Computational Mathetics: Towards a Science of Learning Systems Design

John Self

HAL Id: hal-00190503

https://telearn.hal.science/hal-00190503

Submitted on 23 Nov 2007

DRAFT ONLY
All comments are very welcome

N.B. This document is 148 pages long, with the current page and font settings.
Before printing it, please think of the trees.

Computational Mathetics:
Towards a Science of Learning Systems Design

John Self
Computer Based Learning Unit
University of Leeds
jas@cbl.leeds.ac.uk

November 1995

CONTENTS

1. Introduction

The aim of this report is simply to help put the design of computer-based systems to support learning on a more scientific footing. The aim is simply stated, but its achievement is more difficult. For one thing, it is not at all obvious what "more scientific" means in this context.

The rather clumsy expression "computer-based systems to support learning" will henceforth be abbreviated to "AI-ED systems", that is, "Artificial Intelligence in Education systems" on the grounds that the design principles will be primarily derived from and expressed in the language of Artificial Intelligence. It will become clear that we intend a broad interpretation of the term 'AI-ED system'. We mean any computer-based learning system which has some degree of autonomous decision-making with respect to some aspect of its interaction with its users. This decision-making is necessarily performed on-line, during its interaction with users. Consequently, the system needs access to various kinds of knowledge and reasoning processes to enable such decisions to be made.

Of course, computers may be used as presentational devices, through which carefully pre-designed instruction involving the new technological media is delivered to students. There are, no doubt, considerable potential benefits in this, as computers enable special effects, such as altered time-scales and alluring graphics, to focus students' attention. However, pre-designed instruction assumes that the designer can fully anticipate the reactions of all its users and can build in responses to those reactions, or that users themselves are sufficiently self-aware that they can reliably decide how to use systems (assuming that the required options are, in fact, available). It takes no account of the computer's ability to reason, for itself, about the course of the interaction, as we assume that a good human teacher needs to do.

As we will discuss, the field of Artificial Intelligence in Education (AI-ED) has had a short but chequered history. The initial explorations in the 1970s were marked by the kind of enthusiastic optimism characteristic of AI in general. This early work made significant contributions to both AI and Education. By the 1980s, the successful applied AI work on expert systems and several national programmes seeking to capitalise on AI research led to an imperative to develop AI-ED systems which were practically useful, rather than theoretically interesting. It is notable that few of the AI-ED pioneers ever expressed much confidence that the time was right for practical development. Inevitably, the eventual, perceived failure of the applied projects only confirmed that learning and teaching are intrinsically difficult processes.

Meanwhile, the new technologies, especially multimedia and networking, promised other solutions to what were considered to be serious educational problems. Consequently, it became unwise to continue suggesting AI-ED system development. Now, however, the pendulum is swinging back, again inevitably, as it is realised that the new technologies need to be supported by the kinds of analysis of learning and teaching which AI-ED research carries out.

Below the ebb and flow of research fashion, there has been continuing, if slow, progress in understanding the nature of AI-ED system design. Eventually, this understanding will find a proper place in the design of computer-based systems to help people learn. AI-ED systems are neither a panacea nor an irrelevance - they have a contribution to make. One of our aims is to help develop techniques to clarify its potential contribution.

1.1 The AI-ED context

In order to help place AI-ED systems in a realistic context, let us briefly consider four learning vignettes:

Aboriginal culture

Until recently, Australian aboriginal children would sit under the coolabah tree and listen to stories such as following:

> "Brolga was the favourite of everyone in the tribe, for she was not only the merriest among them, but also the best dancer. The other women were content to beat the ground while the men danced, but Brolga must dance; the dances of her own creation as well as those she had seen. Her fame spread and many came to see her. Some also desired her in marriage but she always rejected them. An evil magician, Nonega, was most persistent in his attention, until the old men of the tribe told him that, because of his tribal relationship and his unpleasant personality, they would never allow Brolga to become his wife. "If I can't have her," snarled Nonega, "she'll never belong to anyone else." One day, when Brolga was dancing by herself on an open plain near her camp, Nonega, chanting incantations from the centre of a whirlwind in which he was travelling, enveloped the girl in a dense cloud of dust. There was no sign of Brolga after the whirlwind had passed, but standing in her place was a tall, graceful bird, moving its wings in the same manner as the young dancer had moved her arms."

The brolga is a beautiful grey bird which dances on the flood plains of northern Australia. Many aboriginal myths explain features of the environment (animals, rocks, stars, and so on) as being derived in some way from human beings. These stories, which were entirely verbal, there being no written form of communication, were accepted as truth and dictated all aspects of aboriginal behaviour. According to Roberts and Mountford (1969), young children did not "receive any formal education as we know it. They appear to do just as they please". At a certain age, a youth was taken from the main camp to live with the old men of the tribe, the sole repositories of tribal law and wisdom, who during many years of training, taught him the laws of his community, the relationship he bore to every member of it, and the secret myths and rituals of adult life.

All cultures have their myths and rituals which are communicated in a similar way. Only the most fervent technologist would imagine that computer systems could or should change these processes in any significant way. Whatever one's views of Australian aboriginal culture, its transmission by some multimedia AI-ED system is a bleak vision.

Funfair physics

The traditional funfair provides many opportunities for children to learn or reinforce concepts of physics. The helter-skelter (a high spiral slide) gives lessons on centrifugal force, gravity and friction: most children can predict the effect of a higher slide or a heavier child, and know the direction they will shoot out at the bottom. The bumper car or dodgem is an exercise in the conservation of momentum: children soon learn where to bump to cause the maximum effect. The big dipper tells them about potential energy and kinetic energy: children know where to sit in the train to experience the greatest acceleration. The coconut shy is about force and impulse. And the ghost train warns children not to believe what they see and feel.

Funfairs are increasingly out of fashion but there are many other activities which enable children to develop intuitive notions of physics. Nowadays, children are more likely to find amusement in computer games, which may show activities violating the laws of physics and much else besides. We can, of course, imagine designing computer games which set out to be faithful to real-world physics and which may therefore lead to sound intuitive concepts. Maybe such games would help in the transition from intuition to a more scientific view of physics. In such a case, the computer game might benefit from some understanding of the nature of 'informal' and 'formal' physics. As we will see, AI-ED systems will be concerned with the nature of intuitive understandings and how they might be changed, and with the degree to which understanding has to be grounded in authentic situations.

The Play of Daniel

University music students might be asked to write an essay on Beauvais' Play of Daniel, a medieval cathedral play. The successful completion of such a task requires the use of a range of skills and knowledge. Students need to know how to write essays in general, which presumes, of course, knowledge of a natural language. They need to be able to adapt the essay to meet the requirements, bearing in mind who will read it and for what purpose. They will need to have a broad knowledge of music in order to be able to make sense of the unusual Play of Daniel. They will also need to know about the social context at the time the Play was composed in order to understand the point of the Play. From all this, they will need to know just what points to emphasise to make a successful essay. Not all this knowledge will be to hand at the time the task is set, so students need also to know how to acquire the knowledge they need. This might involve knowing how to use various computer-based aids for accessing resources.

There are various ways in which computer-based systems might support this activity. Essay-writing is a rather complex skill with which computer-based systems will not (in the near future) be able to give detailed guidance, although various kinds of clerical assistance are possible. Computer-based systems might be able to give advice on how to set about gathering the information probably needed, for example, to determine the date of the music and how it was typically performed. They might also be able to monitor the student's use of the resources and to give advice if the student exhibits problems or deficiencies in her search strategies. Overall, the sheer volume and complexity of the knowledge involved suggests that computer systems will not be able to provide reliable step-by-step guidance for the whole process. However, given the nature of the students involved, this is not what is required, anyway.

Satellite surveillance

Satellite activity analysts are employed by government defence departments to maintain dossiers on the behaviour of earth-orbiting satellites. In particular, they have to provide possible explanations for unusual behaviour, for example, that the satellite is mal-functioning or has been diverted to survey Colombian drug plantations. The satellite's behaviour is displayed on a complex computer graphics screen, which must, of course, be interpreted by the analysts. Naturally, their employers would like the analysts to develop excellent explanatory skills, because the cost of reacting to a faulty explanation can be considerable.

The analyst's task is not a simple one of mapping patterns of observations onto explanations. The data is generally voluminous but also incomplete and possibly unreliable. It would not be adequate to train analysts to recognise

specific situations: they need to be helped to develop general explanation-forming skills, in which they propose hypotheses, gather evidence for those hypotheses, assess the reliability of evidence, and present a convincing argument for their conclusions. It is hard to imagine this training being successful in a context separate from that in which the task is normally performed. The trainee analysts would need full access to the computer display of satellites' behaviour and a way of exploring that system to create and test out hypotheses. In this case, then, the training would necessarily be computer-based, being embedded in the system used for task performance or in an extension of it. The computer-based system would probably need to know about general hypothesis-forming skills if it is to guide the trainee towards improving them. (I am grateful to the Mitre Corporation for showing me a prototype computer-based tutor for this task.)

1.2   What is education?

If a field is to call itself 'AI in Education', then it seems necessary for it to say what it considers 'education' to be. However, despite its name, AI-ED has never been concerned with education in its broad sense but only with the specific issue of learning. We may believe that the whole purpose of education is to promote learning but in reality the process of education includes many activities only indirectly related to learning, as any textbook or conference on 'education' will confirm.

The term 'education' is generally taken to mean 'formal education', that is, 'paid-for education', rather than the 'informal education' that we receive for free from our culture. There is a nostalgic preference for the latter, with the former being considered to stunt individual learning capabilities. These polemic views will not be our concern. We will be concerned only with the nature and effectiveness of the learning processes.

We will avoid simplistic assertions that learning happens in a particular way. Advocates of one method of learning will naturally belittle other methods. However, learning is a complex, many-faceted kind of activity, as our vignettes above indicate. The nature of computer-based support for learning will depend on the context and there is no particular approach which can be categorically labelled as wrong-headed.

Consider, for example, the teaching and learning of a skill such as playing the violin. This has evolved largely outside the formal education system and it shows an amalgam of many different kinds of activity. There is a lot of rather repetitive practice of scales. There is a fair amount of 'academic learning' to develop fluency with musical notation. There are occasional intense one-to-one tutorial sessions. There are some sessions in a group, as music playing is a social activity. There are specialised 'learning environments', such as small-scale violins and the Suzuki method. No one of these learning methods is intrinsically better than the others: they must all be integrated in a successful learning experience.

1.3   What is AI?

The key difference between AI and other forms of computer programming is that AI programs respond intelligently to situations not specifically anticipated by the programmer. In conventional programming, the programmer arranges for anticipated problems to be solved by specifying all the steps towards a solution. In AI programming, the programmer provides the means for the computer to solve problems as they arise. For example, a program to translate between languages could not be written by anticipating all possible sentences and providing translations of them, nor even by listing all the words and their translations and combining them in a simple way. A comprehensive translation program would need to reason about the meaning of the sentences, which implies that it has knowledge about both languages, about the content of the sentences, and about the world, so that ambiguities may be resolved.

AI is both an applied and a theoretical subject. AI applications are very diverse:
- to recognise bridges and buildings from photographs (so that they may be bombed perhaps).
- to work as autonomous robots in dangerous situations, for example, underwater mining.
- to help plan traffic flow.
- to diagnose diseases.
and so on. All these applications require skills we would normally describe as intelligent: some demand specialist knowledge (for example, of obscure diseases); others use everyday knowledge, which we all use without being aware of how complex it is (for example, recognising what is depicted in a photograph).

Important though AI applications might be, we will be more concerned with the theoretical side of AI. AI is not a science that studies objects in the natural world: it studies objects that AI programmers create. In order for these creations to be understood and analysed, their design has to be based on clearly-articulated principles capable of some kind of rigorous analysis. A program for, say, medical diagnosis should be based upon a computational theory of diagnosis. By a 'computational theory' we mean one that is amenable to conventional mathematical analysis and that is oriented towards implementation as a computer program. It should be possible to prove practical results, for example, that under specified conditions, a diagnosis will be possible in a certain time. It should also be possible to carry out empirical experiments with the program, for example, to measure how it performs under different conditions. The

theory develops by coordinating mathematical and empirical studies. In this way, AI leads to the development of new theories, because if an adequate theory already existed it would presumably be programmed in a conventional way.

However, AI would not be a coherent field of study if every application required the development of its own computational theory. It turns out that a theory of diagnosis contains many components which are the same or similar to those required for a computational theory of, for example, planning traffic flow. Both, for example, require forming hypotheses from observations (a rash of purple spots suggests meningitis; a traffic jam suggests a traffic light failure), both require a form of hypothetical reasoning of the "What would happen if .." kind, both perhaps require reasoning from a library of previous cases so that the system does not have to solve every problem from scratch, and so on. Theoretical AI is concerned with the development of methods for analysing such processes independent of any particular application.

1.4   What is AI in Education?

The field of AI in Education is concerned with the application of AI techniques to educational problems. Therefore, AI in Education is part of applied AI, and indeed most practitioners are happy to regard it as so, seeking to develop important, practically useful systems based on AI. Most reports of AI-ED projects give details of the technical design of systems and provide some evidence that the systems are effective. In the next chapter, we will review the status and achievements of AI-ED research.

Our emphasis will be more on relating AI-ED to theoretical AI. As an application area, AI-ED is immensely complicated, not just because of technical difficulties but more especially because education and learning are controversial topics about which there are endless arguments. Any particular AI-ED project has to commit itself to a point of view if it is make any progress in implementing a useful system, which will then, no doubt, be criticised by those with a different view.

AI-ED is interesting because of this constant interplay of ideas and it is important because of the potential contribution to the socially central aim of improving the quality of learning. Contributions to AI-ED come from many directions: primarily from computer science, psychology and educational research, but also from sociology, anthropology, philosophy and the many fields which are the topic of AI-ED systems. Theoretical debate in AI-ED is generally expressed in lowest common denominator terms so that it is accessible to all participants, that is, in informal language. Our aim is to suggest that it is time AI-ED begins to move in the direction that all scientific endeavours take in due course, by developing a formal, technical language which can be used to make arguments more precise and AI-ED system design more analytic. The language of theoretical AI is the most promising starting point, because it already has partial formalisations of processes such as reasoning, learning, diagnosis and dialogue which are central to AI-ED.

1.5   Outline

The next chapter attempts to provide a brief, non-technical review of the AI-ED field. This is somewhat hard to do without lapsing into providing a catalogue of AI-ED systems, the various techniques used to implement them, and the educational philosophies which they demonstrate. It is also difficult because the boundaries of AI-ED are rather vague. The chapter tries to give an impression of the kinds of issues which concern AI-ED researchers at the moment and to indicate the level of practical achievement. The intention is that this chapter provide sufficient background to justify the need for the more theoretical descriptions of following chapters.

Chapter 3 introduces what we have chosen to call 'computational mathetics', for reasons that will be explained there. The general need for computational mathetics and its aims and methodologies are described. In brief, computational mathetics is intended to provide the more formal analyses needed to complement present informal argumentation and design. However, the level of formality is still low compared to other areas of theoretical AI, reflecting the difficulty of AI-ED and the little work so far done in this direction. At least this enables the discussion to be followed by those of a non-formal orientation.

The following chapters each consider a topic within the scope of computational mathetics: representing knowledge, reasoning, metacognition, learning, diagnosis, dialogue and instruction, respectively. These chapters review work in theoretical AI and in AI-ED itself from a computational mathetics perspective. There are two main objectives in these chapters. First, to show that there is a large volume of potentially relevant work which can be adopted and adapted to form a basis for the theoretical analysis of AI-ED research. If it is, we believe that it will lead, in due course, to improvements in AI-ED system design. The second objective is to show that there is much that needs to be done before the aims of computational mathetics may be achieved and hence to provide some targets and challenges for future AI-ED research.

2.   A brief review of AI-ED

The preface to Wenger's comprehensive panorama of AI-ED before 1987 (Wenger 1987) remarked that a similar review of a field then considered to be at an "important threshold of development" would not be possible five years later because there would be too much material to review. There has, in fact, been no general book published on AI-ED since Wenger (1987). All the many books related to the topic which have been published since 1987 have been monographs describing a particular project or edited collections of papers presented at workshops or conferences (Bierman, Breuker and Sandberg, 1989; Birnbaum, 1991a; Brna, Ohlsson and Pain, 1993; Clancey, 1987; Costa, 1992; de Corte, Linn, Mandl and Verschaffel, 1991; Elsom-Cook, 1990; Farr and Psotka, 1992; Frasson and Gauthier, 1990; Goodyear, 1991; Greer and McCalla, 1994; Lajoie and Derry, 1993; Larkin, Chabay and Sheftic, 1992; Mandl and Lesgold, 1988; Moyse and Elsom-Cook, 1992; Polson and Richardson, 1992; Regian and Shute, 1992; Schank and Cleary, 1995; Self, 1988). In addition to this spasm of new books, three new journals have started up (Journal of Artificial Intelligence in Education, Journal of the Learning Sciences, and Interactive Learning Environments) alongside longer-established journals with broader remits (such as the International Journal of Human-Computer Studies and Instructional Science). It is not easy therefore to gain a broad, balanced picture of the contemporary AI-ED field.

This brief chapter can hardly aspire to give such a picture. It aims merely to give a background to the issues which have been discussed in recent years sufficient for appreciating the more technical perspectives of later chapters. It is not organised, as Wenger's book was, as a historical catalogue of systems and projects. Today it is not possible to identify a similar set of classic on-going projects. The AI-ED field may have been on the 'threshold of development' in 1987 but it has, if anything, stepped back from this threshold rather than crossed it. There has been continued development of perhaps smaller-scale systems along 'traditional' lines, as we will see, but there has been much more debate about the direction of the AI-ED field, with many of the pioneers mentioned in the Wenger book leading the attempt to change it.

Any review of AI-ED should logically begin with a discussion of the educational problems which are being addressed before embarking on a survey of how AI might contribute to solutions. The main relevant issues appear to be the following:
- What is the nature of knowledge?
- How may knowledge be learned?
- Should systems instruct, tutor, guide or train students?
- How should new technologies be used in education?
- What are the measures of effectiveness?

Educationalists will debate such issues at great length but it is not the aim of this chapter to contribute to that debate except to the extent that it discusses the AI-ED field's views (implicit and explicit) on them. The review focusses on providing a basis for considering the technical contribution that AI is making and might make to education.

The following sections consider each of the above issues in turn. Each section illustrates a general discussion about AI-ED's views with exemplar AI-ED systems. The sections do not attempt to give a comprehensive account of implemented systems (there are now too many for a short review) and those systems referred to are described only to the extent necessary for the point under discussion. Technical concepts are usually only mentioned, with a fuller discussion to come in later chapters (although we have not interrupted with a multitude of forward references). The chapter ends with a discussion of the main controversies within the AI-ED field today.

2.1   The nature of knowledge

Most AI-ED systems are intended to help their student-users become more knowledgeable in some respect. AI-ED system designers are well aware that education has broader aims - to develop ethical and moral values, to improve attitudes, to nurture better citizens, and so on - but this awareness has only indirectly influenced their system designs. It has been rather assumed (or hoped) that the context in which AI-ED systems will be used will convey these broader goals.

2.1.1   Objectivism

Given the focus on the knowledge-to-be-learned, it seems natural that AI-ED system designers often begin by trying to specify this knowledge as precisely as possible. To achieve this, the full panoply of AI knowledge representation techniques (production systems, frames, semantic networks, predicate logic, and so on) has been applied in AI-ED systems. In so doing, AI-ED designers might be considered to be adopting a philosophy of knowledge called *objectivism*, which holds that the world may be completely and correctly structured in terms of entities, properties and relations and that rational thought consists of the manipulation of abstract symbols viewed as representing reality (Lakoff, 1987). Thus, an AI representation of knowledge might be considered to be an attempt to describe this

structure and the aim of an AI-ED system might be to help learners acquire the entities, properties and relations of this 'correct' propositional structure.

What might be considered the standard approach to AI-ED system design is illustrated by SPENGELS (Bos and van de Plassche, 1994), a straightforward intelligent tutoring system to help Dutch students learn the conjugation and spelling of English verbs, that is, to be able to use the correct form of verbs (such as 'prefer' and 'begin') in sentences such as "He ---- to work with pen and paper." The first step, as no spelling algorithm already existed, was to represent as a decision tree the morphosyntactic and spelling alternation rules taught in different Dutch textbooks. The decision tree effectively asks a series of questions: Is the verb form finite? Which tense is needed? Is the number singular? and so on, leading to a node of the tree showing the correct conjugation. This algorithm then becomes the basis for teaching the student, for deriving correct answers, for checking student answers, for determining misconceptions the student may have, and so on.

Some of the knowledge we wish students to acquire *is* objective because it is knowledge defined (by us) to be correct - for example, the syntax of programming languages or the allowable operations on an algebraic expression. It is no coincidence that the majority of AI-ED systems are concerned with such topics. The comment of GREATERP (Anderson and Reiser, 1985), a beginners' LISP tutor, that "You are within a PROG so you need to use a RETURN" leaves no scope for argument about the correctness of this statement (although there is scope for arguing about whether the student needs to be told so in a particular way and at a particular time).

There are many other domains where it seems necessary to adopt an objectivist view, to some extent. For example, a student on a first-aid course must be given the correct way of dealing with a child suspected of being accidentally poisoned. Again, this does not necessarily mean that a student must just be told the correct way. One can easily imagine that students will understand and remember better if they discover the correct way, in this case, preferably by experimenting with simulated patients, not real ones.

There are also domains where an objectivist view is adopted (temporarily, perhaps) as part of an academic game, by students and teachers. For example, the equations for uniform acceleration, although perhaps understood to be not always applicable, may be adopted as axioms to solve problems. In the case of English verb endings (discussed above), a pedant might point out that these endings are not fixed - they may be different in medieval English or American English - but if the learner's context is understood to be U.K. English, "preferred" is accepted to be correct.

There is a great debate within AI, and specifically within expert systems research, about the extent to which 'correct' knowledge can be specified in areas where it does not obviously exist. In the case of English verb endings it seems a reasonable expectation that correct rules can be specified but in more substantial areas of English use such an approach would probably not be contemplated. To the extent that 'correct' knowledge can be specified, such expert system representations may be used to convey it to students. The prototypical attempt to carry out this programme was the adaptation of the MYCIN expert system for medical diagnosis into the GUIDON tutoring system (Clancey, 1979, 1987).

However, even within domains for which objectivism seems reasonable, it soon becomes apparent that the real learning problems lie not in the objective knowledge but in its relation to less objective knowledge. For example, in programming, the focus moves from the syntax of the language to aspects such as programming design or debugging, where there is no definitively correct knowledge. The BRIDGE Pascal tutor (Bonar and Cunningham, 1988), for example, aims to guide the student through the stages of planning a program, from the initial English-like description through to the Pascal code.

Similarly, in algebra the issue is not so much what the operations are (syntactically) but when they should be applied. AI-ED algebra systems do not just check the correctness of operations (in fact, they often perform the operations themselves as that is assumed not to be the student's difficulty) but provide students with an environment in which they can experiment with the operators. For example, AlgebraLand (Foss, 1987) displays a problem-solving tree of the student's solution attempt. This is intended to make it easier for students to monitor their on-going solution and to reflect on their solution attempts afterwards. AlgebraLand itself gives no explicit tutorial support to these aspects - it only checks that an operator is applicable. The hope is that, relieved of the need to worry about the low-level detail of operator application, students will be more likely to engage in the desired metacognitive activities.

Even when a successful expert system can be built, it does not follow that the expertise embedded in it is a suitable basis for an educational interaction. The expert's performance-oriented knowledge may not be based on a conceptual structuring of the domain which a learner will understand (Clancey, 1984). To put it another way, many studies have shown expert-novice differences which suggest that novices may not learn well from experts. This conclusion is explicit in GREATERP, where the student's solution is not compared to an expert solution but to that of an 'ideal student'. The production rules of GREATERP do not summarise expert knowledge but are aimed to correspond to conceptual units that novices can understand.

2.1.2   Constructivism

A view of knowledge that is often presented as opposed to objectivism is that of *constructivism*, which holds that meaning is imposed on the world by us, rather than existing in the world independent of us. Constructivists therefore emphasise the processes of actively structuring the world and contend that there are many meanings or perspectives for any event or concept, rather than there being a single correct meaning towards which a student must be guided.

It does not follow that an AI-ED system which possesses a purportedly objective representation of knowledge has to adopt an interaction style which violates all constructivist principles. For example, the Socratic dialogues of WHY (Collins and Stevens, 1982) do not simply tell students the 'correct' conception but aim to help them construct one through subtle sequences of counter-examples and so on: "Do you think that any place with mountains has heavy rainfall?" "Yes." "Does southern California have a lot of rain?"

The designer of a system which possesses knowledge which is deemed to be correct may, however, be tempted to use that knowledge (usually acquired after great effort) in a direct knowledge communication mode: "No, southern California doesn't have a lot of rain." To avoid such temptations, an extreme constructivist might argue that an AI-ED system (assuming that it is still to be deemed an AI-ED system) should possess no such knowledge and simply present an environment for the student to explore. The deceptive word here is 'simply', for it is by no means easy to design environments from which knowledge can be discovered. The most well-known such project is LOGO (Papert, 1980, 1993) which has now been subjected to numerous studies and its constructivist foundation has become rather shaky, as the extent to which LOGO needs to be buttressed by other supports has been documented. Similarly, the recent enthusiasm for developing hypermedia and multimedia systems (Kommers, Jonassen and Mayes, 1992) for students to explore has to be tempered by the fact that unaided students have some difficulty in negotiating the vast search spaces. The technologists' manifestation of constructivism as microworlds and other exploratory environments - often contrasted with the tutoring systems of objectivists - is a rather pale image of the comprehensive philosophy of constructivism, which is "a discursive practice that provides the means through which one can describe the social, political and economic circumstances that surround and give meaning to a given piece of educational technology" (Sack, Soloway and Weingrad, 1994).

Advocates of constructivism might argue that *both* agents, the AI-ED system and the student, should adopt a constructivist approach. In this case, the system would not contain a priori correct knowledge but would attempt to discover it in a joint endeavour with the student. The PEOPLE-POWER system (Dillenbourg and Self, 1992) illustrates this approach. The student and the system are supposed to design experiments to be carried out on a simulated political system in order to determine what makes a system democratic in the sense that seats gained in a parliament are proportional to the votes cast for the corresponding parties. Both the student and the system can make (fallible) suggestions and interpretations. There is no target 'correct' knowledge available to the system.

2.1.3   Situationism

Situationism shares some tenets of constructivism but emphasises that the constructed knowledge does not exist in memory but rather emerges from interaction with the environment. It is argued that traditional AI, with its emphasis on symbolic knowledge representations, has assumed (sometimes explicitly, as in the classic Newell and Simon (1972) studies) that those representations have some psychological reality, that is, they correspond, not literally but functionally, with structures in memory. Situationists argue that representations are created in the course of activity but are not themselves knowledge, knowledge being a capacity to interact. Both constructivists and situationists deny that knowledge of the world can be defined independent of a mind, although the former but not the latter might accept that an individual mind creates its own idiosyncratic knowledge structures in memory.

The different perspectives of the proponents of situationism tend to lead to such a disavowal of one another's views that their intersection seems too small to provide an acceptable, brief summary of its principles. Any attempt, especially by non-situationists, to provide a simple statement is invariably met with detailed qualifications, conditions, extensions, and so on. Nonetheless, situationism is presented as a revolutionary philosophy providing a model of representation (which Bickhard and Terveen (1995) call 'interactivism') able to overcome the perceived impasse of contemporary AI (which they say is based on 'encodingism', that is, a presupposition that representation has the nature of encodings).

Situationism is very much a subject of debate within AI and cognitive science generally and within AI-ED in particular (Bickhard and Terveen, 1995; Clancey, 1992a; Hayes, Ford, and Agnew, 1994; Hoppe, 1993; Sandberg and Weilinga, 1992; Vera and Simon, 1993). As yet, there are no AI-ED systems which clearly illustrate its principles. Most discussions of situationism in AI-ED refer to the idea of cognitive apprenticeship (Collins, Brown and Newman, 1989) but cannot point to any exemplar systems in the way that objectivists might point to GREATERP, GUIDON, and so on. Clancey (1993) has, however, listed some principles for designers of such systems, contrasting them with the perceived emphases in objectivism-based AI-ED approaches:
  • Participate with users in multidisciplinary design teams.

- Adopt a global view of the context in which a computer system will be used.
- Be committed to providing cost-effective solutions to real problems.
- Aim to facilitate conversations between people.
- Realise that transparency and ease of use is a relation between an artifact and a community of practice.
- Relate schema models and AI-ED systems to the everyday practice by which they are given meaning and modified.
- View the group as a psychological unit.

2.1.4   Connectionism

C*onnectionism* is another view of knowledge which is presented as a contrast to the symbol-processing version of objectivism. Connectionism holds that knowledge is implicitly represented in the weights and links between large numbers of nodes modelled on neural networks. However, although connectionist representations lack the kind of symbolism characteristic of objectivist representations, they are still very much concerned with representations in memory, rather than in the situation. Connectionist methods have been used to develop components of AI-ED systems, but no AI-ED system follows a wholly connectionist philosophy, presumably because the representations themselves, with only implicit understanding, do not easily support learning interactions.

An interesting question, after reviewing how philosophies of the nature of knowledge relate to students' knowledge, concerns how those philosophies relate to teachers' knowledge (or AI-ED systems' knowledge of how to teach). Some researchers (for example, Clancey, 1987) have attempted to apply the expert system paradigm to teaching knowledge and to specify 'tutoring rules' to be interpreted by an AI-ED system. This, then, reflects an objectivist view that such knowledge exists and can be specified. Most AI-ED systems, however, do not have much explicit knowledge of how to teach: it is largely implicit in the way they react to certain situations. This might appear to be a situationist approach but, in fact, is not because the knowledge is clearly possessed by the system but not in a form which it is easy for observers to analyse.

2.2   The nature of learning

Philosophies of knowledge imply philosophies of learning, to some extent. For example, connectionism, with its assumption that knowledge exists in the weighted links between the nodes of large neural networks, implies that learning is a statistical process whereby the weights are adjusted as many examples and non-examples are encountered. Unfortunately, there is no unequivocal mapping from philosophies of knowledge to philosophies of learning and it is one of the difficulties of AI-ED research that heated arguments about the nature of knowledge often lead to similar conclusions about the nature of learning, teaching and AI-ED system design. For example, an objectivist might not demur from some of the principles derived from situationism listed above, for example, that it is time to move on from 'laboratory studies' to putting more emphasis on 'cost-effective solutions to real problems'.

One of the few attempts to link a theory of learning to that of AI-ED system design is that to relate ACT* (Anderson, 1983) to GREATERP and similar systems. The following principles are said to follow from ACT* (Anderson, Boyle, Farrell and Reiser, 1989):
- Represent the student as a production system.
- Communicate the goal structure underlying the problem-solving.
- Provide instruction in the problem-solving context.
- Promote an abstract understanding of the problem-solving knowledge.
- Minimize working memory load.
- Provide immediate feedback on errors.
- Adjust the grain size of instruction with learning.
- Facilitate successive approximations to the target skill.

These principles do not follow in the mathematician's sense of being derivable from axioms of the theory but are more like implicit implications. Moreover, the principles do not lead directly to design prescriptions. These weak links make it hard to argue that the success (or otherwise) of the systems implemented can be attributed to the psychological theory.

Overall, AI-ED system design reflects a rather eclectic view of the nature of learning, regardless of views of the nature of knowledge. Many different kinds of event and activity can lead to learning and many of them have been supported, to some extent, within AI-ED systems (usually without the dogmatic claims that accompany discussions about the nature of knowledge).

2.2.1   Failure-driven learning

ACT* is an essentially objectivist theory emphasising stored schemas in memory. Recently, ACT* has been modified to encompass some aspects of constructivism (Anderson, 1993) but these modifications have yet to lead to significantly modified principles for AI-ED system design. As they stand, GREATERP and its brothers are

remediationist systems based on the assumption that learning is failure-driven, that is, that the occurrence of failure provides the opportunity for learning.

Many other approaches have the basic idea of failure-driven learning. For example, SOAR (Laird, Rosenbloom and Newell, 1986) is intended to be a comprehensive cognitive architecture based entirely on a process of 'impasse-driven learning'. An impasse is a situation where the architecture has insufficient knowledge to determine how to proceed. The impasse triggers a heuristic search to create a new operator to overcome it. Similarly, VanLehn's theory (VanLehn, 1990) is an impasse-driven one, derived from the influential 'repair theory' (Brown and VanLehn, 1980) originally developed to explain how students learned 'bugs' in subtraction.

2.2.2   Case-based learning

The 'failure' does not have to be a blatant exhibition of lack of success: it could just be some evidence which causes the student to consider whether their current conception is sound. For example, the idea of case-based teaching (Schank, 1990; Schank and Cleary, 1995), derived from the field of case-based reasoning in AI (Kolodner, 1993), is that students learn from stories (cases) presented at the precise point of becoming interested in knowing the information conveyed by the story. For example, DUSTIN is a language-training system with which students enter a multimedia simulated environment in which they interact with (images of) people they will deal with in their work environment. The student attempts authentic tasks, such as checking into a hotel, and on failure is shown a relevant example before re-attempting the task. This approach is an interesting merger of objectivist methods (there are ostensibly correct representations of how to perform the task) with a constructivist philosophy (with rationales such as "in order to assimilate a case, we must attach it someplace in memory") and a situationist style (with the emphasis on authentic tasks).

The 'case' presented to a student may be
- a very short piece of text (as in a counterexample in a WHY dialogue, as above);
- a complex photograph (as in the Sickle Cell Counselor (Bell and Bareiss, 1993), a system designed to teach museum visitors about sickle cell disease);
- a paragraph giving a case history (as in DECIDER (Bloch and Farrell, 1988), which gives summaries of events such as the U.S. invasion of Nicaragua while the student is expressing political beliefs);
- a longish video (as in JASPER (Crews and Biswas, 1993), where students are presented a story in which the characters are faced with challenges that the students must solve).

In the last example, the video becomes a motivating way to present complex problems for students to solve. The use of video supports an avowed constructivist philosophy, emphasising that students construct knowledge in realistic situations rather than receive divorced classroom instruction. Implicit in this approach is a belief in learning by problem-solving, an approach also characteristic of an objectivist philosophy, which would also emphasise that an AI-ED system itself should be able to solve the problems it sets. (In fact, of the systems mentioned in the previous two paragraphs only DECIDER does not have (or cannot work out) a correct solution.)

A learning by problem-solving approach can be rationalised by many different philosophies and supported by many different styles of AI-ED system; for example,
- GREATERP students solve problems and receive immediate feedback on mistakes (the system being able to monitor each step of a solution);
- LOGO students receive feedback from the system when solutions are executed but receive no didactic help;
- WEST students (who learn arithmetic skills in the context of a simple board game (Burton and Brown, 1979)) are given hints from the system if certain constraints are violated;
- JASPER students may receive hints to help them improve their solutions (earlier versions of JASPER had students solving the problems off-line).

So to say that most AI-ED systems reflect a learning by problem-solving philosophy is not very illuminating unless the nature of the problem and the degree of system support are clarified.

2.2.3   Learning through experimentation

A standard scenario is a problem-solving environment in which students perform experiments and are guided by the system in their interpretation. For example, QUEST (White and Frederiksen, 1990) provides a graphic simulation of circuits to enable students to understand principles governing the behaviour of those circuits by performing troubleshooting operations. For such an interaction to be useful to students, the system's interventions must be couched in terms analogous to those of the student: thus, for novices, the system needs representations of naive qualitative physics. The topic of qualitative reasoning is another broad field of AI whose application to AI-ED has still to be explored in detail, although it was arguably initiated by early AI-ED studies of the SOPHIE system (Brown, Burton and de Kleer, 1982).

The tension between extreme objectivist and constructivist/situationist views is well illustrated by discussions about how students might learn the kinds of causal models needed in science. An objectivist might present the standard formulas and require students to apply them to various (textbook) problems; a situationist might expose the student to many real-world instances and hope that generalisations will (implicitly, perhaps) evolve. White (1993) argues that causal models of an intermediate degree of abstraction can foster learning provided that they are:

- Understandable, that is, they build on intuitive notions of causality and mechanism;
- Learnable, that is, they generate explanations of key domain phenomena;
- Transferrable, that is, the objects and actions within them are represented in a decontextualised form;
- Linkable, that is, they help link different levels of abstraction and different model perspectives;
- Usable, that is, they can be used to predict, control and explain physical phenomena.

For example, the ThinkerTools curriculum includes a set of interactive simulations, such as the dot-impulse model, with which students apply horizontal or vertical impulses (using a joystick) to a ball with the objective, for example, of navigating a track to stop on a cross. There are four linked representations of motion: the motion of the ball itself, dots indicating the ball's velocity, arrows whose motion indicates velocities along the x and y axes, and a datacross, which represents a two-dimensional speedometer. The simulation is intended to help students develop the fundamental concepts of Newtonian mechanics such as impulse, velocity, force and acceleration.

2.2.4 Learning through dialogue

When students interact with a simulation, they tend to focus on tweaking the simulation to achieve the desired short-term effect without addressing the mistaken beliefs and conceptions which will continue to cause difficulties in the longer-term. There appears to be a need to engage the student in a dialogue to get at the fundamental misconceptions. This dialogue may be with a human teacher, other students or a computer-based learning environment - but in any case it reflects a constructivist view that knowledge is structured by interpreting events, albeit that this interpretation requires the mediation of other agents rather than isolated cogitation. Along these lines, Pilkington, Hartley, Hintze and Moore (1992) describe an environment with which students express and withdraw commitments during some debate, the system acting as a 'referee' using the guidelines of dialogue game theory to determine the validity of moves. Such an interface, it is argued, might help students not only clarify their conceptions of the domain under debate but also develop general reasoning skills. Eventually, the nature of such a debate may be sufficiently understood that the system itself may adopt the role of a player as well. As Baker (1994) describes, this work is derived from many fields of AI (belief revision, agent theory, distributed AI) and elsewhere (in cognitive and social psychology and the language sciences).

The work on self-explanation, that is, the hypothesis that better students spend more time explaining examples to themselves (Chi, Bassok, Lewis, Reimann and Glaser, 1989), can be interpreted as a theory about the benefits of arguing with oneself. The self-explanation line of research has recently (VanLehn, 1993) been developed into a proposed general methodology for AI-ED research:

- Collect experimental protocols of learners and divide them into good and poor learners (on the basis of outcome measures);
- Investigate what behaviours and processes were different in the two groups (e.g. self-explanation);
- Develop a cognitive simulation model to account for the identified differences (e.g. the CASCADE system (Jones and VanLehn, 1992));
- Design appropriate interventions to cause the more effective behaviour to occur (e.g. strategically hide information to encourage self-explanation);
- Test the resultant AI-ED system.

2.2.5 Learning as a social activity

AI-ED research has not taken much account of the social and cultural settings within which AI-ED systems have to be designed and used. Until recently, the emphasis has been on the technical challenge of constructing interesting systems within research laboratories. However, when such systems are used in classrooms, the effects are not usually as intended (perhaps not surprisingly). For example, Schofield, Evans-Rhodes and Huber (1990) report that when a geometry tutor (based on the Anderson principles itemised above) was tested in schools, both teachers' and students' behaviours changed in not entirely anticipated ways. Although teachers devoted more time to slower students and adopted a more collaborative style and students increased their effort on tasks (all presumably welcome changes), it was also found that the system increased competition among the students. Because students could progress at their own pace (unlike in the normal classroom) and could easily determine the progress of their co-students, a race developed between them - in fact, 40% of the students attributed their greater effort to the increased competition. The self-pacing feature also led to a modification in teachers' grading practices, as it was now less appropriate to mark students on the percentage correct. Instead they tended to assess on the effort invested.

These kinds of observation lead naturally to proposals for 'socio-technical design' (Clancey, 1993), where the emphasis is on designing a system within the social and physical context in which it is intended to be used. Such proposals are

often couched in political terms, presenting such an approach as more 'democratic' because it involves user groups in the decision-making and control of the systems they will use (as opposed to a 'dictatorial' approach in which designs are delivered to users). In particular, user-participatory design, a trend in human-computer interaction and usability research, has recently been applied to AI-ED system design (Murray and Woolf, 1992). The project involved:

- developing a representational framework for domain content and tutoring strategies that was understandable by educators,
- implementing a set of knowledge acquisition tools, and
- involving educators in building the system, through conception, design, implementation and evaluation.

This line of work is part of a broader discussion about the general principles of instructional design theory and knowledge acquisition in AI.

