment for the F-15 aircraft. It is more properly a *coaching* system than an ITS, since there is minimal tutoring interaction, other than response to questions and requests from trainees. During a *reflective follow-up* session, SHERLOCK is used to assess the steps in a trainee's diagnosis by identifying certain factors in the action, called *facets*, which are used by human tutors in assessment. A facet is evaluated as either "good (g)", "bad (b)" or "neutral (n)", and based on this analysis a student's action is designated either "good" or "could be improved". SHERLOCK is then normally required to justify this assessment by way of an explanation.

As described above, SHERLOCK makes no use of case-based reasoning. Explanations are generated by a text planner in which *communicative goals* are interpreted from the query, e.g. "achieve the state where the hearer understands how the status of a component has been determined". These are associated with general *speech acts* that serve the communicative goal, e.g. INFORM, RECOMMEND. Planning an explanation requires selection and association of speech acts for every communicative goal. The text plan is held in a *dialogue history*, and represents a record of the trainee's interaction with SHERLOCK.

Research presented in Rosenblum et al. (1993) seeks to utilise case-based reasoning for retrieving prior explanations from the dialogue history. The purpose of retrieval is not direct reuse of a prior explanation, but rather to provide contextual information for the generation of new explanations that in some way consider the trainee/SHERLOCK interaction record. The instructional basis of this work is the supposition that human tutors frequently refer to relevant previous dialogue when producing explanations. This helps them to reify general principles in the context of the present situation and the student's prior behaviour, which makes for more effective explanation. The prior dialogue is not explicitly included in the new explanation; instead, it serves decision making, such as: How much material can be repeated? Should an alternative explanation

be used? What references should be made to similarities and differences between situations? The goal of this work is to implement a computer model that can take account of discourse context in explanation within an instructional environment.

The case-based component is closely modelled on the HYPO architecture (Ashley, 1991), which was described when discussing CATO. Here, a case represents an action performed by a student and is symbolised by a set of facets with their assessments (g, b or n), plus an evaluation "good" or "could be improved". Additionally, there are pointers in the case to the explanation in the dialogue history that was given before for this action. During the retrieval process, a similarity Directed Acyclic Graph (DAG) is generated that links similar cases. Similarity is judged on the number of matching facets of a certain type, e.g. b facets for answering questions on "could be improved". The distance in the graph of a case from the root case, which represents the current action, is a measure of the number of facets in common with the root case, and therefore a measure of similarity.

This technique allows the present action to be compared to prior actions that are similar in respect of certain factors they contain, and the most similar case to be retrieved. The retrieved case has a pointer that allows the text plan associated with the prior action to be accessed from the dialogue history. This is the first phase of generating explanations that make use of prior dialogue. The second phase has not been implemented as reported. It would require the system to incorporate the retrieved explanation in some meaningful way into a new explanation. An initial suggestion for this uses heuristics to allow, for example, a prior explanation to be rephrased or summarised if the current action is similar to the prior action. Since explanation generation is considered a planning task here, adaption techniques from case-based planning are a possibility for enabling the second phase.

3.15 Discussion

We have described several computer-aided instruction systems that apply ideas and techniques from case-based reasoning. These systems differ on the instructional principles motivating use of case-based reasoning and the specific techniques applied. Predominantly, case-based reasoning has been applied for its indexing capabilities to facilitate retrieval of suitable stories or examples in response to instructional needs. An exploration environment is specially suitable for case-based teaching. Free exploration can be supported by video clips or simulations being presented in response to a student's query. Story retrieval and inter-story indexing are the main research issues in free exploration. A degree of directionality is introduced through the principle of task-centred instruction in which exploration is constrained to elements pertaining to the task in hand. Design of index vocabularies that facilitate story retrieval in the context of issues addressed by the task need additional attention to those issues for free exploration.

Adaption of stories when in video form is a desirable feature not yet implemented, but bridging techniques offer some adaptation capability by changing the context in which a story should be considered. This idea parallels that of *enforcement* (Hammond et al., 1993), in which the real world is manipulated to make it fit better with the internal view of the world held by the planner. In terms of instructional planning, enforcement might prompt the planner to create a situation in which it could satisfy a future need of the student by arranging for the student to want to satisfy that need sooner. This could be achieved by introducing a concept earlier than initially planned to ensure that two or more goals are satisfied concurrently by the same learning experience.