An extreme objectivist might argue that when all the knowledge-to-be-learned and the knowledge-of-how-to-teach-it has been fully specified, the delivery of AI-ED systems to the classroom will not be problematic. The design will take account of all situations and the system will adapt itself accordingly. This assumes that a complete cognitive analysis will subsume the affective dimensions. As Lepper, Woolverton, Mumme and Gurtner (1993) remark, most AI-ED systems only indirectly consider issues such as motivation, whereas studies of human tutors show that they devote more time and attention to motivation and affect than to the strictly cognitive content, especially for certain classes of learners such as remedial students. Human tutors' techniques for maintaining or increasing motivation - based on manipulating the goals of confidence, challenge, control and curiosity - can be seen to be implicitly encoded in some AI-ED systems to the limited extent that some of these techniques seem applicable to such systems.

In any case, situationists would not accept that the goal of explicitly defining all relevant knowledge in deliverable AI-ED systems is a sensible one. It simply does not take account of the fact that the teacher-learner culture is too rich and that the people involved in the use of such systems are able to (indeed, must) contribute to successful design and use of the systems. Because situationists hold that knowledge does not reside in individual heads, they would also move away from one-to-one tutoring systems (which are caricatured as aiming to transfer knowledge to individuals) and encourage more collaborative learning systems, where understanding is developed by group negotiations (as constructivists would accept). Situationists would tend to play down the role of AI *within* computer-based systems, that is, to provide explicit symbolic reasoning, and argue that AI's role is to mediate the collaborative interactions. They would, therefore, seek bridges to work on computer-supported collaborative work and computer-mediated communication. This opens up a debate about the nature of educational institutions and students' activities within and without them which would be too broad to pursue here.

2.3 Styles of interaction

The view of teacher expertise embedded in present AI-ED systems is, it has to be admitted, rather naive. This is because the nature of teacher expertise is not sufficiently clearly known and because AI-ED system designers generally do not have themselves or have access to such expertise. In addition, perhaps surprisingly, the teaching component has often been considered to be of less importance than, for example, the representations of domain knowledge and, therefore, has often been added on as an afterthought.

However, the earlier sections have given many examples of the various teaching styles adopted by AI-ED systems. The old distinction between theories of learning as being descriptive and theories of instruction as being prescriptive is rejected by AI-ED research. For example, VanLehn's proposed methodology, given above, assumes that identifying learning differences will lead directly to prescriptions for instructional interventions.

The criticisms of the styles of present AI-ED systems which are often made are rather misplaced. Few of these systems aims to provide a comprehensive coverage of either a significant part of a curriculum or the range of teaching styles. Rather, each system is an investigation of one style applied to one rather circumscribed topic. Thus, we should consider whether the teaching style of a system is appropriate for the limited aims that its designers have. For example, it is inappropriate to criticise GREATERP for its domineering style of putting students right immediately they stray off the correct path if this is an effective strategy for bringing large numbers of beginning LISP programmers up to a standard of competence after which more subtle strategies may be needed. Similarly, those systems which have a clear training objective, for example, the Space Shuttle Fuel Cell Tutor (Duncan, 1992) and Sherlock (Lesgold, Lajoie, Bunzo and Eggan, 1992), an avionics troubleshooting tutor, where it is essential that students master the operation of complex equipment, may quite justifiably adopt an essentially objectivist approach of defining the knowledge-to-be-learned and ensuring that students acquire it as effectively as possible.

For many AI-ED systems, however, the aims are not so clear-cut. Often there is a 'surface' objective for the student (to write a program to draw a specific shape; to manipulate parameters to maintain an economic simulation in a stable state; to solve a specific algebraic equation) which masks the real objective (to develop various higher-order skills, such as planning and monitoring solution attempts). System interventions directed at the former objective (for example, to point out that a program is incorrect) are irrelevant or harmful if they interfere with the latter objective. Many years ago the designers of the WEST system (Burton and Brown, 1979) proposed instructional guidelines such

as "do not tutor on two successive moves" which only make sense if it is accepted that the system's aims are more than to ensure that the student obtains the 'right answer'. With such systems the balance between 'guiding', 'telling' or 'leaving' the student, and hence the whole vexed issue of the balance of control between the learner and the system, is a continuing debate. The specification of precise and general guidelines has proved elusive and the design of the instructional component of AI-ED systems remains more an art than a science, as it does for other educational systems.

2.4 New technologies in education

Regardless of philosophy, psychology, or any other academic consideration, it is undoubtedly the case that the new technologies increasingly being applied to education have stimulated some of the trends discussed above. For example, the advent of high-fidelity multimedia and virtual reality systems naturally leads to its enthusiasts arguing for the merits of learning through 'immersion in a situation', which is a variation of the situationist's view. Similarly, the availability of high-speed networks permits a degree of distributed, collaborative working which was previously unattainable and this leads to discussions about the intrinsic virtues of 'social learning' mediated by technology.

This review is concerned specifically with the role of AI in education and we will not discuss the technical details of new technologies but only the potential relevance of AI to them. At the moment, the excitement with the new technologies owes nothing to AI. However, as the history of educational innovation shows, new technologies tend not to deliver all that they promise and it is quite predictable that as the limitations of the new technologies become clearer, so AI techniques will be adopted to help overcome them; for example:
- The successful use of multimedia interfaces requires models not only of the media themselves, but of the user, task and discourse (Maybury, 1994), aspects that have long been studied in conventional AI-ED research. No doubt existing work on, for example, student modelling and discourse management will not be immediately applicable and will need to be adapted but this is clearly work with an AI orientation.
- The effectiveness of virtual reality as a learning environment depends fundamentally on the relation between learning and social and perceptual experience, a relationship which is central to AI research. Even at the technical level, preliminary experiments have already shown the need for surrogate 'co-learners' and other intelligent agents in the environment (Shute and Psotka, 1994).
- According to Katz and Lesgold (1993), the demand for computer-supported collaborative learning environments for workplace training can best be met by adapting the coached practice environments, such as Sherlock (Lesgold, Lajoie, Bunzo and Eggan, 1992), originally developed for individual learning. If so, present AI-ED research can be seen as the basis from which the new theories required for these environments will evolve.

Whatever the future holds, recent technological advances have radically changed AI-ED systems. A few years ago, a review such as this would be profusely illustrated with screen images to show student-system interactions, usually involving natural language-like typed communication (see, for example, the illustrations in Wenger (1987)). Now, with graphic interfaces and multimedia, it is virtually impossible to capture on paper the richness and immediacy of such interactions.

2.5 Measures of effectiveness

AI-ED research has the misfortune to be concerned with a field of application of AI (i.e. education) which is riven and driven by demands for evaluation. No other field has such an ethos of evaluation built into it and therefore, compared to other areas of AI, AI-ED needs to take evaluation very seriously. We can distinguish two kinds of evaluation - those of complete systems (summative or external evaluations), which generally attempt to demonstrate some educational benefit of the system as a whole, and those of components or prototypes of systems (formative or internal evaluations), which generally aim to investigate the properties of parts of systems so that may be improved.

2.5.1 External evaluation

The evaluation of any educational innovation, including AI-ED systems, is inherently difficult (Mark and Greer, 1993; Winne, 1993). However, because of the expense of AI-ED system implementation, the demand for successful evaluations is quite reasonably made. It is only recently that AI-ED research has been able to respond to the challenge by carrying out large-scale empirical studies which show the benefits of AI-ED systems in real educational settings, for example, the evaluations of:
- The Geometry tutor, as mentioned above (Schofield, Evans-Rhodes and Huber, 1990).
- SMITHTOWN, a discovery world that teaches scientific inquiry skills in the context of microeconomics (Shute and Glaser, 1990).
- Sherlock, where twenty hours using the system were judged to be as effective as two years 'on the job' (Nichols, Pokorny, Jones, Gott and Alley, 1993).
- A STATICS tutor, where the instructional design effort per hour of instruction time was about 85 hours, compared to 100-300 hours for traditional CAI (Murray, 1993).

- The Space Shuttle Fuel Cell tutor, where NASA trainers were so convinced of its superiority over alternatives that it was adopted without need for a formal evaluation (Duncan, 1992).

2.5.2   Internal evaluation

These evaluation studies, however, use standard educational techniques with AI-ED products but do not themselves use AI techniques. More relevant to this review is the possible use of AI for evaluative purposes.

Any AI-ED system is an implementation of a (usually implicit) theory of learning and instruction. If the theory were sufficiently explicit it could be expressable in executable form and the outcomes from the AI-ED system could be predicted by running the system with 'simulated students'. VanLehn, Ohlsson and Nason (1994) consider the possible uses of simulated students to support teaching training, to enable an AI-ED system to act as a collaborative partner, and to permit formative evaluations. In the last case, for example, one can, in principle, determine which is the better of two proposed instructional designs by seeing which leads to better learning of the simulated students.

In this way, an AI-ED system may be used for formative evaluations before being used with real students. This is standard practice in other fields of computer use, but its usefulness in AI-ED may be doubted because of our lack of faith in the soundness of the theories of learning concerned. The principle, however, seems sound.

Similarly, we can imagine applying the student modelling component of AI-ED systems to assist with the thorny problem of assessment (Martin and VanLehn, 1993). Most AI-ED systems which are not entirely exploratory environments maintain some kind of student model, that is, some representation of what it is believed the student has understood. This student model may be used for many purposes within an AI-ED system, for example, to determine appropriate problems to set, to provide remediative feedback, and so on. Many techniques have been developed to build student models, some derived from well-known AI techniques such as:
- discriminative concept learning (Ohlsson and Langley, 1988), where the aim is to induce the student's problem-solving procedure from observations of his correct and incorrect results;
- resolution from computational logic (Costa, Duchénoy and Kodratoff, 1988), where the technique is used to suggest and prove hypotheses about a student's beliefs;
- neural networks (Mengel and Lively, 1991), where the network is trained to simulate a student's cognitive processes;
- fuzzy logic (Derry and Hawkes, 1993a), to provide an approximate diagnosis, recognising that a student's behaviour is not entirely consistent and induction from it is risky;
- Bayesian networks (Katz, Lesgold, Eggan and Gordin, 1994), also to provide a less precision-oriented approach to student modelling;
- model-based diagnosis (Self, 1993), to cast the student modelling problem in terms of general diagnosis in AI;
- logic meta-programming (Beller and Hoppe, 1993), to reconstruct hypothetical solution paths to check against constraints associated with correct solutions.
- belief revision (Kono, Ikeda and Mizoguchi, 1994), to keep the model consistent with observations;

The need for, and success of, these methods remains a controversial topic within AI-ED research (Lajoie and Derry, 1993; Spada, 1993). To the extent that these techniques are successful and so provide a useful evaluation of an individual student (useful in the sense that it may support individualised interactions) they may be used for assessment purposes. Educationalists must argue about the ethics of computer-based assessment and about whether what can be reliably assessed in this way is in fact what should be assessed, but again the principle seems sound: many AI-ED systems aim to build student models and to the extent that this is possible it may form a basis for assessment of the student and an evaluation of the system's effectiveness in helping that student to learn.

2.6   On-going debates

Education has been a controversial topic for two millenia at least: AI has been equally controversial for rather less long. AI in Education is bound to provoke debate. After two decades, a body of techniques has been developed which are beginning to be consistently re-applied in new systems. Some early AI-ED concepts are now routinely used in off-the-shelf computer-based learning systems: for example, a $50 typing tutor uses a student model with a bug catalogue to generate new practice lessons as needed. Some larger-scale AI-ED systems have been shown to be effective within larger organisations such as the military (as discussed above).

But still there is considerable argument about AI-ED research. Some of the arguments have been touched on earlier. We conclude by mentioning some more general questions:
- Is AI a dangerous metaphor for education? For some critics, AI is seen as supporting a rather behaviouristic approach to learning, in that it aims to adapt the learner the conform to the knowledge embedded in the AI system. As we have indicated, this is a simplistic characterisation of only a sub-class of AI-ED systems.

- Can AI-ED systems support student autonomy and open learning? The more that intelligence is put within AI-ED systems, the more the temptation may be to apply it to control and direct the student's interactions with the system. However, the range of AI-ED systems includes much more than overbearing tutoring systems.
- Will AI-ED systems ever be used in real educational settings? Of course, this depends on what is understood by a 'real educational setting'. The traditional school classroom is perhaps not a very promising setting for many systems, but the organisation of classrooms is changing rapidly. It also seems likely that more learning will occur outside the official classroom as access to computer technology improves and individuals learn at home or at work, for their own interest or career development.
- What will be the impact of new educational technologies and what role will AI play within them? At the moment, we are in a phase where the radically different nature of the new technology has side-lined AI. Eventually, however, there will be a merging of the more software-oriented field of AI-ED with the more hardware-oriented advanced learning technologies.
- Will AI-ED research continue to progress through the rather unprincipled implementation of demonstration systems, or will some theoretical basis for AI-ED system design be developed? Currently only certain components of AI-ED systems are amenable to any kind of theoretical analysis and no comprehensive 'theory of AI-ED systems' is likely in the near or medium-term future.
- Do AI-ED systems reflect a reasonable view of the nature of knowledge and learning? This brings us full circle. As we have discussed, there is no consensus within AI-ED research about these issues and we can find examples of systems which reflect many different philosophies. Apart from the perception of AI-ED research as a field oriented towards producing practically useful systems, AI-ED may also a provide a more technical contribution to such fundamental debates.

3.   Introducing computational mathetics

The previous chapter aimed to provide a non-technical overview of the AI-ED field. You will have seen that the chapter did not use a single α, ϕ, $\forall$, $\varnothing$, or any of the other formal symbols which adorn texts in other fields of AI (and AI-ED is still considered to be a field of AI - for example, "intelligent teaching systems" is one of the topic areas for the major AI conference of 1995). In this, the chapter is not being unduly lenient with the reader, for much the same can be said of all the books mentioned at the beginning of chapter 2. They all have a predominantly wordy style, in which concepts and issues are discussed in informal natural language, perhaps enlivened with the occasional illustrative screen dump, but devoid of any formal definitions, theorems, axioms, derivations, and the like. Superficially, then, AI-ED is very different from general AI, as is apparent from only a glance at the major AI journals and conference proceedings.

This would not matter at all if AI-ED were indeed a very different subject from general AI, whose methodologies were considered inappropriate for AI-ED. On the other hand, if AI-ED is to make a distinctive contribution to education, which I assume is the ultimate aim, then it must come, by definition, from the unique attributes of AI itself. AI-ED researchers must engage in broad educational, philosophical, psychological and sociological discussions but it would be rather arrogant of AI researchers to imagine that whatever expertise they have in AI enables them to resolve such long-standing debates. Whenever opinions are offered on such issues they risk being immediately dismissed as naive and unoriginal by those expert in the respective fields. For example, even the influential situated learning proposals of Collins, Brown and Newman (1989) and Brown, Collins and Duguid (1989) were welcomed with a tone of condescension - both Palinscar (1989) and Wineburg (1989) commented on the lack of references to related earlier work by Dewey, Vygotsky, Bruner and others, with Wineburg remarking that "one hopes that these ideas will interact with their antecedents and surpass them in the rigour of their formulation".

The challenge of contributing to broad educational debates is one that AI-ED must continue to face, but not necessarily in terms set by others. Rather than AI-ED attempting just to contribute in areas which already have established traditions and paradigms - and ones which, if we wished to be argumentative, we could claim have not been entirely fruitful - it could ask, or insist, that the debates be represented in *its* terms, that is, in the language of AI, and so perhaps provide the rigour which Wineburg and others profess to desire. To put it simply, the technical language of AI would be offered as a means of clarifying debates. Moreover, that language of AI, if it is found appropriate, would provide a much more direct link to the design of computer-based systems to promote learning. This, then, is the objective of the following pages but first it is worth reflecting on recent AI-ED developments.

3.1   The need for computational mathetics

Although it would be nice to imagine that the AI-ED field makes rational, scientific progress based on purely academic considerations, we must also take into account the social and cultural context in which AI-ED research is carried out, just as AI-ED itself is increasingly being urged to take into account such factors when designing AI-ED systems. In the twenty years from 1967 or so (when Carbonell and Wexler began their PhD projects (Carbonell, 1970; Wexler, 1970)), AI-ED research proceeded rather serenely, producing a body of work, surveyed in Wenger (1987), which seemed to lay out a panorama ripe for further investigation. In fact, the field then underwent a convulsion in which many of the tenets of the previous work were challenged - a convulsion led, perhaps predictably, by Wenger himself and his foreword-writers, Seely Brown and Greeno, who all, along with others, explicitly disassociated themselves from the 'knowledge communication' theme of the book and its basis in standard symbolic artificial intelligence. Suddenly "intelligent tutoring systems", which had become a broad term encompassing a range of AI-based systems (few of which were actually tutoring systems in the sense subsequently caricatured), became a term to avoid. Instead, AI-ED researchers were urged to widen their horizons beyond the panorama sketched in 1987, by taking account of previously neglected work in disciplines such as psychology, sociology, anthropology, linguistics, philosophy and biology in order to provide a theoretical rationale for the new kinds of learning environment made possible by new learning technologies such as multimedia, conferencing, and virtual reality.

Clearly, this was a timely development, for ITS research was in danger of becoming too blinkered. And yet, there were two odd features with this revolution. First, it was entirely US-led. Of course, this may not be thought odd, for most things are US-led, but pockets of AI-ED research existed in Europe, Canada, Japan and elsewhere and they were almost without exception puzzled by the missionary fervour with which the new paradigms were preached. Having come to many of the same conclusions about the status of AI-ED research themselves, they tended to prefer a more evolutionary development, rather than an explicit disavowal of all previous work. In some cultures, such as the Japanese, there was bewilderment at the idea that they were no longer supposed to talk about 'communicating knowledge'.

The second odd feature was that inspiration for the new approaches came mainly from European scientists and philosophers - Vygotsky, Leontiev, Heidigger, Marx, Piaget, Wittgenstein, Levi-Strauss, Bartlett - many of whom did

not have English as their mother tongue and most of whom were unfortunately no longer with us. This had the effect that the inevitably obscure translations of their original writings could be imbued with all kinds of profound meanings which could not be fully resolved by reading the original text (for English-speakers, anyway) nor, of course, by asking the writers to elucidate. In some cases, the ideas had to some extent been absorbed into European thinking and it was difficult for Europeans to see what all the fuss was about. For example, the new emphasis on 'socio-technical design' seemed at first to ignore the fact that schools at Manchester and in Sweden had been advocating this for forty years. Sack, Soloway and Weingrad (1994) describe how their ideas about AI-ED system design have evolved as a result of their foray "into the wilderness of continental philosophy".

Anyway, it is clear that the significant recent changes in AI-ED research, especially in the US, are partly a product of the US culture, and therefore we should consider the situation as it was in the US in the mid-1980s, when the changes began. The paper which is now considered to be the catalyst for the 'revolution' (Collins, Brown and Newman, 1989, written in 1986) commented that "Current work on developing explicit, cognitive theories of domain skills, metacognitive skills and tutoring skills is making the crucial first steps in the right direction": hardly a clarion call for a revolution! The paper's points are illustrated by referring to standard AI-ED systems of the time. Similarly, Wenger (1987) appeared to indicate that the foundations had been laid and it was time to cross the 'threshold of development'.

However, a more careful reading of Wenger (1987) indicates that he and others had misgivings about the direction of AI-ED research. His organising theme, 'knowledge communication' is first (p7) defined as "the ability to cause and/or support the acquisition of one's knowledge by someone else, via a restricted set of communication operators". However, by the end of the book he has developed a view of knowledge communication in which "both the knowledge states involved in knowledge communication are modified: knowledge communication is viewed as a dynamic interaction beween intelligent agents by which knowledge states are engaged in a process of expansion and articulation" (p431). Similarly, as one reads his individual project descriptions, which are uniformly and exaggeratedly complimentary, it is striking that many of the protagonists were in fact withdrawing from the original aims of their projects. Subsequently, many have explicitly or implicitly disavowed their earlier work and have advocated different methods or have moved out of AI-ED entirely (Brown, 1990; Clancey, 1992b, 1993; Collins, 1988; Greeno, 1989; Lave and Wenger, 1991; Sleeman et al, 1989; Soloway et al, 1992).

There may be an element of getting one's criticism in first here. By the late 1980s, intelligent tutoring systems research had developed a reputation for failing to deliver what it promised or, rather, what had been promised on its behalf (the acronym ITS was sometimes rendered as 'invisible tutoring systems'). AI-ED researchers knew better than outside critics the reasons for this perceived failure and it was perhaps politically sensible to admit the error of one's ways and to propose radically different solutions. Whatever the reason, there was a sense of despair. Sack, Soloway and Weingrad (1994) comment that
> "we made large catalogs of bugs we observed in student programs, gave them very long and complicated names, and then organized them into taxonomies. Unfortunately, these lists of bugs with formal identities never made it out of the laboratory and into the classrooms ... our old bug taxonomies are of no educational interest ... [now] we want to put powerful, real-world tools in the hands of students so that they might have the opportunity to create transitional objects which will serve to introduce them to the society-at-large."

Similarly, Clancey (1993) considers that
> "Despite the use of advanced computer technology in the 1990s, the dominant form of instructional design in schools and industry is 1960s-style page-turning presentation. No commercial authoring tool has the complexity of GUIDON. At the same time, researchers in industry are finding that expert-system techniques, hatched in university laboratories, are inadequate for developing useful programs that fit into people's lives.... After more than a decade, I felt that I could no longer continue saying that I was developing instructional programs for medicine because not a single program I worked on was in routine use."

These 'failures' may be more ones of unrealistic expectations. In O'Shea and Self (1983) we had predicted that
> "computerised tutors will play a minor role in education for many years to come. A few tutorial programs will be generally available in 1992, but there will be little incentive to develop more. The type of application where there is an incentive is in areas where failures in training result in great cost. For example, if errors in operating nuclear power plants could be reduced by training with an expert teaching system (like SOPHIE but incorporating a simulation of a power plant) then such a system might well be developed."

As mentioned in chapter 2, we do have a few off-the-shelf tutors, like SPENGELS, and most of the significant on-going AI-ED projects are in the area of expensive training projects, such as Sherlock and the Space Shuttle tutor. Our conclusion in 1983 was that ITS research should continue, despite this 'failure' to deliver products, because of its academic interest and longer-term potential.

However, in the US, at least, there was considered to be some urgency to find other ways of solving educational problems. In this context, the Institute of Research on Learning and the Institute for the Learning Sciences were

established and they naturally had to advocate methods different to the failed ITS ones. The former's aim appeared at first to continue the line of ITS research:

"The institute will work on artificial intelligence systems for traditional classroom learning, as well as for training in the workplace. The focus of our research will be on how children and adults think and learn, and on expert computer systems that can coach them the same way a personal tutor would" (from the inaugural speech of George Pake, the first director of IRL, on November 12th 1986).

The IRL would work in association with cognitive scientists, sociologists, education professionals and anthropologists - the last because "anthropologists are experts in human beings as social animals ... Most of what we learn, we learn with others". The distinctive approach of IRL was thereby established, to become stronger over the following years.

The Institute for the Learning Sciences aimed to go "beyond today's generation of simulators and 'intelligent tutors'" and develop 'discovery systems' (Schank and Edelson, 1989):

"A 'discovery system' encourages a student to become an active learner by forcing him to generate hypotheses, test them, and revise them. The system develops the student's capabilities as a case-based learner by providing him with relevant cases at appropriate moments. It enables the student to learn through failure and encourages creativity by inviting him to pursue any hypothesis without attaching a stigma to possible failure. Finally, it allows the student to learn through experience by providing him with the opportunity to explore a simulated environment, while at the same time allowing him to learn from the experience of others through exposure to relevant cases."

Changes in AI-ED do not occur in a vacuum: they are part of more general attempts to change educational practice. A paper by Cohen (1989) is sometimes quoted to support the new mission of computer technology in education. This essay traces a history of teaching practice, discusses the 'new pedagogy' and considers the difficulties facing its adoption. Here are three quotations which might help us to understand what is happening in US education, and hence in AI-ED. First,

"Consider first the view that knowledge is purely objective - that it is discovered, not constructed. This notion has deep roots in medieval Europe. Recall that educated men of that age worked from hand-copied manuscripts that had survived the collapse of a glorious Empire, or found their way into Europe from more sophisticated eastern civilizations... Scientists and philosophers in the seventeenth and eighteenth centuries worshipped a rational Nature. They believed in the objectivity and authority of sciences that would open nature's lawful heart to investigators... During most of the modern age, then, there was little argument about the objectivity of knowledge, nor about the great authority of such knowledge... Only very recently have these old and deeply rooted ideas been broadly questioned."

History is in the eye of the beholder and alternative views are possible. The above quotation concerns an alleged prevailing philosophy of knowledge which if it did prevail then it did so in Europe, America and elsewhere until science itself indicated that it was untenable through the uncertainty principle, relativity, chaos theory and Goedel's theorem. However, the philosophy of objectivism does not necessarily entail a particular pedagogy, nor vice versa. The most influential teacher for Western society was arguably Jesus Christ, whose methods clearly followed the 'new pedagogy' rather than an objectivistic philosophy. His enigmatic parables were couched in 'authentic' situations and their meaning had to be constructed by listeners, as indeed they still are today. Similarly, in Chinese society, the writings of Confucius are still revered and re-interpreted after 2500 years. It is fairly clear that before medieval times versions of the new pedagogy were dominant. From medieval times almost all those who wrote about educational philosophy (such as Erasmus, Locke, Rousseau, Comenius and Froebel) espoused views which were more constructivist than objectivist. In fact, in all the recent constructivist writings that I have read I've yet to see a direct quotation from any educational philosopher arguing that knowledge should be 'transmitted'. Instead they invariably infer and attribute a philosophy from the practice. It could well be that practice was dictated by social and other factors rather than by philosophy. After all, what else could medieval monks do but laboriously transcribe rare texts? Cohen quotes the American icon Mark Twain's story of learning to become a Mississippi riverboat pilot to indicate that there was a popular rebellion against 'formal education'. Similar stories are common in European literature. For example, there can be no more biting satire on 'knowledge transmission' than Jonathan Swift's Gulliver:

"I was at the mathematical school, where the master taught his pupils after a method scarce imaginable to us in Europe. The proposition and demonstration were fairly written on a thin wafer, with ink composed of cephalick tincture. This the student was to swallow upon a fasting stomach, and for three days following eat nothing but bread and water. As the wafer digested, the tincture mounted to his brain, bearing the proposition along with it."

Similarly, Charles Dickens' Gradgrind is made a ridiculous figure for his 'facts, facts' philosophy. In short, popular educational philosophy never did support objectivism.

The second quotation is, following on a discussion of the impediments to adopting the new pedagogy:

"But where is it written that change will occur if only the 'obstacles' are removed? It is easy to understand why such an assumption would be common among educators, in view of many reformers' insistence that adventuresome teaching is possible anywhere. The idea that change is normal is particularly easy to understand among a people that embraces the idea of progress as avidly as Americans do. But why should researchers

adopt these assumptions? Why should we accept that improvement is to be expected, or that change is the normal state of affairs? It may seem un-American, but perhaps stability is to be expected in teaching."

We should not pass over the phrase 'adventuresome teaching'. This term is not defined anywhere but it is another loaded term (like 'constructivism' itself) which can only be denied by painting oneself into a corner labelled 'unadventuresome teaching' (or 'destructivism', 'obstructivism' or, even worse, 'instructivism'). If we paraphrase it to, say, 'risky teaching' then it is not so obviously a good thing. Anyway, the main point is that American culture "embraces the idea of progress" more than others, that change is conflated with progress, and that the 'new pedagogy' is, by definition, change.

Finally:
> "Recent efforts to make teaching more adventurous thus are a modest and recent chapter in a much larger and older story. Our struggles .. are only a few episodes in a gathering collision between inherited and revolutionary ideas about the nature of knowledge, learning and teaching... Efforts to sort out the intellectual content and practical implications of both traditions have only just begun, under the pressure of conflict and challenge. This is true even in the United States: While it is the nation most deeply committed to the new pedagogy, efforts to try the new ideas out in practice here still are isolated and quite fragmentary. Other countries, like France, Germany, or Spain remain largely untouched by new instructional ideas and practices. It seems reasonable to suppose that we are working on the frontiers of this great collision."

So, the new pedagogy is in the great American tradition of frontier-crossing. While in the US change is the normal state of affairs, other cultures ignore new ideas, for some reason. If everyone's cultural context were one in which, as in Chicago, on an average day 19% of the schoolchildren are playing truant then we might all believe in the need for adventuresome teaching (it would be an adventure to find the children, at least).

Actually, constructivism is a broad philosophy with adherents in the arts, humanities, social sciences and only relatively recently in education and technology. The term (or at least the Russian equivalent of it, konstruktivisma) was apparently first used by Russian artists in the early 1920s, who, in the aftermath of the revolution and the World War, were reacting against the straightjacket of production art and Marxist doctrine. Gan (1922) wrote that "constructivism is a phenomenon of our age". A 'Manifesto of International Constructivism' was signed in September 1922. The meaning of the term has evolved as it has been used through the century by artists, architects, linguists, and others, but its core still resonates with its recent adoption by educational technologists:
> "What can be stated quite categorically about constructivism is that it rejects the comfortable assumption of a 'given' harmony between human feeling and the hostile world. In contrast, it implies that man himself is the creator of order in a world that is neither sympathetic nor hostile, and that the artist must play a central role in determining the type of order that is imposed." (Bann, 1974).

Intriguingly, the term 'situationism' also has a pedigree in the arts. There was an exhibition of situationist art held at the Centre Georges Pompidou and the Institute of Contemporary Arts in April 1989. The Situationists were an avant garde group of the 1950s and 1960s whose activities had a revival of interest in the late 1980s. The key features of their work were iconoclasm, a sense of purpose, an aura of radicality and by inference authenticity. So-called Specto-situationists claimed to have played a key role in stimulating the Paris riots of May 1968. I cannot resist giving a short extract from a transcript of a conversation between the art critics Ralph Rumney and Stewart Home, discussing the exhibition: "It was what was actually done that was important, far more than the theory. Theories are evanescent. Situationist theory was intentionally inspissated, to make it difficult to understand and extremely difficult to criticise." "And also to give an impression of complete originality!"

The Cohen essay is about education in general and says nothing directly about AI-ED. However, we can see the same re-writing of history happening over the much shorter timescale of AI-ED. I do not believe that many ITS researchers of the 1970s and early 1980s held the philosophies now attributed to them (if so, where are the quotes?). Like the medieval monks, they were doing what they could with the technologies of the time. I also do not believe that the foundations laid then will turn out to be completely irrelevant to the design of AI-ED systems to support the new pedagogy, whatever it turns out to be.

This discussion of the recent history and context of AI-ED research unfortunately violates the very principles which we aim to advocate. It engages in a vague polemical debate about very general issues which do not necessarily impact directly on the design of systems. The purpose of the above discussion is not to denigrate alternative approaches, for that would be to join an unnecessarily confrontational debate setting up false dichotomies. It is essential for AI-ED research to take better account of the concepts of 'situation', 'context', 'community', 'discourse', 'social learning', and so on, but these terms need to be defined, not used as mantra. The language of AI is offered as a way of defining and then analysing such concepts. That will be the aim of 'computational mathetics'. First, however, it will be fun to consider two analogies, risky though they always are.

3.2   An analogy with aeronautics

It has been said that the design of computer-based learning environments is a form of 'educational engineering'. If only it were so. The 'educational engineering' term is based on a derogatory characterisation of engineering as the undisciplined design of devices by tinkering with them until they appear to work. Modern engineering is different. Let us consider the case of aeronautical engineering.

After Greek mythology, Leonardo da Vinci (1452-1519) was the first person to develop detailed proposals for a flying machine, all based on flapping wings. These could never work because it is not possible in this way to attain the energy output per unit weight that birds achieve. The breakthrough - to base the design on a soaring bird not a flapping one - was made by Sir George Cayley (1773-1857), who in 1799 designed an airplane, indicating the lift and drag forces in his drawings, whirled his models through the air to carry out experiments, and published his results in the *Journal of Natural Philosophy* in 1809. Although this work explained the fundamental concepts necessary for flight, it was forgotten and the later aviation pioneers proceeded in ignorance of it.

Otto Lilienthal (1848-1896) was the first person to fly heavier-than-air craft in a more or less reliable fashion. He published a book on *Bird Flight as the Basis of Aviation* in 1889 and considered that the only way to develop a deeper understanding was to engage in actual flying experiments himself. Accordingly, he made 2000 glides, before he crashed and died in 1896. The Wright brothers systematically studied the writings of Lilienthal and others and began their own test flights of gliders in 1900. They built a wind tunnel to study wing design, discovering the crucial concepts of wing warping and adverse yaw. In 1903 they developed a generally sound theory of propeller operation. By 1905 they had made over 100 powered flights for durations up to 38 minutes, but the US War Department refused to believe that flight was possible and rejected an invitation to a demonstration. However, the test flights continued as occasions of great drama and apprehension - and danger, for Orville Wright had two serious accidents and killed an observer in 1908. Very soon, the test flight was displaced by more specialized environments, such as wind tunnels, and the elaboration of the theory of aeronautics.

Aeronautics is largely concerned with the flow of fluids over surfaces and therefore is a branch of fluid mechanics, which had been comprehensively studied in the 19th century. The first attempts to apply fluid mechanics to aeronautics produced theories which, although apparently correct, gave answers which did not agree with the results of tests. An important element, viscosity, had been omitted. Today, aeronautics is a "blend of beautiful theory and empirical fine tuning" (Shevell, 1983). The theory is expressed in the language of mathematical physics and includes a wealth of specific technical concepts, such as downwash, ground effect, and wake vortex turbulence, and its own sub-theories, such as the theory of circulation. The theory provides substantial confidence in the safety and efficiency of aircraft before any flight is attempted. If there are variations in design which need to be investigated then this may be done in specially designed test environments, not in a full-scale test flight. The successful development of aeronautics has been possible even though fluid mechanics itself is far from complete, with basic questions such as the nature and cause of turbulence still not wholly answered.

In comparison, AI-ED system design is about where airplane design was in 1905. Perhaps a thousand 'test flights' have been made and most have crashed. Empirical studies have been carried out on our 'target analogy', the human teacher, although these are likely to have missed the crucial aspects. Special test environments are beginning to be built, as we will see. The US Department of Defense is beginning to believe that AI-ED systems can solve some of their training problems. Unfortunately, there is unlikely to be a manuscript hidden in the archives which lays out the basic theoretical principles for AI-ED system design (although some people are looking). In fact, there is no ready-to-hand theory which is likely to be adaptable for our purposes, as fluid mechanics was for aeronautics. Our argument will be that AI is the best source for such a theory, although we can be sure that we will have more work to do than just identify missing components, such as viscosity. It may be that AI-ED system design is so fundamentally different from airplane design that no corresponding theory is attainable: we may see.

3.3   An analogy with computational linguistics

The ability to learn has some similarities with the ability to use language:
- They are both universal abilities in that they are developed by all normal individuals in all cultures.
- There are believed to be universal principles which hold for both abilities. The observation in 1786 by Sir William Jones, Chief Justice in Bengal, that modern languages "have sprung from some common source which, perhaps, no longer exists" was considered deeply insightful and is sometimes regarded as marking the birth of 'linguistics'. The search for linguistic universals, that is, properties of language which are necessary and innate, is very much part of modern linguistics. The assumption of 'learning universals' seems to have gone unremarked.
- Although all individuals develop both abilities they may be improved by a deliberate educational effort. In English-speaking countries 'English language' is a core of the primary and secondary curriculum, and courses such as 'study skills' are sometimes offered although more often teaching about learning is distributed opportunistically around other courses.

- Both abilities can be applied individually or within groups. It is arguable that the purpose of both abilities is to enable individuals to participate and contribute in society. At least, it is clear that both cognitive and social aspects have to be considered.
- Both abilities can be possessed (to some extent) by agents other than humans, for example, chimps and computers.

Linguistics, the scientific study of language, has been a recognised discipline for two centuries or more, with several established sub-disciplines: anthropological linguistics, applied linguistics, biological linguistics, clinical linguistics, computational linguistics, educational linguistics, ethnolinguistics, geographical linguistics, mathematical linguistics, neurolinguistics, philosophical linguistics, psycholinguistics, sociolinguistics, statistical linguistics, and theolinguistics (Crystal, 1987). Computational linguistics provides the theoretical and technical underpinning for the development of computer-based language systems, such as those for machine translation, natural language front-ends for information retrieval, and so on. The achievements of computational linguistics are a major reason why there are now many hundreds of computer-based language systems in routine use. A modern textbook on computational linguistics (such as Gazdar and Mellish, 1989) says very little about particular applications. It also says little about related sub-disciplines such as psycholinguistics and sociolinguistics. As Gazdar and Mellish (1989) say in their preface:

"The book is formally oriented and technical in character, and organized, for the most part, around formal techniques. The perspective adopted is that of computer science, not cognitive science. We have no claims to make about the way the human mind processes natural language ... This is a book about natural language processing techniques, not about their application."

The earliest books on computational linguistics make strange reading today. For example, almost half of Hays (1967) deals with elementary data structures and transient hardware, the remainder discussing miscellaneous applications, such as concordances. Gazdar and Mellish, in what is still only an 'introduction to computational linguistics', can write over 500 pages of technical material. In comparison, AI-ED system design is about where the design of natural language systems was in 1970. As remarked above, current AI-ED books are primarily non-technical descriptions of various applications.

3.4 The definition of computational mathetics

The scientific study of learning, in so far as it exists today, is distributed around psychology, education, sociology and artificial intelligence. It does not form a coherent, integrated field of study and does not contribute reliably to the design of computer-based learning systems. While not wishing to suggest that such a discipline be created, we can daydream about how the design of AI-ED systems would be different if it were.

According to the Shorter Oxford Dictionary, the word 'mathetic' is an adjective meaning "pertaining to learning", from the Greek 'manthanein', "to learn". We may coin the noun 'mathetics' to mean "the study of matters pertaining to learning", in analogy with linguistics, physics, aesthetics, and so on. Then the field of 'computational mathetics' would be "the study of matters pertaining to learning, and how it may be promoted, using the techniques, concepts and methodologies of computer science and artificial intelligence" (Self, 1992), in analogy with the definition of computational linguistics (Gazdar and Mellish, 1989). (Papert (1993) also finds the need to invent the word 'mathetics' for "a course on the art of learning". In fact, the coinage is not new. There was, I understand, a short-lived *Journal of Mathetics* in the 1960s.)

Computational mathetics would be, like computational linguistics, technically and theoretically based and oriented towards, but independent of, practical applications. In particular, it would be oriented towards the eventual design of computer-based systems to promote human learning. As a field, it would be related to but clearly distinct from educational mathetics, neuromathetics, psychomathetics, sociomathetics, and so forth. Most of what is written today about AI-ED systems could be considered to belong in those co-fields, not within computational mathetics. Those co-fields must continue to develop for they are complementary to, not in competition with, the field of computational mathetics.

Computational mathetics would be explicitly concerned with 'learning', not 'education' in its broad sense. This is a realistic view of what AI-ED systems aim to do, despite the 'education' in the title. The business of education involves much more than just that of learning, for example, the range of administrative and pastoral activities. These are outside the scope of AI-ED systems and therefore of computational mathetics. Computational mathetics would not be concerned solely with human learning and therefore would not be wholly within psychology. It would also not be concerned solely with the design of computer programs to learn (i.e. machine learning) but with the interaction between two or more agents, one or more of whom it is intended should learn. Computational mathetics would differ from other fields in scope but, more importantly, in approach.

The term 'computational mathetics' was first used as a private joke to ward off gullible colleagues who tended to mock anything to do with AI and education and especially both. The term also comes in useful to avoid those conversations

that tend to occur in public houses whenever AI or education is mentioned. At first, the term sounds impressively pretentious. With use, however, it seems perfectly natural and what it connotes entirely worthwhile and sensible.

3.5   The approach of computational mathetics

To justify a neologism such as 'computational mathetics' it seems necessary to indicate what might be added to what we already have. Let us therefore consider the current style of theorising in AI-ED research and development. Here are four illustrative sets of principles which their authors propose for AI-ED system design (the first two are repeated from chapter 2):

Anderson, Boyle, Farrell and Reiser (1989):
- Represent the student as a production system.
- Communicate the goal structure underlying the problem-solving.
- Provide instruction in the problem-solving context.
- Promote an abstract understanding of the problem-solving knowledge.
- Minimize working memory load.
- Provide immediate feedback on errors.
- Adjust the grain size of instruction with learning.
- Facilitate successive approximations to the target skill.

Clancey (1993):
- Participate with users in multidisciplinary design teams.
- Adopt a global view of the context in which a computer system will be used.
- Be committed to providing cost-effective solutions to real problems.
- Aim to facilitate conversations between people.
- Realise that transparency and ease of use is a relation between an artifact and a community of practice.
- Relate schema models and AI-ED systems to the everyday practice by which they are given meaning and modified.
- View the group as a psychological unit.

Leinhardt and Ohlsson (1990):
- Begin instruction by activating relevant previous knowledge.
- Mark the beginning of a new lesson segment.
- Tell the student the nature of the lesson segment.
- Label knowledge items.
- Mark reference-preserving shifts of expression.