The principle of enforcement is to impose stability on the world so that a planner can function in that world with the minimum of reasoning. However, intelligent tutoring has as its motivation to get away from uniformity and rigidness in its view of the world and how it responds to students. Bridging and enforcement techniques might prove useful for adaptation in story-based teaching, where it is difficult to adapt stories themselves. But there has to be careful consideration of what can be stabilised and to what degree, because if enforcement and bridging are taken too far something resembling programmed instruction may result.

A distributed story-base may also assist adaptation by reducing a story into primitive elements (clips) that can be combined on retrieval. The added problem this approach entails, however, is defining an index vocabulary suitably general to allow any primitive clip to be relevant to more than one instructional situation. Primitive clips need to be indexed by the subgoals they satisfy and the various contexts they are relevant to. There is also the problem of maintaining continuity throughout the entire story, and ensuring fidelity of the story to the instructional objectives of the plan. Considerably more knowledge of the contents of stories and their primitive clips than presently represented in a story index is needed to enable a complete story to be adapted by "editing" together several disparate primitive clips. Creating and maintaining such a rich indexing mechanism would require a considerable effort.

Task managers (planners) are not widely used in case-based teaching, and their introduction will pose problems for student modelling since contents of stories are not known to the system. A favoured and necessary assumption taken with case-based teaching is to infer students' goals to be "learning information required to enable them to traverse an impasse". Although this assumption imparts some guidance for pedagogical decision making, it is certainly insufficient to reliably model a student's intentions. The self-education aid has sought to address this limitation by allowing students to formulate their own goals, but even here there is only weak student modelling, since a student's goals are inferred within the context of a pre-determined causal model of goals and plans.

Other applications have sought to test ideas on how students learn from examples by modelling the knowledge acquisition process as a student model or simulated student. Representation of declarative knowledge as memory organisation packages organised in a discrimination network represent a student's model of the domain as a subset of the expert's. Procedural knowledge and causal knowledge are not addressed by case-based representations, nor are misconceptions. However, hybrid knowledge representation as applied in the simulated student does incorporate these types of knowledge, but how a pedagogical component might make use of a case-based cognitive learning model is not addressed. These are clearly concerns for future research.

Natural language processing for knowledge acquisition using memory structures has aided student modelling by analysing students' self-explanations of worked-out examples. Work has suggested this is technically achievable, but there is insufficient support that the resulting models are a true indication of the student's understanding. Inconsistencies can occur due to students' inability to articulate explanations, even though they have an understanding of the material. Also, the process of articulating an explanation is hypothesised to cause learning so though analysis of the explanation might indicate students' present level of understanding, it says little of their subsequent knowledge state following the self-explanation effect. Finally, as we discuss in the next section, case-based advisory systems for instructional planning have been developed that retrieve plans in response to a human tutors requirements. The next step is to incorporate this capability within an autonomous tutoring system. To do so, we must address the problems of case adaptation, credit assignment in plan failure, cooperative planning between the student and the tutor, and integration of plan generation and execution. None of which has yet been undertaken in case-based teaching.

4 Case-based lesson planning

We are presently researching how case-based planning techniques based on the CHEF methodology could be applied to select and sequence lessons within a tutoring system. Planning is a major area of interest in intelligent tutoring and, until recently, most planners had to regenerate plans for situations they had planned for previously when faced with the same problem again. When planning instruction there are certain similarities between situations a planner must deal with, e.g. the same instructional objectives might have to be taught. In these cases, it would be beneficial to be able to reapply an old plan in the new situation, and this feature is provided by Case-based Planning. CHEF (Hammond, 1989) demonstrates the worth of case-based reasoning for sequencing sub-goals to obtain a global goal. The same capabilities are needed for sequencing

individual lessons to make a curriculum for tutoring. With the *learning from examples* tutoring strategy, for example, the correct presentation of lessons in the right context is crucial to achieve the desired result. To be effective, a curriculum plan must not be static, and instead should adapt to any goal failures and feedback from students during the teaching session. This feature requires a powerful plan creation and modification mechanism that can handle the dynamic requirements of individual students. Before we discuss the requirements of an instructional planner any further, we briefly present CBMIP as an addendum to the survey of applications.

4.1 CBMIP

The Case-based Mathematics Instructional Planner (CBMIP) (Du et al., 1991) is a decision aid for preparing lesson plans for the domain of ordinary differential equations. It is based on the CHEF methodology, and incorporates four basic components that allow CBMIP to retrieve and adapt previous lesson plans to meet the requirements of new instructional goals. The goals are organised in a discrimination net, and each goal has an indicator to a plan that is relevant to achieving the goal. Terminal nodes in the discrimination node are the goals themselves, and higher nodes define specific characteristics of differential equations. Therefore, a plan presents a goal that incorporates specific characteristics.

4.1.1 Fundamental components of a case-based planner

SELECTOR: retrieves a plan that is the best match for a set of objectives. Five *characteristics* are used as indices to identify an Instructional Goal that meets the present objectives: equation type; solution range; solution type; coefficient of equation; and variable type of equation. The plan associated with this instructional goal is retrieved, that is, the plan that facilitates the teaching of this instructional goal is retrieved from memory.

A plan is represented as a sequence of *teaching actions*, the goal it achieves, characteristics it is relevant to, and characteristics it is not relevant to. Each teaching action is represented by its name, prerequisite knowledge needed before it is effected, and post-requisite knowledge expected to be known after it is effected. An example of a teaching action is shown below:

Taction35
Action-Name: Finite Base Function Set (FBFS)
Precondition: Independent Set (ISET), Orthogonality (ORTH)
Effect: Finite Base Function Set (FBFS)

A plan looks like the following:

Plan4
Plan-Name: Instruct Basic Finite Element Analysis
Goal-Achieve: Goal33
Rel-Characteristics: approximation, (global infinite), (special general), (constant variable exp poly trig), (single multiple)
Not-Rel-Characteristics: exact, local
Tactions: LSP, FSP, NORM, etc...

MODIFIER: satisfies goals that the retrieved plan does not meet by applying modification rules. The actions to modify have to be identified, and strategies to do the modification specific to the plan have to be applied.

REPAIRER: to accommodate modification mistakes that might occur, a REPAIRER is used to correct conflicts in a plan. Repairs include addition, deletion and reordering of actions in the plan. The completed plan is output to the user.

STORER: places in memory the results of the SELECTOR, MODIFIER and REPAIRER. It stores the plan using the same indexing as is used by the SELECTOR, i.e. objectives satisfied by the new plan.

At this stage, CBMIP produces a plan that is suitable for classroom use for all students. It is suggested that a standard plan can be individualised for a particular student for use within a tutoring system by repairing it under advice from a student model; however, no details are provided of how this is done. We are researching how a plan can be designed specially for students' individual needs as the normal process of planning. In the following section, we discuss the limitations of the CHEF methodology as utilised in CBMIP for individualised instruction, and suggest more general requirements of an instructional planner that need to be addressed by case-based instructional planning systems. Of special consideration is a failure memory not incorporated in CBMIP, and plan evaluation.

4.2 Issues to be addressed by case-based instructional planning

A case-based planner must be able to predict problems and plan to avoid them; this is what sets case-based planning apart from other planners that reuse stored plans. A case-based planner must recognise when its normal indexing and modification processes will result in a failed plan, and it needs to know how to avoid the problem. In CHEF, a FAILURE memory is used to infer the possibility of a failure in a new situation. A failure is defined as a particular state of the world that has come about or failed to come about as a result of the running of the plan. In the light of a failure, CHEF identifies plans that target or avoid the identified problem. This is possible when plans are indexed by the problems they avoid. To identify when in future a repaired plan can be used to avoid problems, the planner must be able to predict when those problems are likely to reoccur. To do this it needs to know what feature of the original plan caused it to fail so it can look for this feature in other situations, and thereby predict a similar failure. This is the problem of *credit/blame assignment*.

CHEF has a simulation that can run a plan and determine the results. Therefore, it can identify the consequence of combining actions, which it assigns as a cause of failure and links to the memory of that failure. This is difficult in tutoring because the planner cannot determine the results of a plan without actually running a tutoring session, since the cause of failure could lie outside the planner's world, i.e. with the student. Features of the student that contribute to a failure may not be easily predicted, so identifying the cause of a plan failure is a major problem to be overcome in instructional planning. Where a domain model is available as in CHEF's simulation, a plan can be run and its results analysed. In human learning, however, there is insufficient knowledge of theories of learning that relate instruction to how this instruction effects students' internal representations of domain knowledge to allow reliable simulations to be created in many ill-structured domains. Simulated students have been designed (e.g. Vanlehn et al., 1994) that aim to mimic students' learning from instruction and provide feedback on the plans effectiveness, but they operate in limited well-structured domains (e.g. mathematics), and there is little consideration of the individual nature of how different students learn from instruction.