Reusser (1993):
- Design and use computer-based tools pedagogically, that is, as cognitive instructional tools for mindful teachers and learners in a culture of problem-solving.
- Extend and empower the minds of intentional learners.
- Provide learners with some guidance according to the "principle of minimal help."
- Have students construct and externalize their mental models.
- Provide students with intelligible and effective representational tools of thought and of communication.
- Promote the use of comprehension-related strategies.
- Encourage reflective and self-directed learning.
- Extend the use of computer-based instructional tools into a supportive classroom culture of collaborative learning.

These principles are a different kind of theoretical entity to those encountered in aeronautics or computational linguistics books. But regardless of that, we may ask: Where do they come from? In what ways were they derived? How do they relate to one another? Are they sound? Are they useful?

The Anderson principles are supposed to be corollaries of ACT*, a psychological theory of cognition. The principles are derived from the theory by a process of discussion (in English). It is not possible to prove, in a mathematical sense, that the principles do follow from the theory, that they are all the principles that follow, that they are the most important principles, or that contrary principles cannot be derived. The Clancey and Reusser principles follow from an even more tenuous argument from what is more a philosophy than a theory about learning. The Leinhardt and Ohlsson principles were specified after empirical observations of classroom teachers.

Taken together, they provide 28 principles for AI-ED system design. Or are some of the principles the same or even contradictory? Is "Encourage reflective and self-directed learning" the same as "Provide immediate feedback on errors" or do they contradict one another - if the latter, how, precisely? A scientific field cannot advance through the

ad-hoc accumulation of unrelated principles. It seems encumbent on anyone proposing a new principle to say clearly how that principle supplements or overrides previously-held principles.

The soundness of the principles is hard to determine. For one thing, the principles are stated as recommendations not as propositions, that is, a principle such as "Encourage reflective and self-directed learning" is not of a form which can be said to be true or false. We must presumably read the principles as "If you <recommendation> then <result, for example, students will learn more>". Therefore, one way of evaluating the soundness of a principle is to implement a system according to the recommendation and see if the promised result ensues. According to Ohlsson (1991), Anderson, Conrad and Corbett (1989) "empirically validates" the Anderson principles, although Anderson, Boyle, Corbett and Lewis (1990) comment that "we do not really know what features of our tutors produced these positive outcomes nor do we know how optimal our tutors are". (As far as I am aware, no similar empirical validation has been claimed for any of the other principles.).

This kind of validation is problematic for several reasons:
- The 'result' is never explicitly stated, as we see. A scientific theory should make precise predictions: we need to know which students will learn what, how, and when.
- The 'recommendation' is not stated sufficiently clearly that a system implementation follows directly from it. Consider "Encourage reflective and self-directed learning". The early papers on cognitive apprenticeship referred to AlgebraLand as an exemplar system designed to promote reflection. However, the limited empirical studies that were carried out indicated that the desired reflection hardly ever occurred (Foss, 1987), as discussed further in chapter 6.
- It is not possible to implement a system to accord with only a single recommendation - inevitably, a whole set of other recommendations has to be adopted as well. Therefore, even if the promised result materialises it cannot be reliably attributed to a particular recommendation.
- There are ethical difficulties in carrying out experiments with students in realistic situations (not mitigated by the fortunate fact that most such experiments yield a 'no significant difference' outcome).
- It may be inefficient and costly to implement a complete AI-ED system to investigate a single principle.

Surely, nothing could be as unarguable as a successful empirical evaluation. Bhuiyan (1992) reports that users of his system (PETAL) scored 60% correct, compared to 0% for the control group (maybe we could say that they performed infinitely better). His project is exemplary in following the standard AI-ED research paradigm: carry out empirical studies of real learners; develop a theory to explain what causes good and bad learning; develop a system to promote good learning; carry out a comparative study to show that students do indeed learn better using the system, and hence validate the theory. Perhaps the results can be explained, almost regardless of the theory developed. Imagine that (1) a learning task is of the 'ah I see' variety, where performance improves in a quantum step from 0% to 100%; (2) a system with a specially designed interface enables students to 'see' the concept in time t_1, which is less than the time t_2 it takes under normal conditions; (3) the post test is carried out at a time t, between t_1 and t_2. Then the system users will score 100% and the non-users 0%. Maybe the PETAL experiment meets these conditions (in my experience, the concept of recursion, the domain of PETAL, comes close to meeting the first condition). It is important to reduce learning time if possible but it is not clear that any deep cognitive theory is validated by the experimental results. Of course, Bhuiyan did not set out to design a misleading experiment, but it is possible to see how impressive empirical results might be contrived, if that were the only criterion (which does not say much for the research methodology).

The AI-ED field is necessarily multi-disciplinary, involving aspects of computer science, education, psychology, and, to a lesser extent, other fields. It is not my purpose to imply that some disciplines are misguided in their methodologies or somehow incompetent in not providing what is needed. Nor is it the aim to argue that AI-ED should become 'computational mathetics', for want of a term. According to Ohlsson (1991),

"The educational literature contains few if any ideas about learning. The coarse level of analysis employed in most other writings on education only suffices for the formulation of principles which are so vague as to be useless."

As a result, he argues that AI-ED should simply become a part of cognitive psychology. Instead, I see AI-ED as a field where what I have called psychomathetics, sociomathetics, and so on must all continue to contribute but where the area of computational mathetics has been relatively neglected and may soon be able to play a stronger role. Established disciplines have relatively agreed-upon methodologies which differ from one another as well as from that of computational mathetics.

Computational mathetics would not just aim to make the vague theories and principles of other disciplines more precise, rigorous, formal and computationally useful. Its primary aim is to serve AI-ED system design, not to formalise general theories for other disciplines. For example, Lepper, Woolverton, Mumme and Gurtner (1993) found that human tutors tended to make indirect responses to student errors (such as "So, you think it's 126?") but it could well be that such responses would be inappropriate for AI-ED systems and therefore of little significance in computational mathetics. Computational mathetics should contribute its own techniques, concepts and methodologies to the field of AI-ED.

It would be idealistic but nonetheless desirable for computational mathetics to aim to be neutral with respect to the controversies of its associated fields such as psychomathetics and educational mathetics. In analogy with computational linguistics, computational mathetics would aim to make no claims about the way the human mind learns (or at least make it clear where it is making such claims). Maybe an appropriately defined formal language in computational mathetics could be used to express any relevant theory of psychomathetics or any other -mathetics. Computational mathetics would also take no position with respect to broad educational issues such as: What is the purpose of education? How can educational innovation be promoted? Again, the aim is not to imply that such questions are unimportant, but that they may be usefully separated from more technical, computational questions.

3.6 The language of computational mathetics

It is time to begin making choices in order to be more specific about the nature of computational mathetics. The basic notation which we will use is that of agent-oriented programming (Shoham, 1993). Rather than begin with a formal definition of its syntax and semantics, we will first give a few simple illustrative expressions in the notation in order to provide an intuitive feel for it and then refine the notation in later chapters.

The expression
```
Believes(John,Composer(Fidelio,Beethoven))
```
might be intended to indicate that John has been ascribed the belief that the composer of Fidelio was Beethoven. That is what it might indicate to us but, of course, it will hardly indicate this to a computer program which may not have a mapping between, say, the symbol `Composer` and the concept of 'composer' and even if it did it would not have such a rich appreciation of what the concept means. The general form of expression of which this is an instance is:
```
Modality(agent,proposition)
```
where the three parts need some preliminary explanation.

An *agent* is "an entity whose state is viewed as consisting of mental components such as beliefs, capabilities, choices, and commitments" (Shoham, 1993). Agenthood is in the eye or mind of an observer who views an entity. The observer finds it useful to ascribe beliefs, etc. to the entity. Such an ascription enables the observer to reason about the entity, for example, to make predictions about how the entity will behave. Ascribing a belief (etc.) to an entity is not making any claim about what that entity physically possesses, in some sense. Making such ascriptions is a common explicatory device. For example, a recent documentary on the nature of lightning commented that "the lightning does not know the ground is there at all". Clearly, there is no implication that an entity such as lightning is physically capable of possessing anything which we could reasonably call knowledge.

For our purposes, the two main classes of entity are students and programs. Just as we can ascribe beliefs to students, so we can ascribe them to programs:
```
Believes(program,Composer(Fidelio,Mozart))
```
It might seem odd to ascribe beliefs to a designed entity such as a program: we imagine that we could just inspect the design and see directly what it believes, i.e. what the designer has designed it to believe. However, programs are complex and the contents of its 'mental components' change while the program is running. We will find it useful to make ascriptions such as:
```
Believes(program,Believes(John,Composer(Fidelio,Beethoven)))
```

The term 'agent' has become widely used in computer science and AI, without a universally accepted definition. Shoham's definition above allows the term to refer to entities which are human or software, and that is the sense we have used previously, as an abstraction of the classes of intelligent entity which includes both humans and programs. According to Wooldridge and Jennings (1994a), in computer science, the term 'agent' is generally restricted to computer systems which possess the properties of:
- autonomy - being able to operate without the direct intervention of humans or others,
- social ability - being able to interact with other agents,
- reactivity - being able to perceive and respond to their environment, and
- pro-activeness - being able to exhibit goal-directed behaviour by taking the initiative.

We can accept this definition, as the properties are all ones which we would wish both programs and students to possess. In AI, the term 'agent' generally means (as Shoham's definition indicates) that an entity is conceptualised in 'human-oriented' mentalistic notions such as belief, knowledge, obligation, desire, and so on. From an AI perspective, the consideration of such notions seems necessary to provide the properties listed above, although this is debatable. As far as AI-ED is concerned, the term 'agent' is convenient as it does not pre-empt discussion of the possible roles and status of the participants (humans and programs) involved.

A *proposition* is, for the moment, simply a statement to which one may sensibly respond 'true' or 'false'. We may not know which response we should give, but we know that one or the other is appropriate. "I am not married" is a proposition, but "Will you marry me?" is not. We will use predicate logic with modal operators to express propositions (chapter 4).

The *modality* denotes the kind of 'mental component' which is ascribed to the agent with respect to the proposition. Typical modalities are `Believes`, `Knows`, `Accepts`, `Is-aware-of`, `Wants`, `Intends`, and `Is-committed-to`. For example, the expression

    Wants(program,Believes(John,Composer(Fidelio,Mozart)))

might denote that we have ascribed to the program the 'want' or goal that John believes that the composer of Fidelio was Mozart. As we can anticipate, the precise interpretation of such modalities will be difficult.

We will also need *performatives*, that is, operators which specify some kind of communication between agents or some kind of updating of what has been ascribed to agents. For example,

    Tell(program,John,Composer(Fidelio,Mozart))

might initiate some statement from the program to John, with some consequent change of beliefs ascribed to him. This will be discussed further in section 4.4 and afterwards.

The agent-oriented notation seems a possible basis for developing computational mathetics as it promises precision yet breadth. The idea of an agent as an autonomous, rational entity enables us to view both programs and students as communicating individuals, each with their own goals, with some degree of independence, and with some ability to reason for themselves and about others. Moreover, it may be possible to capitalise on the work done on multi-agent systems and related fields in artificial intelligence. Although the simple examples above illustrate a 'knowledge transmission' view of AI-ED, there is nothing in agent-oriented programming which prohibits other views, and we will seek to extend the notation to encompass those views in due course.

At this stage, we have only an informally-illustrated notation. We have no semantics to go with it. However, there is no need to expect or insist that we later define a complete formal semantics. This would be futile, for the 'real meaning' of any notation cannot be fully captured in culture-free symbolic representations, however complex. However, we can hope for sufficient agreement on our interpretations and for sufficient content in our representations that our analyses might be informative and useful.

By opting for an agent-oriented notation, we imply that the appropriate level for theorising about AI-ED systems is at what Newell (1982) called the 'knowledge level' rather than at the 'symbol level' or 'program level' or some level concerned with physical, mental structures. The knowledge level is where we ascribe knowledge, goals and actions to an agent. The knowledge level is said to lie "immediately above" the 'symbol level', where representations are specified. However, it is not possible to say anything very precise about the knowledge level without adopting some symbolic notation to say it. The only real distinction between the knowledge level and the symbol level lies in the kinds of symbols used. For our purposes, the point of operating at the knowledge level is to suggest that we are concerned with knowledge level entities which are symbol-independent, although we must use some symbols to say anything about them. Therefore, we have no particular commitment to the symbols actually used.

3.7   The aims of computational mathetics

The main aim of computational mathetics is to enable theories of learning, instruction and anything else of relevance to be expressed in a formal language so that designs for AI-ED systems to meet specified objectives can be derived analytically. This aim is unattainable now and possibly unattainable in principle. However, we can consider how aspects of the main aim might be achieved.

In any endeavour, the role of 'formality' passes through up to six stages:
1. To begin with, practitioners deny that it is necessary, possible or appropriate to try to express their activities in any kind of artificial, precisely-defined language. The content of their activities is just too complex to be adequately described at all without the full subtlety of natural language.
2. Eventually, perhaps, some recurring patterns in their activities will be recognised and in the interests of brevity some symbol will begin to be used to denote that pattern. In this second stage, there will be much confusion and argument over which patterns to symbolise and what the symbols 'mean'.
3. In due course, some consensus may emerge and all practitioners will become obliged to use the agreed-upon symbols. So, for example, in chemistry HCl denotes hydrochloric acid and in music k denotes a crochet. During this stage, the symbols are just abbreviations for concepts which could be written out in full, and in fact to begin with any use of the symbols will be accompanied by a translation into the full version, for example, to explain that

        Believes(John,Composer(Fidelio,Beethoven))

    means that "John has been ascribed the belief that the composer of Fidelio was Beethoven".
4. In the fourth stage, the notation itself becomes a vehicle to work with, regardless of the translated 'meaning' of the expressions in the notation. So, for example, a chemist will begin to carry out symbolic manipulations directly on chemical equations

        $2HCl + Na_2Co_3 = 2NaCl + H_2O + CO_2$
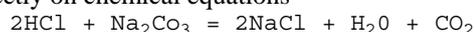
    without pausing to consider the real meaning of the symbolic operations. For the result of a series of operations to be meaningful, each individual operation, in general, must be meaningful. When the abstract symbols are

particularly powerful (for example, the ring notation for certain aromatic compounds) they seem to inspire the appropriate operations. It is only in the fourth stage that any real benefit begins to come from formalisation.

5. In the fifth stage, it is realised that the operations themselves can be formalised, that is, we can agree on a notation for the operations. In this case, we can, if we wish, represent the operations within a programming language and have a computer program carry out the manipulations.

6. As the operation of a computer program may be rather opaque and hence its output considered untrustworthy, we might identify a sixth stage (although it is not fundamentally different from the fifth) in which the content of the operations and the operations themselves are so precisely defined that they could in principle and maybe in practice be written down so that we could formally derive outcomes and engage in meta-theoretic activities such as proving that certain outcomes can or cannot be derived, or how efficiently outcomes can be derived.

Not all endeavours can or should pass through all stages. For example, most parts of chemistry have not passed beyond the fourth stage - that is, (as far as I know) in most cases the operations have not been defined so that programs or ourselves can carry out rigorous derivations of results.

Where among the stages does AI-ED lie? For some AI-ED researchers, it is resolutely at the first stage. The few who have attempted any kind of formalisation have not been able to progress beyond the second stage, because there is no agreement on whether, what and how to formalise. The terminology, even without precise definitions, is barely established. There are some efforts, as we will describe, which might be considered to be at the fifth stage, as they involve the use of computer simulations to make predictions, but they by-pass the stages of saying precisely how the simulations work.

At the moment, AI-ED systems are created by repeated iterations through a loop in which informal theories lead to experimental systems which are empirically evaluated (top of Figure 3.1). With the present state of AI-ED theory, the need for empirical evaluations seems inescapable. The potential problems of evaluation-based research were discussed in section 3.5.

AI-ED research should aim to eliminate (or at least greatly reduce) the need for empirical evaluations, not embrace them within the design process. With aircraft, the maiden flight is a demonstration, not an evaluation. All the essential properties have been theoretically determined, reducing the role of empiricism to fine-tuning the theory. Moreover, the empirical tests are carried out not in the real world but in environments specially designed to test aspects of the theory. Of course, we are a long way from an AI-ED theory from which designs may be formally derived (bottom of Figure 3.1).. However, there is an intermediate strategy (middle of Figure 3.1) which may enable us to move in this direction. Instead of deriving the design of a complete AI-ED system by some mathematical analysis, we may able to carry out such an analysis on components of a system and/or derive theoretical outcomes by computer simulation, rather than by analysis.
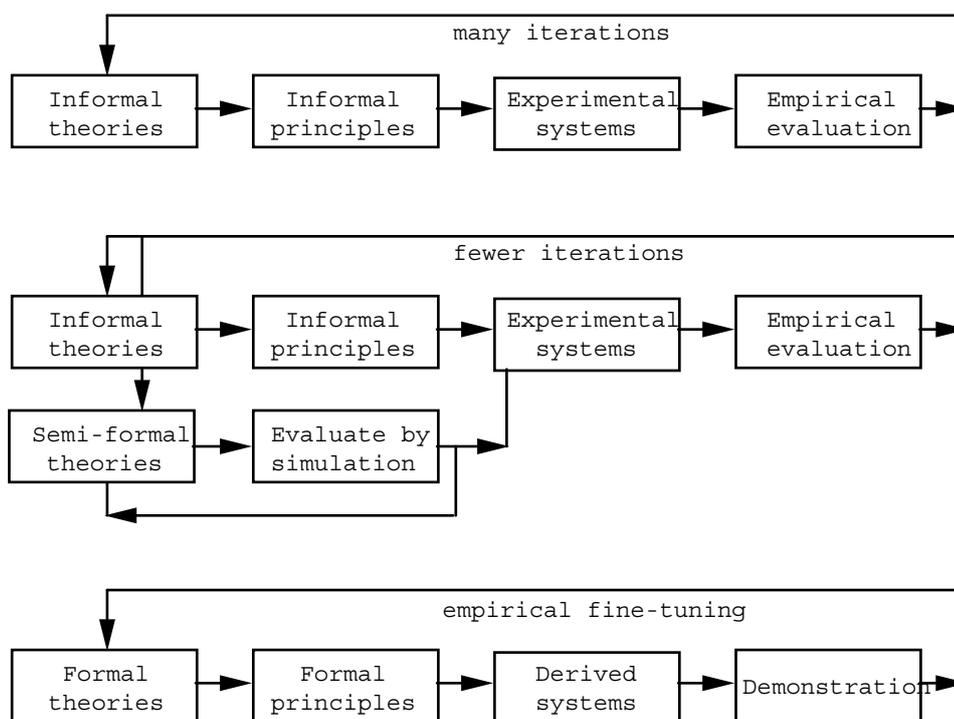

Figure 3.1. The progression from informal empiricism to formal demonstration

It should hardly be necessary to point out the potential benefits of formalisation to any scientific endeavour (which I assume AI-ED system design to be) but because it has been neglected heretofore a brief list may be worthwhile:

- The use of a formal language may help clarify otherwise vague principles.
- The use of an agreed formal language can make it easier for researchers to understand and build upon the work of others.
- An analytical study of a component of AI-ED systems can lead to precise statements about the power and shortcomings of that component and enable comparative studies of various proposed implementations of that component. Thus, formal tools may help us manage the complexity of AI-ED systems.
- An analysis of proposed systems may eliminate or reduce the need for costly experimental implementations.
- Knowing the theoretical properties of proposed systems, such as their likely run-time inefficiencies, makes it easier to determine where practical compromises should be made.
- An appropriate formal language can serve as a high-level specification language and can be directly implemented in logic-based programming languages, enabling the efficient study of prototypes.
- It is possible that AI-ED may be able to capitalise upon the large body of formal work already carried out in AI.

Genesereth and Nilsson (1987) comment that "successful preparation in mature fields of science and engineering always includes a solid grounding in the mathematical apparatus of that field". Two sceptical responses to this are possible. First, AI-ED (and perhaps AI in general) is not a 'mature field' . Therefore, there is no 'mathematical apparatus' and it would be premature to try to establish one, or perhaps even misguided for in a new field what can be formalised first are inevitably the simplest, possibly least important, and maybe irrelevant aspects of the field. Secondly, AI-ED (and perhaps AI in general) is not a field of science or engineering at all. Within AI there has always been a vigorous debate between the 'neats' (the 'formalists') and the 'scruffies' (the 'experimentalists'). As always seems to be the case, these are not mutually exclusive options: the important question is how we combine the virtues of both approaches. AI-ED, because of its dependence on contributions from education, psychology and other disciplines, might be claimed to be more of a social science than a natural science or engineering. Some social scientists will argue that the ontological differences between humanity and the rest of nature (for example, that human beings are 'free agents' who create their own world) are such that the social world is not amenable to the kind of allegedly objective and reductive analysis characteristic of natural science explanations. As far as AI-ED is concerned, we can say that many perspectives are necessary and welcome: my own view is that it is time that the perspective of computational mathetics was applied. Debates about the scope of computational mathetics will not, by definition, be resolved by general discussion but by a determined effort to widen its boundaries.

Before embarking on this, some expectations may need to be lowered. The version of computational mathetics which follows does not achieve anything near the aims laid out above. In fact, in most cases it is not possible to do much more than indicate those areas of formal AI which promise to be useful for computational mathetics. Many of these areas are research fields in their own right and it would take a large-scale, coordinated programme to investigate in depth their applicability to AI-ED. But we must be realistic - computational linguistics has made progress despite the fact that its theoretical concepts are not complete, correct, or cognitively valid, and so may computational mathetics.

4.   Representing knowledge

As the main aim of designers of AI-ED systems is that students come to know something (not necessarily something which the designers have completely specified beforehand) and the core activity of AI has been knowledge representation, we will begin the development of computational mathetics by looking at knowledge representation techniques from an AI-ED perspective. Any standard AI textbook (such as Rich and Knight, 1991; Winston, 1992; Ginsburg, 1993) provides a more general and complete coverage of knowledge representation. In this chapter the aim is to consider the foundations of knowledge representation in so far as they specifically affect AI-ED. We will consider whether the established techniques of AI can be extended to encompass different views of the nature of knowledge. In this chapter we are mainly concerned with developing notations which can be used to represent knowledge, belief, and so on in computational mathetics; the following chapters consider how knowledge can be used and acquired.

4.1   Behaviour, belief and knowledge

Imagine that you are driving in the dark and a car approaching you flashes its lights. In order to react appropriately to this communicative act, you ascribe beliefs and goals to the oncoming driver: he has recognised you and is saying 'hi'; or he is warning you of some hazard (an accident ahead, or a police speed trap, or that your own lights are off); or he is, on the contrary, encouraging you to go ahead (for example, to turn ahead of him) where normally you would not; or he actually intended to signal to turn but hit the wrong switch; and so on. Depending on the situation, some of these ascriptions are more plausible than others. At all events, you may (or may not) adapt your own behaviour. And all this happens literally in a flash - you do not pull over to contemplate at length the semiotics of car light flashing.

This is as simple a communicative act as there can be but it illustrates the essential characteristics of AI-ED system communications. Both (or all) participants are or should be continually ascribing mental components to one another and adapting their own behaviour as a result. Usually these ascriptions are not entirely reliable because of the shortage of information and time to consider it. Nonetheless, even though unreliable, the ascriptions can make a crucial difference to subsequent behaviour.

We need now to consider the relation between behaviour, belief and knowledge. This is philosophical quicksand but the designers of AI-ED systems have no choice but to build upon it.

Imagine now that every night at 10 o'clock I pick up my walking stick and my dog bounds up wagging its tail. We might well say that the dog believes we are going for a walk. Does the dog believe that we always go for a walk at 10 o'clock? If a dog does not understand the concept of '10 o'clock' can it be said to believe something about it? Does it really understand the concept of 'going for a walk' either? My dog not being keen on philosophical discussion, we cannot resolve this. So imagine instead that I tell Mary that "praseodymium is ductile" and she says 'yes'. Can we ascribe:

```
Believes(Mary,Ductile(praseodymium))
```
Maybe not, for she may not understand the meaning of 'ductile' or 'praseodymium '. Perhaps all we can say is that:
```
Believes(Mary,"praseodymium is ductile" is true)
```
assuming that she trusts me to say what is true, or perhaps that she believes that I say what I believe to be true (although I may be mistaken):
```
Believes(Mary,Believes(me,meaning("praseodymium is ductile")))
```
With Mary, we could try to unravel this, but the point is that it can be tricky in even the simplest cases to write appropriate ascriptions of the form `Believes(a,p)`. In general, the `p` should be in the language of the person ascribed to, `a`, not that of the ascriber.

The notation `Believes(a,p)` means that the belief `p` has been ascribed to the agent `a` by some observer, not that `a` believes `p` (although we will often use the latter phrase as a convenient abbreviation). This ascription is the result of a belief of the observer, and if the observer is a second agent `b` communicating with `a` then we might need to specify: `Believes(b,Believes(a,p))`. This too is an ascription of some observer. If this observer is the agent `a` then we could specify this too: `Believes(a,Believes(b,Believes(a,p)))`. This nesting of beliefs stops when we postulate that an ascription is made by some agent who is not a participant in the interaction and to whom beliefs do not need to be ascribed.

As we have taken 'belief' as the basic epistemic concept we cannot define it in terms of other concepts. All we can say, rather circuitously, is that if an agent has a belief `p` then it is disposed to behave as if `p` were true and that if an agent behaves in a certain way an observer is disposed to ascribe the belief `p` to it. The use of the word 'disposed' emphasises that there is no guaranteed mapping between belief and behaviour.

There are three senses of the word 'belief' in English which we do not mean to inherit. One is its use to imply that the ascriber of a belief `p` to an agent `a` actually believes that `p` is false. For example, "John believes the moon is a star"

might be taken to imply that the speaker believes that the moon is not a star. Secondly, 'belief' is sometimes used in contexts precisely because the belief is one for which the agent has or can have no evidence, for example, "John believes he is the reincarnation of Mohammed". Thirdly, in English, we can say someone believes in abstract concepts ("John believes in Marxism") rather than a specific proposition. It is matter of considerable debate whether the former can be reduced to a set of the latter. By `Believes(a,p)` we just mean that, in our context, it helps in understanding and predicting behaviour to ascribe the belief `p` (a proposition) to the agent `a`.

The conventional definition of 'knowledge' is that it is 'justified, true belief'. This is disputed philosophically but it will serve our purposes. Obviously, we need to consider what counts as 'true' and 'justified'. Conventionally, again, a proposition is 'true' if it 'corresponds to the facts', but again we can worry about what is meant by 'the facts' and 'corresponds'. It seems clear that we would not want to specify:

         `Knows(John,Composer(Fidelio,Beethoven))`
if we, as observer, knew as an incontrovertible fact that Beethoven did not compose Fidelio. The ascription:

         `Knows(Mary,Ductile(praseodymium))`
seems less clearcut, as we need a precise definition of the predicate `Ductile`. For most interesting propositions, such as `Honest(Churchill)`, the 'facts' are difficult to determine.

In the case of AI-ED systems, one of the agents concerned (the program) has two kinds of belief: those which it is has been explicitly given by its designer and those which it has determined 'for itself' while running, for example, beliefs about the student using it. As the system has no independent means of determining whether the former set of beliefs corresponds to the facts, we often speak loosely of the system 'knowing' those propositions. The system can have no justification for holding those beliefs, other than the fact that the designer has specified them. However, for the latter set of beliefs, and all the student's beliefs, we can require that an acceptable justification be given before we agree that a true belief is in fact knowledge. For example, a student may say that she believes that all metals are ductile and all her behaviour might accord with this belief but unless she can give some convincing justification for believing it (for example, that she has noticed that many metals are ductile, or she has a theory about the way the electron structure of metals causes them to be ductile) we might be reluctant to say that she *knows* that all metals are ductile.

This is a complicated issue because it means that for a proposition to make the transition from belief to knowledge an agent needs to keep or create convincing justifications for that belief. Human agents tend not to retain the original justifications (if any) for their beliefs - they tend to (re)construct them when needed. Maybe it matters only that the constructed justifications be convincing, whatever that may mean. At least, it seems that the transition to knowledge requires some degree of articulateness about that knowledge.

It might seem that if a student behaved exactly as if she held a particular true belief `p` then, to all intents and purposes, she may be said to know `p`. However, when some students were asked to use a simple simulation to predict the resultant speeds of two colliding inelastic balls of variable speed and mass, those students who were able to make predictions entirely in accord with the principle of conservation of momentum were unable to verbalise anything corresponding to that principle. In such a circumstance, it seems unreasonable to say that the students know that principle.

In the case of computer agents which have been designed to 'know' `p` with the intention that a student user should also come to know `p` there is clearly a problem, for if the computer agent cannot articulate any justification for `p` then the student may be unable to construct one. Even if a justification can be articulated then it must be expressed, as discussed above, in terms meaningful to the student. For example, in general, it is not sufficient that a program be able to apply the principle of conservation of momentum - it needs to be able to give some justification or explanation of it in terms of concepts believed or understood by the student. This is the basis for the black box - white box distinction which separates performance-oriented AI systems from AI-ED systems (Brown, Burton and de Kleer, 1982).

Newell (1982) defined knowledge to be "whatever can be ascribed to an agent, such that its behaviour can be computed according to the principle of rationality" where the principle of rationality says (circularly) that an agent will carry out a certain action if it has knowledge that one of its goals can be achieved by that action. This functional definition is less suitable for human agents than it is for computer agents, where the concepts of 'rationality' and 'computed' can be given precise operational definitions. For students, unfortunately, the relation between knowledge and behaviour is not so straightforward, not least because the notion of rationality is complicated (as discussed in the next chapter).

In the AI and AI-ED community discussions about the nature of knowledge are often merged with those about the behaviour resulting from the knowledge and the processes by which the knowledge is acquired - which we discuss in separate chapters because for students, at least, there is no necessary connection between the three. At this stage, we have not violated any principles of objectivism, constructivism, situationism, or any other -ism. For example, we can agree that "knowledge is not a thing or set of descriptions .. [but] a capacity to coordinate and sequence behaviour" (Clancey, 1995). Our `p`s and `Believes(a,p)`s are clearly not knowledge. We do not even need to claim that they are representations of knowledge. All we need to say is that they are representations which agents might find useful in

deciding how to behave. The claim that the action made possible by knowledge is situated in the sense that it is constrained by an agent's understanding of its place in a social process challenges the claimants to try to specify precisely how this understanding constrains action.

The constructivist view that knowledge "cannot simply be received by students but must be constructed anew by them" (Self, 1990), i.e. dynamically as they behave, need not be interpreted as a radically different philosophy of knowledge. It is a view of knowledge acquisition rather than of knowledge itself, unless it is interpreted as a view that no knowledge exists other than at the instant it is (re)constructed. There is no way of showing one way or the other whether knowledge exists in some sense before some event causes it to be constructed, because any 'knowledge-detecting activity' is an event which may cause the very knowledge one is trying to detect to be constructed. However, there seem to be some knowledge-acquiring events which strain the idea of 'construction' as a complex, active process in which an agent assembles and synthesises something significant on the basis of the social context and, maybe, prior knowledge. If I look out the window and see that it is raining then for all except pedantic purposes I know that it is raining, without much constructive processing. If I ask someone looking out of the window then they may 'transmit' to me the knowledge that it is raining. If a student asks me the atomic number of carbon then, in some circumstances, I may reasonably be said to transmit to her the knowledge that it is 6 without any significant knowledge construction on her part. This is not to say that all, or even much, knowledge can be transmitted in this way.

The constructivist view that there is no knowledge of the world independent of what is constructed by us is a separate philosophical claim. Again, it is one which can never be disproved as it is always us doing the constructing. The idea that all knowledge is hypothetical and incomplete, although perhaps literally true, does not seem very helpful. I can assert that I know all there is (which isn't much) to know about when buses are supposed to leave my village Brookhouse for Lancaster and I can base my actions on this knowledge. I will miss the bus if I worry overmuch about whether this knowledge is correct or complete. Similarly, in most contexts where the knowledge will be used, it seems possible to say that one has correct and complete knowledge of, say, the atomic numbers of the chemical elements or of the present participle of the French word 'devoir'. Perhaps the constructivist view is not that knowledge itself is (always) hypothetical but that the processes by which it is acquired are hypothetical, i.e. depend on the development of hypotheses, but that is a different claim.

4.2   Propositions and logic

The object of a belief or other mental construct is a *proposition* which is represented in some language-independent, mind-independent, abstract, symbolic notation, or *logic*. The term 'logic' is not meant to suggest that what is represented is infallibly correct but that the notation and the operations which can be carried out on it are precisely defined. Most of our notations will be variations and extensions of *predicate logic*, the lingua franca of AI, which we therefore give a brief summary of.

A *sentence* in predicate logic, such as `Composer(Fidelio,Mozart)`, is composed of different kinds of symbol: constants, variables, operators, and punctuation symbols such as commas and parentheses.

A *constant* may be of three kinds:
- an *object constant*, such as `Fidelio` and `Mozart`, which is used to name a specific element in the universe of discourse;
- a *relation constant*, such as `Composer`, which is used to name a predicate on the universe of discourse, i.e. a relation which maps a specified number of elements onto true or false;
- a *function constant*, which is used to name a function on members of the universe of discourse, such as `father` in the sentence `Carpenter(father(Jesus))`.

A *variable* is used to express properties of objects without naming the objects. There are two kinds of variable, those which are *universally quantified* and those which are *existentially quantified*. The former is illustrated by the sentence $\forall x\ Red(x)$, which might be intended to mean that every object in the universe of discourse is red. The latter is illustrated by the sentence $\exists x\ Red(x)$, which might be intended to mean that there is at least one object in the universe of discourse which is red.

An *operator* allows us to form more complex sentences from simpler ones. For example, in
        `Composer(Fidelio,Mozart) & Austrian(Mozart)`
which might be intended to mean that the Composer of Fidelio was Mozart and that Mozart was Austrian, `&` is an operator. The standard operators of predicate logic, that is, `&` (and), ∨ (or), ∅ (implies), × (is equivalent to), and ~ (not), are defined by specifying how the truth value of the complex sentence depends on the truth values of its components. If we use φ and ψ to denote two sentences in predicate logic then
        φ `&` ψ is true if and only if φ and ψ are both true;
        φ ∨ ψ is true if and only if at least one of φ and ψ is true;
        φ ∅ ψ is true unless φ is true and ψ is false;

φ × ψ is true if and only if φ ⊘ ψ and ψ ⊘ φ, i.e. φ and ψ are both true or both false;

~φ is true if and only if φ is false.

We can use operators in sentences involving quantifiers, for example,

        ∀x Toadstool(x)⊘ Poisonous(x)

which might be intended to mean that 'if x is a toadstool then x is poisonous' or 'all toadstools are poisonous'.

A mathematical or formal logic text would now give a precise syntax defining how all these symbols may be combined to form acceptable sentences in predicate logic. We will rely on intuition in this respect, and only comment on a few points where intuition may fail us.

What we consider to be the objects, relations and functions in our universe of discourse is up to us and not part of the syntax of the logic. An object constant does not have to be a simple, tangible, individual thing. For example, in `Atomic-number(carbon,6)` the constant `carbon` does not refer to an individual object but to a concept or class of objects. As we write sentences in predicate logic using even more unusual kinds of object and (in due course) make inferences from them, we need to be careful that undesired or even contradictory conclusions are not reached.

We should note that variables in predicate logic stand for anonymous objects but not anonymous relations or functions. So, for example, we cannot write ∀x x(Fred), which might be intended to mean that all the 1-argument relations in the universe of discourse apply to Fred. Also, relations can only be applied to *terms*, i.e. objects, variables, or functions of terms. So, for example, one cannot apply a relation to another relation, as in

        Virtuous(Honest)

which might be intended to mean that the property of being honest is virtuous, or to a sentence, as in

        Simple(Composer(Fidelio,Mozart))

which might be intended to mean that the sentence `Composer(Fidelio,Mozart)` is simple. Some of these restrictions will not hold in some of the 'nonstandard logics' introduced later.

Some of the language sketched above is redundant. For example, as ∃x Red(x) is equivalent to ~∀x ~Red(x), we can make do with only one quantifier (and unless otherwise indicated our variables will be universally quantified). Similarly,

        ∀x Toadstool(x)⊘ Poisonous(x)

is equivalent to

        ∀x ~Toadstool(x) ∨ Poisonous(x)

Therefore it is usual to convert sentences into a standard form, such as *conjunctive normal form*, i.e. as a conjunction of disjunctions of (negations of) expressions of the form `P(t1,t2, ..)` before any computational operation is carried out. There is a standard algorithm for carrying out this conversion (Genesereth and Nilsson, 1987).

In order to determine the truth or otherwise of a sentence it is necessary to give an *interpretation* to the constants in the sentence, for example, to interpret `Mozart` as a reference to a particular person, Wolfgang Amadeus Mozart, who lived from 1756 to 1791. With this and the other obvious interpretations, `Composer(Fidelio,Mozart)` is false. In our examples so far, the intended interpretations have been obvious, but this will not always be the case. If a sentence is true regardless of the interpretation of the constants in it, then it is called a *theorem*. For example,

        Composer(Fidelio,Mozart) ∨ ~Composer(Fidelio,Mozart)

is a theorem. The sentence `Composer(Fidelio,Beethoven)` is true, with the obvious interpretation, but is not a theorem. Most of the putative theorems which we shall be interested in will be of the form:

        φ1 & φ2 & φ3 & ... ⊘ ψ

where φ1, φ2, φ3, ... are premises (i.e. sentences assumed to be true) and ψ is a possible conclusion. In the next chapter, we will consider ways in which such sentences may be proved to be theorems (or not).

Because the premises will, in general, contain universally quantified variables, theorem-proving will involve a process of *substitution* which should be distinguished from that of interpretation. A substitution involves the replacement of a variable by a term. A substitution and interpretation which makes a sentence true is said to *satisfy* that sentence. A sentence is *satisfiable* if there is some substitution and interpretation that satisfies it; otherwise, it is *unsatisfiable*. If a sentence is satisfied by all substitutions and interpretations then it is said to be *valid* or a *theorem*. If a variable is universally quantified then we should be able to replace the variable by any term whatsoever and the sentence, if it is a theorem, will be true. Conversely, if we can find one substitution for which the sentence is false, then it is not a theorem. This would seem to imply the need for a systematic search of all possible substitutions, which promises problems, as there are, in general, an infinite number of terms (from the recursive definition allowing terms to be functions of terms). We will return to this in the next chapter.

The above has outlined the syntax of predicate logic. It is possible to give a formal definition of the semantics of predicate logic but we will use an informal *sentential semantics*, whereby what is true is just what is defined to be true as a premise or can be shown to be true by applying rules of inference (to be defined later).

At this point, it needs to be repeated that this brief review of predicate logic is not meant to imply that the goal of computational mathetics is simply to adopt the logicist ambition that all knowledge in AI be represented in a completely use-independent way in predicate logic. McDermott (1987), Birnbaum (1991b) and others have criticised logicism in AI. Computational mathetics should welcome whatever notations and languages turn out to be useful, but as predicate logic has been the starting point for almost all formal work in AI we can anticipate that it will be a basis for computational mathetics. As we will see, we will adopt notations which transcend standard predicate logic whenever convenient.

4.3   Modal representations

Predicate logic may be suitable for a single agent to reason *with* its knowledge (as in AI generally) but in AI-ED we need an agent to be able to reason *about* its and other agents' knowledge and other ascriptions.

The `Believes(a,φ)` notation does not conform to predicate logic syntax because we have a sentence φ within the scope of what looks like a relation constant. `Believes` is in fact not a relation constant but a *modal operator*, i..e. an operator that takes sentential arguments. In English, the following constructions all seem to require the use of modal operators: "It is possible that ...", "It is necessary that ...", "I am sure that ...", "I doubt that ...", "I know that ...", "I accept that ...", "I desire that ...", "I hope that ...", "I fear that ...", "I am happy that ...", and so on. The syntax of a *modal logic*, with a modal operator `M`, can be defined straightforwardly by saying that a sentence is:
- a predicate logic sentence.
- a sentence constructed from modal logic sentences using the logical operators &, ∨, ∅, ~, and so on.
- ∀x φ or ∃x φ where φ is a sentence.
- `M(φ)`, possibly with non-sentential arguments as well, where φ is a sentence. Normally, we would indicate the agent as the first argument, as in `M(a,φ)`.

Henceforth, for brevity, we will use `B` for `Believes`, `K` for `Knows`, `c` for the computer system or program, `s` for an arbitrary student, and `a` for any agent (human or computer).