During the problem of credit assignment, the planner must decide if the plan has failed due to planning mistakes or due to incomplete knowledge of the student. A planning mistake differs from failures due to incomplete knowledge in that a mistake is an action taken by the planner that leads to an undesirable outcome. A mistake is characterised by the fact that the undesirable outcome is *avoidable* if a different action was taken, and the planner had complete knowledge of the student to allow it to take that alternate action if it chose. The problem of deciding if the fault is in the planner's knowledge of planning or due to some unpredictable idiosyncrasy of the student is a difficult matter. Any failures due to incomplete knowledge will only become evident if the student fails to reach the target goal. Blame assignment in such situations is more properly a student modelling problem than a concern for the planner. In any case, CHEF does not handle this kind of situation. It handles five classes of problems (TOPs), which are used to select appropriate repair

strategies. Problems due to external factors, however, are not considered. The five TOPs are defined below, with examples of envisaged planning problems that must be repaired in the domain of instructional planning:

(1) Side-effects SE.
(2) Desired-effects DE.
(3) Side-feature SF.
(4) Desired-feature DF.
(5) Step-parameter SP.

Examples of planning problems in curriculum planning
(SE) required achievement level of student is not reached to allow a subsequent goal to execute. An unwanted side-effect is the presence of misconceptions as a result of some instruction.

(DE) the achievement of a goal makes a subsequent goal unnecessary or does not allow it to run, e.g. student wants to learn two or more unrelated concepts and the planner will only teach one line of reasoning (plan) at a time. This occurs if the student selects a target objective related to one type of task and also indicates sub-goals that relate to another task. Consequently, if the planner enables the sub-goals to execute, the target objective will not be able to run. Conversely, students indicate they do not want to learn a particular enabling objective, thereby making it impossible for the plan to succeed in having them acquire the target objective.

(SF) the presence of a non-essential or non-goal satisfying feature of a goal causes the plan to fail, e.g. the requirement of a goal to impart an ATTITUDINAL skill is not-essential to being able to perform that goal, and if students do not attain or demonstrate that Attitude and the planner insists they do a failure can occur. Applies also to non-essential prerequisites.

(DF) the presence of an essential feature causes a violation, e.g. conflict in prerequisites of two goals, i.e. two goals require each other as prerequisites OR student does not want to demonstrate an essential prerequisite.

(SP) problems due to bad parameters of a step, e.g. number of examples, time spent on revision, level of difficulty of material, presentation style, etc.

To these five we add a sixth class of problems that a case-based planner should consider when assigning blame and selecting repair strategies: Extraneous Features (EF):

(EF) problems due to knowledge-independent factors present in the student, e.g. lack of motivation, tiredness, and low interest, i.e. affective factors that are not predictable from the planner's knowledge of the student.

Extraneous features are unanticipated events that are usually imperceptible. Some extraneous features that effect a student's performance, and consequently the success of an instructional plan, are lack of motivation, forgetting, fatigue and level of interest. Poor performance can be blamed sometimes on extraneous features, and sometimes not. It is important for a planner to be able to recognise situations when a plan failure is due not to planning mistakes but to unanticipated extraneous features. A problem arises in determining cause of plan failure because extraneous features are not represented in any domain model of the student, and therefore the planner cannot reason about such things. Deductive student modelling is not suited to representing and reasoning about such features, and it is necessary to look for other representation methods to accommodate them. A case-based approach to diagnosis based on co-variance of extraneous features with specific overt behaviour may provide the necessary reasoning ability. In this type of student model we do not require causal justification of an extraneous feature with any observed behaviour, only that it has been observed before and was considered a contributing factor to the failure. It could be imagined a system along the lines of HYPO in which an argument is generated for the importance

of an extraneous feature in the outcome of a case (instruction plan). It seems intuitively amenable to apply the dialectical techniques of legal reasoning to argue for the significance or otherwise of specific factors in a case's description.