This simple syntax does, however, mask a number of difficult technical and philosophical issues which indicate that modal logics must have very different properties to those of ordinary predicate logic. As computational mathetics is concerned with the development of belief and knowledge it can hardly ignore such issues, although in the interests of practicality it may well be necessary to make compromises in defined ways. Among the issues are the following:
- Modal logics of belief and knowledge are *referentially opaque*, that is, two equivalent terms cannot be interchanged. In predicate logic, if it is known that two terms are equal then either may be replaced by the other in any context. For example, if we have the premises:
  ```
  aspirin = acetylsalicyclic acid
  Medicine(aspirin)
  ```
  then it follows that
  ```
  Medicine(acetylsalicyclic acid).
  ```
  But in a modal logic, from
  ```
  aspirin = acetylsalicyclic acid
  B(s,Medicine(aspirin))
  ```
  it does not follow that
  ```
  B(s,Medicine(acetylsalicyclic acid)).
  ```
- In predicate logic the truth value of a complex sentence depends only on the truth values of its component sentences. Therefore, we can replace any component sentence by one with the same truth value. Clearly, one cannot do this in a modal logic. For example, if we have two theorems `t1` and `t2` and `B(a,t1)` then we cannot infer `B(a,t2)`.
- We need to be careful with the use of quantifiers in modal logics. The syntax allows us to write both
  ```
  K(s,∃x Composer(Fidelio,x))
  ```
  and `∃x K(s,Composer(Fidelio,x))`.
  Do these two sentences capture useful differences? In English, the first means perhaps that the student `s` knows that someone composed Fidelio (but not necessarily who) and the second that there is someone (but we don't who it is) and the student knows of that person that he or she wrote Fidelio. Philosophers have discussed this as the 'de dicto' and 'de re' distinction. If this distinction is important to us we cannot move the quantifiers inside and outside the scope of modal operators at will.
- Similarly, modal operators do not necessarily commute with the logical operators. For example,
  ```
  B(s,~Composer(Fidelio,Beethoven))
  ```
  is clearly not the same as
  ```
  ~B(s,Composer(Fidelio,Beethoven)).
  ```
  It is not so clear whether or not
  ```
  B(s,Composer(Fidelio,Beethoven) ∨ Composer(Fidelio,Mozart))
  ```
  is equivalent to
  ```
  B(s,Composer(Fidelio,Beethoven)) ∨ B(s,Composer(Fidelio,Mozart))
  ```

- Modal logics are not necessarily *consequentially closed*. That is, if φ1, φ2, φ3, ... ∅ ψ then it is not necessarily the case that `M(ψ)` follows from `M(φ1)`, `M(φ2)`, `M(φ3)`, ... For example, a student may not believe everything that follows from what she believes. A modal logic which is consequentially closed is said to be *logically omniscient*.
- In predicate logic a set of premises has to be consistent otherwise anything follows. In a modal logic of belief we can allow a set of beliefs to be inconsistent - as the logic may not be closed, this is not necessarily a problem. However, we might expect a set of beliefs to have some degree of coherence, for example, an agent maybe cannot believe a proposition and that it doesn't believe it:

  ```
  ~B(a,(p & ~B(a,p)))
  ```
- The syntax allows sentences to be nested to any depth and an obvious extension would allow different modal operators and agents to be used, so allowing sentences such as

  ```
  B(c,Accepts(s,K(c,B(s,φ))))
  ```

  i.e. the computer system `c` is ascribed the belief that the student accepts that the computer knows she believes φ. We can be sceptical that we will be able to specify what kinds of inference we would like to make from such sentences and that we will need such sentences at all. But for any child who understands the Hansel and Gretel story, at the point when the trail is to be laid, we seem to need to make ascriptions equivalent to:

  ```
  B(child,Intends(mother,abandon))
  B(child,Intends(Hansel,trail))
  B(child,B(Hansel,Intends(mother,abandon)))
  B(child,B(mother,B(Hansel,Intends(mother,walk))))
  B(child,B(Hansel,B(mother,B(Hansel,Intends(mother,walk)))))
  ```
- We can try to use modal representations to clarify what is meant by some difficult constructions in English. For example, "John knows whether Mozart composed Fidelio" is perhaps

  ```
  K(John,Composer(Fidelio,Mozart)) v K(John,~Composer(Fidelio,Mozart))
  ```

  Maybe "John knows who composed Fidelio" is

  ```
  ∃x K(John,x=composer(Fidelio))
  ```

  We will consider the sentences "John knows about Fidelio" and "John knows how to compose Fidelio" later.

A satisfactory treatment of modal logics requires the use of advanced theoretical apparatus. Sufficient progress has been made (Halpern and Moses, 1992) that it is possible to imagine their use as the theoretical basis for AI-ED system design, as AI-ED systems are essentially concerned with the development of a student's beliefs and knowledge. Practically, efficient reasoning in modal logics remains a problem because there are no standard theorem-provers as there are for predicate logic (one approach, in fact, is to convert modal logic sentences into ordinary predicate logic). These kinds of theoretical and practical limitations help to clarify what can be realistically attempted with AI-ED systems.

The semantics of modal logics is usually defined in terms of *possible worlds* (Hintikka, 1962). The intuitive idea is that besides the true state of affairs there are a number of other possible states of affairs, or possible worlds. The worlds are connected by an accessibility relation `R` which may be defined to satisfy various constraints. For example, it may be transitive, i.e. `uRv` and `vRw` implies `uRw`, where `u`, `v` and `w` are worlds. In each world, a proposition is given a truth value. An agent in a world `w` is said to believe a proposition if it is true in all worlds accessible to `w`.

The modal logic based on a transitive accessibility relation (called 'weak S4') can be given a sound and complete proof theory comprising the following rules of inference and axioms:

R_1	necessity	`p => Mp`
R_2	modus ponens	`p & p∅q => q`
A_1	tautologies	`p`, where `p` is a propositional tautology
A_2	distribution	`Mp & M(p∅q) ∅ Mq`
A_3	positive introspection	`Mp ∅ MMp`

Other logics, perhaps less suitable for representing belief, have different accessibility relations and use one or more of the following axioms:

A_4	knowledge	`Mp ∅ p`
A_5	negative introspection	`~Mp ∅ M(~Mp)`
A_6	consistency	`Mp ∅ ~M~p`

We may note in passing that modal logics of knowledge and belief (with and without axiom A_4) omit any consideration of the need for justifications (as discussed above). The completeness and complexity of variations of these modal logics is discussed by Halpern and Moses (1992). As the semantics associated with different accessibility relations can be equivalently represented by a set of inference rules and axioms, we will use a sentential semantics (as above) rather than possible worlds (Konolige, 1988). We assume that an agent has a base set of beliefs and a (possibly incomplete) set of inference rules. An expression `M(a,φ)` is true if and only if φ is a member of the base set or can be derived from that set using the agent's inference rules.

Further consideration of this reasoning process will be deferred to the next chapter, except that we should comment that the nature of this process will be different for different kinds of agent. In the case of AI-ED, we have a program agent which might aim to reason as efficiently as possible on its own behalf, but when it is reasoning about a student's reasoning process, it is necessary to take account of psychological aspects. For example, any modal logic of belief which includes A_1 and A_2 (as does every modal logic using the possible worlds approach) suffers from the problem of *logical omniscience*, that is, the agent believes all implications of its beliefs. This is considered psychologically implausible. Therefore some kind of 'limited reasoning' might be defined (as discussed in chapter 5). Also we might need to consider short-term and long-term memory issues, which (roughly speaking) correspond to beliefs which are 'active' and 'inactive'. We might also consider how more global attributes of agents, such as that they are gullible, arrogant, narrow-minded, and so on, might be related to properties of their reasoning and other processes. (section 6.9).

4.4   Situation calculus

Our sentences in predicate logic and modal logic have so far been timeless, that is, we have not specified when sentences such as `B(John,Composer(Fidelio,Beethoven))` are supposed to hold. This is inadequate for AI-ED systems, whose express purpose is to change beliefs and knowledge.

Suppose we wanted to represent a simple 'transmission' principle: "If a student does not know something and we tell her it then she will know it." The following attempt:

       `~K(s,φ) & Tell(teacher,s,φ) ∅ K(s,φ)`

is a logical nonsense. The operators in predicate logic do not have an implicit temporal semantics: `&` and `∅` do not mean 'and then' and 'causes'. The sentence is in fact equivalent to `~p & q ∅ p`. Also, `Tell` is not naturally a relation constant but a performative (section 3.6).

One approach to tackling this is to use the idea of a *situation*, that is, an entity which is supposed to denote a snapshot of the universe and which can be regarded as a term in predicate logic. Then, to say that a student does not know a proposition in a particular situation, say `s27`, we may write:

       `~K(s,φ,s27)`

where we have introduced an extra argument to denote the situation in which the assertion holds. Now, to represent the above principle, we need to relate the situations before and after the telling act. This we may do by regarding the performative `Tell` as a function mapping a situation into a new situation. So,

       `tell(teacher,s,φ,s27)`

is the situation obtaining by applying the `tell` function when in situation `s27`. Then the transmission principle can be written:

       `~K(s,φ,t) ∅ K(s,φ,tell(teacher,s,φ,t))`

where `t` represents a situation variable. In this way, we may develop a language for expressing instructional principles, although, of course, any worthwhile principle will be much harder to express than the above. We will consider this further in chapter 10.

A variation on the above notation is to rewrite expressions of the form `P(t1,t2,..,t)` as `Holds(p(t1,t2,..),t)`, which asserts that *state* `p(t1,t2,..)` holds in situation `t`. For example, the above rule becomes:

       `~Holds(knows(s,φ),t) ∅ Holds(knows(s,φ),tell(teacher,s,φ,t))`

In other words, the relation constant `P` is re-expressed as a function `p`, so that `p(t1,t2,...)` can be regarded as a predicate logic object. The process of converting relations into objects is known as *reification*. Although the notation is more wordy, it allows us to quantify over and apply functions to states. Also, it allows a more natural notation for modifiers of states, such as

       `thoroughly(knows(s,φ))`

than the use of an arbitrary number of extra arguments, as in

       `K(s,φ,thoroughly,t).`

The concept of a situation, which have glibly introduced as an object in predicate logic, is of considerable potential philosophical and practical importance. As is well known, the meaning of indexicals such as 'here' and 'you' is situation-dependent, and the same may be said of almost any concept or proposition. For example, a 'billion' is different in England and America:

       `Holds(billion=10⁹,America) & Holds(billion=10¹²,England)`

A student would need to know which context she is in in order to carry out arithmetic involving the concept of a billion. Not only do words mean different things in different contexts, but people have different abilities in different contexs:

       `Holds(can-subtract(Fred),darts) & ~Holds(can-subtract(Fred),classroom)`

We often ask students to work within a specific context, i.e. to adopt a set of assumptions temporarily. For example, we might ask an economics student, who perhaps believes most in Keynesian principles, to nevertheless adopt a Marxist point of view for the purpose of some exercise.

While the concept of a situation or context is ubiquitous, for everything we say is said within a situation, and we can try to write expressions capturing our intuitions, it is very hard to specify a satisfactory semantics for situations (although, thankfully, we can achieve something without it). For example, it is not clear that an expression `Holds(p,t)` should even have a truth-value:

```
Holds(Lorentz-transformation=...,Newtonian-mechanics)
```

may rather be considered meaningless. One way to try to define a semantics for `Holds(p,t)` is to try to specify a set of rules of inference and axioms, as we indicated with weak S4 for modal logic. For example, we could wonder about:

```
Holds(holds(p,t1),t2) ⊃ Holds(holds(p,t2),t1)
```
```
Holds(holds(p,t),t) ⊃ Holds(p,t)
```

and so on, and, using the modal operators:

```
Holds(knows(a,p),t) ⊃ K(a,Holds(p,t))
```

Meanwhile, computationalists are already using the basic ideas of situations and contexts for practical purposes. For example, we can specify that some subset of a knowledge base holds only in a specified situation and so work more efficiently on only a portion of the knowledge base. We can also specify relationships among situations so that, for example, properties may be inherited. For example, from

```
B(s,Holds(exist(black-holes),universe))
B(s,Subset(solar-system,universe))
B(s,Holds(p,t) & Subset(t1,t) ⊃ Holds(p,t1))
```

we may infer

```
B(s,Holds(exist(black-holes),solar-system))
```

As the example indicates, we need to be careful in defining such relationships. As we will see, we can allow propositions associated with a situation to be inconsistent, and we can also associate rules of inference, even unsound ones, with a situation, giving great flexibility in representing how an agent reasons.

So, the technical concept of a 'situation' has been well-established in theoretical AI since the *situation calculus* was introduced by McCarthy and Hayes (1969). It has been refined and extended to provide a broad basis for reasoning about time and change. It is not clear that it is broad enough to encompass the notion of 'situation' in situationism. Here, a situation is "not a physical setting" but more to do with a 'community of practice' or a 'culture' (Clancey, 1995). One can hardly deny that culture is important, but what precisely does it mean? Brown, Collins and Duguid (1989) say that a culture is "a system of beliefs about some aspects of the world", which, on the surface, does not seem very different from what we have above. Saying that the situation calculus cannot in principle be extended to capture the rich notion of a culture that situationists have in mind is like saying that mere words cannot capture the taste of wine. However, unless we can be more precise about 'culture' then AI-ED systems will simply have to swallow whatever vague notion of culture is built into them (for it would be a mistake to assume that any system could be culture-free). Situated learning theorists argue that learning is or should be a process of enculturation. This is fine, but which culture do we mean? There are a very large, if not infinite, number of cultures. Consider the students in the four scenarios described in section 1.1: what, precisely, are the cultures into which they are enculturating themselves? Situationists seem to discuss cultures as though, unlike knowledge, they already exist in some objective sense. If asked, however, they would no doubt argue that a culture, like knowledge, has to be 'constructed' - which sounds quite a challenge for a learner. Otherwise, AI-ED systems would be engaged in a process of "culture transmission" which sounds even more invidious than mere "knowledge transmission". The general point is that vague polemical arguments about the role of such all-encompassing concepts as 'culture' have limited utility for AI-ED unless they are grounded in technical definitions amenable to theoretical analysis and practical application, especially when those arguments use or re-use terms like 'situation' which are already part of the technical lexicon.

4.5   Structured representations

As the above discussion of situations, contexts and cultures indicates, the propositions which are ascribed to an agent as beliefs or knowledge are rarely to be considered independent members of an unstructured set. A set of propositions may become structured in basically two ways: by partitioning the propositions into (possibly overlapping) subsets; and by specifying relationships between pairs of propositions.

Let us define an agent's *belief-set* `Beliefs(a)` to be the set of beliefs that have been ascribed to the agent `a`:

```
Beliefs(a) ≡ { p | B(a,p) }
```

Similar sets can be defined for the other modal operators, such as a *knowledge-set*:

```
Knowledge(a) ≡ { p | K(a,p) }
```

An agent's vocabulary `V(a)` may be defined to be the set of object, relation and function constants referred to in the agent's belief-set, knowledge-set, etc.:

```
V(a) ≡ { c | c is an object, relation or function constant
                & c occurs in Beliefs(a) or Knowledge(a) or ... }
```

An AI-ED system is concerned with some 'domain': we talk loosely of programs in 'the domain of algebra' or 'the domain of meteorology'. An agent's domain-vocabulary `DV(a)` is some subset of `V(a)` which the agent considers to

be part of the language for that domain. Obviously, the contents of `DV(a)` will differ for different agents. An agent's beliefs about a domain, `Domain-beliefs(a)`, is the subset of `Beliefs(a)` which concerns elements of `DV(a)`:

```
Domain-beliefs(a) + { p | B(a,p) & ∃c (c in p & c a DV(a) }

Domain-knowledge(a) + { p | K(a,p) & ∃c (c in p & c a DV(a) }
```

(In the following, we will restrict consideration to the `Believes` operator.) The set `Beliefs(a)` also contains the agent's beliefs about the problem at hand and its ongoing solution, denoted by `Problem-specific-beliefs(a)`. So, we can define an agent's background beliefs by:

```
Background-beliefs(a) + Beliefs(a) - Domain-beliefs(a) - Problem-specific-
    beliefs(a)
```

When solving problems in a domain an agent may need recourse to beliefs in `Background-beliefs(a)`, i.e. to propositions which appear unrelated to the domain or the specific problem. This is because `Domain-beliefs(a)` may contain references to constants which are not in `DV(a)` but which are referred to in other propositions in `Beliefs(a)`. These beliefs are indirectly or implicitly related to the domain. We could define:

```
Implicit-DV(a) + { c | ∃p c in p &
                     (p a Domain-beliefs(a) or p a Implicit-domain-beliefs(a)) }

Implicit-domain-beliefs(a) + { p | B(a,p) & ∃c (c in p & c a Implicit-DV(a) }
```

The distinction between `Domain-beliefs(a)` and `Implicit-domain-beliefs(a)` may be helpful in cases where an agent does not bring to bear beliefs which are not explicitly, obviously useful, and is related to the discussion of what constitutes a 'context' in the previous section.

If we consider a particular constant `c` (say, `Fidelio`), then we can form a set of the agent's beliefs (or knowledge) which are explicitly about `c`, i.e. have a reference to `c`:

```
Beliefs-about(a,c) + { p | B(a,p) & c in p }

Knows-about(a,c) + { p | K(a,p) & c in p }
```

These sets might be considered to represent the agent's beliefs or knowledge about the 'concept' `c`. That is, what an agent beliefs (or knows) about a concept is just the set of propositions which the agent believes (or knows) about that concept. For example, if we take the concept of 'metal', we can imagine that for a novice chemist:

```
Beliefs-about(a,metal) = { Isa(iron,metal), Isa(steel,metal), ...
                          ∀x Isa(x,metal) ∅ Shiny(x), ... }
```

For an expert chemist, some propositions will be in terms of the electron structure of metals. This seems to accord with the intuition that a concept is not a 'black-and-white' notion, that one knows all or nothing about. We could plausibly define:

```
Knows-of(a,c) + ~(Knows-about(a,c) = {})
```

Computationally, there are clearly potential practical benefits in 'gathering together' propositions which appear to be related, rather than have to search for them in an unstructured set. This is the basis for all the standard AI work on frames, scripts, schemas, semantic networks, and so on, which is obviously relevant, but not specific, to computational mathetics, and so will not be elaborated on here. In formal terms, there is no difference in these representations, because they can all be converted into one another and predicate logic, although there is much discussion of their merits in particular applications.

The idea of a *frame* begins to illustrate the second way of structuring belief sets mentioned above, that is, by defining relationships between propositions. A frame is essentially a set of relation constants (or *slots*) which are applied to (or filled by) particular *values* to define a certain concept. A frame (e.g. `iron`, `steel`, ..) may be defined to be a subclass of other frames (e.g. `metal`, `noble-metal`, ..), in which case it may inherit values from the superclass (just as from ∀x Isa(x,metal) ∅ Shiny(x) we may infer `Shiny(iron)`). It may not always inherit values, for sometimes values are only default assumptions which may be overridden. For example, most metals are solid at room temperature (we will consider this in the next chapter).

4.6   Multiple representations

AI-ED research has always emphasised the need for 'multiple representations' of knowledge, appreciating that the different agents (such as students and experts) have different views and that in most domains different views are necessary (such as different economic theories). We must distinguish three different uses of the term:

- For different representations of the *same* knowledge, for example, the use of predicate logic or semantic networks to represent the same knowledge of metals. In this case, the concern is to enable conclusions to be reached more efficiently, not to find different conclusions.
- For representations of *different* knowledge of the *same* agent, for example, to view a prison as a place for incarcerating criminals or for rehabilitating them. This is the idea of context or situation, as discussed in section 4.4. Here, the problem is to find an 'appropriate' view to enable conclusions (which might differ from those of another view) or to coordinate the use of more than one view.

- For representations of *different* knowledge of *different* agents, for example, to view a metal in terms of visible characteristics (such as shininess and hardness), as a student might, or in terms of electron structure, as a teacher might. In this case, the issue is to coordinate or negotiate communication between the two agents.

Computational mathetics is most concerned with the second and third uses. For example, the aim of some AI-ED systems is expressed in terms of enabling a student to progress through a succession of representations corresponding to increasing levels of expertise in the domain (White and Frederiksen, 1990). In addition, a number of studies (for example, DiSessa, 1987) have shown that students bring idiosyncratic views of their own to problem-solving and these have to be taken account of when interacting with the student, for example, when giving explanations.

For example, Kamsteeg (1994) describes (in English) six "naive mental models" which students may have about heat and temperature and which we can try to define in terms of their beliefs:
- A 'simple particle model':
   ```
   Beliefs(s) = { Temperature(body) ∝ number-of-'heat particles' }
   ```
   i.e. heat is a kind of substance which holds a certain amount of temperature.
- A 'refined particle model':
   ```
   Beliefs(s) = { Temperature(body) ∝ number-of-'heat particles' / Size(body) }
   ```
   i.e. as above except that bigger bodies need more heat particles for the same temperature rise.
- A 'resistance model':
   ```
   Beliefs(s) = { Temperature(body) ∝ number-of-'heat particles'
                                     / Size(body) x Heat-resistance(body) }
   ```
   i.e. as above except that bodies have a 'heat-resistant layer' off which heat particles are 'bounced' and lost. The more heat-resistance a body has, the less its temperature rises for a given amount of heat.
- A 'magnet model':
   ```
   Beliefs(s) = { Temperature(body) ∝ number-of-'heat particles'
                                     x Heat-holding-capacity(body) / Size(body) }
   ```
   i.e. as in the refined particle model except that bodies have different capacities for holding on to their heat particles.
- A 'simmer model':
   ```
   Beliefs(s) = { Temperature(body) ∝ number-of-'heat particles'
                                      / Evaporation-rate(body) x time
                 Temperature(environment) ∝ number-of-'heat particles'
                                      x Evaporation-rate(body) x time }
   ```
   where `time` is the time since the heat particles were added to the body (assumed all at the same time, for simplicity), i.e. a body loses heat particles to the environment.
- A 'battery model':
   ```
   Beliefs(s) = { Temperature(body) ∝ number-of-'heat particles'
                                     x quality-of-'heat particles' }
   ```
   i.e. some heat particles are of better 'quality' than others and are not so quickly lost (just as some batteries lose volts faster than others).

The precise definitions of these models is obviously more complex. Some of them, for example, make subtly different predictions about changes of temperature over time, which we haven't addressed above. We should note also that the six models are not all mutually exclusive: they can be combined in various ways to give more models than just those listed above. The ambition to use such models to explain and predict student behaviour is fraught with considerable difficulty. Among the problems, some of which will be discussed later, are:
- Students do not use such models completely consistently - they may switch from one model to another, or misapply a model.
- The models themselves may be inconsistent or incomplete.
- It is often hard to ascribe a particular model to a student on the basis of her behaviour.
- Students may not be able to discuss such models directly, at least not in the terms which we have used in the model descriptions (for example, the analogies with resistors, magnets and batteries are ones which we, as observers, might make, but which students themselves may not).

Nonetheless, students do not tackle problems of elementary thermodynamics in a random fashion. They do, to some extent, try to develop and apply coherent theories. For an AI-ED system, it seems plausible that it would be useful for the system to have some conception of the student's current model(s).

This example from physics might be taken to imply that the student is seeking to develop the one 'correct' model. The same idea of multiple representations can (or should) be used in domains, like economics, where no one model is distinguished as correct and where the students task is more to apply, combine and compare a range of models. In summary, multiple representations are important in AI-ED for two distinct reasons: to improve problem-solving efficiency, because different views of a domain are possible, and to develop better psychological models of students. The reasons for adopting multiple representations influence how we deal with theoretical and practical limitations in using with them.

4.7   Social knowledge

Our notation allows us to nest modal operators concerning different agents, for example, to write
```
        B(a,B(b,p))
```
to mean that we have ascribed to agent `a` the ascription of belief `p` to agent `b`, or more simply that `a` believes `b` believes `p`. This provides our means to describe the interrelationships and interactions between members of a society of agents, an understanding of which "is vital for story understanding, in automated teaching and in intelligent interactive systems" (Davis, 1990), all components of computational mathetics. However, our current ability to describe these aspects formally is very limited.

As previous sections discussed, we may partition an agent's beliefs or knowledge into views or contexts. Clearly, there is likely to be some overlap between these views and those of other agents. It will be useful to identify those propositions which are common to the belief-sets or knowledge-sets of all agents in a society. For example, as they are common, such propositions may not need to be communicated or explained to one another. If
```
        ∀x K(x,p)
        ∀x1,x2 K(x1,K(x2,p))
        ∀x1,x2,x3 K(x1,K(x2,K(x3,p)))
```
and so on, then we say that p is common knowledge to the set of agents:
```
        Common-knows({a1,a2,a3,...},p)

        Common-knowledge({a1,a2,a3,...}) + { p | Common-knows({a1,a2,a3,...},p) }
```
This set of agents is sometimes called 'any fool'. As with single-agent modal logic, we can try to define some axioms for common knowledge, for example,
```
        Common-knows({..,a,..},p) ∅ K(a,p)

        Common-knows(x,p) ∅ Common-knows(x,Common-knows(x,p))

        Common-knows(x,p) & Common-knows(x,p∅q) ∅ Common-knows(x,q)
```

The idea of common knowledge has been studied in fields as diverse as distributed computer systems (Halpern and Moses, 1992) and philosophy (Strawson, 1971). Implicit in the notation is a view that the (common) knowledge of a set of agents is the intersection of the individual agent's knowledge. Of more potential relevance to AI-ED, given the enthusiasm for collaborative learning, is the view that the knowledge of a set of agents is the *union* of the individual agent's knowledge. Unfortunately, no formalisations of this view seem to exist. We could imagine that the set's knowledge can be used to solve a problem not solvable by any single agent, perhaps leading to each agent believing all propositions necessary to solve that problem:
```
        B(a1,p,t) & B(a2,p∅q,t) ∅ B({a1,a2},{p,p∅q},t)

        B(x,{φ,φ∅ψ},t) ∅ Solves(x,ψ,{φ,φ∅ψ},t)

        Solves(x,y,z,t) ∅ ∀a,p (a a x & p a z ∅ B(a,p,solved(y,t)))
```

There has been more work on multi-agent collaborative or cooperative planning than there has on multi-agent problem-solving. This may have relevance to computational mathetics as student and teacher agents have common and conflicting goals which should be taken into account. Issues that need to be considered include:
* The division of tasks between the various agents.
* The communication of relevant information between agents.
* The prediction of how other agents will react to a request.
* The coordination of plans which may hinder or obstruct one another.

Some of these issues will be considered in later chapters.

4.8   Procedural representations

The idea touched upon in the previous section that knowledge exists (only) as a social construct is offered as a radical alternative to the standard cognitive science view that knowledge is an individual cognitive construct. Another possibility is that knowledge exists (only) in action:

   "Action is involved in knowledge ... in the sense that knowledge is a form of action, and that action is one of the terms by which knowledge is acquired and used." (Hofstadter, 1962)

The relation between knowledge and action has also concerned philosophers. For example, Ryle (1949) is often selectively quoted for writing that "It is ... possible for people intelligently to perform some sorts of operations when they are not yet able to consider any propositions enjoining how they should be performed" in order to justify a 'knowledge without representations' philosophy (Brooks, 1991). In fact, Ryle was arguing that mental concepts refer not to unwitnessable activities in the body or mind but to dispositions to behave in certain ways in appropriate circumstances (much as we have assumed). One can have behaviour without (symbolic) knowledge and also knowledge without behaviour. Whitehead (1932) referred to 'inert ideas', that is, knowledge which appears to exist (perhaps because it can be stated) but which cannot necessarily be applied when it should.

This split between explicit knowledge and behaviour is not necessarily desirable. The fact that humans can behave without knowing and can know without behaving is not a reason that AI-ED systems should aim for such an outcome. Sometimes it is sufficient to be able to behave without being able to articulate propositions underlying that behaviour. For example, I can use my phone without being able to say precisely how, by giving a description of the layout of buttons. But then I never use my phone without it being literally to hand, so that the layout of buttons is not a problem. However, if I were incapacitated (perhaps after being attacked by a burglar) then it might be useful to be able to articulate an explicit procedure executable by my four-year-old daughter (if I had one).

Ryle (1949) commented that "efficient practice precedes the theory of it". To which, one may respond "not always". Sometimes there is a lot of efficient practice and no theory; sometimes there is more inefficient practice than there is efficient practice; sometimes a good theory precedes efficient practice. Usually, practice and theory develop together in mutual support. In some ways, the rationale for our own discussion about AI-ED mirrors this debate. We might believe that AI-ED system designers can successfully design AI-ED systems without being able to consider any propositions enjoining how they should be designed. Or we might believe that the development of a good theory might contribute to the efficient practice of AI-ED system design.

Situationists believe that the link between mental structures and behaviour is the wrong way round in conventional cognitive science and AI, which emphasises transition 1 in this diagram, i.e. the view that behaviour follows from instantiating and interpreting underlying mental structures.

```
Individual, cognitive
mental structures              2
                          1        Behaviour
           Situation        3
```
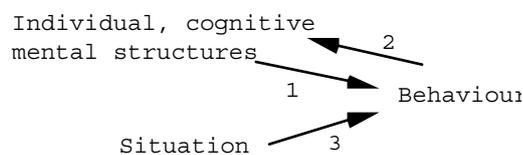
Situated cognition, on the other hand, focusses on transitions 2 and 3. Hanks (1991) writes that there is a "potentially radical shift from invariant structures to ones that are less rigid and more deeply adaptive. One way of phrasing this is to say that structure is more the variable outcome of action than its invariant precondition ... It (i.e. behaviour) involves a prereflective grasp of complex situations".

The field of AI has debated the 'declarative/procedural controversy' (Winograd, 1975) almost to exhaustion. In order to situate the situated cognition position, it is worth briefly summarising the use of production systems, a common representational scheme in AI-ED and in expert systems and psychological modelling. With this scheme an agent is ascribed a set of production rules, or condition-action rules. The 'condition' is expressed as a list of patterns which are matched against items in a working memory (the agent's problem-specific knowledge). The 'action' is a set of individual actions which change the contents of the working memory. Typically a problem is solved by repeatedly applying rules whose conditions match items in the changing working memory. For example,

```
Knows-how(a,to-boil-an-egg):
    Water-in-pan-is-boiling ∅ Put-egg-in & Turn-egg-timer
    Pan-is-empty ∅ Add-sufficient-water
    Egg-timer-has-run-through ∅ Remove-boiled-egg & Turn-gas-off
    Pan-has-water & Gas-is-off ∅ Turn-gas-on
    Egg-timer-is-running ∅ Do-the-daily-crossword
```

As rules only fire if their conditions are satisfied, the order of the rules (in simple cases) does not matter and therefore it is possible to add rules to handle further situations. The refinements of production systems are discussed in any standard AI text. That the distinction between 'procedural' knowledge and 'declarative' knowledge is hazy is clear from considering the similar notation of the Prolog programming language, where a single definition can serve as both a proposition, returning true or false, and as a procedure computing a value.

Usually, production systems contains many hundreds of rules. Their main virtues are said to be that the rules can be independently associated with individual actions, and that the rules can be independently articulated. The virtues and limitations of production systems for AI-ED purposes have been thoroughly discussed (Anderson, Boyle, Corbett and Lewis, 1990; Clancey, 1987). The main point to make here is that the approach does indeed emphasise transition 1 (in the above diagram) but it does not entirely neglect the other two transitions. The rules themselves do not entirely determine behaviour - they are adapted by the 'situation' represented by the working memory. Also, the actions may, if desired, change the mental construct itself, by amending existing rules or adding new ones. Of course, these processes may not fully capture what situationists have in mind.

Similarly, as the above quote from Hanks (1991) shows, situated cognition does not entirely ignore the role of mental structures in determining behaviour. However, we may ask:

- What exactly is a "less rigid and more deeply adaptive" structure? Is it a different kind of structure to the productions systems, blackboards, frames, and so on developed in AI? All of these are adaptive to some extent - but what extra or different does "deeply adaptive" mean?
- How is such a structure used to provide "a prereflective grasp of complex situations"? How does this differ from, for example, a production rule which carries out a very 'superficial' matching of the situation to provide a 'grasp' of it?
- What kind of structure is it that "is more the variable outcome of action"? Is the structure produced as *outcome* of action very different in kind to that postulated in standard cognitive science as the *input* to action? If not, is it acceptable to use standard schemes to represent such outcome structures?
- How long do these outcome structures last? Can they be used as the 'less rigid' structures providing 'prereflective grasps' of later situations? If not, why do agents bother to create such structures?

As more precise answers to these kinds of questions are developed, it is likely that situated cognition will become seen mainly as a theory emphasising aspects of cognition which have been neglected in previous approaches and which are particularly important in some circumstances, but not as a theory warranting the wholesale jettisoning of theories and techniques already found useful in other circumstances.

5.   Reasoning

In this chapter we will consider how agents reason with mental components such as beliefs. We will interpret the word 'reason' broadly to include any process by which an agent derives conclusions from premises during the course of problem-solving. The process need not be considered sound or rational in the logical sense, because, of course, some agents are not always sound and rational. We will try to be explicit about these reasoning processes because:

- In current AI-ED systems these processes are often buried in executable code and not amenable to any kind of theoretical comparison.
- An AI-ED system which has access to explicit reasoning processes may be able to reason about them and not just about beliefs and knowledge. The system may be able to discuss and explain such reasoning processes, possibly in a domain-independent way, and so move beyond solely domain-related issues.
- It may be possible to customise representations of reasoning processes in order to describe different agents (for example, different students) or the same agent at different times (if, for example, a student learns a new reasoning process).
- By identifying reasoning processes appropriate for different agents we may be able to separate computational and cognitive issues, which are currently intertwined. For computer agents, it may be that efficiency considerations are most important; for student agents, psychological validity may be most important.
- In order to discard beliefs, it may be useful to record how they were derived, which is only possible if the reasoning processes can be monitored.

A range of reasoning processes will be described, with a modest degree of formality. It is not the aim to develop a complete, 'correct' representation. Rather the aim is to develop illustrative notations of sufficient precision that we may determine which properties of reasoning processes are relevant to particular AI-ED applications. Considering the computational complexity of the various notations might help clarify the compromises necessary to achieve practical performance. Just as the previous chapter tried to consider belief and knowledge independently of the uses to which it may be put (the subject of this chapter), so in this chapter we will consider the reasoning processes that an agent may have available without considering which of the processes an agent may choose to apply in any particular problem-solving context (which is considered in the next chapter).

5.1   Reasoning schemata

reasoning schemataA representation such as

        B(a,Metal(x) ∅ Shiny(x))

which is intended to denote that an agent a has been ascribed the belief that all metals are shiny, does not indicate how the agent may use this belief to derive conclusions. We could imagine that the agent might in different situations, such as the presence of an object b which is or is not a metal or is or is not shiny, infer Shiny(b) or ~Shiny(b) or Metal(b) or ~Metal(b), respectively. Some of these inferences are valid in standard logic, and some are invalid but nonetheless plausible. For more complex beliefs it is naturally harder to say what inferences might follow.

We will use the notation

        Reasons(a,φ1, φ2, φ3, ... => ψ)

to indicate that we have ascribed to agent a the 'reasoner' φ1, φ2, φ3, ... => ψ. The reasoner φ1, φ2, φ3, ... => ψ indicates that if we have ascribed beliefs φ1, φ2, φ3, ... to the agent then it may be necessary to also ascribe the belief ψ, as the agent considered to hold those beliefs may make that inference. So, for example, from

        Reasons(a,φ∅ψ, ψ => φ)
        B(a,Metal(x) ∅ Shiny(x))
        B(a,Shiny(steel))

we may need to make the ascription

        B(a,Metal(steel))

As this simple example indicates, a reasoner is a schema which needs to be instantiated to match particular beliefs (we will consider this further below). We use the term 'reasoner' to avoid the connotations of phrases such as 'rule of inference' or 'operator'. As we did in section 4.5 for beliefs, so we may define an agent's *reasoner-set* Reasoners(a) to be the set of reasoners ascribed to the agent:

        Reasoners(a) + { r | Reasons(a,r) }

We may now illustrate some reasoners which may be relevant to AI-ED.

5.1.1   Reasoning in standard logics

The reasoner:

        φ∅ψ1, ~φ∅ψ2 => ψ1 ∨ ψ2

or equivalently

        ~φ ∨ ψ1, φ ∨ ψ2 => ψ1 ∨ ψ2

describes the rule of inference called *resolution*, which is known to be sound and complete in predicate logic. To apply the rule all the sentences have to be in conjunctive normal form (section 4.2). The rule of resolution is the basis for standard theorem-provers in predicate logic, the programming language Prolog, and the rule-matching mechanism of production systems. The precise definition of resolution, explaining how variables may be substituted to make expressions unify, is given in any theoretical AI textbook (such as Genesereth and Nilsson, 1987).

A special case of the rule is when $\psi1$ and $\psi2$ are both empty:

```
~φ,φ => false
```

i.e. the agent may infer `false` if there are two propositions in the belief-set which directly contradict one another. As resolution is complete (i.e. any sentence logically implied by a set of sentences can be derived by repeated applications of the rule of resolution), if a set of sentences $\phi1$, $\phi2$, ... ϕn is contradictory then `false` may be derived from them. In such a case, ~($\phi1$ & $\phi2$ & ... & ϕn) is a theorem, by definition; or, equivalently, $\phi1$ & $\phi2$ & ... $\varnothing$ ~ϕn is a theorem. Therefore, to show that an expression of the form $\phi1$ & $\phi2$ & $\phi3$ & ... $\varnothing$ ψ is a theorem (as discussed in section 4.2) one may negate the conclusion ψ and show that $\phi1$ & $\phi2$ & $\phi3$ & ... ~ψ leads to `false` after applying resolution. Although this procedure is not guaranteed to terminate (as predicate logic is an undecidable system), there has been much research on developing efficient theorem-provers.

This reasoner may be used by a system agent to reason about its own knowledge, or to reason about what is implied by the beliefs ascribed to a student. However, the reasoner is of little use if the system needs to reason about the student's beliefs in the same way that the student does or might, because it is implausible, to say the least, that a student will apply anything comparable to the rule of resolution. The reasoner is also of little use if the system needs to reason about its own knowledge and explain its reasoning to a student (as resolution is not likely to be comprehensible). Thus, resolution cannot be used to model students' reasoning processes or to explain reasoning processes.

Rules of inference for a logical system may be defined which are intended to give proofs that are closer to those that human theorem-provers produce. We could, for example, have reasoners such as:

```
φ & ψ => φ
modus-ponens: φ∅ψ, φ => ψ
modus-tolens: φ∅ψ, ~ψ => ~φ
φ∅ψ, ψ∅φ => φ×ψ, ...
```

Such a 'natural deduction' set of reasoners might be more useful for modelling the reasoning processes of students than the rule of resolution. For convenience, we can attach names (such as `modus-ponens` and `modus-tolens` above) to individual reasoners, as indicated above.

If the aim of the reasoner-set is to provide an ascription to a student which corresponds to problem-solving performance then it might be necessary to include reasoners which are unsound:

```
φ∅ψ, ψ => φ
φ∅ψ, ~φ => ~ψ
```

Apart from having faulty reasoning schemata, students may also not have a complete set of 'correct' reasoners. All the issues which arise when attempting to model students' knowledge (chapter 8), arise also with respect to their reasoning processes, although few AI-ED systems attempt to model the latter. (Note that the process of reasoning unsoundly is different to the process of reasoning with inconsistent premises, the latter being discussed briefly in section 5.4.)

The previous examples suggest that our reasoners are just rules of inference in a formal logic system. However, we would also consider what are in some domains called 'operators' to be examples of reasoners. For example, in algebra, we might consider that an equation such as `x=c*(e)` where `c` is an integer and `e` an expression, can be rewritten as `x=c*e` where `c*e` denotes the result of applying a multiplying-out operator. We could represent this by:

```
Equation(x=c*(e)) => Equation(x=c*e)
```

In this case, because we are likely to need to keep a record of the operators applied, we might use the situation variable introduced in section 4.4:

```
Equation(x=c*(e),t) => Equation(x=c*e,multiply-out(t))
```

We could similarly represent normal rules of inference, to label them and keep a track of problem-solving performance:

```
(φ∅ψ,t),(φ,t) => (ψ,modus-ponens(t))
```

where (ϕ,t) refers to the condition of having ascribed `B(a,φ,t)` to the agent.

AlgebraLand (Foss, 1987) provides a menu of such operators. In this case, it is assumed that students understand all the operations denoted by the menu items. Therefore, the problem of ascribing an appropriate reasoner-set to the student is bypassed. Many simulation-based systems adopt a similar approach. For example, a simulated chemistry lab might provide a menu of chemical operations for students to select from. It would be assumed that students know what the operations are and that their problem is more one of determining an appropriate sequence of operations (in this case, perhaps, to prepare a specified chemical). In more open environments, where students can carry out their own operations, there is obviously a difficult diagnosis problem for a system to ascribe reasoners to a student. For example, if a student is attempting to solve a symbolic integration problem (and there is no prescribed menu of

allowable transformations) then it can be hard to determine an operation whereby the student has moved from one expression to another, especially as the transformations may be incorrectly applied.