The problem of blame assignment is even more acute when evaluation of a plan is performed after the entire plan has been executed. If a plan is created that incorporates several instructional objectives, any failure on the student's part could be attributable to any part of the plan. Retrospective analysis of the plan cannot reliably identify what part of the instruction presented to the student was responsible for failure in learning. For this reason, it is important that plan generation and plan execution are performed iteratively. Feedback from the execution of a plan should be used to design subsequent instruction. To facilitate this, instructional planning should be performed incrementally. Instead of preparing a plan that addresses the entire set of instructional objectives, it is necessary in an intelligent tutoring system to generate and execute a small portion of the plan at a time, and to use the results of this to design subsequent instruction. This approach to planning is similar to Model Tracing as used in some tutors (e.g. LISP Tutor–Anderson et al., 1985), in which an *opportunistic* approach is taken where the student's interactions with the tutor decide the course of instruction. With the case-based approach we favour, planning is performed at two levels: global level—a plan is retrieved that specifies the high-level objectives that need to be taught and a provisional sequence in which they should be presented; and local planning—plan generation and execution are combined to present instruction that takes the student from a specified start state to a specified end state, as defined by the global plan.

For tutoring it is desirable to monitor the plan during execution to identify where a failure occurs, and to know when to make changes to the plan to account for the student's changing needs. Planning and execution must be combined because we cannot predict every contingency, and we do not have a complete and accurate view of what results the plan will produce in the real world. We must also deal with new goals that are added during interaction with the student. Some of these may have to be incorporated in the plan immediately, and others will be *suspended* (Hammond et al., 1993). A suspended goal can be activated when some of its prerequisite goals are satisfied during interaction. The prerequisite hierarchy of instructional goals will provide the knowledge needed to recognise when a suspended goal can be activated, but a problem in opportunistic planning is limiting the amount of testing done to recognise when prerequisite goals are satisfied.

The use of a global plan should simplify the process of instructional planning provided the plan is produced from experience of past plans. The global plan can act as a standard or ideal plan that avoids previously encountered problems. It can be retrieved with little reasoning to satisfy a set of goals (at a high level). The addition of an opportunistic planning capability means that, although a global plan can be retrieved to take students from their current knowledge state to the desired knowledge state, details of the plan (contents and sequence) at the local level can be decided in the light of new information and goals identified during interaction. Therefore, it is desirable to have an opportunistic planning capability as well that can capitalise on new situations. For example, in some cases the planner might be able to satisfy goals sooner than expected, e.g. when the student achieves higher than expected test score, or learns faster or chooses not to follow a particular course.

To identify a step that caused failure, it is necessary to have continuous monitoring so errors can be identified as they happen, and the step that caused it is identified as being the step immediately prior to the failure occurring. An assumption here is that the student does not forget, which of course is an untruth. If a student does forget previous instruction, it amounts to a failed precondition of the later step. This kind of problem is difficult for the planner to plan for, since it cannot tell what the student has forgotten, and therefore, it is difficult for it to determine which preconditions have failed and when. It cannot plan for a failure because it is unaware of any failure. It might be necessary to make the working assumption that students do not forget because it is not clear how a case-based planner would effectively handle forgetting when assigning blame for a plan failure, other than continuous retesting of prior instruction. We will not say any more about blame assignment, except to reiterate that to repair plans due to incomplete knowledge of the student the

planner has the difficult job of identifying causes external to its knowledge of the world. Only when the case-based planner knows *why* a failure has occurred can appropriate repair strategies be selected and invoked.

There are certain additional repair strategies that might be appropriate for tutoring: counter-planning strategies are needed, which could be used to deal with planning failures resulting from interactions with the student, e.g. conflicting goals between what students want to learn and what the tutor suggests they learn. There might be a need to alter the initial set of goals the planner is handed to overcome some problems where selection of target objective is incorrect. Interaction between the planner and the student model may allow the initial goal set to be altered if the original cannot be satisfied or feedback from the student "suggests" a different target objective is of more interest.

Other problems faced by case-based instructional planning include: where a case is not available and no similar case can be retrieved, the planner is unable to produce any plan because it does not have a facility to plan completely from scratch. If it tries to adapt a deficient plan it could produce a poor plan. Like any human teacher, the planner will be biased by the events it has been subjected to. It will plan by what it "personally" knows has worked for it, although this might be inappropriate in some other circumstances that are not encountered or accounted for by the planner. It will tend then to plan as if the unusual situation did not exist or fail to produce a plan. Since the planner learns and becomes better as it encounters more instances of producing and repairing plans, initially it is a relatively poor planner. The planner will tend to produce less than optimal plans because of its limited experience and case-base. The paradoxical situation is that for a planner to improve it must produce poor plans that fail, which speaks badly for early users of the instructional system. Although this may be acceptable in the artificial world of computer simulations, it is not acceptable where the development of people is dependent. There are serious ethical issues to be considered when deciding where to stop manual testing and improvement of the planner in the laboratory and when to apply it to real students, knowing that initially it might produce poor plans, with the consequences that has on students' development.