5.1.2   Reasoning in nonstandard logics

There is a large variety of 'non-standard' logics which attempt to overcome some of the apparent shortcomings of predicate logic. For example, the following theorems

$\phi\varnothing(\psi\varnothing\phi)$

$(\phi\ \&\ \sim\!\phi)\varnothing\psi$

may appear paradoxical if we consider that $\phi\varnothing\psi$ should indicate some kind of causality between ϕ and ψ. It is possible to define logics with different notions of validity than predicate logic. For example, *relevance logic* (Anderson and Belnap, 1975) denies the above paradoxes and requires that one proposition entails another only if there is an element of causality that relevantly connects them. In this logic, we have rules of inference such as:

$(\phi,\alpha),\ (\psi,\alpha) => (\phi\ \&\ \psi,\alpha)$

$(\phi,\alpha),\ (\psi,\alpha\ B\ \beta) => (\phi\varnothing\psi,\beta)$

In this logic (ϕ,α) means that the sentence ϕ was derived from sentences in its 'origin set' α. The first rule says that if ϕ and ψ are sentences with the same origin set, then we can deduce $\phi\ \&\ \psi$ and associate it with the same origin set. Relevance logic has been used in work on belief revision (section 7.2.4) and is presumably partly motivated by a feeling that human reasoning follows such schemata rather than those of standard logic. Several other nonstandard logics will be introduced later.

We may also adapt standard logics to take account of the fact that many studies have indicated that humans, even those with training in formal logic, do not find it easy to apply abstract reasoners of the above kinds. Holland, Holyoak, Nisbett and Thagard (1986) develop the idea of *pragmatic reasoning schemata* which are intermediate between abstract rules and domain-specific rules. A pragmatic reasoning schema is a "set of generalized context-sensitive rules which, unlike purely syntactic rules, are defined in terms of classes of goals (such as taking desirable actions or making predictions about possible events) and relationships to these goals (such as cause and effect or precondition and allowable action)", for example:

```
Goal(x) & Precondition(x,y) => Must-satisfy(y)
Goal(x) & Precondition(x,y) & Can't-satisfy(y) => Can't-do(x)
```

Pragmatic reasoning schemata are abstract rules in that they apply to a wide range of content domains but they are constrained by particular inferential goals and event relationships. They are, to some extent, 'situated'. The degree to which students can be usefully ascribed such pragmatic schemata is a matter of debate.

Reasoners may be more or less abstract, and more or less domain-independent. Formal logic rules of inference are intended to be abstract and domain-independent, pragmatic reasoning schemata less so. Operators such as `integrate-by-parts` are abstract but fairly domain-dependent (of course, it depends how we define the 'domain').

Abstract reasoners may be made more 'concrete' by instantiating them. For example,

```
Rainy∅Umbrella, Rainy => Umbrella
Planet(x)∅Orbits(x,sun), Planet(neptune) => Orbits(neptune,sun)
```

are both instantiations of the `modus ponens` reasoner. A person might believe the proposition `Rainy∅Umbrella`, i.e. if it is rainy take an umbrella, but only if she actually behaved as the rule implies in relevant situations might we wish to ascribe to her the reasoner above. If a student behaved in such a way that we might ascribe both the above reasoners to her we would not necessarily wish to ascribe the general form of the rule (as there might be other situations in which modus ponens is not applied, for some reason). There is a range of possible abstractions of a particular reasoner. For example, we might have:

```
Planet(x)∅Orbits(x,sun), Planet(c) => Orbits(c,sun)
P(x)∅Q(x,b), P(c) => Q(c,b)
```

where P and Q are arbitrary relations. (Note that the second reasoner is not in predicate logic syntax - to make it so, we could reify the relations, as discussed in section 4.4. Also the variables b and c are not normal variables, for the intention is that they stand for any object constant. These are difficult technical issues which would need to be addressed in any full theoretical analysis or implementation.)

Given a domain-vocabulary, we could proceed to define a `Domain-reasoners(a)` set analogous to the `Domain-beliefs(a)` of section 4.5. We could also define a set of `Abstract-reasoners(a)` which contains those reasoners in `Reasoners(a)` which do not refer explicitly to any object, relation or function constant, such as the last reasoner above. We could define the other reasoners in `Reasoners(a)` to be in `Concrete-reasoners(a)`, although they may still have object, relation or function variables within them (as in the last but one reasoner above).

A member of `Abstract-reasoners(a)` could not be a member of `Domain-reasoners(a)` because it does refer to any member of the agent's vocabulary. After instantiation, an abstract reasoner could be applied to any number of domains. It is arguable that an agent which may be ascribed an abstract reasoner, rather than a series of instantiations of it, is a more capable problem-solver, showing a deeper understanding of the general processes of deriving

conclusions from premises. On the other hand, the specific instantiations may be more efficient to apply. Acquiring a more abstract reasoner may be a step towards 'transfer' (section 6.7), i.e. the ability to apply skills acquired in one domain to a second domain. Hutchins (1991) gives the example of a pilot who has learned situation-specific skills of the form "When the dial indicator reaches a certain point, then drop the wing flaps", bypassing the official 'abstract' rule given in the technical handbook which requires a simple calculation involving height and speed. The former reasoner is adequate until the pilot is transferred to a different plane.

5.1.3   Reasoning in modal logics

So far in this section we have only considered the case where a reasoner is expressed in terms of sentences in predicate logic (or something close to it). The `Reasons(a,φ1, φ2, φ3, ... => ψ)` notation indicates we might infer `B(a,ψ)` from `B(a,φ1,)`, `B(a,φ2)`, `B(a,φ3)`, ... If the sentences φ1, φ2, φ3, ... , ψ do not contain any modal operators then we may consider only how an agent might reason in predicate logic (or something similar) and omit the `B(a,..)`s from our notation, as we have above. However, matters are more complicated if we have, as we do in AI-ED, beliefs about other agent's beliefs (or about one's own beliefs), such as

        `B(c,B(s,Metal(x)⊘Shiny(x)))`

i.e. the computer system believes that the student believes all metals are shiny. Let us imagine that we also have

        `B(c,B(s,~Metal(mercury)))`
        `B(c,B(s,Metal(iron)))`
        `B(c,B(s,Shiny(mercury)))`

If we have

        `Reasons(c,φ⊘ψ,  φ => ψ)`

then nothing follows from the above four belief ascriptions. We could try:

        `Reasons(c,B(s,φ⊘ψ),  B(s,φ) => B(s,ψ))`

or maybe

        `B(c,Reasons(s,φ⊘ψ,  φ => ψ))`

These two expressions differ in who the reasoning is ascribed to. In either case, we might want to infer:

        `B(c,B(s,Shiny(iron)))`

However, we cannot simply take one agent's reasoners and effectively ascribe them to a second agent, because the second agent may reason differently. If, for example, we have

        `B(c,Reasons(s,φ⊘ψ,  ψ => φ))`

then perhaps we should be able to infer

        `B(c,B(s,Metal(mercury)))`

and hence that the system believes that the student may derive a belief which contradicts an already held belief.

We may define the set of beliefs and the set of reasoners which an agent `a` ascribes to an agent `b`:

        `Beliefs(a,b) + { p | B(a,B(b,p)) }`

        `Reasoners(a,b) + { r | B(a,Reasons(b,r)) }`

We may define a function `Interpret` which maps a reasoner-set and a belief-set into a new belief-set. `Interpret` is a program which defines which reasoners actually generate new beliefs: it is considered further in chapter 6. In the above example, there are nine possible applications of the function `Interpret`, most of which are of potential relevance to AI-ED:

- `Interpret(Reasoners(c),Beliefs(c))` - describes how the program reasons about the domain;
- `Interpret(Reasoners(c),Beliefs(s))` - describes how the program would reason about the domain using the student's beliefs (which, of course, are generally not known);
- `Interpret(Reasoners(c),Beliefs(c,s))` - describes how the program would reason about the domain using the beliefs ascribed to the student by the program;
- `Interpret(Reasoners(s),Beliefs(s))` - describes how the student reasons about the domain;
- `Interpret(Reasoners(s),Beliefs(c,s))` - describes how the student would reason about the domain using the beliefs ascribed to her by the program;
- `Interpret(Reasoners(c,s),Beliefs(c,s))` - describes how the program believes the student would reason about the domain using the beliefs ascribed to her by the program.

In general, we identify a set of belief-sets (the beliefs ascribed to an agent, or ascribed by one agent to another) and a set of reasoner-sets (the reasoners ascribed to an agent, or ascribed by one agent to another). For a given belief-set Δ and reasoner-set `R`, a proposition `p` is provable, with respect to this belief-set and reasoner-set, if and only if `p` a `Interpret(R,Δ)`. If we wish, we could say that `p` is true according to this belief-set and reasoner-set. At this stage, we make no assumptions about the completeness or consistency of the belief-sets and reasoner-sets, although obviously some undesirable properties will cause problems for `Interpret`. In particular, we expect the reasoner-sets to be limited in various ways, for computational and psychological reasons.

This arbitrarily nested, multi-agent notation is more complex than the formal modal logics for which theoretical treatments have been developed. Most of this work has used variations of the axioms and rules of inference given for

a uni-agent modal logic in section 4.3. All of the axioms have some plausibility (which is why they are suggested, of course) but all seem to have computational and philosophical limitations. For example, the 'negative introspection' axiom applied to belief:

$$\sim\!B(a,p) \varnothing B(a,\sim\!B(a,p))$$

seems to support the intuition that if I don't believe a proposition then I believe that I don't. We might need such an axiom when a student becomes aware of her own lack of knowledge. But how do we determine `~B(a,p)`? There are an infinity of such negative sentences - both computer and human agents are unlikely to have it 'stored' in any sense and will have to derive it, perhaps by showing `B(a,~p)`, a process related to the problems of logical omniscience mentioned in section 4.3. Philosophers labour to re-express the axioms in order to capture the nuances of the notion of 'belief'. One suspects that this is an ultimately futile exercise, just as it is for linguists to try to define grammars which precisely define all and only the sentences of a natural language. However, just as limited grammars are useful in computational linguistics, so limited axioms for modal logics may be useful in computational mathetics.

In this section we have considered only the problem of defining axioms and rules of inference for the `Believes` modal operator. Similar definitions would be needed for any other modal operators which we feel we need in computational mathetics, and we would also need axioms for reasoning about combinations of the operators. We would also need to represent the dynamic aspects of such ascriptions, because, as discussed above, the whole point of AI-ED is to facilitate such change. All this is obviously a formidable task, especially for a multi-agent logic. However it may only be through developing such definitions that the subtle differences between modal operators can be explicated and precise specifications of AI-ED systems given.

For example, Baker (1994) argues that at least in negotiative learner-teacher dialogues a more useful ascription to students is that of 'acceptance' rather than 'belief'. Of course, merely changing the name of the ascription makes not a scrap of difference. To make a difference, we (ideally) should provide axioms and rules of inference for `Accepts` which define how it differs from `Believes`. Informally, the main differences are intended to be that:

- An agent can *decide* to accept a proposition but not to believe it. Thus, there is a different relation between `Decides` and `Accepts` than between `Decides` and `Believes`, where `Decides` is some kind of modal operator dealing with an agent's decisions. What can be accepted presumably depends on what is already believed.
- "To accept a proposition is to be willing to use it in one's reasoning, but not necessarily to adopt it in any reasonably stable sense (like belief)". So acceptances are relatively temporary. An acceptance 'dies' for one of two reasons - there is evidence to drop it, or it transforms into something more stable, like belief.
- Acceptances are more readily dropped than beliefs. As propositions are accepted only 'for the sake of the argument', if the argument indicates that they are unsound then they are more readily discarded.
- More successful acceptances transform into beliefs, not through the passage of time but after the accepted proposition has been found sufficiently useful for problem-solving. For AI-ED systems, this is the intention because we would not want students to just accept any and all propositions and never come to believe anything.
- "What a speaker says .. is not necessarily a direct reflection of what he believes but .. rather of what the speaker is prepared to accept." So discussions of dialogue (chapter 9) should be expressed in terms of `Accepts` rather than `Believes`.

Certainly, students are uncomfortable with ascribing beliefs to themselves. If asked, they prefer not to admit to 'beliefs', which seem to require some reasoned commitment which they are not yet prepared to make. Unlike experts and AI systems, students do not anticipate problem-solving success and therefore are fully prepared to find that propositions used in problem-solving are in fact unsound. Rather, they are prepared 'to go along with' (i.e.accept) propositions and see how things work out. Therefore it seems likely that computational mathetics will benefit from a more precise definition of acceptance than the informal statements above, although the philosophical logicians (such as Cohen, 1992) who have considered the nature of acceptance and belief have yet to tackle this.

5.2   Limited reasoning

Even the simplest of the reasoning schemata given above has potential problems. For example, the resolution rule may be applied forever in an attempt to derive a particular conclusion, as predicate logic is undecidable. We cannot just apply all reasoners repeatedly until no further conclusions can be drawn (for example, the introspective rules in modal logics are recursive). This would be unreasonable computationally (as it would take much too long, in general), psychologically (as human agents do not draw all possible conclusions from their beliefs) and philosophically (it seems strange to say that an agent believes a proposition if it takes it several minutes of intense reasoning to determine that it does). Therefore it is necessary to consider various kinds of 'limited reasoning', that is, reasoning in which only a subset of the potential applications of reasoners are actually carried out. There are two kinds of limitation - those in which the set of beliefs is restricted and those in which the limited time available is taken into account. The latter is considered in chapter 6. Concerning the former, we will consider the ideas of implicit and explicit belief and of local reasoning.

5.2.1   Implicit and explicit beliefs

Levesque (1984) distinguishes between implicit belief and explicit belief, the former being what we have just called 'belief' and the latter being a belief which the agent considers relevant or which has been 'activated' by the agent. Explicit beliefs are the effective beliefs of the agent (effective in the sense that they may be used to reason with to solve problems). The set of implicit beliefs provides an upper bound on the set of explicit beliefs: no agent can explicitly believe more than it implicitly believes. Some say that the explicit beliefs describe a 'realistic' agent and the implicit beliefs an 'ideal' agent, though there is little ideal in being thoroughly inefficient. Others, of a psychological orientation, draw analogies with working memory and long-term memory (which we will not pursue here).

The logic uses two modal operators, `I-believes` and `E-believes`. Implicit belief is modelled by possible world semantics (as above) but explicit belief requires a modified version. Levesque adopts the idea of a 'situation' from situation semantics (Barwise and Perry, 1983), which is not the same as the 'situation' in section 4.4. In this kind of situation a proposition may be true, false, both or neither. A complete, coherent situation, that is, one in which all propositions are true or false, corresponds to the standard possible world, but we may also have a 'partial world' in which a proposition may be neither true nor false and an 'incoherent world' in which it may be both.

By specifying a semantics similar to that for relevance logic, it can be shown that explicit belief does not suffer from the problem of logical omniscience. For example, the following formulae are all satisfiable:

```
E-believes(a,ϕ) & E-believes(a,ϕ∅ψ) & ~E-believes(a,ψ)
E-believes(a,ϕ) & E-believes(a,~ϕ)
E-believes(a,ϕ & ~ϕ)
~E-believes(a,ϕ ∨ ~ϕ)
```

The properties of explicit belief follow from the incoherence and incompleteness introduced in situations. The former leads to the possibility of believing unsatisfiable propositions and the latter to the possible lack of belief of valid propositions. In this way, it is possible to define formal logics to express some aspects of the inconsistencies and incompletenesses which students display. However, many subtle issues remain. For example, if reasoning is considered to be carried out with respect to the situations thought possible by the agent, is it reasonable to allow incoherent situations as being possible? Also, the effect of imperfect reasoning in classical logic is achieved by assuming perfect reasoning in a non-classical logic, which seems rather odd.

Fagin and Halpern (1987) attempt to isolate the advantages given by Levesque's notion of incomplete situations by defining a syntactic *awareness* function. The intuition is that an agent cannot believe a proposition if it is not aware of it. Instead of using incoherent or partial worlds, Fagin and Halpern use standard possible worlds with the awareness function to filter out those formulae of which the agent is unaware. A world `w` supports the truth of `E-believes(a,p)` if all the worlds the agent considers possible in `w` support the truth of `p` relative to the set of primitive propositions of which the agent is aware in world `w`. Implicit belief is as before and explicit belief is similar to Levesque's except that (1) an agent's set of explicit beliefs is closed under implication and (2) an agent cannot hold inconsistent explicit beliefs. For example, `E-believes(a,p & ~p)` is not satisfiable.

Consider the following belief-set:

```
E-believes(c,Foreigner(s))
E-believes(c,Foreigner(x) ∅ ~Aware(x,sconce))
E-believes(c,Prerequisite(sconce,college-etiquette))
E-believes(c,~Aware(s,y) & Prerequisite(y,z) ∅ ~Infer(s,z))
```

i.e. the system explicitly believes the student to be a foreigner, that all foreigners are unaware of the concept of a sconce (a fine imposed at Oxbridge), which is a prerequisite for understanding Oxbridge college dining etiquette, and that a student cannot infer propositions if she is unaware of a prerequisite concept. In such a case the system cannot show that the student can infer `college-etiquette` because she is unaware of a prerequisite concept (`Infer` is a 'meta-level predicate' which is considered further in chapter 6).

A major benefit for student modelling purposes offered by this logic of awareness is that, as we see above, it allows nested beliefs, which Levesque's logic of implicit belief does not. Fagin and Halpern's logic of *general awareness* allows the modal operator `Aware` to apply to non-primitive propositions. As an agent explicitly believes a proposition if it implicitly believes it and it is aware of it, we have:

```
E-believes(a,ϕ) + I-believes(a,ϕ) & Aware(a,ϕ)
```

This version of explicit belief retains many of the properties of implicit belief, relativised to awareness. For example, the rules and axioms R_1, A_2 and A_3 in the weak S4 logic become:

R_1	necessity	`p => Aware(p) ∅ E-believes(p)`
A_2	distribution	`E-believes(p) & E-believes(p->q) & Aware(q) ∅ E-believes(q)`
A_3	positive introspection	`E-believes(p) & Aware(E-believes(p)) ∅ E-believes(E-believes(p))`

To these general axioms, we might wish to add restrictions to provide desired properties of the logic. For example, we might specify that an agent is unaware of (any proposition that mentioned) another agent, or that an agent is aware only of a certain subset of propositions.

5.2.2 *Local reasoning*

The restriction of an agent to only a subset of propositions can be elaborated to provide a logic of *local reasoning* which differs from the logic of general awareness in that it enables an agent to hold inconsistent beliefs. The idea is that an agent's belief-set may be partitioned into a set of clusters such that any cluster is internally consistent but may contradict a different one. The idea has intuitive plausibility for it seems that we never use our beliefs about, say, nuclear reactors and football at the same time (except perhaps after reading this sentence). In AI-ED, many studies have shown that students have difficulty because they adopt an inappropriate point of view on a problem, or they switch between points of view without taking account of apparent inconsistencies between them. The formal idea of local reasoning is related to that of situation and context discussed in section 4.4.

In this logic, `E-believes(a,p)` is used to denote that agent `a` explicitly believes `p` in some 'frame of mind', and `I-believes(a,p)` to denote that agent `a` implicitly believes `p`, that is, believes `p` if all its frames of mind are pooled. This version of implicit belief satisfies axioms A_1 and A_2 of weak S4 but not axiom A_3, and the version of explicit belief is not closed under implication and therefore not subject to the problem of logical omniscience. The formula

> `E-believes(a,p) & E-believes(a,p⊃q) ⊃ ~E-believes(a,q)`

is satisfiable because `a` might believe `p` in one frame of mind and `p⊃q` in another but never be in a frame of mind where it puts these facts together. Moreover, an agent may hold inconsistent beliefs because it might believe `p` and `~p` in different frames of mind. However, agents do not believe in incoherent worlds, i.e. `E-believes(a,p & ~p)` is impossible.

As with the logic of general awareness, we may impose conditions to capture various properties. For example, Fagin and Halpern define a 'narrow-minded agent' to be one who when in one frame of mind refuses to admit it may occasionally be in another. For such an agent

> `E-believes(a,(~(E-believes(a,p) & E-believes(a,~p)))))`

is valid even though

> `E-believes(a,p) & E-believes(a,~p)`

is consistent. In addition, although the logic of local reasoning assumes an agent can do perfect reasoning within each cluster, we can add an awareness function to the structure for local reasoning to provide a model of belief which is not closed under valid implication.

Huang (1994) presents a version of limited reasoning which is intended to be more suitable for AI-ED. He uses three modal operators, `Attends`, `E-believes` and `I-believes`. The last two are as before: explicit beliefs may be inconsistent as in the logic of local reasoning, and implicit beliefs are closed under implication. `Attends(a,p)` denotes that the agent `a` is 'attending to' the proposition `p`. The set of propositions being attended to constitute a distinguished frame of mind, the one the agent is actually using. This set is defined to be logically consistent and closed under implication, unlike awareness above. These properties are captured by the following rules of inference and axioms

R_1	`p => Attends(p)`
R_2	`p & p⊃q => q`
R_3	`p⊃q => E-believes(p)⊃E-believes(q)`
A_1	`p`, where `p` is a propositional tautology
A_2	`Attends(p) & Attends(p⊃q) ⊃ Attends(q)`
A_3	`~E-believes(false)`
A_4	`Attends(p) ⊃ E-believes(p)`
A_5	`E-believes(p) ⊃ I-believes(p)`
A_6	`I-believes(p) & I-believes(p⊃q) ⊃ I-believes(q)`

Imagine that we have the following ascriptions

> `E-believes(s,Shiny(mercury))`
> `E-believes(s,Liquid(mercury))`
> `E-believes(s,Shiny(x)⊃Metal(x))`
> `E-believes(s,Liquid(x)⊃~Metal(x))`

From these we cannot, using the above rules of inference, derive

> `E-believes(s,Metal(mercury))`

or

> `E-believes(s,~Metal(mercury)`.

If we also have

> `Attends(s,Shiny(mercury))`

```
        Attends(s,Shiny(x)∅Metal(x))
```
i.e. the student is attending to the shininess but not the liquidity of mercury, then we can derive
```
        Attends(s,~Metal(mercury))
```
and
```
        E-believes(s,~Metal(mercury))
```

It should be re-emphasised at this stage that it is not the aim to develop from among this variety of limited reasoning mechanisms one which is 'correct'. This is an unattainable aim: some philosophical or computational objection can assuredly be raised against any proposed scheme. Rather, the aim is to develop a general framework within which any such scheme can be explicitly defined and theoretically analysed. In AI-ED, we have a limited agent (the computer system) reasoning about the beliefs of another limited agent (the student). The problems are complex but some formalisation may help us avoid burying assumptions within opaque algorithms.

5.3   Nonmonotonic reasoning

If a sentence ϕ is a valid conclusion from a set of premises Δ and from any superset of Δ then the inference process is said to be 'monotonic'. Resolution in predicate logic is monotonic, because a derived conclusion cannot become invalidated by the adding of further premises. Reasoning is *nonmonotonic* if the above condition does not hold, that is, if a conclusion derived from a set of premises may become invalid if further premises are added. The development of nonmonotonic reasoning in AI reflects the fact that most agents, certainly human ones, do not reason using the monotonic methods of standard logic. They draw conclusions which they are prepared to withdraw if further evidence undermines the support for them. Limited reasoning, as discussed above, is often nonmonotonic because relaxing the limitations often prevents previously-drawn conclusions from being valid, but it does not have to be as, for example, we can have limited reasoning using resolution. Nonmonotonic reasoning has also been studied in philosophy as *defeasible reasoning*, where the emphasis has been on considering the process of reasoning as one of constructing arguments and on considering how an argument may be defeated (the relation to argumentation will be considered in section 9.6).

Nonmonotonic reasoning is usually discussed with respect to a single agent, and we need to consider that aspect too. Multi-agent reasoning in AI-ED is deeply and unavoidably nonmonotonic, for the following reasons:
- The system's beliefs about the student's beliefs can never be confirmed and must therefore always be considered subject to revision. Even apparently objective facts such as students' assertions about what they know or descriptions of students' actions need to be regarded as provisional (because they may not fully understand terms they use, or they may make slips in performing tasks, for example).
- Because of the 'bandwidth problem' systems will rarely have access to sufficient data to permit reliable inferences about students' beliefs, and consequently will have to make default assumptions which may later have to be withdrawn.
- Students do occasionally learn (and forget) and therefore what the system believes of the student at one time will not necessarily hold at a later time.
- Both the system and the student will, in the cause of efficiency, have recourse to nonmonotonic reasoning when reasoning on their own behalf about the domain.

Within AI generally, the field of nonmonotonic reasoning is vast and active but there has been little explicit linking to AI-ED (some tentative explorations are reported in Kono, Ikeda and Mizoguchi (1994) and Giangrandi and Tasso (1995)). In this section, we will just summarise the main approaches and point out the potential relevance to computational mathetics. Of course, the effect of nonmonotonic reasoning may be achieved through computational techniques (such as semantic networks) which are efficient but not completely understood, rather than through the use of formal systems which are generally intractable. However, formal approaches to nonmonotonic reasoning may yield benefits in terms of clarity and correctness, and provide useful tools for specifying and describing nonmonotonic systems, in particular, for that limited class which are actually covered by relatively ad-hoc techniques.

5.3.1   Circumscription

There are two basic approaches to nonmonotonic reasoning: model-theoretic and proof-theoretic. Model-theoretic approaches are based on the idea that anything that does not follow is assumed to be false ('model' here is used in the mathematical sense: a model of a theory T is any structure M such that T is true in M). Proof-theoretic approaches use non-standard (nonmonotonic) logics to derive conclusions through inference rules.

The closed-world assumption (as used in Prolog) is the simplest example of a model-theoretic approach and the various formalisations of *circumscription* the most comprehensive. Under the closed-world assumption, the negation of any sentence ϕ which cannot be derived from Δ is assumed to be valid. In general, this assumption is too sweeping: being unable to prove something does not make it false. The closed-world assumption has certain undesirable properties - for example, if Δ = {P(a)∨P(b)} then both ~P(a) and ~P(b) follow from the assumption, giving the set

`{P(a)∨P(b),~P(a),~P(b)}` which is inconsistent. (Prolog avoids this by not allowing disjunctive premises.) In the general case, the closed-world assumption is not computable, as the proof of φ from Δ is not computable.

The basic idea of circumscription is that one considers not all models of `T` but only those which are minimal with respect to a specific property `P`. For example, if we have:

```
Δ  =   {   Metal(iron),
           Metal(copper),
           Metal(mercury),
           Liquid(mercury),
           Metal(x) & ~Liquid(x) ∅ Ductile(x) }
```

to represent the belief-set of a student who believes the facts indicated and that metals which are not liquid are ductile, then we cannot logically conclude from Δ that `Ductile(iron)`, because we do not have `~Liquid(iron)` or, more generally, that the student believes that all metals except mercury are not liquid. Applying circumscription to minimize the predicate `Liquid` we obtain the extra premise:

```
           Liquid(x) -> (x = mercury)
```

using which the conclusion `Ductile(iron)` now follows.

Circumscription is achieved by means of an axiom schema:

```
       Circum(T,P) :   T(Φ) & ∀x(Φ(x)∅P(x)) ∅ ∀x(P(x)∅Φ(x))
```

where `T(Φ)` is the result of replacing all occurrences of `P` in `T` by the predicate expression Φ, `P` is an n-ary relation and `x` abbreviates $x_1, ... x_n$. Informally, this states that if Φ satisfies the conditions satisfied by `P` and every n-tuple that satisfies Φ also satisfies `P`, then the only n-tuples which satisfy `P` are those which satisfy Φ. For the simple example above, the outcome is the same as for the closed-world assumption. More complex forms of circumscription have been defined, for example, prioritised circumscription allows the predicates which are to be minimised to be placed in an order of relative importance: this, for example, could be used to allow us to conclude `~Ductile(mercury)`, which does not follow from `Circum(Δ,Liquid)` as above. The precise relationships between the various forms of circumscription have been the subject of many studies.

Any ascription of beliefs to a student is bound to make (implicit) use of some circumscription-like scheme as it would be unreasonable to require explicit representation of all that which the student does not believe. However, formally, the matter is more complex than even circumscription can handle. For example, if there is no proposition of the form:

```
       B(c,B(s,p))
```

then is the missing default assumption

```
       ~B(c,B(s,p))
       B(c,~B(s,p))
or     B(c,B(s,~p)),
```

all of which mean subtly different things? The four possibilities almost correspond to the values 'true', 'false', 'unknown' and 'fail' of the four-valued logic suggested by Ikeda and Mizoguchi (1994) (discussed in section 8.2.3).

5.3.2   Default logics

The application of circumscription results in further predicate logic expressons and therefore allows the usual theorem-proving mechanisms to stay unchanged. Proof-theoretic approaches to nonmonotonic reasoning, on the other hand, include within the logic extra rules which allow nonmonotonic inferences to be drawn. For such logics, a new semantics must be defined. These nonmonotonic logics focus on the notion of normality, that is, on rules which tend to apply unless there are exceptions.

For example, *default logic* allows a theory to contain ordinary sentences and defaults, i.e. expressions of the form

```
       p : q ∅ r
```

which may be read as "if `p` is derivable and the sentence `q` is consistent (i.e. its negation is not provable) with the theory, then the default rule is applicable and the conclusion `r` may be drawn." This is nonmonotonic because a sentence `q`, previously consistent with a theory, may become inconsistent if new sentences are added to the theory. The two most common forms of default are where `q = r` or `q = r & s`, for example:

```
       Physics-graduate(s) : Knows-about-momentum(s) ∅ Knows-about-momentum(s)
       Metal(x) : ~Liquid(x) & ~Eq(x = mercury) ∅ ~Liquid(x)
```

(Note that although we have used the same symbol ∅ to denote nonmonotonic inference as monotonic inference, they have very different properties. For example, in monotonic inference if `p∅q` and `q∅r` then we can infer `p∅r`, but this is not so for nonmonotonic inference: consider p='Fred is a schizophrenic', q='Fred is an adult', r='Fred is employed'.)

Such defaults act as rules of conjecture, allowing inferences which would not be possible under ordinary logic rules. The conclusion has the status of a belief which may need to be withdrawn if the assumption becomes inconsistent. The potential circularity (arising from the fact that what is provable in a default logic both determines and is determined by what is not provable) is avoided by requiring that an *extension* of a particular theory (1) contains all the

known facts, (2) be closed under the implication rules, and (3) contains the consequents of all defaults which apply within that extension. In general, there are many possible extensions for a given default theory. Informally, an extension describes an acceptable set of beliefs that an agent may have about an incompletely specified world and therefore is similar to the concept of a possible world. As determining whether a formula is within an extension is undecidable, the implementation of default logic seems problematic, although for most practical cases (such as the kinds of default illustrated above) implementations are possible. For example, the LNT system (described in section 6.3) has an underlying default logic written in Prolog.

As an example, consider the following set of two default rules and three premises:

```
Δ  =  {   Shiny(x) : Metal(x) ∅ Metal(x),
          Liquid(x) : Nonmetal(x) ∅ Nonmetal(x),
          Shiny(mercury),
          Liquid(mercury),
          ~(Metal(x) & Nonmetal(x)) }
```

One extension is to apply the first default rule to infer `Metal(mercury)` which then inhibits application of the second default rule. Or we could apply the second rule to infer `Nonmetal(mercury)` and so inhibit the first rule. In this case, there is no basis for preferring one extension over the other (so perhaps it is best to infer nothing).

5.3.3   Autoepistemic logics

Default logic involves an agent reflecting upon its own knowledge, in particular, to consider whether a proposition is consistent with what it believes. It enables the expression of arguments such as 'If I had a brother then I would know it and as I don't know it, then I do not have one', which seems reasonable except for the more lurid soap operas. The notion of consistency is, however, outside the language of default logic. We could instead attempt to capture the notion within the logic, as is done in an *autoepistemic logic*. For example, we could rephrase:

```
p : q ∅ r
```

by the sentence:

```
B(a,p) & ~Infer(a,~q) ∅ B(a,r)
```

i.e. "if p belongs to the agent's belief-set and ~q does not follow from it, then r is believed." Not surprisingly, given this translation, various formal equivalences between variations of default logic and autoepistemic logic can be established (Konolige, 1988; Lin and Shoham, 1992). The specific version:

```
~Infer(a,p) ∅ B(a,~p)
```

is a kind of extension of the .introspection axioms of modal logics, involving the meta-level predicate `Infer`, to be considered in chapter 6. In general, an autoepistemic logic enables an agent to make inferences on the basis of its own knowledge or beliefs (or ignorance). The main point here, however, is that we have a link through modal logics of belief to other aspects such as limited reasoning, as discussed above, and a prospect of being able to tie together all the threads, in due course.

Unfortunately, as discussed by Levesque (1990), this link is weakened by the fact that work on autoepistemic logics and default logics has defined different semantics for the notion of belief to that used in belief logics. Levesque attempts to overcome this by defining a second modal operator, in addition to `Believes`, namely `Only-believes`, such that `Only-believes(a,p)` is to be read as "p is *all* that is believed by a" or perhaps "*only* p is believed by a", and then by developing a semantics for a language with two such operators. Subsequently, this operator is modified to `Only-believes(a,p,n)` for "only p is believed by a about n", just as circumscription takes place with respect to a predicate and not the whole belief-set.

5.3.4   Multi-agent nonmonotonic reasoning

As mentioned above, almost all the work on nonmonotonic reasoning has considered only the single-agent case. As Lakemeyer (1993) remarks, "This focus on single agents is somewhat surprising, since there is little doubt that agents, who have been invested with nonmonotonic reasoning mechanisms, should be able to reason about other agents and their ability to reason nonmonotonically as well." Multi-agent nonmonotonic reasoning arises naturally in AI-ED, where we have at least two agents (program and student) reasoning nonmonotonically about one another, but knowledge representation researchers illustrate their ideas with simple problems involving Jack, Jill, birds and penguins.

Lakemeyer presents an axiomatization of multi-agent nonmonotonic reasoning based on Levesque's 'only logic', mentioned above. An example can be re-expressed in AI-ED terms. Let a computer program c believe φ and suppose c makes the following assumption: unless I believe that a student s believes φ assume that s does not believe it. Then if this assumption is all that c believes then it indeed believes that s does not believe φ. Formally,

```
Only-believes(c,(~B(c,B(s,φ))∅ ~B(s,φ))) ∅ B(c,~B(s,φ))
```

is derivable from the axioms of the logic. The complex technical machinery apparently needed to derive such an intuitively obvious conclusion is perhaps a warning to AI-ED that ad-hoc mechanisms are likely to have theoretical shortcomings.

Clearly, difficult technical issues remain and most work in this area continues to focus on the detailed properties of different formalisations rather than on considering possible applications to areas such as AI-ED. However, some general implications for computational mathetics may be listed:

- Although some formalisations are theoretically intractable, practical implementations for realistic applications are possible and are beginning to be developed (Donini et al, 1990). The original motivation for nonmonotonic reasoning, to enable agents to jump to conclusions, i.e. to reason faster, has yet to be satisfied by the rather complex formalisations so far developed.
- As nonmonotonicity often arises through an agent reasoning about its (or other's) beliefs, the meaning of the belief operator depends on the context (i.e. the other beliefs). `Believes` functions as an indexical and expressions should ideally be indexed. Formally, this is an aspect which has not been considered much.
- Nonmonotonicity is intimately related with other aspects of computational mathetics: for example, determining which belief(s) to withdraw (section 7.2.4) is likely to depend to some extent on which beliefs were derived by ordinary inferences and which by defaults; similarly, limited reasoning may be achieved by not reflecting too deeply about a set of beliefs but by deriving default assumptions.
- We can use logical notations to describe nonmonotonic reasoning without making any psychological claims.
- To repeat a general point, formal characterisations of nonmonotonic reasoning begin to provide us with a way of precisely describing and analysing aspects of AI-ED which at present are proposed, described and implemented in an ad-hoc manner.

5.4   Reasoning with inconsistent knowledge

The previous section considered the problem of deriving a possibly unreliable conclusion from a set Δ of premises, assumed reliable. Alternatively (or perhaps as well), we could view the problem as one of deriving a reliable conclusion from a possibly unreliable set of assumptions, assumed true until a contradiction is detected. In AI-ED, for example, our sources of information are not fully reliable. Our beliefs about a student derive from statements of the student, analysis of her problem-solving (complicated by the presence of slips), and perhaps from a teacher. Even experts do not always agree. Therefore, it is quite likely that a realistic Δ will be potentially inconsistent. In ordinary logic any conclusion follows from inconsistent premises.

Roos (1992) describes a logic for reasoning with inconsistent knowledge, whose basic idea is quite simple and can be illustrated by adapting the example from section 5.3.2. We have four assumptions

```
Δ  =  {  1.  Shiny(x) ⊘ Metal(x),
         2.  Liquid(x) ⊘ ~Metal(x),
         3.  Shiny(mercury),
         4.  Liquid(mercury) }
```

from which we can derive `Metal(mercury)` (from `1` and `3`) and `~Metal(mercury)` (from `2` and `4`), which is contradictory. To restore consistency, one of the assumptions has to be rejected (in this case, any of the four could be rejected). To enable an assumption to be selected, a 'reliability relation' `<` is defined, so that (in the simplest case) the least reliable assumption is rejected. Here, we might have `1<2<3<4`, in which case assumption `1` will be rejected and we will conclude `~Metal(mercury)`.

In hypothetical or counterfactual reasoning an agent proceeds by making an assumption believed to be false - in the above terms, it is given a low reliability. Then when, as hoped, a contradiction is derived the least reliable assumption is rejected as false, as originally intended.

In general, it is necessary to consider how the reliability of a derivation depends on the reliability of its assumptions (it may not always be best simply to discard the weakest link). Also, as the retained assumptions are still only assumptions they may later be rejected - in which case, it may be advisable to reinstate an assumption previously rejected for being less reliable (for example, if we later reject assumption `2` then maybe `1` should be reinstated). This is a standard concern in belief revision.

5.5   Probabilistic reasoning

The idea of a reliability relation naturally suggests associating with each sentence a number which is a measure of the 'reliability' of that sentence. Similarly, the various approaches to nonmonotonic logic seem to have some statistical basis. For example, the default assumption "Metals shine" seems to be related to the empirical fact that the large majority, say 99%, of metals actually do shine. However, nonmonotonic logicians have insisted that, in situations where nonmonotonic reasoning is necessary, "statistical reasoning has no part to play whatsoever" (Reiter, 1987a). These situations are apparently ones in which the default assumption captures some communicative convention to the effect that, in this case, whenever a metal is mentioned, unless something explicitly to the contrary is stated, we are expected (or required) to infer that it shines. In nonmonotonic reasoning, when an agent believes something (whether by default assumption or not) it believes it totally, even though it might need to be retracted later.

Still, such a convention is unlikely to be adopted if it flouted statistical reality and there are certainly other contexts where it is appears to be necessary to reason directly about the statistical likelihood or probability of sentences. Historically, AI has been very reluctant to approve the use of numbers for probabilistic reasoning. AI was conceived with the realisation that computers could process complex, 'meaningful', symbolic structures, not just numbers: it seems a retrograde step to resort to the latter. The AI-ED field has been less reluctant, as education itself is fixated with the need to provide numbers. Overall, though, the desirability of using numbers in reasoning remains controversial.

Intuitively, it seems that we attach a 'degree of strength' to our beliefs. I am quite sure that Beethoven wrote Fidelio and less sure that he wrote Prometheus - I am even less sure that he wrote both. If pushed, I could attach numbers to my confidence: I am 90% sure that Beethoven wrote Fidelio, and so on. First, we have to consider how to represent such statements. Attaching a probability to each sentence:

        `Composer(Fidelio,Beethoven,.9)`

is not adequate: where would we put the number in a sentence such as `Metal(x) ⊘ Shine(x)`? We seem to need an expression such as:

        `Prob(Metal(x) ⊘ Shine(x),.75)`

where `Prob` is a modal operator, as it takes a sentence as an argument. Like other modal operators, `Prob` would be referentially opaque. However, because the probability of a sentence is not an objective characteristic of that sentence but a property associated with it by an agent, we can dispense with `Prob` and simply extend the `Believes` operator:

        `B(a,Metal(x) ⊘ Shine(x),.75)`

As before, we can nest such expressions:

        `B(c,B(s,Metal(x) ⊘ Shine(x),.75),.95)`

i.e. the program is very confident (.95) that the student is fairly sure (.75) that metals are shiny. Sometimes we may prefer to specify a range rather than a precise number:

        `B(a,Metal(x) ⊘ Shiny(x),[.75,1])`

The basic approach of probabilistic reasoning is to define how the probability of composite sentences depends on the probabilties of its constituent sentences, just as the truth value of an ordinary predicate logic sentence is defined (section 4.2). Predicate logic can be regarded as a special case of probabilistic logic where all probabilities are 0 or 1. Unfortunately, as we should suspect from the general discussion of modal logics in section 4.3, the probability of ϕ & ψ does not depend only on the probabilities of ϕ and ψ. For example, if we have

        `B(a,Short(Fred),.3)`
        `B(a,Policeman(Fred),.05)`

then we cannot calculate the probability `p` such that

        `B(a,Short(Fred) & Policeman(Fred),p)`

without knowing (what `a` believes about) the joint occurrence of being short and a policeman.