Another problem is in judging the validity of a single plan that has been successful. Was it due to good design, a bright student, did the student have help, was it just luck, or was it a combination of all these things? There is a danger in placing too much reliance on one or two successful plans in a domain like instruction where little is understood and measurable about the causes for success and failure.

A question the planner must address when evaluating a plan during execution is "When is it wrong?" At what point should it decided the plan in effect is insufficient and needs modification before continuing with instruction? The planner must decide if to persevere with the current plan because the student is progressing adequately, or attempt to alter the plan. It needs to predict problems and perform self-analysis of its current performance in planning for the student's needs so that it can take remedial action before catastrophic results, i.e. the planner must perform damage limitation.

Also, a crucial part of planning is the ability to go back and make changes to earlier decisions in light of new information. However with instruction, earlier decisions would have been taught to the student before new information is available. In some case-based systems, (e.g. PERSUADER—Sycara, 1988), if some part of the retrieved case cannot easily be adapted an alternative is to abandon it and select the next best plan in the case-base and try to adapt that. With the incremental planning approach needed for instruction, it is not possible to abandon a plan if it cannot be adapted sufficiently without having to discard some of the last lessons actually taught to the student. This might mean that several hours of instruction are wasted. Plans might also need to be abandoned if modifications to a current step interfere with previous steps that have already been applied in the real world. So the previous step (or earlier ones) might have to be abandoned and replaced to enable the plan to proceed beyond its present impasse. When prior steps are abandoned there could still be interference if the abandoned steps are already in place in the student's cognitive model. This can result in having to include additional "remedial" instruction in

the plan to get the student to the desired prerequisite state, and will cause added confusion when performing blame assignment.

4.3 Conclusion

We have outlined several problems facing use of case-based reasoning for instructional planning, and have made suggestions how some may be tackled. In our work we are researching combining plan-based with opportunistic planning techniques for a task-oriented teaching approach. The planner's role is to prepare a sequence of tasks that reflect the knowledge the student already has and the necessary concepts to be learned. Since teaching is done as a sequence of tasks, evaluation can be performed at each stage (i.e. at the end of each task), and the course of instruction from there can be designed within the constraints of the global plan. The major research issues here are in developing a representation of tasks that allows the planner to design new tasks that require the student to demonstrate specified skills, and enables appropriate examples to be retrieved as and when the student needs information. The planner will design tasks to be presented to the student by piecing together suitable performance objectives that together form a task the student wants to be able to perform. Our long-term objective is to address the important capability a case-based instructional planner must have, namely to recognise why a planning failure has occurred when the cause of failure lies within the student.

References

Abelson, RP, 1981. "Psychological status of the script concept" *American Psychologist* July.

Anderson, JR and Reiser, BJ, 1985. "The LISP Tutor" *Byte* 159–175.

Ashley, KD, 1991. "Reasoning with cases and hypotheticals in HYPO" *International Journal of Man-Machine Studies*.

Ashley, KD and Aleven, V, 1991. "A computational approach to explaining case-based concepts of relevance in a tutorial context" In: *Proceedings of Workshop on Case-Based Reasoning*, 257–268, DARPA, Morgan-Kaufmann.

Aleven, V and Ashley KD, 1992. "Automated generation of examples for a tutorial in case-based argumentation" In: *Intelligent Tutoring Systems, Second International Conference, ITS '92*, 575–584, Springer-Verlag.

Ashley, KD and Aleven, V, 1992. "Generating dialectical examples automatically" In: *Proceedings Tenth National Conference on Artificial Intelligence AAAI-92*, 654–660.

Bareiss, ER, Porter, BW and Wier, CC, 1988. "Protos: an exemplar-based learning apprentice" *International Journal of Man-Machine Studies* **29** 549–561. (Also in: Kodratoff, Y and Michalski, R (eds), 1990. *Machine Learning: An Artificial Intelligence Approach, Volume III*, 112–127, Morgan Kaufman.)

Bariess, R, 1991. "The Story Archive: A memory for case-based tutoring" In: *Proceedings of Workshop on Case-Based Reasoning*, 269–279, DARPA, Morgan-Kaufmann.

Bell, BL and Feifer, RG, 1992. "Intelligent tutoring with dumb software" In: *Intelligent Tutoring Systems, Second International Conference, ITS '92*, 615–624, Springer-Verlag.

Bell, BL and Bariess, R, 1993. "Sickle Cell Counselor: Using a goal-based scenario to motivate exploration of knowledge in a museum context" In: *Proceeding of Artificial Intelligence in Education, AI-ED 93*, 153–160, Association for the Advancement of Computing in Education.

Bumbaca, F, 1988. "Intelligent computer-assisted instruction: a theoretical framework" *International Journal of Man-Machine Studies* **29** 227–255.

Chi, M, Feltovich, PJ and Glaser, R, 1981. "Categorization and representation of physics problems by experts and novices" *Cognitive Science* **5** 121–152.

Chi, MTH, Bassok, M, Lewis, MW, Reimann, P and Glaser, R, 1989. "Self-explanations: how students study and use examples in learning to solve problems" *Cognitive Science* **13** 145–182.

Clancey, WJ, 1987. *Knowledge-Based Tutoring: The GUIDON Program.*, MIT Press.

Du, Z and McCalla, G, 1991. "CBMIP—a case-based mathematics instructional planner" In: Birnbaum, L (ed), *Proceedings of 1991 International Conference on the Learning Sciences*, 122–129, Association for the Advancement of Computing in Education.

Edelson, DC, 1991. "Oh, the stories I could tell: managing an aesopic teaching dialogue" In *Proceedings of Workshop on Case-Based Reasoning*, 280–291, DARPA, Morgan-Kaufmann.

Edelson, DC, 1992. "When should a cheetah remind you of a bat? Reminding in case-based teaching" In: *Proceedings Tenth National Conference on Artificial Intelligence AAAI-92*, 667–672.

Elsom-Cook, MT, 1990. "Guided discovery tutoring" In: Elsom-Cook, MT (ed), *Guided Discovery Tutoring—A framework for ICAI research*, Paul Chapman Publishing.

Farrell, R, 1988. "Facilitating self-education by questioning assumptive reasoning" In: *Proceedings of AAAI-88*, 2–6.

Fernandez-Valmayor, A and Chamizo Fernandez, C, 1992. "Educational and research utilization of a dynamic knowledge base" *Computers and Education* **18** (1–3) 51–61.

Hammond, KJ, 1989. *Case-Based Planning—viewing planning as a memory task*, Academic Press.

Hammond, K, Converse, T and Marks, M, 1993. "Towards a theory of agency" In: Minton, S (ed), *Machine Learning Methods for Planning*, 351–396, Morgan Kaufmann.

Hayes-Roth, F, 1977. "Learning by example" In: Lesgold, AM, Pellegrino, JW, Fokkema, SD and Glaser, R (eds), *Cognitive Psychology and Instruction*, 27–38, Plenum Press.

Hempel, CG, 1942. "The function of general laws in history" *The Journal of Philosophy* **39** 35–48. (Reprinted in Hempel, CG, 1965. *Aspects of Scientific Explanation: and other essays in the philosophy of science*, The Free Press.)

Johnson, LW and Solloway, E, 1985. "PROUST: An automatic debugger for Pascal programs" *Byte*. (Also in Kearsley, G, 1987. *Artificial intelligence and instruction—applications and methods*, Addison-Wesley.)

Kolodner, JL, 1983. "Maintaining organization in a dynamic long-term memory" *Cognitive Science* **7** 243–280.

Kolodner, JL, 1991a. "Improving human decision making through case-based decision aiding" *AI Magazine*, 52–68.

Kolodner, JL, 1991b. "Helping teachers teach science better: case-based decision aiding for science education" In: Birnbaum, L (ed), *Proceedings of 1991 International Conference on the Learning Sciences*, 274–280, Association for the Advancement of Computing in Education.

Kolodner, JL, 1992. "An introduction to case-based reasoning" *Artificial Intelligence Review* **6** 3–34.

Kolodner, J, 1993. *Case-Based Reasoning*, Morgan Kaufmann.

Koton, PA, 1989. "A method for improving the efficiency of model-based reasoning systems" In: Horn, W (ed), 1990. *Causal AI Models: Steps Toward Applications*, 273–282, Hemisphere Publishing.