Most schemes for carrying out probabilistic reasoning therefore proceed by defining axioms for calculating probability-like measures. The field is large and contentious, as intuitions about probability are difficult to pin down. We will briefly mention three examples:

- The MYCIN expert system (Shortliffe, 1976) uses 'certainty factors' ranging from -1 to +1 rather than probabilities and combination rules such as:

    `CF(φ & ψ) + minimum(CF(φ),CF(ψ))`

    `CF(ψ) + CF(φ) x CF(φ⊘ψ)`

- Probabilistic logic (Nilsson, 1986) uses bounds on probabilities because in general that is all that can be calculated given probabilities of constituent sentences. However, if the bounds of `p` and `q` are [.5,1] then the bounds of `p&q` would be [0,1], as it would if the original bounds were [0,1] - thus, the likely distribution of the probability is not captured.
- Fuzzy logic (Zadeh, 1987) differentiates between probability and the gradual membership of a set. For example, we may say that the probability that it will be sunny tomorrow is .4 and that today is to a greater or lesser extent a member of the set of the sunny days. Most concepts, it is argued, do not correspond to relations which map constants onto the values true or false but have vaguely defined boundaries. Fuzzy logic defines ways of determining the degree of membership for composite concepts.

These and many other schemes are supported by comprehensive theories which should clarify when each scheme is appropriate, for the choice between them is not arbitrary, as they provide different conclusions.

In AI-ED, we find, as usual, that the terms are sometimes adopted but rarely the theories that go with them. For example, Derry and Hawkes (1993a) use 'informal fuzzy reasoning' to represent how a student solves algebra word problems. Whether or not a student's solution attempt has a particular feature is indicated by a number in the range [0,1], or more precisely by one of seven values in that range (i-1)/6, i=1..7, given a linguistic gloss as {`definitely not x, not x, rather not x, neither x or not x, rather x, x, definitely x`}. Similarly, Katz, Lesgold, Eggan and Gordin (1994) represent the degree to which a student possesses a skill or concept by using a 'fuzzy variable' which can take one of the values {`no knowledge, limited knowledge, unautomated knowledge, partially automated knowledge, fully developed knowledge`}. More precisely, a

probability is assigned to each of these five values. For example, the student's ability to use a handheld meter might be represented by the vector (0,0,.3,.5,.2). The vector for a global skill (such as test equipment usage) is determined by a weighted polynomial of the vectors representing its component skills (such as the ability to use the oscilloscope, digital multimeter, and so on). The properties and benefits of such a scheme are not formally stated.

5.5.1 Bayesian networks

Naturally, in the face of all these proposals, there has been a vigorous rearguard action from those familiar with long-established probability theory. Its current adaptation, Bayesian networks, has become more than a rearguard: it is now the standard approach to uncertain reasoning (Pearl, 1988; Charniak, 1991). Bayesian networks are based on the familiar Bayes' law:

```
Prob(p|q) = Prob(p) * Prob(q|p) / Prob(q)
```

where `Prob(p|q)` is the probability of p given that q holds. This follows immediately from the axiom that

```
Prob(p & q) = Prob(p) * Prob(q|p) = Prob(q) * Prob(p|q)
```

In passing, we note that we can use this notation to express default rules that may be overridden:

```
Prob(~Liquid(x)|Metal(x)) = high
Prob(~Liquid(x)|Metal(x), x=mercury) = low
```

In fact, standard probabilistic reasoning is nonmonotonic in the sense that it may deliver different probabilities as new evidence is acquired.

AI and AI-ED differ from the kinds of context in which classical probability theory is usually applied in that they are not concerned with events (such as coin tossing) which are repeatable and about which statistics may be gathered. In AI-ED we are unlikely to estimate the probability that a student will answer a particular problem correctly by asking her to solve that same problem 100 times. Instead, we have to use 'subjective probabilities', that is, estimates of probabilities based on (unreliable) evidence. This, together with the fact that very large numbers of such estimates have to be made to tackle realistic problems, causes some to doubt the usefulness of probabilistic approaches. Also, it is hard to ensure that such probability estimates are consistent with one another.

This last shortcoming of classical probability is overcome with Bayesian networks, which are guaranteed to be consistent. A Bayesian network is a directed acyclic graph, a simple example of which is shown in Figure 5.1. The nodes are variables representing states of affairs and taking the values true or false, in the simplest case. The link points from a 'cause' to an 'effect'. For example, the state of affairs of a student being able to subtract (`sub`) causes the state of affairs where a student can solve a particular problem (`ssub`). The causal connections are not intended to be complete - for example, a student may be able to subtract and multiply and still not be able to long-divide.
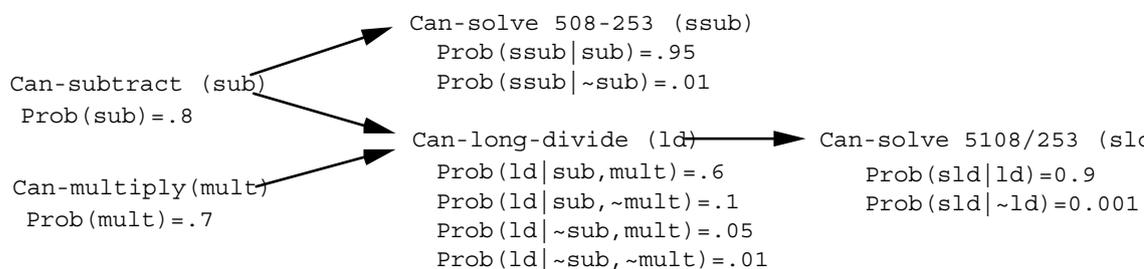
Figure 5.1. A simple Bayesian network

A Bayesian network can be used both to predict (for example, whether a student will be able to solve a long-division problem) and to diagnose (for example, to determine the probabilities of the student being able to subtract and multiply after observing that she has failed to solve a long-division problem). This is an advantage over logic-based methods, where it is sometimes necessary to have rules in both directions, for example, we have used both the rules: `Metal(x)` ∅ `Shine(x)` and `Shine(x)` ∅ `Metal(x)`.

A Bayesian network must be initialised with relevant a priori probabilities of the root nodes and the conditional probabilities of their successors (as shown in Figure 5.1: for example, the probability that the student will solve the given subtraction problem even if they cannot subtract is .01 - a guess, presumably). The structure of the network specifies which states of affairs are directly causally related to others, and therefore, by omission, which states are independent of one another and hence for which conditional probabilities do not need to be specified. For example, being able to multiply (`mult`) is not directly related to `ssub` and so `Prob(ssub|mult)` is not needed.

As evidence is acquired that, say, a student can solve the subtraction problem but not the long-division one, then probabilities associated with the adjacent nodes can be recalculated and so new values propagated through the network. The algorithm is based on Bayes' theorem and is given in full in Pearl (1988), along with a thorough

analysis. The general algorithm for arbitrarily complex graphs is intractable, but efficient methods exist for simply connected graphs and various approximate methods have been devised for others.

Proposals and preliminary attempts to apply Bayesian networks to AI-ED are described by Villano (1992), Sime (1993) and Katz, Lesgold, Eggan and Gordin (1994).

5.6   Qualitative reasoning

As Pearl (1988) remarks, "Although probabilities are expressed in numbers, the merit of probability calculus rests in providing a means for articulating and manipulating qualitative relationships that are found useful in normal discourse." As we saw in the previous section, some researchers prefer to attach qualitative labels (such as `rather` and `definitely`) to points or ranges on the probability scale. It is a natural step to define reasoning schema which deal directly with such qualitative labels: "if she knows very little about `x`, then she will almost certainly be unable to solve problem `y`"; "if we run faster in the rain, then we get less wet", and so on.

Qualitative reasoning is important in AI-ED for two main reasons. First, qualitative reasoning is an integral part of expert problem-solving. It is not just that students, who do not know the quantitative formulae emphasised in school curricula, have to resort to qualitative reasoning during an unfortunate, preliminary stage before they learn 'better'; it is more to the point that experts use qualitative reasoning both before and during problem-solving in order to determine what kind of quantitative analysis is appropriate and whether it is proceeding satisfactorily. Therefore, even in apparently quantitative domains like physics and maths, students need to develop abilities in qualitative reasoning and AI-ED systems need to perform qualitative reasoning in order to provide comprehensible explanations and to diagnose students' problem-solving behaviour.

The second main reason derives from the fact that many AI-ED systems are based on simulations of some quantitative process. These simulations are often very complex software systems, which perhaps existed before an AI-ED system was built around them. Students cannot be expected to be able to understand the detailed calculations carried out by such simulations: they need more qualitative explanations to be provided. Of course, it is difficult for a system to integrate and coordinate the quantitative and qualitative aspects of such simulations.

The field of qualitative reasoning in AI, which was initiated by limitations of the SOPHIE tutoring system (de Kleer and Brown, 1981), has been almost entirely concerned with reasoning about the physical world. Qualitative reasoning is even more prevalent in some of the social sciences, where quantitative theories are less developed - for example, in economics ("if inflation rises, then unemployment increases") and sociology ("if unemployment rises, then crime increases") - but so far AI's qualitative reasoning techniques have not been applied much to these domains.

Unlike the field of nonmonotonic reasoning, where most of the research effort goes on making detailed comparisons of the various proposals, in qualitative reasoning the different approaches seem to be pursued in parallel and it is quite difficult to determine exactly how they relate to one another and what their particular benefits are. Even a superficial review would be too comprehensive for our purposes (the original papers describing the main methods are de Kleer and Brown (1984), Forbus (1984) and Kuipers (1986); Bredeweg and Winkels (1994) gives a brief summary of these methods; Weld and de Kleer (1989) is a collection of papers on qualitative reasoning). For illustrative purposes, we will simply give a short summary of Qualitative Process (QP) theory, selected because it is the only one of the main theories which its developer is attempting to apply to AI-ED (Forbus, 1991).

QP theory focusses on the *processes* in a system (for example, the flow of heat from one place to another), rather than, say, its physical components. A process is said to be active (or not) in a situation. For a process to be active certain preconditions have to hold, as indicated by this rule:

```
Holds(connected(x,y),t) & Value(temp(x),t)>Value(temp(y),t) ∅
            ∃p Process(p,heat-flow(x,y)) & Active(p,t)
```
A process influences parameters of the system, for example, heat flow from `x` to `y` changes the amount of heat of `x` and `y`:

```
Token(p,heat-flow(x,y)) & Active(p,t) ∅
            Value(influence(p,heat(x),t)<0 & Value(influence(p,heat(y),t)>0
```
Some parameters are directly influenced by processes, others only indirectly (through other parameters). For example, the temperature of a body is related to the heat it contains. In QP theory such relations are expressed as qualitative proportionalities: `R1∝R2` means that, other things being equal, an increase in `R1` will cause an increase in `R2`. For example,

```
temperature(x)∝heat(x)
```
(cf. the naive mental models of heat and temperature outlined in section 4.6).

From the description of processes a system's behaviour from an initial state can be predicted. The applicable processes and their influences are determined, and these are propagated through the system, giving a new situation, and so on,

until no further changes occur. QP theory is being applied to develop 'self-explanatory simulations' (Forbus, 1991) which combine qualitative and quantitative representations to generate predictions and explanations of a system's behaviour. The aim is to enable such simulators to be built automatically by expressing the relevant domain theory in the QP language. Then explanations would follow from the QP interpretation, in terms of the causal links between processes and influences.

Other applications of qualitative reasoning to AI-ED include:
- The QUEST system (White and Frederiksen, 1990), mentioned in section 2.2.3, where the main emphasis is on the student's progression through a series of domain models (from qualitative to quantitative) by the judicious choice of problems to avoid the development of misconceptions.
- The QMaPs theory (Van Joolingen, 1995), in which a system is described by specifying relations or constraints that hold between system parameters. For example, a water tank with a constant inflow and an outflow that monotonically increases with the water level would be described by the following relations:
    ```
    deriv(level,time,netflow)
    sum(netflow,outflow,inflow)
    constant(level,inflow)
    M+(level,outflow)
    ```
    As with QP theory, such descriptions form runnable models which can generate explanations and predictions and be integrated with quantitative processes.
- The Sepia system (Ploetzner, 1995), which emphasises the complementary roles of quantitative and qualitative representations. The qualitative knowledge that when there is a non-massless body on a plane then there is a normal force on the body due to the plane is written in Sepia's notation (simplifying a little) as:
    ```
    Instance(x,body)) & ~Massless(x) & Instance(y,plane) & On(x,y) ∅
                        Instance(force(x,y,fn),normal-force)
    ```
However, in a pattern that is becoming familiar, these beguiling new notations are not precisely defined and related to the longer-established qualitative reasoning theories developed in mainstream AI. It is not clear why the established theories are considered inadequate for AI-ED purposes and what benefits the new notations provide.

5.7   Reasoning about time and action

Methods for qualitative reasoning such as QP theory 'quantise' time by assuming that during a process the system passes through qualitatively identical states. This is inadequate if it is necessary to regard time as continuous, for example, to reason about the duration of processes or about whether one event happens before or after another. It is also inadequate if there are actions, external to the system, which might affect predicted outcomes.

The first problem has led to the development of the field of *temporal reasoning*. In this case, it is not clear that temporal reasoning has specific relevance to AI-ED (except in domains where students have to reason about when something will happen, for example, in medical diagnosis, to determine when a drug should be administered). Therefore, we will content ourselves with referring the reader to a review of temporal reasoning methods (Vila, 1994) and commenting that the standard methods use techniques with which we have become familiar:
- The use of extra arguments in predicate logic expressions to indicate the time or period when a term holds:
    ```
    Composer(Fidelio,Beethoven,1805)
    ```
    As with other special arguments like situations and probabilities, a difficulty is that the arguments are not special at all according to the syntax and therefore nonsensical expressions can easily be written.
- The reification of temporal terms:
    ```
    Holds(composer(Fidelio,Beethoven),1805)
    ```
    with the advantage mentioned before that it is possible to predicate and quantify over such terms.
- The introduction of further modal operators:
    ```
    Was(Composer(Fidelio,Beethoven))
    ```
    As before, for each temporal operator introduced, a semantics has to be defned.

The second problem is of more obvious relevance to AI-ED because we have agents performing actions intended to effect predicted outcomes. Qualitative reasoning assumes that events will unfold 'uneventfully' to the predicted conclusion, but we have agents whose aim is to change this predicted course of events. AI-ED systems only function in contexts where predictions indicate that some action is needed, i.e. something has not yet been learned. (Interestingly, young children's qualitative reasoning is more in terms of agents performing actions to affect the state of the world than in terms of events unfolding naturally. For example, when asked about the wind, they might say that it is caused by the trees waving at us. We continue to be inclined to attribute otherwise mysterious events to the actions of hidden agents.)

The standard approach to dealing with actions, based on the situation calculus, was introduced in section 4.4 and will be discussed further in later chapters. For the moment we just point out the relationship of the notorious *frame problem* to nonmonotonic reasoning. Previously, we had a rule:
```
~Holds(knows(s,φ),t) ∅ Holds(knows(s,φ),tell(teacher,s,φ,t))
```
The frame problem concerns the difficulty of inferences such as:

$$\text{Holds(knows(s,ψ),t)} \oslash \text{Holds(knows(s,ψ),tell(teacher,s,φ,t))}$$

i.e. that a student continues to know what she knew before a teacher tells her something else. Common sense suggests that almost all of an agent's knowledge and beliefs carries over from one situation to another. Of course, it is impractical to list explicitly all the knowledge and beliefs which persist - an agent makes default assumptions that they do:

$$\text{Holds(p,t) \& ~Abnormal(action,p,t)} \oslash \text{Holds(p,action(t))}$$

i.e. if p holds in situation `t` and there is nothing abnormal about an action then p will continue to hold in situation `action(t)`. For example, in the case of belief, an abnormal action could be one which leads the agent to hold a contradictory belief. Similarly, we assume that the absence of a belief persists by default: if an agent does not believe something in a certain situation then it will only come to believe it through some abnormal action (for example, one which causes it to learn it). Needless to say, the frame problem continues to be a source of great theoretical and practical difficulty (Ford and Hayes, 1991).

5.8   Diagrammatic reasoning

So far in this chapter it has been argued that computational mathetics will need to capitalise on various areas of mainstream AI (nonmonotonic reasoning, probabilistic reasoning, qualitative reasoning, etc) which have been relatively neglected in AI-ED. Although these areas still have many open problems and have not yet addressed some aspects which computational mathetics needs, a promising degree of theoretical consensus has been developed. Inevitably, though, computational mathetics will need more than AI has yet been able to provide a foundation for. One such topic concerns an agent's ability to reason with diagrams, icons, figures, maps, graphs, illustrations, pictures, and so on, which we will call *diagrammatic reasoning*. With the increased use of graphical interfaces and multimedia in AI-ED systems, the role of diagrammatic reasoning is becoming much more important.

Philosophers and psychologists have long been interested in the nature of mental imagery but in AI consideration of diagrammatic reasoning has been relatively neglected compared to the symbolic, logicist view of reasoning. An obvious difficulty is indicated by the fact that the various kinds of diagrams are often referred to as 'external representations', that is, they are external to a computer system which therefore has no satisfactory internal representation of them. To be sure, any diagram can be represented in, say, predicate logic, but a system's facility in reasoning with those representations scarcely matches that of human ability at diagrammatic reasoning (whatever that means, precisely).

Diagrammatic reasoning is rich and ubiquitous. It ranges from the use of simple conventions (for example, that the length of a line represents the duration of some activity, or that things tend to 'flow' from top to bottom or left to right of a page) to the use of complex, special-purpose diagrams (for example, circuit diagrams for equipment, visual programs, and architectural drawings). Moreover, current computer systems, with their capability for animation, user-controlled zooming, annotated overlays, and so on, provide much more than the equivalent of a drawing.

At this stage, it is not clear to what extent an AI-ED system will need itself to have some facility at diagrammatic reasoning. It is possible that it will be sufficient for AI-ED system designers to be so well-versed in the properties of diagrammatic reasoning that they will be able to design systems which catalyse in the student precisely those diagrammatic reasoning abilities which are needed. This seems rather optimistic for various reasons:

- Diagrammatic reasoning ability is not entirely innate; it is largely a learned ability. Students, being students, may not yet have learned the requisite ability. For example, a student trying to solve an optimisation problem using PERT networks may have difficulty through misinterpreting the semantics of the diagram, rather than through misunderstanding the problem itself. An AI-ED system which aimed to give advice to such a student could hardly do so without itself being able to monitor diagrammatic reasoning performance.
- As there are individual differences in students' diagrammatic reasoning ability, it is likely to be necessary for systems to be able to dynamically generate appropriate diagrams rather than simply present pre-specified ones.
- If a student is not merely to 'receive' a diagram from a system but is to be actively engaged in constructing her own diagrams (as constructivists would prefer, despite the fact that at present many of them are engaged in designing presentational multimedia systems) then it seems likely to be beneficial if the system could itself understand, i.e. reason with, those diagrams.
- Diagrams differ greatly in their appropriateness for a particular problem. A system ought to be able to detect when a student has adopted an inappropriate diagram and to give some explanation of why it is inappropriate.

These arguments echo the general ones for the use of explicit representations in AI-ED. It is much too early to say whether they will be adequately achievable in the case of diagrammatic reasoning. At the moment, only speculative comments on how diagrams facilitate reasoning and problem-solving are possible.

The most patent and potent characteristic is that diagrams directly indicate certain relations (such as proximity and relative size) so that inferences involving those relations are so immediate that they barely seem like inferences. What relations are 'directly indicated' must be learned - for example, of the three complex diagrams mentioned above, relative size is important in architectural drawings, but much less so in visual programs or circuit diagrams.

Diagrammatic reasoning may involve re-drawing the diagram and adding symbolic annotations. The latter indicates that an adequate description of diagrammatic reasoning must explain how it is integrated with symbolic reasoning (as usual, there is no hard-and-fast distinction to be drawn).

Diagrammatic reasoning can be used for problems which are not intrinsically spatial. One may solve a problem expressed in natural language by representing the problem statement as, for example, Venn diagrams, then manipulating the diagrams, and converting the 'answer representation' back into natural language. In general, then, the process involves adopting some diagrammatic representation which can be reasoned with much more easily than (and converted into and from) the original representation. The relevant non-visual relations of the problem statement need to be mapped onto directly indicated relations in the diagram (such as membership in the case of Venn diagrams).

The reasoning steps carried out during diagrammatic reasoning may or may not be comparable with those which might be carried out in the original or any other representation. For example, a Venn diagram proof may differ from a predicate logic one. It may be that there are certain kinds of symbolic reasoning process which it is considered important for a student to master and for which we can try to design analogous diagrammatic reasoning processes which are relatively simple for a student to understand. This is the motivation for what Merrill and Reiser (1994) call 'reasoning-congruent' learning environments. Other systems (for example, GEOMETRY (Anderson, Boyle and Yost, 1985); AlgebraLand (Foss, 1987); BRIDGE (Bonar and Cunningham, 1988)) have used proof-like diagrams to enable students to plan and monitor their own problem-solving. The main aim of reasoning-congruent environments is to provide the student with access to the invisible behaviour of the objects of reasoning. To this end, six principles are identified which we can add to the catalogue presented in section 3.5 and which serve as a further challenge to computational mathetics:

- Make students' own reasoning explicit by having them make predictions of behaviour.
- Render behaviour visible by allowing student to access normally invisible states.
- Minimize the translation process from the students' internal plans to the external representation of the solution.
- Have the structure of a partial solution remind the students of where they are in their solution plan and the search space of the domain.
- Allow students to focus on subproblems on the way to solving the entire problem, thereby avoiding premature commitment and exploiting independence of subgoals.
- Proactively guide problem-solving by encouraging students to use a more profitable set of tools for solving problems.

In practice, the principles are illustrated by the GIL system, with which students build representations of a Lisp program by connecting icons representing program constructs in a graph, rather than a text-based representation

AI-ED systems that allow students to develop graphical representations on screen do so by providing a palette of icons which can be put together in restricted ways. Cox and Brna (1995) consider the extent to which a system may help a student construct and use their own representations, bearing in mind that students are liable to invent idiosyncratic notations with unfathomable or non-existent semantics. They identify four stages during which a system may be able to provide assistance:

- Problem comprehension. Student diagrams are often prematurely constructed and reveal an inadequate understanding of the problem statement.
- Representation selection. A good diagram represents not only the problem but also facilitates the reasoning necessary to solve it. Students may be unable to predict whether their selected method achieves this.
- Diagram construction. Students may need help in ensuring that all relevant information is expressed in the diagram, or in switching to an alternative representation if some kind of impasse is reached.
- Diagram use. A system should be able to check whether a student's reasoning steps and answer are consistent with her diagram.

Of course, as usual, it is a separate pedagogic decision as to whether a system should actually make interventions such as these.

5.9   Distributed reasoning

According to Shoham (1993), agent-oriented programming "promotes a societal view of computation, in which multiple agents interact with one another" but so far in this chapter we have imagined an agent reasoning in isolation from other agents. Recently in AI-ED there has been more emphasis on the possible benefits of various kinds of collaborative problem-solving and learning and therefore we should consider 'distributed reasoning', where two or more agents reason together to solve a problems.

To be concrete, let us imagine two possible scenarios. In the first (student-program-student) we have two students working together, using an intermediary program, to solve some problem. We would need to disambiguate the word 'together': does this mean physically (side-by-side) or conceptually (apart, but communicating via the program)? The role of the program is also crucial - is it merely a conduit for communication, is it the focus for problem-solving, or does it play some constructive part in facilitating the collaboration? In the second scenario (student-program-program) we have one student working with a program to solve some problem, this program genuinely collaborating with the

student, with a second program playing any of the roles indicated for the first scenario. Obviously, we can have more students and programs, but these two scenarios capture the essential features of distributed reasoning in AI-ED, namely:
- that different agents may well have different goals and will have incomplete knowledge of other agents' goals;
- that there may be variable scope for independent action by any agent;
- that different kinds of communication between agents are necessary.

Most of the relevant work has been carried out in the field of distributed AI, another burgeoning field of AI, with many applications but few yet to AI-ED. The main reasons why distributed AI has become important, and how these relate to AI-ED, can be summarised as follows:
- The advent of multi-processor systems and fast computer networks makes it necessary to consider how computation may be distributed. In AI-ED, it is now possible to consider students working together at a distance, synchronously or asynchronously.
- Distributed problem-solving may improve reliability, as components may duplicate or provide extra functions. Group problem-solving is likely to be more efficient than individual problem-solving, because individuals may bring to bear different expertise and perspectives. An overall problem may be solved even though no one individual has sufficient competence to solve it alone. This is not so obviously as beneficial in AI-ED as it is in distributed AI, for in AI-ED the specific problem is incidental. The aim is rather for each student to develop general problem-solving expertise and this may be inhibited if a colleague contributes too much.
- Some problems simply cannot be solved in isolation: they demand working with others (and some argue that most real-life problems are of this kind). For example, a flight simulator may help train an individual pilot but in reality a pilot's skills involve working with co-pilots, navigators, and air traffic controllers.
- Sometimes it is necessary to consider a group of agents as a 'community' with its own goals, different from and not some kind of union of the individual goals. Each individual agent may be unaware of the community goals. For example, a university composed of students and teachers all with their own goals might be considered to be a community with its own goals. Overall, the objective might be to meet the goals of the community, rather than those of the individuals.

However, AI-ED differs from distributed AI in one important respect. In distributed AI, communication and coordination between agents is considered to be an overhead. These processes involve spending time on activities other than directly solving the problem. Therefore, they should be engaged in only to the extent necessary to maximise the efficiency of a problem-solving process. In AI-ED, on the other hand, efficiency of problem solution is not the main concern. Advocates of collaborative problem-solving are more concerned that the processes of communication and coordination should promote various kinds of metacognitive activity which are beneficial to learning. For example, in the student-program-student scenario, one student may require the other to justify a proposed problem-solving step before agreeing to proceed: the process of generating such an explanation may, in some circumstances, improve the problem-solving abilities of both students, although it may actually interfere with the current solution process. Some of these factors will be considered in chapter 6 (on metacognition) and chapter 9 (on dialogue).

At the beginning of this chapter, we discussed how an agent `a` may solve problems by applying reasoners from the set `Reasoners(a)` to its set of beliefs `Beliefs(a)` and, through the `Interpret` function, how an agent may reason 'on behalf of' another agent. We did not discuss how an agent selects reasoners to apply, because that involves consideration of the goals of the agent (which will be discussed in the next chapter). However, we can consider in outline how we can extend this to deal with distributed reasoning.

If an agent has complete knowledge of the beliefs, reasoners, goals, intentions, and so on of all other agents (and infinite time to consider the matter) then it could in principle determine some optimum action. In practice, this is infeasible as it is not possible to make fully reliable ascriptions to other agents. The simplest approach is to appoint one agent, who is assumed to have adequate knowledge of the other agents, to a role of coordinator, responsibe for allocating tasks to achieve coherent behaviour towards some goal, its own, usually. This might be the role of the program in the student-program-student scenario (or a teacher in a classroom). If the program is asked to give advice to two students `s1` and `s2` about the next step (or reasoner) to apply or to comment on a proposed one, it would need to use `Beliefs(c,s1)`, `Reasoners(c,s1)`, `Beliefs(c,s2)`, and `Reasoners(c,s2)` with the `Interpret` function, taking into account `Goals(c)`, `Goals(s1)` and `Goals(s2)`. An example of such a 'referee' program in the AI-ED context will be discussed in section 9.3. Of course, there is the danger that such a referee would become unable to maintain its knowledge of all the other agents and would become a decision-making bottleneck.

In general, agents can have only partial knowledge of one another's beliefs and goals. Therefore, genuinely joint problem-solving involves a complex process of considering possible contributions to meet possibly conflicting goals. This is obviously a problem-solving process in its own right, or, expressed differently, it is process which is 'meta' to the original problem-solving activity. As we have not yet considered how an individual agent performs meta-problem-solving, this will be deferred until the next chapter.

6. Metacognition

AI-ED research has increasingly emphasised the importance of metacognitive skills. For example, the cognitive apprenticeship manifesto (Collins, Brown and Newman, 1989) says that: "We must first recognise that cognitive and metacognitive strategies and processes are more central than either low-level skills or abstract conceptual and factual knowledge". Other AI-ED research (for example, Shute and Bonar (1986); Derry and Hawkes (1993b); Lajoie (1993)) has aimed to develop systems to improve students' metacognitive abilities. Educational theorists such as Dewey, Vygotsky and Piaget all emphasised various aspects of metacognition, and more recently Schoenfeld (1987) has stressed its role in mathematics education. Brown (1987) comments that "the processes which have recently earned the title metacognitive are central to learning and development". In AI, the related concepts of metaknowledge, metareasoning and meta-level architectures have been extensively discussed (Maes and Nardi (1988), Genesereth and Nilsson (1987)). Therefore, computational mathetics should consider the nature of metacognition in some detail.

The first generally accepted definition of metacognition was that of Flavell (1976): "Metacognition refers to one's knowledge concerning one's cognitive processes and products or anything related to them .. metacognition refers, among other things, to the active monitoring and consequent regulation and orchestration of these processes in relation to the cognitive objects or data on which they bear." Brown (1987) offers "Metacognition refers loosely to one's knowledge and control of [one's] own cognitive system".

Unfortunately, phrases like "anything related to them", "among other things" and "refers loosely" present an open door through which a great many activities have passed to be labelled as 'metacognitive'. These include:
- being aware of one's own competence and incompetence, such as anticipating that one will be unable to solve a problem;
- knowing about general cognitive limitations, for example, that human memory is fallible, and being able to devise strategies to overcome them;
- recognising a problem, that is, identifying a problem as similar to an earlier one;
- taking account of the characteristics of the problem, for example, that a task is dangerous and needs to be tackled carefully;
- generating a plan, that is, sketching out a sequence of operations before execution;
- using general problem-solving techniques, such as quickly scanning a problem statement to get an impression of its likely difficulty;
- allocating resources, for example, deciding how much time to allocate to certain processes;
- being able to consider one's own knowledge and to apply it to problems of different kinds;
- modifing cognitive processes, that is, learning, by, for example, reflecting on how next time to avoid difficulties encountered;
- applying sound strategies, for example, in a scientific investigation to modify only one variable at a time;
- checking the consistency of incoming or calculated data;
- sharing control between cognitive and metacognitive processes;
- monitoring a solution process, for example, asking oneself if the goal is becoming nearer and knowing when to abandon a process;
- evaluating the solution, for example, by double-checking with a different method;
- evaluating the solution process, for example, by considering time wasted up a blind alley;
- being able to explain something to oneself or others.

All these activities are to some degree 'extra' to or transcend problem-solving. They involve the allocation of cognitive effort to activities which are not directly concerned with the problem at hand. Of course, the hope is that the investment in such extra processes will lead to eventual benefit with this problem and/or later problems.

Despite this litany of apparently essential skills, metacognition is not an unqualified benefit. Its claimed importance derives from a view that it is necessary for an agent to know more than a set of facts and how to apply them to solve specific problems. The agent should also be able to reason rationally about the problem-solving process itself. This, it is assumed, will enable the agent to improve problem-solving performance (that is, to learn), to develop transferrable skills, and to engage in discussions (such as tutorial interactions) about such processes. These may sound like platitudes but they are questioned by those who doubt that activity derives, or should derive, from a rational reasoning process: instead, action might follow in response to the situation in which the agent finds itself. Others may even question the implicit educational aim of fostering rationality. We cannot resolve such issues, but we can concede that metacognitive mechanisms must be applied with caution: an agent that spent too much time at the meta-level might accomplish less at the problem-level.

As the activities listed above are, by definition, to some extent problem-independent, discussions of metacognition are entangled with debates about the existence of general cognitive skills (as opposed to problem-specific knowledge). The fact that general cognitive skills can never be applied in their generality but must always be contextualised to the particular problem setting (Perkins and Salomon, 1989) has led some psychologists to deny their existence or at least

to give them a minor role in problem-solving performance. Whatever is meant by the nature of psychological reality, from the perspective of computational mathetics this denial is like denying the existence of 'metality' because only specific instances of metality (such as iron and copper) exist. If an agent behaves in such a way that it is useful to do so, we will readily ascribe metacognitive abilities to it. Computational mathetics is concerned with the evidence that is needed to make the ascription (a diagnosis problem - chapter 8), the representation of what is ascribed, and the role such an ascription might play in AI-ED systems.

One obvious potential role is to enable an AI-ED system to, in some way, teach metacognitive skills. This again opens a fierce debate about how, or even whether, such general skills can, in fact, be taught. The case for attempting to teach general skills which can be applied to unfamiliar problems is clear: the feasibility of doing so is questioned by many empirical studies that seem to indicate that students are unable to learn general skills, taught as general skills (for example, Pressley, Snyder and Cariglia-Bull, 1987). Arguments that learning Latin, logic, programming, and so on, improves students' general reasoning and planning abilities have not been supported by empirical evidence. On the other hand, teaching domain-specific knowledge seems to lead only to brittle knowledge able to handle only formulaic problems. Perkins and Salomon (1989) reach the reasonable compromise that general skills are best taught in a domain-specific form, while indicating their general applicability.

Anyway, that is a debate for educational research. The role of computational mathetics is, on the assumption that metacognition is important for AI-ED system designers, to develop a more precise language for discussing the many facets of metacognition and to develop practically useful ways of dealing with metacognitive ascriptions. In this chapter we will begin to develop a notation for discussing metacognition but we must be realistic about our aims. For one thing, discussions of metacognition in the educational and psychological literature are very discursive, with no attempt at formality and precise definitions. As the above list indicates, there is a huge range of activities covered by the term 'metacognition'. Some of them overlap with others in undefined ways. Some may be more amenable to rigorous analysis than others. In AI and computer science, those aspects of system performance which are comparable to metacognitive activities are often not explicitly defined but are buried in interpreting code, as we will see. Moreover, we must acknowledge the severe difficulty of making reliable metacognitive ascriptions. Often an agent engaged in some metacognitive activity (such as reflecting on a recent failure to solve a problem) is exhibiting *no* behaviour - and there is therefore very little evidence from which to make any ascription. Consequently, AI-ED system designers who aim to develop students' metacognitive abilities often try to devise ways for students to 'externalise' metacognition, without thereby inhibiting it.

Let us first ground the discussion by referring to the only (and not widely-disseminated) study of the degree to which AlgebraLand did, in fact, foster the development of metacognitive skills, as anticipated by advocates of cognitive apprenticeship (Collins and Brown, 1988; Brown, Collins and Duguid, 1989). With AlgebraLand, as a student applies operators to equations so a tree is built on the screen to show the various problem-solving steps, halts and continuations. This reification of problem-solving processes provides an opportunity for students to "examine their own floundering in order to formulate self-monitoring strategies that would help to detect and prune nonproductive approaches to similar problems" and to "reflect on problem-solving and evaluation strategies in the context of their use" (Collins and Brown, 1988). Did they take the opportunity?

Foss (1987) considers the extent to which students reflected on (after) and monitored (during) a problem solution. Regarding reflection, she comments that "naturally, students need some guidance while they review their search trees" (and perhaps concedes by omission that unguided reflection is unsuccessful or simply doesn't happen). Three types of review task were used to supplement AlgebraLand:
  • Students were asked to annotate their tree in terms of given equation solving plans.
  • Students were asked to re-solve the equation, omitting unnecessary steps.
  • Students were asked to compare their solution with an optimal solution.
If the system is not intended to comment on these extra activities (as seems to be the case here) then it does not need to represent or perform any metacognitive processes itself. No data is reported that students became better equation solvers or better at reflecting in general as a result of these de-briefing exercises.

If a solution tree has more than one branch then the student has clearly monitored her solution process to some extent, as she has decided to abandon one path and start another. Using post-problem interviews, Foss tried to determine when and how students decided to abandon a path and gave the following list of conditions (Foss, 1987):
  • Surface features of the equation, such as an increase in complexity.
  • Repeated states, that is, when a previously generated state is re-generated.
  • Repeated application of inverse operators.
  • Noticing shorter paths, that is, seeing a state on another path which may be closer to a solution.
  • Temporal cues, that is, after long pauses to decide on an operator.
  • Difficulty in deciding on an appropriate operator.
  • Subjective certainty factor drops below threshold.
  • Expectation violation, for example, when the application of an operator produces a surprising result.
  • Plan-action conflict, that is, the actions used to satisfy a goal are ineffective.

- Conflicting subgoals, that is, earlier accomplished goals undone while satisfying current subgoal.

This list has been quoted in full because it is indicative of the state of the art of metacognitive studies. AlgebraLand is described as a standard system and the importance of metacognition has been much emphasised and yet there appears to have been no attempt to clarify these intriguing results obtained some time ago. The conditions are vaguely described and not independent. More precise definitions of them and data as to their relative occurrence and effectiveness would be welcome. Moreover, the conditions are presented in terms of algebra but are (one suspects) instances of more general conditions. It would be interesting to know to what extent more generally-stated conditions apply to other kinds of problem. As for reflection, no data is reported that students became better equation solvers or better at monitoring in general as a result of using AlgebraLand.

The list of metacognitive activities given earlier also shows a distressing vagueness. It is clear that many of them (such as allocating resources) are related to others (such as generating a plan). AI-ED system designers are exhorted to take fully into account the importance of metacognition, although educationalists and psychologists seem reluctant or unable to say precisely what it is. Actually, what has been called metacognition is an amalgam of many different things, and they will never be satisfactorily disentangled until a precise technical language is adopted for doing so. We will work towards our own language by first briefly reviewing AI work on meta-level architectures and metaknowledge.

6.1   Meta-level architectures

A *meta-level architecture* defines a computational system in terms of an *object-system* and a *meta-system* (Maes and Nardi, 1988). The object-system contains a model for reasoning about the world to solve problems; the meta-system contains a representation of the object-system, which it reasons about (Figure 6.1). The object-system and its meta-system representation are *causally connected*, meaning that any change in one causes a change in the other.
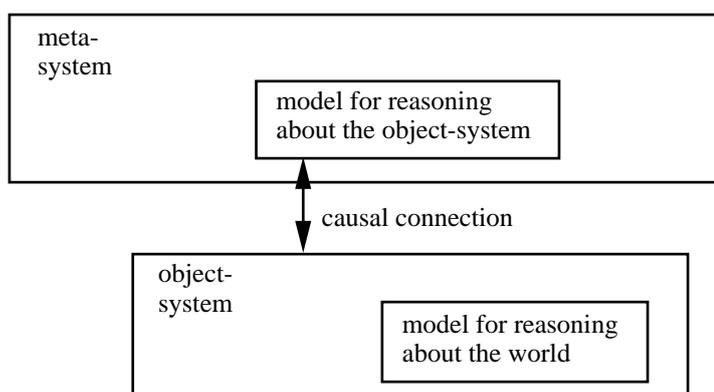

Figure 6.1. Object-system and meta-system

The purpose of such architectures has mainly been to permit control of object-system computations. An intelligent agent should be able to oversee its own problem-solving processes. It should be able to assess the merits of particular methods and use the results of such assessments to decide upon subsequent steps.

In practice the architectures are often realised by means of *meta-circular interpreters*, which are explicit representations in the language itself of the interpreter of the language used to define the object-system and meta-system. Such systems are necessarily causally connected but the disadvantage is that the meta-system representation must be a complete and efficient representation of the operation of the system. Alternatively, in *declarative reflection* the causal connection is maintained by explicit specifications incorporated in the interpretation process. Some systems carry out all inference in the object-system, having no separate meta-system interpreter and evaluating meta-predicates at fixed points in the computational cycle. Others perform all inference in the meta-system, simulating the object-system interpreter. It is also possible to have separate interpreters for the two systems.

Many classical AI systems can be re-described in terms of meta-level architectures, to no obvious purpose except to show that the basic idea (stripped of the jargon) is not particularly radical. For example, TIERESIAS (Davis, 1980) has rules in the meta-system which ordered and pruned rules in the object-system. SOAR (Laird, Rosenbloom and Newell, 1986) has three levels: a 'problem space level', a 'production level' that applies productions from long-term memory to objects in working memory, and a 'preference level' that selects between candidates or signals impasses, which are overcome by recursively applying the three levels to a new sub-goal.

6.2   Metaknowledge

It is sometimes argued that what a student knows is less important than what she knows she knows or does not know. Knowledge of one's own limitations can be a reason for acting, to acquire knowledge, and for not acting, to avoid contemplated actions outside one's competence. We would like to be able to handle everyday statements such as "I don't know anything about art" and "Anything I don't know about isn't worth knowing". In AI, such issues are considered as *metaknowledge*, that is, knowledge about knowledge. Of course, we can also have 'metabelief' and plausibly meta- some of the other mental ascriptions, such as metacommitment and metagoal, although these terms are not in use. We have already met some techniques for dealing with metaknowledge, for example, in autoepistemic logic and the introspection axioms in modal logic. The basic idea, which we will elaborate on below, is to reason about the contents of a set of sentences representing belief or knowledge, rather than about the individual sentences. We will make a distinction, which is not always made clear in AI, between metaknowledge and *metareasoning*, i.e. reasoning about reasoning, which is more concerned with how a meta-level architecture is interpreted.

6.3   Metacognitive schemata

metacognitive schemataIn the previous chapter, we discussed the idea of a set of reasoners being applied to a set of beliefs, through an `interpret` function, to solve a problem. Now we must consider and extend the definition of this interpretation process. Let us first consider an artificially simple case: an agent reasoning, using modus ponens and modus tolens, about metals. The reasoners themselves do not determine which conclusions will be drawn - for this, we need to specify, for example, that modus ponens will be applied first and then modus tolens. A rule for selecting a reasoner to apply will be called a 'monitor'. In general, then, we will ascribe to an agent a set of monitors M. Whereas the set of reasoners R maps the agent's beliefs Δ into a new set of beliefs Δ', the set of monitors M maps RxΔ into (RxΔ)'. In general terms, a reasoner is a function of its 'lower level' Δ and produces results at this lower level, and a monitor is a function of both its lower levels (R and Δ) and produces results at its lower levels.

We will represent this by the notation:
```
        { monitors { reasoners { beliefs }}}
```
For example, we might have:
```
        { Applicable(modus-ponens) ∅ apply(modus-ponens),
          Applicable(modus-tolens) ∅ apply(modus-tolens),
          ...        { modus-ponens: ϕ∅ψ, ϕ => ψ,
                       modus-tolens: ϕ∅ψ, ~ψ => ~ϕ,
                ...        { Metal(iron),
                            ~Shiny(sulphur),
                            Metal(copper),
                            Metal(x) ∅ Shiny(x) } } }
```
In this case, it is intended to indicate that the agent will derive the conclusions `Shiny(iron)` and `Shiny(copper)` before the conclusion `~Metal(sulphur)`.

We will not pause to develop a formal notation for monitors, as this is difficult technically. First, we need to clarify the intentions informally. The notation above is supposed to indicate that if modus-ponens is applicable then it will be applied before modus-tolens. Of course, this intention is only implicitly indicated by the ordering of the monitors (alternatively, we could have attached weights to the monitors). Making this intention explicit would involve defining a further outer set of rules, to define how a monitor is selected to apply. This will be considered later. (Readers may be alarmed at the prospect of an infinity of chapters each making explicit how rules of the previous chapter are applied.)

Below, we give some illustrations to indicate an approach towards explicitness in discussing some aspects which, though said to be important for AI-ED, are always discussed informally. We will see in outline how some of the metacognitive processes listed at the beginning of this chapter can be addressed. Adding more detail to any one of these aspects - for example, to check the consistency of ascriptions - would be a major research undertaking. Some will be considered further below. First, we just comment on some general points.

The discussions of sections 4.5 and 5.1 about the domain-relatedness and abstractness of beliefs and reasoners can be echoed for monitors. A monitor such as:
```
        Applicable(modus-tolens) ∅ apply(modus-tolens),
```
might be considered to be a member of `Domain-monitors(a)` because `modus-tolens` would be a member of the agent `a`'s domain vocabulary. A monitor such as:
```
        More-complex(state) ∅ abandon-path,
```
might be considered to be a member of `Abstract-monitors(a)`. Intuitively, it seems that many monitors will be abstract and concerned with general problem-solving skills, and that AI-ED systems will be concerned with the transition from concrete to abstract monitors.

The above example shows an ascription of monitors, reasoners and beliefs to a single agent (a student, say) by an observer. As discussed previously, in AI-ED we have to consider at least two agents (a student and a system) and, at

minimum, the ascriptions of one (the system) to the other (the student). We will denote the ascription to an agent `a` by `Asc(a)`, so that:

```
Asc(a) = { m1, m2, .. { r1, r2, .. { b1, b2, .. } } }
```

indicates that `Believes(a,b1)`, `Reasons(a,r1)`, `Monitors(a,m1)`, and so on. An ascription of one agent `a` to another `b` will be denoted by, so that

```
Asc(a,b) = { m1, m2, .. { r1, r2, .. { b1, b2, .. } } }
```

indicates that `B(a,B(b,b1))`, `B(a,Reasons(b,r1))`, and so on. AI-ED systems are concerned in various ways with considering the relations between (at least) `Asc(s)`, `Asc(c)`, and `Asc(c,s)`, that is, with comparing the (ascriptions which we might make to describe the) way that the student solves problems, the way the system solves problems, and the way the system believes the student solves problems. We will pursue this in chapter 8.

We can use this notation to represent self-ascriptions of individual beliefs, reasoners and monitors. For example, the contents of `Asc(s,s)` could indicate that the student believes that she reasons with modus tolens. But we cannot, as it stands, represent more global properties of the sets of beliefs, reasoners and monitors. For example, we could use

```
Asc(s) = { m1, m2, .. { r1, r2, .. { Cold(Mars), B(s,Cold(Mars)), .. } } }
```

to represent that the student believes that Mars is cold, and believes that she believes that, but we cannot naturally represent that she believes nothing about Neptune. This last statement is about the contents of the set of beliefs, namely that there is no mention of Neptune there. Propositions about the set of beliefs will be written in square brackets at the head of the set:

```
Asc(s) =
{ m1, m2,
  ...        { r1, r2,
             ...        { [Ignorant-of(Neptune),...]
                        Cold(Mars),
                        ... } } }
```

Computationally, it is expensive to (re)determine this property and its existence as a global attribute can avoid the need for detailed searches of the set. Of course, if the contents of the set of beliefs changes, then it may be necessary to check that the comment still holds.

We may want to make explicit how the property is determined, for example:

```
Asc(s) =
{ m1, m2,
  ...        { ∀b (b in Beliefs(s)∅~Mention(b,c)) => Ignorant-of(c),
             ...        { b1, b2,
                        ... } } }
```

We are here ascribing to the student a reasoner for determining whether she is ignorant of some concept. The conclusion here is a kind of meta-belief, which we should not just include among the basic beliefs. It is about the set of beliefs. The reasoner is no longer matching individual beliefs but referring to the set of beliefs.

Similarly, we can consider a student who cannot solve quadratic equations, say. This is a statement about the set of reasoners ascribed to her - they do not provide a means of solving quadratics:

```
Asc(s) =
{ m1, m2,
  ...        { [Cannot-solve(quadratics)]
             r1, r2,
             ...        { b1, b2,
                        ... } } }
```

This notation provides the means for dealing with attitudes and aptitudes which transcend specific beliefs, reasoners and monitors (section 6.9). In AI-ED, we often assign students to classes (typically 'novice', 'intermediate' and 'expert'). These are labels defining global properties of the ascriptions made. In principle, we can attach comments as precise as we wish. For example, we could describe a student as persistent, irrational and an expert on Unix but a novice on the World Wide Web. These are comments about the monitors, reasoners and beliefs, respectively, ascribed to her:

```
Asc(s) =
{ [persistent, ...]
  m1, m2,
  ...        { [irrational, ...]
             r1, r2,
             ...        { [expert(Unix), novice(World-Wide-Web), ...]
                        b1, b2,
                        ... } } }
```

We need, as usual, to be careful about who has been ascribed what: there is a world of difference between the student believing that she is a Unix expert and the system believing that she is. The former would be represented in `Asc(s,s)`, the latter in `Asc(c,s)`. `Asc(s)`, as above, is an observer's ascription to the student.

From the definition of a monitor as something that maps RxΔ into (RxΔ)' we can distinguish two types of monitor: those that produce new beliefs and those that produce new reasoners. The examples above are of the former type. New beliefs may also be generated by learning processes, such as generalisation. For example:

```
{ Concrete(x) ∅ generalise(x),
    ...        { ...        { Planet(neptune) ∅ Orbits(neptune,sun),
                               ... } } }
```

might lead to the new belief:

```
Planet(y) ∅ Orbits(y,sun),
```

A monitor may generate new reasoners in various ways, for example, by reacting to the inadequate performance of existing reasoners, by perhaps generalising them. The result of a generalisation is an entity at the same level as the object generalised; the process of generalising is an entity at a meta-level. Of course, the precise definition of generalisation is complex (section 7.3). We might also represent a process of 'compilation' whereby reasoners are coalesced into a more efficient reasoners. For example,

```
{ Reasoner(x1,x2 => x3) & Reasoner(x3,x4 => z5) ∅ new-reasoner(x1,x2,x4 =>
x5)),
    ...        { p∅q, p => q,
                 q, ~qVr => r,
                 ...        { b1, b2,
                               ... } } }
```

might lead to the new reasoner:

```
p∅q, p, ~qVr => r
```

We should note that the curly bracket notation does not allow the expression of all the sentences we need. We can represent B(a,p) and B(a,~p), but not ~B(a,p), and similarly for multi-agent ascriptions. We cannot quantify over expressions, use logical operators, and so on. In such cases, we must resort to the full notation.

Finally, in this section, we should remark that, although the terminology and notation may be different, this three-level architecture is comparable to the distinctions which cognitive scientists make in describing the kinds of knowledge needed for problem-solving. Roughly corresponding to our beliefs, reasoners and monitors are: static domain, inference and task knowledge (Breuker and Wielinga, 1989); problem-situation knowledge, procedural knowledge and declarative knowledge (de Jong and Ferguson-Hessler, 1986); declarative, procedural and conditional knowledge (Anderson, Boyle, Corbett and Lewis, 1990); and conceptual, procedural and utilizational knowledge (Smith, Greeno and Vitolo, 1989). As even the terms indicate, the precise relationship between all these concepts is hard to specify.

6.3.1   Problem-solving

In this section, we will give some brief illustrative examples of our notation.

(a) Symbolic logic

Students of symbolic logic are required to use rules of inference to derive expressions in a logical system. For example, Twidale (1989) describes a system to support students carrying out natural deduction proofs in propositional logic using the rules given in Lemmon (1965). In this case, the set of beliefs contains beliefs about the goal and the on-going proof. All the beliefs are problem-specific beliefs, because symbolic logic is, by definition, domain-independent. The reasoners are the rules of inference specified by Lemmon. The student's task is to determine an appropriate sequence of applications of these reasoners, and more generally to develop a set of 'monitors' to enable her to carry out efficient proofs in general. A set of heuristics are suggested to guide the student: "assume any assumptions stated in the goal, then apply any of modus-ponens, modus-tolens, and-elimination or double-negation, then if you're trying to prove X->Y use the conditional-plan, and so on" So, for a competent agent, we might ascribe:

```
{ make-assumptions,
  Applicable(modus-ponens) ∅ apply(modus-ponens),
  ...
  Goal(X->Y) ∅ plan(conditional),
  ...        { assumption: ...,
               modus-ponens: φ∅ψ, φ => ψ,
               modus-tolens: φ∅ψ, ~ψ => ~φ,
               conditional: ...,
               ...        { Goal(PvQ->~(~P&~Q)),
                            Assumption(P),
                            ~~P,
                            ... } } }
```

Although there are many details under-specified, we can make some general points. The monitors can function in a data-driven or goal-driven fashion. An agent might apply any reasoner that is applicable to the current data - this is like a novice logician blindly applying any rule of inference she can in the hope that the required expression will result

(behaviour that is actually observed with beginners). Or an agent may be driven by the goal (or sub-goals) to set up plans. All the previous discussion about beliefs and reasoners (for example, that an agent might need to keep track of which reasoners have been applied, e.g. ~~P from Assumption(P) using double-negation; that an agent, especially a student, may have incomplete or incorrect reasoners; and that the actions will need to be expressed using a notation like that of the situation calculus) continues to apply and will not be repeated in this chapter.

(b) Algebra

In this case, the reasoners are the operators which are available and the beliefs concern the equation to be solved and general knowledge, for example, about arithmetic. Monitors are concerned with the application of reasoners. For example, if a student says "Whenever I see any brackets I multiply them out" she is expressing a monitor, referring to a reasoner concerned with the actual operation of multiplying out. A competent student might use a strategy of isolating the target variable (by getting rid of associated numerals), after collecting up multiple occurrences of the variable. The conditions for abandoning a search path specifed by Foss (1987), listed earlier, are also monitors. Thus, we might have an ascription, in outline:

```
{ Has-brackets(exp) ∅ remove_brackets(exp),
  Multiple-occurences(var,exp) ∅ plan(collect-variables),
  Single-occurrence(var,exp) ∅ plan(isolate-variable),
  More-complex(exp) ∅ abandon-path,
  ...        { remove-brackets: ...,
       ...          { 6X+9X+4 = 26-7,
                      26-7 = 19,
                      ... } } }
```

Once such an outline is sketched it becomes a challenge for greater precision, which is the hallmark of the approach of computational mathetics. For example, we might immediately ask what is meant by abandon-path, and begin to elaborate ideas how about an agent changes the equation currently focussed upon. However, at this point, precision is not the objective, not least because precision in itself is not a virtue - we need a clearer idea of what AI-ED systems need increased precision for. At the moment, we have an informal picture of an agent being ascribed monitors to account for behaviour such as deciding which operator to apply, setting up a plan of action, abandoning a path, and so on, the monitors themselves being represented in a production system-like notation.

We can make further general points, however. For example, which items should be ascribed to which level is not fixed but depends upon the agent. A novice, for example, might need to form a plan to isolate a variable, this plan being achieved by the application of several primitive operators, whereas an expert might have 'compiled' a single (macro)operator to accomplish this step. In the latter case, our ascription might be:

```
{ Has-brackets(exp) ∅ remove_brackets(exp),
  Single-occurrence(var,exp) ∅ isolate-variable(exp),
  ...        { remove-brackets: ...,
               isolate-variable: ...,
       ...          { 6X+9X+4 = 26-7,
                      ... } } }
```

We can anticipate that one process of learning, that is, the agent moving from simple to more complex operators, might be represented by a process of editing such a description (chapter 7).

Also, we can see that we have scope for capturing the generality of monitors. The above examples are all in terms of algebra but we could generalise the monitors (if we considered that an agent behaved in such a way that it might be ascribed a more general rule) and imagine them to be instantiated to algebra. For example, we might have a monitor:

```
More-complex(state) ∅ abandon-path,
```

to express the rule that whenever during problem-solving of any kind we generate a more complicated state then we should abandon that method and try something else. Of course, an effective agent will refine this general rule, because it is sometimes necessary to make matters temporarily worse, but it is a plausibly useful problem-independent heuristic.

(c) Chemistry

An organic chemistry student is asked to synthesise, that is, specify a sequence of chemical reactions to produce a specified compound, for example, 2-chloro-2,4,4-trimethylpentane. The student is given an (implicit) list of possible ingredients - any compound found in a chemistry lab, such as ethylene, propylene, etc. - and an (implicit) list of 'operators' - those chemical reactions described in a textbook, such as cationic polymerisation, alkylation, etc. The student should bear in mind various heuristics- maximise the yield, minimise the time (or number of reactions) to generate the target compound, minimise general inconvenience, maximise safety, minimise cost, and so on.. In our terms, these heuristics are monitors - they help to determine which chemical reaction to carry out. The chemical reactions themselves are reasoners - they are capable of operating on chemicals to produce new chemicals. The

properties of the chemicals and the reactions constitute the basic knowledge of the domain. In this case, the control regime for the monitors is one of selecting the best available option, rather than the first that is applicable:

```
{ Goal(x) & Alkyl(x) ∅ plan(alkylation),
  Applicable(reaction,y)   &   Available(y)   &   Danger(reaction,low)   ∅
apply(reaction),
  ...        { cationic-polymerisation: ...,
    alkylation: ...,
    ...        { Goal(2-chloro-2,4,4-trimethylpentane),
      Colourless(propylene),
      Danger(cationic-polymerisation,low),
      ... } } }
```

In this case the amount of detail needed is immense - several hundred pages of an advanced textbook, plus more elementary knowledge of chemistry and other commonsense knowledge - but almost all that which is specified in the textbooks is at the lower two levels, whereas the knowledge which really distinguishes an expert chemist lies more at the third level. The more that this knowledge can be made explicit, then maybe the more likely it is that an AI-ED system could help students acquire such knowledge. It is perhaps possible that a representation such as the above could be elaborated (at the monitor level) to form the basis for an organic chemistry simulation which aims to give planning and monitoring advice to students.

(d) Music

We can now re-consider the student asked to discuss Beauvais' Play of Daniel (section 1.1). This is a different kind of problem to that in mathematics and science but we can try to distinguish, in general terms, the kinds of knowledge which need to be brought to bear to tackle it. At the domain level, we have the host of beliefs which the student may have about the Play of Daniel (who composed it, when, details of the music itself, etc.), about music generally (forms of music, how music is performed, etc.) and about the world generally (social conditions when Daniel was composed, the nature of religious ceremonies, etc.). This is a vast body of knowledge to represent in computational form - but, thankfully, we might not have to, at least, not in great detail, for two reasons. First, an AI-ED system itself may provide access, through a multimedia system, to this kind of knowledge. In so far as the system needs to know it, it may be able to ask the student. For example, after the student has read the relevant pages, the system can ask: "ok, so when was Daniel composed?". The second reason follows: the system is not really concerned with the domain level per se. It is rather assumed that the student is competent to understand all the domain level concepts - what she is more likely to need advice with are strategic questions about which domain level entries are most important.

In this case, there are more or less specific rules for inferring new information from what is known. For example, we might infer the approximate date of a piece of music from the notation in which it is written, or the musical instruments for which it is composed. There are many kinds of 'operation' which one may perform to determine, for example, various contrasts: secular/sacred, serious/comic, traditional/non-traditional, etc. For example, the use of non-liturgical inclusions illustrates the secular/sacred contrast. It may be that a student will need advice in this area, for she may have gathered the relevant 'raw facts' but not noticed an important conclusion that follows from them.

More likely, the student will need help in deciding what facts to gather. There is an infinity of facts that could be stated about Daniel: the student needs to know which facts are most likely to lead to the kinds of conclusions which will contribution to a successful critical essay. It may be appropriate to look for contrasts with other music, to explain the musical style, to describe the plot, to place the piece in its artistic and social context, and so on. Of course, no computer-based learning environment can hope to provide infallible advice in these respects. On the other hand, some advice is conceivable. Some of it is 'data-driven' - for example, if the student records the fact that Daniel was composed in the 12th century then this could trigger a number of follow-on questions, such as: Was anything like it composed before?, Was the music written down?, Was anything happening in the 12th century that 'caused' this composition?, and so on. Some of it is 'goal-driven' - for example, to write an essay on any piece of music it would seem necessary to establish when it was composed in order to put it into historical context.

At the risk of being too simplistic, we could illustrate this in our framework:

```
{ Goal(criticise(x)) ∅ plan(category(x)),
  Goal(criticise(x)) ∅ plan(meaning(x)),
  Goal(category(x)) ∅ plan(date(x)),
  Goal(meaning(x)) & ~Composer(x,y) ∅ plan(read-composer(x,y)),
  Applicable(x) ∅ apply(x),
  ...        { modus-ponens: φ∅ψ, φ => ψ,
    category: Isa(x,b)∅Category(x,c) & Isa(a,b) => Category(a,c),
    ...        { Goal(criticise(daniel)),
      Isa(daniel,feast-of-fools-play),
      Isa(x,feast-of-fools-play) ∅ Category(x,jocular),
      ... } } }
```

The first two monitors, for example, say that if the goal is to criticise something then you might set up a subgoal of determining its category or its meaning. Later monitors might that if the (sub)goal is to determine the meaning of a piece of music you might set up subgoals to determine its composer or its performer. The last monitor listed simply says that if any reasoner is applicable, then apply it (undirected data-driven reasoning). If a computer-based advisory system has access to such a representation then it might be used to advise a student who does not know how to proceed, or to comment on the student's plans for proceeding.

6.3.2   Metareasoning

This three-level framework can be used to describe metareasoning techniques as developed in AI and to provide limited reasoning (as discussed in section 5.2). The general idea is that the monitoring level specify properties of the reasoning level which determine how it is interpreted with respect to a set of beliefs. The aim is to clarify the nature of cognitive activity during problem-solving and specifically, for student modelling purposes, to enable the system to explicitly model and reason about different aspects of a student's competence. Unless these components are declaratively specified, they cannot be dynamically changed by the system (to model changes in the student or to adapt the general framework to an individual student) and they cannot form the focus of instructional interactions.

Consider the following ascription:

```
{ Easy(x) ∅ apply(x,3),
  Difficult(x) ∅ apply(x,1),
  ...        { [Difficult(modus-tolens), Easy(modus-ponens)]
               modus-ponens: ϕ∅ψ, ϕ => ψ,
               modus-tolens: ϕ∅ψ, ~ψ => ~ϕ,
               ...        { Cold(Neptune),
                            Cold(Pluto),
                            Cold(x) ∅ Lifeless(x),
                            ~Lifeless(Mars),
                            ~Lifeless(Earth) } } }
```

Ignoring the considerable technical problems, the intention is that the monitors indicate that easy reasoners are applied (at most) three times and difficult reasoners (at most) once. This particular agent is believed to consider modus-tolens a difficult rule and modus-ponens an easy one. Interpreting these schemata we obtain the derived beliefs:

```
Lifeless(neptune)
Lifeless(pluto)
~Cold(mars)
```

but not `~Cold(earth)`, assuming that the schemata are applied from the beginning of the belief-set.

The idea of a metalanguage has been much studied in AI and in mathematics (for example, to overcome logical paradoxes). At first in AI, metareasoning was used to shorten proofs obtained using simple, uniform deduction strategies such as those based on resolution, by, for example, looking at syntactic structure rather than repeatedly applying inference rules. Metareasoning has since been applied to many areas of AI. We will illustrate the method by two examples related to student modelling.

Aiello and Micarelli (1990) describe a system called SEDAF to help students learn how to graph mathematical functions by solving for characteristics of the function. The system's architecture can be described, in outline, by the following ascription:

```
{ Answers(p,dontknow) & Proof(p,d) & q a d ∅ ~B(s,q),
  Answers(p,yes) & Proof(~p,d) & q a d ∅ ~B(s,q),
  Answers(p,yes) & Proof(p,d) & q a d ∅ B(s,q),
  ...        { resolution: ϕ∅ψ1, ~ϕ∅ψ2 => ψ1 ∨ ψ2
                           { Stationary(x,f) & Decreasing-left(x,f) &
                                     Increasing-right(x,f) ∅ Minimum(x,f) ,
                             Denominator-zero(x,f) ∅ Pole(x,f) ,
                           ... } } }
```

The set of beliefs describes the properties of the domain. The system's reasoning schemata are not stated but in fact amount to the rule of resolution. In order to associate a meta-level with the reasoners, a meta-level predicate `Proof(p,d)` is defined which asserts that d is a derivation or proof of proposition p (using the specified reasoners on the specified beliefs). `Proof` is defined by a suitable set of meta-axioms, so that if `Proof('p','d')` is a theorem of the meta-theory then d is a derivation of p (using the reasoners and beliefs) where `'p'` and `'d'` are the representations of p and d at the meta-level (Weyhrauch, 1980). The first monitor specifies that if the student does not know the answer to a problem which the system can solve then the system believes that the student does not know propositions used in the derivation of the answer. (In general, she would not know one or more of those propositions, and the system might begin some dialogue to work out which.) Note that the same reasoning schema (resolution) is used throughout, which is not necessary formally: we could try to develop a representation which better corresponds to the student's reasoning processes.

The underlying reasoning of the LNT system (Van Arragon, 1991) is based upon a version of resolution (linear resolution) and uses the meta-level predicate `Infer`. `Infer(p∅q)` means that `q` may be inferred from `p`, from the beliefs using linear resolution. It is used at the meta-level in monitors of the form:

```
conds ∅ ~Infer(p∅q)
```

where `conds` defines the conditions under which the inference cannot be made. Therefore, this use of `Infer` provides a kind of inhibited or limited reasoning. The following ascription might describe a student who will not make an inference of two or more steps and so will not infer `~Liquid(mercury)` and will be unaware of the (potential) inconsistency of her beliefs:

```
{ Infer(p1∅q1) & Infer(p2∅q2) & q1•q2 ∅ ~Infer(p∅q),
  ...        { linear-resolution: ...
                    { Metal(mercury),
                      Liquid(mercury),
                      Metal(x) ∅ Solid(x),
                      Solid(x) ∅ ~Liquid(x),
                      ... } } }
```

This was the basic idea used in the foreigner example of section 5.2.1 to handle a more interesting case of limited reasoning, that of lack of awareness. Other types of limitation can be similarly expressed. For example, a particular reasoning step may be too difficult for novices to carry out:

```
Difficult(p∅q) & Novice(s) ∅ ~Infer(p∅q)
```

These two meta-level predicates (`Proof` and `Infer`) just ask questions about derivations in the lower level. In general, monitors can also impose restrictions on what can be derived. They could, for example, assume that the lower level will draw the inferences it is capable of, unless the meta-level knows of constraints which prevent them being drawn. This approach is independent of the underlying reasoning procedures - but to actually design such a meta-level we have to commit ourselves to a particular underlying procedure. Both SEDAF and LNT have been implemented in Prolog, in which it is relatively easy to write a meta-interpreter to define meta-level predicates such as `Proof` and `Infer`. Such implementations demonstrate both the methodological advantages of having a formal specification and the practical advantages of concise, rapid prototyping. However, the extra layer of interpretation may lead to inefficiency, although techniques are being developed to help overcome this (Donini et al, 1990).

6.4 Planning

To develop more detail on metacognition we can adopt the standard division of metacognitive activities into those that happen before, during and after problem-solving - planning, monitoring and reflection, respectively. Standard though it is, it is not a very precise division, because problem-solving typically involves setting up sub-problems to solve and hence an activity that occurs after the sub-problem is solved is 'reflecting' from the perspective of the sub-problem but 'monitoring' from the perspective of the original problem. Instead, the division could be based on the timescale over which the activity is concerned. For us, planning will concern the consideration of events in the near and relatively distant future; reflecting will concern the consideration of events in the near and relatively distant past; and monitoring will concern the consideration of events in the near past and near future. Given the vague boundaries, we can anticipate some overlap in the three kinds of activity.

Planning is concerned with "finding a series of actions that can be expected to have a desirable outcome" (Ginsberg, 1993). Planning is relevant to AI-ED for two main reasons. First, AI-ED systems may need to plan to achieve the desirable outcome of a student learning. This is considered in chapter 10. Secondly, students may need to plan to solve problems and systems may need to model their planning processes. For an illustration, imagine a trainee car mechanic using a simulation to learn how to change the wheel on a car (a version of this problem has become a benchmark for AI planning research (Barrett and Weld, 1994)). There are various actions available to her: jack the wheel, unbolt the lugs, etc. We would not expec t a competent mechanic to simply apply the (simulated) actions to see if they are useful, but to form some overall plan, to, say, first remove the old wheel, and so on. A good AI-ED system would be able to communicate with the trainee about plans and not just about actions.

A good plan need not be complete and fully specified before any action is carried out. Consider a student solving an integration problem. She might plan to use a $t=\tan(\theta/2)$ substitution and then review the new situation. To anticipate the second step would involve determining the outcome of the first step, and therefore developing a complete plan would be tantamount to solving the whole problem in the head before solving it for real. Our trainee mechanic might develop a more detailed plan because the outcomes of the various actions might be considered more predictable. Also, the need for planning depends on the ease of carrying out and undoing actions. As far as AI-ED is concerned, we need to be aware of the considerable difference between, say, changing a wheel in a simulation and on a real car. Many AI-ED systems make a feature of the fact that it is easy for students to apply and undo operators and hence to experiment. If the intention is that students develop planning abilities then this feature may be counter-productive. For example, AlgebraLand aimed to develop metacognitive skills but reduced the need for students to plan by making it inexpensive to recover from mistakes.

At first glance, it might seem that we need no special mechanisms to deal with planning: we could just represent everything in predicate logic, carry out a constructive proof that our goal can be achieved, and from the proof extract the actions needed to reach the goal. It is not that simple, for various reasons which we can relate to AI-ED:

- As planning involves reasoning about action, we will need to use the situation calculus and face up to the problems mentioned earlier, such as the frame problem. We will probably need to use nonmonotonic reasoning techniques, as many things are unchanged by actions. Putting a wheel on will probably leave the jack there; removing the jack may not leave the car where it was. There is a particular problem of subgoal interaction, where one action undoes the desired effect of an earlier action.
- Planning needs to be integrated with performance, as even the best-laid plans may go awry when executed. It should not be necessary to re-plan from scratch if the previous plan can be adapted to overcome the unanticipated situation. An instructional plan may need to be modified if a student demonstrates an unexpected misunderstanding. It may happen that unexpected events ease the execution of a plan.
- Few problems are so novel that a complete plan needs to be derived afresh. Often, some similar problem has been solved in the past and the plan for that problem may be adaptable ('case-based planning' (Hammond, 1990)). The difficulty lies in specifying problems so that similarity can be determined and plans so that they may be amended.
- Planning in AI usually assumes that there is only one agent with goals and actions. In AI-ED, we have at least two agents with their own goals and actions, and these may be in conflict, to some extent.
- It is usually assumed that planned actions are carried out in series but, of course, it may be possible to carry some out in parallel. In general, it is not easy to predict the interactions between such actions even given definitions of their effects if carried out separately.
- An agent will typically have more than one goal - some local (such as 'get a new wheel on'), some more global (such as 'gain a qualification' and 'keep healthy'). The goals may not just be in terms of desirable states of the world - they may be in terms of minimising or maximising some objective (such as time and cost) or preventing some state of the world.
- Not only does the goal or goals impose constraints on the plan obtained but there are typically constraints on the planning process itself. A doctor cannot spend an indeterminate time on devising a plan for a thorough diagnosis - a patient may need speedy action rather than that which a lengthy analysis might show is optimum.

Despite all these complexities, Ginsberg (1993) predicts that planning is the area of AI research that will make the most progress in the next ten years. Here, we will outline how the formal representations of planning relate to the previous schemes developed and to AI-ED generally. Plans and goals may be ascribed to agents like the other mental constructs we have discussed. Up till now, we have included them among the set of beliefs, but we may now wish to separate them:

```
{ m1, m2,
   ...      { r1, r2,
               ...      beliefs:{ b1, b2, ... }
                        goals:  { g1, g2, ... }
                        plans:  { p1, p2, ... } } }
```

It might be useful to partition monitors and reasoners according to whether they operate on beliefs, goals or plans (of course, some may operate on more than one).

Goals and plans correspond to modal operators and hence have the properties discussed before, for example, of being referentially opaque. For example, we may have:

```
Goal(a,visit(capital-of-Germany),t)
```

that is, agent `a` has in situation `t` the goal of visiting the capital of Germany, without necessarily having:

```
Goal(a,visit(Bonn),t).
```

As with the other modal operators, we can try to define axioms to capture the properties of plans and goals, for example,

```
Goal(a,g,t) Ø K(a,Goal(a,g,t),t)
Goal(a,g,t) & B(a,gØh,t) Ø Goal(a,h,t)
```

Attempts to develop theoretical treatments of plans and goals usually begin by adopting a set of primitive modal operators and then proceed to define new concepts in terms of them. Cohen and Levesque (1990a) opt for four primitive operators: `Believes`, `Goal`, `Happens` (what event happens next) and `Done` (which event has just occurred). Implicit in the last two is a view that the world can be described as a linear sequence of events. The properties of the primitive operators are defined by a set of propositions, in the case of `Believes` from weak S4 modal logic, in the case of `Happens` and `Done` from dynamic logic (Harel, 1979), and in the case of `Goal` an additional set, including:

```
Goal(a,g) Ø ~Goal(a,~g)
Goal(a,g) & Goal(a,gØh) Ø Goal(a,h)
B(a,Happens(a,e)) Ø Goal(a,Happens(a,e))
```

Further modal operators (such as `Eventually`, `Always`, `Before`, `Later`) are then defined directly in terms of the primitives, and gradually a more complex edifice built up. For example, an 'achievement goal' `A-goal` (that is, a goal

which the agent believes to be currently false, as opposed to a 'maintenance goal' which the agent believes to be true) may be defined:

```
A-goal(a,g) + B(a,~g) & Goal(a,Later(g))
```
and then a 'persistent goal':

```
P-goal(a,g) + A-goal(a,g) & Before((B(a,g) ∨ B(a,Always(~g)),~Goal(a,Later(g))))
```
that is, an agent a has a persistent goal to achieve g if a wants g to be true later, and a believes g is currently false, and a believes that this state of affairs will continue until a believes g is true or will always be false. An 'intention' to do an action is defined by:

```
Intends(a,action) + P-goal(a,Done(a,B(a,Happens(action))?;action))
```
The semantics which Cohen and Levesque define for '?' and ';' means that this can be interpreted as saying that an agent intends to do an action if it has a persistent goal to believe it is about to do the intended action and then doing it. We can also define an intention in terms of achieving a desired state of affairs (such as `Happy(a)`) rather than carrying out a particular action. Pollack (1990) considers that an agent a has a plan p={p1, p2,...} to achieve g if:

- a believes that executing p1, p2,.. in order will entail performance of g;
- a believes that each pi plays a role in the plan;
- a intends to execute p1, p2,.. in the specified temporal order;
- a intends to execute p as a way of doing g;
- a intends that each pi plays a role in the plan.

In case this brief excursion into philosophically complex issues such as intention seems like a diversion from computational mathetics, it may be remarked that Bauer, Biundo, Dengler, Koehler and Paul (1993) report the use of an interval-based temporal logic with the modal operators `Next`, `Sometimes`, and `Always` to handle plan generation and plan recognition in an intelligent help system aimed at providing help to users of software systems. The system attempts to derive a plan from its domain knowledge by proving a specification formula. Planning is therefore seen as a kind of automatic programming.

However, the above analyses have not addressed the problem of interleaving planning and acting, which is conceded to be "hard to implement and even harder to formalize" (Davis, 1990). This neglect, with its unreasonable assumptions about the nature of planning and its practical intractability for time-constrained planning, has led to arguments that the underlying assumption that agents decide on goals and plans and reason about them (indeed, the underlying assumption of symbolic AI itself) is misguided. Instead, agents' behaviour is to be seen as generated by simple, direct processes stimulated by the environment.

So, *reactive planning* involves little inference or prediction but relies upon a library specifying an appropriate action given a goal and current situation. The view is that most behaviour is 'routine' in that most tasks, once learned, can be accomplished straightforwardly. Kaelbling and Rosenschein (1990), for example, describe a 'situated automata' in which a declarative specification of knowledge in a modal logic is compiled into a digital machine capable of efficient action. Pollack (1992) expresses scepticism about the wider feasibility of such knowledge-compilation techniques. The phrase "once learned" above suggests that we may be even more sceptical from the point of view of computational mathetics, because it is precisely the transition from unlearned to learned tasks with which we are concerned. In AI-ED we are not just aiming for models that perform efficiently - we need models of students' planning processes on tasks for which their behaviour is not yet routine. Still, the recent emphasis on the *use* of plans rather than on planning per se leads to a consideration of monitoring activities, which has been stressed in studies of metacognition but relatively neglected in AI.

6.5   Monitoring

For once, we have a term, 'monitoring', which does not correspond to an established area of AI research, at least, not under that name. However, Genesereth and Nilsson (1987) discuss the "process of suspending the process of reasoning, reasoning about that process, and using the results to control subsequent reasoning" (unfortunately calling that process 'reflection'). In a way, monitoring problem-solving progress has been part of AI from the beginning, as AI problems are characteristically those which cannot be solved in a straightforward way. So, for example, the simple idea of an evaluation function to select the next node to expand in a search tree could be regarded as a monitoring activity, although it seems more an intrinsic part of the problem-solving process than a 'suspension' of it. Similarly, the SOAR architecture has a technique of *universal subgoaling* whenever the problem-solving process becomes baulked which might also be considered a monitoring activity.

There is no hard-and-fast boundary but it seems that the term 'monitoring' should be reserved for activities which are significantly different from those of the 'normal' problem-solving process. The latter we have regarded as the application of reasoners to beliefs. Typically, a monitoring activity involves consideration of problem-solving events in the near past and future. The list of conditions of when to abandon a solution path in AlgebraLand (given earlier) describe monitoring activities:

```
{ Number-of-applicable-reasoners(n) & n>5 ∅ abandon-path,
```

```
Last-result(x) & Surprising(x) ∅ abandon-path,
...        { r1, r2,
           ...        { b1, b2,... } } }
```
Determining the number of applicable reasoners involves peeking ahead but not actually applying any of them; considering the last result is obviously a reference to a recent event.

It is a desirable feature of monitoring activities that they be efficient to carry out but have the prospect of effecting significant improvements in overall performance. They tend to involve reasoning about inferences without actually (yet) making the inferences. Formal treatments therefore involve the use of metareasoning techniques, as discussed in section 6.3. Such formalisations involve considerable technical difficulties. For example, to say in a meta-level that an object-level concept (such as modus tolens) is difficult we cannot simply say

```
Difficult(φ∅ψ, ~ψ => ~φ)
```

because the symbols within the brackets are not acceptable parameters. We have to name the concept, as we did earlier, and say `Difficult(modus-tolens)`, in which case we have to somehow relate the name to the concept, or we must quote the concept:

```
Difficult("φ∅ψ, ~ψ => ~φ")
```

In the latter case, we then have to be prepared to 'unquote' variables, because we might want to say, for example, that a variable in one expression is the same as that in another. Then it becomes necessary to consider variables to be of different 'types' (as suggested in section 5.1.2) so that, for example, a metareasoner does not confuse an object and a relation. Once such a technical apparatus is defined it is possible to give a rigorous statement of, for example, the process of reasoning in predicate logic using the rule of resolution.

However, although this is mathematical satisfying because it gives us a full definition of otherwise implicit processes, there is a danger of missing the point from a computational mathetics perspective. In the worst case, all that is achieved is that a relatively simple reasoning process is transformed into a more complex metareasoning process, a single step of the former now corresponding to many steps of the latter. As Ginsberg (1993) says "control of reasoning is a domain where it is better to draw inaccurate conclusions quickly than to draw accurate conclusions slowly".

The fact that realistic agents have limited resources to solve problems has led some to advocate a new definition of rationality in AI. The standard AI definition of rationality (discussed in section 4.1) is that a rational agent will carry out a certain action if it has knowledge that one of its goals can be achieved by that action. This definition takes no account of the resources an agent might need to expend to conclude that its goals will be achieved. Instead, Russell, Subramanian and Parr (1993) propose the concept of *bounded optimality*, in which the utility of a decision is a function of both its quality and the time taken to choose it. In such a scheme, there has to be explicit consideration of the expected benefits and costs of potential actions. Moreover, there has to be explicit consideration of the expected benefits and costs of deliberations about those actions (and so on, in principle). At the moment, there are few established techniques of resource-bounded reasoning which can be co-opted for the purposes of computational mathetics.

6.6   Reflecting

The term 'reflection' has been so generally used in the educational and psychological literature that it has come to mean little more than the everyday notion of 'quiet, careful and long contemplation'. Locke (1690) contrasted two sources of ideas: sensation and reflection, the latter being "the perception of the state of our own minds" or "the notice which the mind takes of its own operation". Dewey (1938) defined reflection to be "the active, persistent and careful consideration of any belief or supposed form of knowledge in the light of the grounds that support it". Vygotsky (1978) wanted reflection to be "the transferral of argumentation to an internal level". Piaget (1976) referred to "reflected abstraction", which is "linked to both awareness and conceptualisation" (and contrasted with "empirical abstraction", which is "an activity carried out on the objects themselves"). These are rather all-purpose definitions. We will restrict the notion of reflection to refer to the consideration of the problem-solving process, after the completion of that process, with the aim of improving subsequent problem-solving activities.

6.6.1   Reflective learning

Consider the following extract (Self, 1995) from two students contemplating their solution trace from an AlgebraLand-like system:

> It looked quite easier that way .. than this way [right hand branch], this way obviously took more thinking for us.
> Yeah. Cos as a general rule you get all .. the things you dislike on the right hand side first don't you.
> Yeah.
> In this case, get rid of that nasty 2, and then you expand, cos you wouldn't sort of expand .. if there was an x where that 2 is then we would've expanded wouldn't we.
> Yeah.

But you just don't, that's the general rule, you just don't.
..

This appears to be two students reflecting on their use of monitors. They propose a monitor ('first get the things you dislike on the right hand side') and then refine that rule. To take a simpler example, imagine a student who has (been ascribed) the monitor 'first multiply out brackets' who, after contemplating a tortuous solution attempt, realises that in some circumstances, for example, with

```
        y x (y - 3) + 5 x (y - 3) = 0
```

or, in general, when the expressions within the brackets are the same, it might be better to collect up terms. In terms of our three-level framework, this appears to involve the use of a fourth level, meta to the monitoring level, because it operates on monitors.