Lajoie, SP and Lesgold, A, 1992. "Apprenticeship training in the workplace: computer-coached practice environment as a new form of apprenticeship" In: Farr MJ and Pstoka, J (eds), *Intelligent Instruction by Computer: Theory and Practice*, Taylor and Francis.

Newell, A, 19??. "The knowledge level" *Artificial Intelligence* **18**, 87–127.

Ohlsson, S, 1993. "Impact of cognitive theory on the practice of courseware authoring" *Journal of Computer Assisted Learning* **9** 194–221.

Redmond, M, 1990. "Distributed cases for case-based reasoning: facilitating use of multiple cases" In: *Proceedings of the National Conference of Artificial Intelligence AAAI-90*.

Redmond, M, 1991. "Improving case retrieval through observing expert problem solving" In: *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*, 516–521.

Reimann, P, Wichmann, S and Schult, TJ, 1993. "A learning strategy model for worked-out examples" In: *Proceedings of Artificial Intelligence in Education AI-ED 93*, 290–297, Association for the Advancement of Computing in Education.

Rosenblum, JA and Moore, JD, 1993. "Participating in instructional dialogues: finding and exploiting relevant prior explanations" In: *Proceedings of Artificial Intelligence in Education AI-ED 93*, 145–152, Association for the Advancement of Computing in Education.

Riesbeck, CK and Schank, RC, 1989. *Inside Case-Based Reasoning*, Lawrence Erlbaum.

Schank, RC, 1975. *Conceptual Information Processing*, North-Holland.

Schank, RC and Abelson, RP, 1977. *Scripts, Plans, Goals, and Understanding*, Lawrence Erlbaum.

Schank, RC, 1986. *Explanation Patterns—Understanding Mechanically and Creatively*, Lawrence Erlbaum.

Schank, RC, 1991. "Case-based teaching: four experiences in educational software design", Technical Report No, 7 Institute for the Learning Sciences, Northwestern University.

Schank, RC, Ferguson, W, Birnbaum, L, Barger, J and Greising, M, 1991. "ASK TOM: An experimental interface for video case libraries" In: *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society*, 570–575.

Schult, TJ, 1993. "Tutorial remindings in a physics simulation environment" In: *Proceedings of AI-ED 93*, 105–112, Association for the Advancement of Computing in Education.

Schult, TJ and Reimann, P, 1993. "Making case-based tutoring more effective" In: *Proceedings First European Workshop on Case-Based Reasoning EWCBR-93*, 385–390.

Sokolnicki, T, 1991. "Towards knowledge-based tutors: a survey and appraisal of intelligent tutoring systems" *The Knowledge Engineering Review* **6** (2) 59–95.

Stevens, A, Collins, A and Goldin, SE, 1992. "Misconceptions in students' understanding" In: Sleeman, D and Brown, JS (eds), 1982, *Intelligent Tutoring Systems*, Academic Press.

Sycara, K, 1988. "Using case-based reasoning for plan adaptation and repair" In: *Proceedings Workshop on Case-Based Reasoning—DARPA*, 425–434.

Vanlehn, K, 1987. "Learning one subprocedure per lesson" *Artificial Intelligence* **31** 1–40.

Vanlehn, K, Ohlsson, S and Nason, R, 1994. "Applications of simulated students: an exploration" *Journal of Artificial Intelligence in Education* **5** (2) 135–175.

Watson, I and Marir, F, 1994. "Case-based reasoning: a review" *The Knowledge Engineering Review* **9** (4).

Weber, G, Waloszek, G and Wender, KF, 1988. "The role of episodic memory in an intelligent tutoring system" In: Self, J (ed), *Artificial Intelligence and Human Learning: intelligent computer-aided instruction*, Chapman & Hall.

Weber, G, 1991. "Explanation-based retrieval in a case-based learning model" In: *Proceedings of The Thirteenth Annual Conference of the Cognitive Science Society*, 522–527.

Weber, G, 1993. "ELM: case-based diagnosis for program code in a knowledge based help system" In: *Proceedings First European Workshop on Case-Based Reasoning EWCBR-93*, 391–395.

Wenger, E, 1987. *Artificial Intelligence and Tutoring Systems*, Morgan Kaufmann.

White, BY and Frederiksen JR, 1987. "Qualitative models and intelligent learning environments" In: Lawler, RW and Yazdani, M (eds), *Artificial Intelligence and Education, volume one*, Ablex.