Therefore, we will ascribe to an agent a set of 'reflectors' `Rf`. Whereas the set of reasoners `R` maps the agent's beliefs Δ into a new set of beliefs Δ', and the set of monitors `M` maps `RxΔ` into `(RxΔ)'`, the set of reflectors `Rf` maps `MxRxΔ` into `(MxRxΔ)'`. In general terms, then, a reasoner is a function of its 'lower level' Δ and produces results at this lower level, a monitor is a function of both its lower levels (`R` and Δ) and produces results at its lower levels, and a reflector is a function of all three of its lower levels (`M`, `R` and Δ) and produces results at its lower levels. We can extend our notation to include reflectors, as illustrated by:

```
        reflectors: { ref1: condition ∅ refine(m1), ...
                monitors:  { m1: Has-brackets(exp) ∅ remove-brackets(exp),
                           m2: More-complex(exp) ∅ abandon-path, ...
                      reasoners: { remove-brackets: ..., ...
                             beliefs: { 6X+9X+4 = 26-7,
                                      26-7 = 19, ... }
                             goals:   ...
                             plans:   ... } } }
```

It is hard to give a formal description of reflectors as we are, after all, concerned with processes which edit other processes. The idea in the above case is that the `condition` of `ref1` carries out some analysis of the problem-solving trace, which is represented by problem-specific beliefs (goals and plans), which record which reasoners were applied to which beliefs under the recommendation of which monitors, and concludes in this case that monitor `m1` has led to an inefficient solution and should therefore be refined. The `refine(m1)` determines how `m1` should be changed and (if we are optimistic) leads, if executed, to the new monitors:

```
        m1a: Has-brackets(exp) & Different(terms-in-brackets) ∅ remove-brackets(exp),
        m1b: Has-brackets(exp) & Same(terms-in-brackets) ∅ collect-products(exp),
```

From the definition, a reflector can change reasoners and beliefs as well as monitors, which seems reasonable because we can imagine an agent deciding, after solving a problem and perhaps deriving an apparently wrong result, that its reasoners or beliefs were, in fact, unsound. But also from the definition, a monitor can change reasoners and beliefs too, which again seems reasonable, as agents may come to such realisations during as well as after problem-solving. Thus, there will be some similarity between some monitors and some reflectors, the difference lying mainly in the conditions which activate them. However, reflectors can change monitors, which monitors themselves cannot do, and this is clearly a different kind of process from changing a reasoner or a belief.

At this stage, we are merely trying to distinguish various processes which have been conflated and provide a framework for defining them. It must remain questionable whether dialogues such as the above could be unravelled sufficiently for specific reflectors, monitors, and so on to be expressed in a computationally manageable form. Protocols of students using systems such as AlgebaLand show that most of the discussion is at the belief and reasoner levels, despite the fact that such environments have been specifically designed to promote reflective and monitoring activities. If an environment had some explicit understanding of the upper levels then it would be more likely that it could at least make appropriate instructional interventions to provoke the desired activities.

As each new level is introduced, the discussions of the lower levels are echoed for that new level, although it gets progressively harder to give convincingly precise illustrations (at the top-most level, we are after all concerned with the purpose of life and other non-trivial matters). For example, we may distinguish concrete and abstract reflectors, although we may suspect that not much that is concrete will float up to such levels. Similarly, domain-specific and domain-independent reflectors may be defined, although the former may be uncommon. Also, just as we considered monitors which were functions of the sets of reasoners and beliefs (rather than individual reasoners and beliefs), so we can consider reflectors which are functions of the set of monitors (rather than individual monitors). For example, a student who says "Whenever there are very many things I could do I never know what to do" is commenting on the absence of a suitable monitor.

The way in which control is shared among the levels cannot be pre-determined, because it must depend on the nature of the problem and indeed of the agent (some agents are more impulsive, others more reflective, for example). We imagine that most of the time an agent is applying reasoners, under the guidance of monitors, to derive new beliefs, and that occasionally the lower levels will set 'triggers' (perhaps corresponding to verbalisations such as "I am

confused", "Why am I doing this?", "I have finished", "That took much too long") which may be detected and acted upon by appropriate reflectors.

This four-level framework has been given (Self, 1995) the ridiculous acronym of Dormorbile (for DOmain, Reasoning, MOnitoring and Reflection Basis for Intelligent Learning Environments), in honour of the Dormobile Company which went bankrupt in 1994 after thirty years' production of the fondly remembered dormobile, a van-cum-bedroom-cum-house.

6.6.2   *Self-explanation*

A reflective activity which has recently been emphasised, because it is one of the few which has been empirically shown to have learning benefits, is that of self-explanation (Chi, Bassok, Lewis, Reimann and Glaser, 1989). When students were presented with example problems with solutions it was found that students previously identified as good problem-solvers produced more explanations of the examples to themselves and more reliably assessed their own understanding than poor problem-solvers.

An example is invariably incomplete - in our terms, it does not indicate all the monitors, reasoners and beliefs which were used to generate the solution, nor how these various components were combined to generate the solution. Generating a self-explanation involves the application of a reflector which attempts to fill in those gaps (by identifying the missing components or specifying how they relate to one another) or at least to recognise the existence of the gaps. It seems entirely reasonable that such an activity would improve subsequent problem-solving performance, at least on similar problems.

As the process of self-explanation is considered beneficial, it is natural to wonder if it can be taught. Bielaczyc, Pirolli and Brown (1993) describe an experiment involved students explaining to themselves someone else's solution attempt and a teacher overseeing this process and intervening when necessary to clarify what is meant by an explanation. The main focus then is on `Reflectors(s)`, the student's processes that reflect on a problem solution. The teacher is engaged in a fifth level, which oversees and is intended to change the student's reflective processes.

We can anticipate that some examples will be better than others (depending on the student involved) at generating productive self-explanations. Almost complete examples leave little to explain; very brief examples leave too much to explain; and for intermediate examples, some gaps are likely to be more productive than others. Moreover, the mode of presentation of the example can have an effect. For example, Wilkin (1994) shows that the benefits of diagrammatic reasoning (discussed in section 5.8) can be counterproductive in causing poor learners to rely on diagrammatic features of example diagrams rather than generate conceptual self-explanations.

6.7   Transfer

As metacognition concerns partly domain-independent skills, it seems to be intimately related with the contentious topic of transfer - "the degree to which a behaviour is repeated in a new situation" (Detterman, 1993). Thousands of psychological studies of transfer have led to a conclusion that "there has been no positive evidence of general transfer besides a few highly questionable studies" (Singley and Anderson, 1989) - a conclusion which seems to invalidate a fundamental premise of the modern educational system, that behaviour learned in classrooms will transfer to the world outside classrooms.

We can illustrate the issues by briefly reviewing a study of transfer by White (1993). A curriculum was devised around a series of computer microworlds concerned with how forces affect the motion of objects (briefly mentioned in section 2.2.3). Typical computer-based activities involved students applying impulses to screen objects to produce a desired motion. Subsequently, students were asked (among other things) to solve a 'transfer task': "Suppose that we have two identical rivers with two identical boats trying to cross those rivers. The only difference is that one river has a current flowing and the other does not. Both boats have the same motors and leave at the same time. Which boat gets to the other side first?" The experimental subjects did as well as high school physics students (six years older) and better than high school students who had not yet taken their physics course.

Assuming for the moment that transfer does in fact occur, the prevalent view of how it occurs is as follows. A student learns a representation $r1$ for mediating behaviour in the initial situation $t1$. In a new situation $t2$ transfer will occur if either $r1$ can be used to behave in the new situation or if $r1$ can be adapted to a new representation $r2$ (on the basis of the differences between $t1$ and $t2$) suitable for $t2$. In the first case, $r1$ must be sufficiently abstract that it may support behaviour in both situations, assuming the situations are significantly different. In the second case, the student must have mechanisms for adapting her representations. In both cases, the student must realise that the first situation is relevant to the second. Various theories adopt different versions of this view, using, for example, different representations such as schemata (Reed, 1993) and production systems (Singley and Anderson, 1989). Such a view has been implicit in our development of various schemata for reasoners, monitors and reflectors.

However, doubts about the occurrence of transfer follow directly from the definition above: how new does a situation have to be to count as new?, and how close does behaviour have to be to count as a repetition? In the experiment described above, for example, to what degree is the river crossing problem a new problem? These loopholes allow doubters to assert that "studies that claim transfer often tell subjects to transfer or use a 'trick' to call the subject's attention to the similarity of the two problems" (Detterman, 1993). From the point of view of the student, the use of such hints or tricks is not surprising - after all, when they come to the situation `t2` they have literally thousands of preceding situations which might be relevant, not just the `t1` which might seem obvious to an experimenter.

If transfer is rare then this is clearly a challenge for education. We might conclude, as Detterman (1993) does, that as transfer does not occur we must teach students 'the facts', that is, teach them directly what we want them to learn. We will refrain from entering the realms of emotional educational philosophy. Interestingly, the same starting point (the empirical observation that transfer is difficult to achieve) is also the basis for the radically different philosophy of situated learning and cognitive apprenticeship, which places issues of metacognition and transfer at its core.

At first glance, the notion of transfer presents a paradox for situated cognition, for if internal representations are denied then what is there to transfer? Brown, Collins and Duguid (1988) argue that "a situated theory of knowledge challenges the widely held belief that the abstraction of knowledge from situations is the key to transferability" but give no clear alternative view. Greeno, Smith and Moore (1993) discuss the 'transfer of situated learning' in a chapter three times longer than any other chapter in a book on transfer (Detterman and Sternberg, 1993). Clearly, the issue is complex and so it will be safer to give Greeno et al's own conclusions. In their view,

"Symbolic cognitive representations can play an important role in transfer, but they are considered as instrumental parts of the activities that occur in the initial learning and transfer situations, rather than being fundamental and ubiquitous... Our focus is on activities rather than representations. Transfer, in this view, depends on transformations of activity and is enabled by structural invariance in the interactions of agents in situations; these interactions can be described as action schemata, referring to the organizing principle of the activity rather than to symbolic cognitive representations."

Vera and Simon (1993), however, conclude that although transfer "is a fundamentally important problem, which calls for continuing and expanded study", it "has nothing to do with the adequacy of symbolic systems as theories of intelligent action, in schools or in the real world."

The topic of transfer is a good illustration of the approach of computational mathetics, which is (or should be) completely neutral with respect to the above arguments. Whether transfer occurs or not is a matter for psycho-mathetics. The educational implications of whatever degree of transfer exists is a matter for educational mathetics. Computational mathetics is concerned with saying as precisely as possible whatever needs to be said about transfer, so that it may be applied to the design of AI-ED systems. It is apparently conceded by situationists that symbolic representations have some role in transfer, so the framework we have developed need not be discarded yet. The computational mathetics question is how the new concepts that are introduced can be represented - bearing in mind that these would be representations for our and our systems' benefit, not representations of any kind of psychological reality.

What precisely, then, are these 'action schemata' that Greeno, Smith and Moore (1993) refer to? Previously, we included 'problem-specific beliefs' among the set of beliefs. These include beliefs about the problem situation (in short, the 'situation') and the problem states reached and operators applied during problem-solving (in short, the 'actions'). Making no assumptions about how these are represented and remembering that this is an ascription (by us) to an agent, so that, for example, the `situation` is our ascription to the agent of how it views the situation and not a representation of how the situation objectively is, we may also separate them out, if we wish:

```
reflectors: { ref1: condition ∅ refine(m1), ...
        monitors:  { m1: Has-brackets(exp) ∅ remove-brackets(exp),
                m2: More-complex(exp) ∅ abandon-path, ...
            reasoners: { remove-brackets: ..., ...
                    beliefs:   ...
                    actions:   ...
                    situation: ...
                    goals:     ...
                    plans:     ... } } }
```

If the situated learning view is (simply) that, in some cases, transfer is mediated by action schemata rather than any other schemata, then there appears to be little problem in conceding that. The challenge is to define ways of representing the action schemata so that notions which Greeno et al discuss, such as invariance and affordance, can be made sufficiently precise that an AI-ED system could, in principle, reason about situations and actions to make decisions concerning transfer. If it is argued that actions and situations should not be represented in a rigorous form but discussed in long non-technical papers then that is just a methodological preference. If it is argued, as no doubt it is, that they are much more subtle concepts than the above indicates then there is even more reason for trying to say precisely how.

6.8 Distributed metacognition

Teasley and Roschelle (1993) present an analysis of a case study of two students engaged in the "collaborative construction of new problem-solving knowledge". The students' task is to adjust two vectors (denoting initial velocity and acceleration) so that a simulation replicates a target motion. The main theoretical concept used in the analysis is that of a *Joint Problem Space* (JPS). The JPS is a "shared knowledge structure that supports problem-solving activity by integrating (a) goals, (b) descriptions of the current problem state, (c) awareness of available problem-solving actions, and (d) associations that relate goals, features of the current problem state, and available actions".

Collaborative problem-solving is considered to involve jointly constructing a JPS. Conversation between collaborators is concerned with constructing and maintaining a JPS, that is, with introducing and accepting knowledge into the JPS, with monitoring for divergences in meaning, and with repairing divergences that impede the collaboration. Collaborators are engaged in continually building, monitoring and repairing the JPS, aiming for some convergence of meaning and a knowledge structure able to support problem-solving. Actually, it is noted that this process is not continual as collaborative problem-solving involves periods where partners are disengaged, perhaps while one partner is thinking about ideas too ill-formed to introduce into shared work.

Teasley and Roschelle (1993) discuss the role of discourse in mediating collaboration but give few details of the JPS. The only non-natural language descriptions of this knowledge structure are two 'socially-distributed productions', that is, productions whose content is developed by the partners in stages:

```
If the Goal is to adjust the initial speed and the speed 'going up' is too slow
   then make the veocity vector bigger.
If the Goal is to adjust the initial speed and the initial dot spacing is
   greater
   then make the veocity vector longer.
```

The JPS appears to be a version of the concept of common knowledge as developed in distributed AI (section 4.7). A proposition in the JPS is believed by both partners, and believed by both partners to be believed by the other, and so on. If we denote the two partners by `p1` and `p2` and consider the JPS to be a 'virtual agent' `jps`, then collaborative problem-solving is considered to involve the partners comparing `Asc(p1,jps)` with `Asc(p1,p1)`, and `Asc(p2,jps)` with `Asc(p2,p2)`, respectively, i.e. with comparing what they consider to be joint beliefs with their own beliefs. An ascription to `jps` makes the philosophical point that joint goals cannot be analysed solely in terms of individual goals and beliefs. A joint goal is held by both (or all) partners and is mutually believed to be a joint goal.

Teasley and Roschelle appear doubtful of the need or feasibility of developing a precise description of the contents of the JPS, because of the subtleties of natural language collaborative discourse, the complex nature of its integration with on-screen activity, and the fact that students bring idiosyncratic conceptual knowledge which may not correspond to any representation. Instead, they use the JPS notion as a 'conceptual resource' (for them) to understand how students collaborate and do not attempt to incorporate it within their system (the Envisioning Machine, EM) to enable it to have any such understanding. There is no explicit JPS on screen for the students to work with. Thus, although they discuss the benefits of EM (and similar computer-based systems) as a mediator of collaboration - namely, that it provides a means of disambiguating natural language and resolving impasses and that its display invites and constrains students' interpretations - in fact, EM is more a conduit for collaboration than a mediator of it. The dictionary definition of 'to mediate' is 'to intervene in order to bring about agreement' and EM makes no such interventions as it has no understanding which would enable it to.

At this stage, it is an open question whether the aim of computational mathetics for precision will not only improve our conceptual resources but also enable systems themselves to reason appropriately to determine interventions, in this case, to promote collaborative learning. It is at least plausible that some on-screen representation of the JPS would be useful. It might enable students to state more clearly what they believe to be in the JPS and so reduce ambiguities. It might also enable the system to monitor the contents of the JPS, for example, to check that the JPS is consistent with observations. It might detect vagueness or incompleteness in the JPS. For example, it could wonder about the difference, if any, between `bigger` and `longer` in the two socially-distributed productions given above. As always, it is a separate question as to what kind of intervention might be appropriate in such a circumstance.

Interestingly, most related formal AI has concerned the specific problem of collaborative planning, though plans are not mentioned as parts of a JPS. This work attempts to distinguish carefully between situations where two (or more) agents have their own plans for achieving either a shared goal or their own goals. The agents need to coordinate the components of their plans to achieve the joint goal or all goals (if possible). The possible roles of a mediator need also to be defined. The mediator may be one of the problem-solving agents or an extra agent specifically given the task of overseeing the coordinated planning activity. In either case, the mediator's ability depends on its knowledge of the other agents' goals and plans (and of their knowledge of each other's goals and plans), and also on its knowledge of the beliefs and knowledge from which these goals and plans have been derived, because a successful mediation will probably involve consideration of the grounds for any proposed action. In addition, any formalisation of collaborative

activity needs to take account of the different nature of agents - human or computer-based - as they have different capabilities for rationality, reliability, self-centredness, and so on.

Grosz and Kraus (1993) present a formulation of collaborative planning derived from Pollack's model of individual planning and using the modal operators of intentionality of Cohen and Levesque (1990a), discussed in section 6.4. They also introduce the notion of a 'potential intention', i.e. an intention an agent is considering adopting but to which it is not yet committed. Wooldridge and Jennings (1994b) develop a formalisation based on the concepts of commitment and convention, the latter being a specification of the conditions under which a commitment might be abandoned and how an agent should behave in such a circumstance (and is a kind of monitor, in our terms). The notational details are too complex for our purposes here. Let us just summarise some of the questions that might need to be considered when developing a formalisation adequate for the purposes of computational mathetics:

- To what extent do agents actually have (complete) plans and are these plans known to other agents?
- Assuming that plans are carried out by actions, to what extent do the agents have an agreed understanding of these actions? Can they communicate with each other about the actions?
- What kinds of negotiation are possible between agents? Are there, for example, any 'social laws' which prevent any one agent being delegated an unfair share of activity to carry out?
- Can the problem-solving activity be devolved into independent, concurrent activities, so that the activity is more one of cooperation than collaboration?
- Are the mediating interventions (if any) directed solely towards solving a specific problem, or may they relate to longer-term learning benefits?

6.9   Attributes, aptitudes and attitudes

Until recently, AI-ED research has tended to emphasise the role of knowledge, implicitly agreeing with Chi, Glaser and Rees (1982) that students' difficulties "can be attributed mainly to inadequacies of their knowledge base and not to limitations in either the architecture of their cognitive systems or processing capabilities". Now, there is more attention on cognitive and metacognitive processes, some of which may not be related to knowledge per se but to more or less intrinsic characteristics of the individual learner. Some researchers have argued that an AI-ED system needs to be aware of such characteristics in order to provide individualised learning experiences. Our problem, from a computational mathetics perspective, is to determine how to represent those characteristics which AI-ED systems need to be aware of.

Folk psychology suggests a wide range of possibly relevant characteristics: individuals might be described as persistent, compulsive, systematic, sagacious, and so on, but not all adjectives (for example, greedy, fragrant, devout) need feature in a dictionary of computational mathetics. However, the huge literature on individual differences has generally been rather sceptical of the prospect of reliably identifying and adapting to such differences (Corno and Snow, 1986). Nevertheless, let us try to disentangle the kinds of characteristic which might be involved and speculate on how they might be handled within computational mathetics.

Agent-oriented programming is not intrinsically opposed to the notion of 'characteristics' despite the previous focus on the role of knowledge or belief and how it is processed. An agent is considered to possess 'attitudinal states' (such as wants, intentions, and likes) and to pursue its activities in the light of those attitudes. Just as it is useful to ascribe knowledge to an agent, so it may be useful to ascribe characteristics (if, for example, it enables predictions of the behaviour of the agent or enables some other agent, such as an AI-ED system, to adapt to that agent). There is no conflict with the spirit of agent theories - though there is considerable difficulty in integrating the ideas satisfactorily.

A characteristic (such as 'persistent') is not the kind of thing which can be incorporated within any of our levels. Rather, it is a label which describes the content of such levels. Therefore, we will, as anticipated in section 6.3, say that an agent `a` which has been given the ascription `Asc(a)` may be ascribed the characteristic C if some associated boolean function `c` is true if applied to `Asc(a)`. For example, the boolean function associated with `persistent` might return true if no monitor contains `abandon-path`. We will attach such a label to the highest level in our nested framework to which the associated function refers, for example:

```
reflectors: { [reflective, ...]
                 ref1: condition ∅ refine(m1), ...
        monitors:  { [persistent, ...]
                       m1: Has-brackets(exp) ∅ remove_brackets(exp), ...
                reasoners: { [irrational, ...]
                              remove-brackets: ..., ...
                   beliefs:  { [unix-expert, ...]
                                b1, b2, ... }
                   actions:   ...
                   situation: ...
                   goals:     ...
                   plans:     ... } } }
```
This notation is intended to indicate that an observer has made an ascription such that `Unix-expert(a)`, `Irrational(a)`, etc. are considered to hold.

Once such a definition is proposed it becomes, as is the style in computational mathetics, a challenge for precision. For example, is someone a unix-expert only on the basis of what they know but does the label imply some expertise in using that knowledge? Would 'fuzzy functions', returning a value in the range 0 to 1, be better than boolean functions? Maybe, whatever precision is needed will evolve from such computational descriptions rather than through the informal educational and psychological literature, where such terms are used as though it is obvious what they mean. Formal definitions of terms such as 'narrow-minded', 'compulsively reflective', 'arrogant', 'persistent', 'cooperative', 'gullible', and so on already exist in the AI literature. Such terms are used (perhaps semi-seriously) without any psychological claims, but at least they provide a benchmark against which educational psychologists can try to define their own terms when they use them.

Before considering particular kinds of characteristic, some general points can be made. First, from the point of view of implementation, it is hard to imagine an AI-ED system continually reapplying a large number of such functions every time some behaviour occurs in order to determine if any have become true. Rather, we imagine such ascriptions to be 'interrupt-driven', that is, to be initiated by relatively simple-to-evaluate 'trigger functions' which detect particular events and directly cause the ascription or provoke some more lengthy analysis. For example, if a student is detected to use the unix command 'grep' then she might be immediately ascribed the label `unix-expert`.

There must be some 'payoff' from making such an ascription, and this may be of two kinds. First, the ascription usually allows a large number of inferences to be drawn. For example, once a student is labelled as a `unix-expert`, then she may be assumed to know a great many things about Unix. Obviously, these inferences are default assumptions (section 5.3). Secondly, the ascription may play a role in the system's instructional strategy. For example, it may well be more convenient to have a rule of the form 'if student is a unix-expert then ..' than a complicated rule in terms of the beliefs ascribed to the student.

The permanence or otherwise of such ascriptions differs:
- some characteristics (e.g. knowledge of a particular law of physics) are transient and it may be the AI-ED system's aim to change them,
- some (e.g. blindness) may be permanent but nonetheless of pedagogic concern if not focus,
- some (e.g. the status of being a unix-expert) seem to be long-term once acquired,
- some (e.g. persistence) it may be only implicitly part of the system's aim to change,
- some (e.g. anxiety) may be situation-dependent.

The value of relatively permanent characteristics may be determined off-line, through psychological tests. On-line interrogation concerning characteristics is likely to be of little use. The on-line assignment of student characteristics on the basis of a particular event (or series of events) can be problematic. Overall, there is a difficult problem in keeping an ascription of characteristics in line with observations. In section 7.2 we will consider the problem of 'belief revision', which, as the term suggests, is concerned with the revision of the `beliefs` components, but so far very little work has been done on the revision of other components.

6.9.1   Stereotypes

The stereotype is one of the most common kinds of characteristic to be ascribed to agents (Rich, 1989; Hustadt, 1994; Kay, 1994). An agent ascribed a particular stereotype may also be ascribed various other mental components, either by definition or by default. Normally, the stereotypes are arranged into a hierarchical structure (such as Figure 6.2), which permits the inheritance of properties. A set of propositions is associated with each node, representing a stereotype, such that if the agent is ascribed that stereotype then it is also ascribed those propositions, together with those propositions attached to any encompassing stereotypes. We might also assume that the agent should not be ascribed those propositions attached to sub-stereotypes. In general, an agent may be assigned to stereotypes along several dimensions, leading to the possibility of inconsistencies in the default assumptions. In the absence of information about the agent, it might be assigned to the 'any fool' stereotype, such that it is assumed to believe only those things which any fool believes, that is, so-called common knowledge.
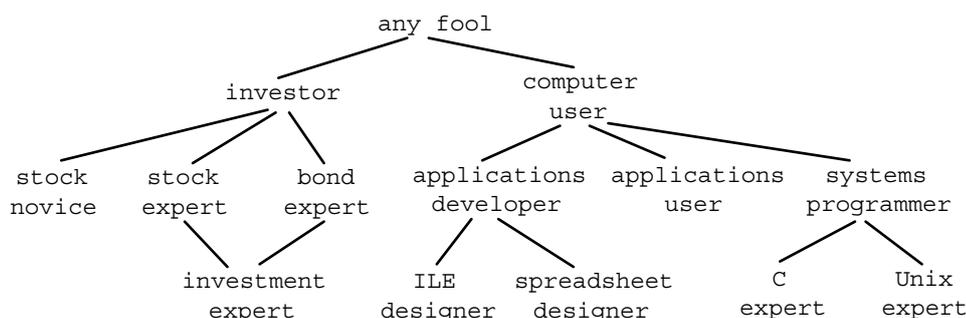

Figure 6.2. A stereotype hierarchy.

As with other ascriptions, stereotypes may be ascribed by one agent to another, to any depth, for example,

        B(Hansel,B(mother,gullible(Hansel)))

Also, we can imagine the need to ascribe stereotypes to virtual agents denoting groups rather than individuals, as in the previous section:

        B(Russia,B(China,decadent(USA)))

As with other nested ascriptions, we need to be careful about the inferences we make. The definition of a stereotype (such as `decadent`), the kinds of assumptions that follow, and the stereotype hierarchy might well differ from one agent to another. In this case, we would need to ascribe all these components to agents as well, and not regard them as objectively given as 'correct'.

At this point, it seems desirable to disassociate ourselves from the negative connotations of stereotypical reasoning for commonsense inference, which is a matter of its everyday use, not its formal properties. Formally, stereotypical reasoning is just a version of default reasoning which endorses an unusually large number of default assumptions.

Stereotypes are useful for initialising user models generally, especially for systems which do not anticipate much subsequent change in the content of the models (which should not be the case for AI-ED contexts). For student modelling, where the focus is on the subsequent dynamic tracking of changes in the user (i.e. student), stereotypes are not of much use beyond the initialisation stage because they do not permit the necessary fine-grained analysis.

6.9.2   Aptitudes

Corno and Snow (1986) distinguish three kinds of aptitude which have a bearing on learning:
- *Cognitive aptitudes*, i.e. those concerned with (prior) knowledge and intellectual abilities. Many stereotypes are of this kind.
- *Affective aptitudes*, i.e. concerned with values, including issues of motivation, anxiety, autonomy, and self-concept. These will be considered further in the next section.
- *Conative aptitudes*, i.e. concerned with wants, intentions, etc. and, in the educational context, cognitive and learning styles. Many such styles have been studied, usually in terms of contrasts, such as holist v. serialist, reflective v. impulsive, and convergent v. divergent. These attributes seem to refer to global properties (rather than the performance properties, as with cognitive attributes) of a meta-level of an ascription. For example, a 'holist' style refers to the general strategy of the learning component. Similarly, 'reflective' refers to the number and type of meta-level interruptions on the base-level problem-solving.

It is probably conative aptitudes which believers in aptitude-treatment interactions (Snow, 1990) are most optimistic of applying to AI-ED. A representative study (although in fact there have been very few studies to represent) is reported by Shute (1993). She focusses on two aptitudes, working memory capacity (WM) and general knowledge (GK), and two treatments, 'constrained' and 'extended', which differed only in the number of problems presented to the learner. The WM aptitude is related to the potential size of the various sets ascribed to the learner and depends on the active-passive distinction, which we have largely ignored. The GK aptitude is concerned with what we called 'background knowledge', knowledge not specific to a particular domain or problem. Both aptitudes were measured by a battery of standard cognitive tests. It was found that low-WM, high-GK students learned more from the constrained environment, and that high-WM, low-GK students learned more from the extended environment.

6.9.3   Affects

The 'affective dimension', concerned with a student's motivation, emotions, and feelings, is often described as though it were a dimension orthogonal to the 'cognitive dimension' which AI, cognitive science, AI-ED and computational mathetics tend to emphasise. AI-ED research is said to neglect the affective dimension, considered by many to be the more important of the two. For example, Lepper, Woolverton, Mumme and Gurtner (1993) write that

> "Implicit in the design of many current computer tutors is a conception of the tutoring process derived from a purely cognitive analysis of teaching... issues of student affect, motivation, and attention are simply not considered in such models".

This criticism is apparently valid but nonetheless odd. On the one hand, critics deny the need for the kinds of detailed, explicit representations of domain and pedagogical knowledge which AI-ED researchers try to develop, arguing that a suitably designed environment can behave appropriately with only implicit knowledge, but, on the other hand, they criticise AI-ED designers for not explicitly considering and representing motivation, etc., without allowing that AI-ED systems might, through their behaviour, exhibit an adequate implicit knowledge of such factors. After all, there are many tales of students astonishing teachers with their enthusiasm for voluntarily using systems (AI-ED and otherwise) where neither the designers nor the systems themselves explicitly considered issues such as motivation.

Nevertheless, the challenge to be explicit about affective factors is well worth facing, if only to try to clarify the bountiful literature on the topic. Lepper et al (1993) consider that the four major affective goals are:
- confidence - to enhance the learner's level of confidence;

- challenge - to produce an appropriate level of challenge for the learner;
- control - to maintain in the learner a sense of personal control;
- curiosity - to elicit from the learner a high level of curiosity.

It is interesting that these goals are expressed as goals for a teacher with respect to a learner, rather than in terms of goals of a learner. Although the study of human tutors by McArthur, Stasz and Zmuidzinas (1990) led them to consider that "many tutorial actions appeared to fulfil a motivational function while also accomplishing some other [informational] purpose", Lepper et al propose that a tutor needs two complementary diagnostic models, one cognitive and one affective.

Del Soldato and du Boulay (1995) take this recommendation literally. Their system maintains two independent student models. The model of the student's motivational state consists of just three variables:
- `confidence` - determined by whether the student succeeds or fails with or without help, and whether she asks for help and accepts help offered;
- `effort` - determined by the number of attempts to obtain a solution;
- `independence` - determined by the number and nature of tutor interventions.

These variables are functions of the overt behaviour of the student and tutor, and not of any ascription to explain that behaviour, and so could be associated, as described above, with the `actions` component of our framework. Clearly, a deeper analysis is possible, though maybe unnecessary. For example, `effort` is presumably related to the `persistent` characteristic discussed above.

As with all the other aspects we have discussed, affective factors have two potential roles in computational mathetics: they might be needed to describe the agents involved, and they might also be needed because the problem which the student (and the system) is considering concerns those factors. Elliott (1993) describes a training system designed to teach novice account executives how to sell telephone-book advertising. It is assumed that solutions to such problems involve considering the effect that actions might have on the emotions of the people involved. Clients are considered to be of four personality types (`dominant`, `political`, `steady` and `wary`). Eighty-six domain rules express how actions affect clients' emotions, for example:

```
If the seller takes charge with a dominant client then the client will become
    angry
```

Twenty-one 'emotion types' are identified, such as `joy`, `distress`, `happy-for`, `gloating`, and `resentment`. Superficial though such an approach may seem, it was "successful in giving agents the ability to appraise situations so that their concerns led to emotions consistent with their intended personalities." However, it was "not well suited for representing dispositional behaviours rooted in more cognitive motivations. A better solution would have to additionally integrate some representation of individuals' long term planning .. strategies."

7. Learning

As learning is the intended outcome of using any AI-ED system, computational mathetics needs to help develop well-articulated theories of learning so that we and systems may understand and perhaps predict those learning outcomes (or lack of them). Perhaps, if a theory of learning is sufficiently precise, it may lead directly to a theory of instruction, that is, if a theory of learning can predict the potential learning outcomes from possible instructional actions, then a theory of instruction may select or generate actions which are in some way optimal (chapter 10). Moreover, if a student learns, as is the intention, while using an AI-ED system, then a theory of learning will be needed to update any ascription of knowledge to the learner.

The topic of learning has been a major field of psychology for over a century. In AI, there has been a profusion of research on machine learning, with some researchers considering an ability to learn to be a necessary characteristic of any AI system. Interestingly, the educational literature has relatively little discussion of learning, but even so there is a huge volume of work to try to relate to computational mathetics.

In previous chapters we have alluded to various learning processes. In terms of our framework, we have reasoners changing beliefs, etc., monitors changing reasoners and beliefs, reflectors changing monitors, reasoners and beliefs, and so on. These changes are assumed to occur before, during or after problem-solving. In some cases, for example, a reasoner changing the situation, we imagine the change to be relatively transient. Such a change might not persist beyond the current problem. Other changes, for example, a reflector changing a monitor, perhaps after lengthy consideration of a problem-solving failure, we imagine to be more long-term: indeed, it appears to be the whole point of such a change to affect subsequent problem-solving. Such a distinction is not clearcut. For example, an agent might well recall the details of a problem-solving situation when tackling a similar problem later (this is the basis for case-based learning, discussed below).

In addition to learning *through* problem-solving, almost as a side-effect, we might also view learning *as* problem-solving. An agent might learn intentionally, through setting up a goal to learn. Maybe this requires the consideration of a dual 'problem space', one devoted specifically to learning rather than to some other problem. But not all learning is intentional. Some learning occurs without awareness and incidentally to some other activity. It is arguable whether this other activity can always be construed as a form of problem-solving. In so far as learning is intentional, then the agent regards it, like other metacognitive activities, as an investment. It requires a commitment of cognitive resources in the hope of benefits in subsequent problem-solving. Of course, not all such intentions will be realised. Indeed, we might comment that, for students, real learning is quite rare and therefore a good model of human learning will perform poorly as a learner. Even more cynically, we must recognise that some students have an intention not to learn, i.e. they prefer to meet some short-term goal such as completing an assignment while investing the minimum effort on any longer-term learning objectives.

As well as learning which involves changing the contents of the various mental ascriptions, there is also a form of learning which involves a kind of 'sedimentation' through the levels. For example, an expert problem solver might have a reasoner corresponding to a 'macro-operator' which a beginner might regard as several smaller operators to be applied in sequence with careful monitoring. As experience is gathered, this monitor plus the smaller operators will coalesce into the macro-operator. In our framework, the coalescing process would be defined as a reflector, operating on monitors and reasoners and producing a new reasoner. This is a form of *knowledge compilation* (Anderson, 1983), leading to more efficient problem-solving.

Within a level, we might not consider the vocabulary fixed. For example, an agent might initially describe reasoners in terms of, say, specific simple arithmetic operations and then, in due course, learn concepts such as 'commutative', 'exponential', and so on, which enable the reasoners to be re-expressed. Again, this may lead to an increase in the efficiency, but not the scope, of problem-solving.

In general, then, we might consider all learning to involve a form of (usually non-deductive) inference. From what an agent already knows, it infers more, in order to improve later problem-solving. Generally, such inferences are nonmonotonic and not restricted to domain knowledge. However, we should not regard learning as a monolithic process and assert that agents learn 'by making mistakes', 'by doing', 'from stories', or any other single activity. Agents learn different kinds of thing, at different times, by different processes and for different purposes. Different kinds of agent (computer and human, specifically) have different learning capabilities, and for human agents these capabilities change with age and maybe with context and culture.

In this chapter, we will distinguish four types of learning, although as usual the boundaries are not clearcut. An agent may learn in three ways - directly from the environment (*perceptual learning*), by some internal cognitive process, or by communicating with other agents (*social learning*). In the second case, we distinguish processes which involve an

analysis of a single piece of evidence (*analytical learning*) and those which involve some kind of induction over several pieces of evidence (*inductive learning*).

7.1   Perceptual learning

Philosophers and cognitive scientists have laboured to clarify how beliefs may be acquired through perception, with little help from AI researchers. More than other forms of reasoning (considered in chapter 5), perception is an unconscious process. However, perception is not an automatic process but involves some form of reasoning, as is shown by the fact that what we come to believe through perception depends on what we already believe and on the conditions under which the perception is made.

Musto and Konolige (1993) attempt a formalization in terms of autoepistemic logic. Their basic axiom is:
```
        Perceives(a,p) & ~B(a,Defeater(Perceives(a,p)∅p))  ∅ B(a,p)
```
that is, if agent `a` perceives `p` then, provided `a` does not believe there is some 'defeater' of the inference, `a` believes `p`. There are considered to be two classes of defeater of `Perceives(a,p)∅p`, namely:

- `B(a,~p)`, that is, `a` already believes the negation of `p`.
- `B(a,Abnormal(p))`, that is, `a` believes that there is some abnormal condition under which `p` has been perceived. For example, `Perceives(a,Pink(elephant))∅Pink(elephant)` might be defeated by a belief that `a` is inebriated or that the elephant has been viewed in a vivid sunset.

They also propose the following causal rule:
```
        p & ~Abnormal(p)  ∅  Perceives(a,p)
```
which is clearly an over-simplification, and a set of default rules to resolve conflicts between perceptions and memory or between two perceptions. This formalisation has been applied to mobile robot perception but not to any AI-ED context.

7.2   Analytical learning

Analytical learning methods involve the detailed analysis of a much smaller number of observations than inductive learning methods. To facilitate such an analysis, analytical methods tend to be based more on the use of prior knowledge to explain and generalise from the observations. Often what is learned is a deductive consequence of this prior knowledge.

7.2.1   Failure-driven learning

Many theories of learning, both in cognitive science and in machine learning, are based on the idea that learning is stimulated by some kind of 'cognitive conflict', that is, some mismatch between what is expected and what actually happens. This 'failure' causes some change to avoid its repetition.

For example, the SOAR architecture (Laird, Rosenbloom and Newell, 1986) proposes that all human learning occurs through a process of 'chunking' after failure. Whenever there is a problem-solving impasse, SOAR enters a meta-level to overcome the difficulty. When the meta-level returns, a new production rule is generated linking the impasse conditions with the remedy found. Subsequently, whenever those conditions are encountered the new rule will apply without any resort to the meta-level. SOAR's learning ability has been made use of in a system mainly concerned with recognising student plans, discussed further in the next chapter (Hill and Johnson, 1995).

An impasse-based theory of learning which has been more influential in AI-ED research is *repair theory* and its successors (Brown and VanLehn, 1980; VanLehn, 1982, 1988, 1990). The basic ideas and evolution of repair theory can be explained with respect to our framework:
```
        reflectors: { ...
                monitors:  { m1: ...,
                         m2: ..., ...,
                         impasse ∅ change-situation & generate-patch,
                         ...
                reasoners: { p1: ...,
                         p2: ..., ...,
                         patch1: ...,
                         patch2: ..., ...
                        beliefs:    ...
                        actions:    ...
                        situation:  ...
                        goals:      ...
                        plans:      ...  } } }
```
A student is assumed to apply productions (`p1`, `p2`, ...) under the control of monitors (`m1`, `m2`, ...) to change the beliefs, situation, etc. When an impasse is reached, i.e. no productions are applicable, a 'repair' changes the situation so that problem-solving may continue. The repair mechanism is supposed to be domain-independent and to cause only small changes. In the first version of the theory, the problem-solving productions were to be unchanged. Consequently, as

nothing has been learned, the same impasse will be reached in similar situations. Some indeterminacy in the repair heuristics may lead to a different repair, causing 'bug migration', the prediction and subsequent detection of which was considered to provide strong support for the theory (VanLehn, 1988). However, as some stable bugs are manifested, it became necessary to postulate that the repair mechanism generates 'patches', that is, simple impasse-repair pairs which (as with SOAR) are added to the set of productions and are henceforth applied in similar impasse situations. The productions and the patches were considered to be two distinct types of reasoner, the latter being relatively abstract (so that they applied to multiple occurrences of an impasse) and operating only on the situation. Eventually, however, this distinction came to be seen as unprincipled and the reasoners were then considered to consist of rules and 'malrules' (Sleeman and Smith, 1981), the latter being identical in form to the rules but generating incorrect rather than correct answers. Moreover, both rules and malrules were considered to be learned by the same mechanism, induction over teachers' examples and incorrect examples produced by local problem-solving. This inductive learning was considered to occur at impasses, overcome by external help (from a teacher or textbook), usually generating rules, or by repairs, usually generating malrules.

7.2.2   *Explanation-based learning*

When an agent reasons to reach a conclusion which turns out to be wrong then it may reason about its reasoning process in order to determine where the difficulty lies and how to overcome it. This is clearly a form of failure-driven learning. However, such a reflective process is not restricted to failures: an agent may reason about a derivation which led to a correct conclusion in order to streamline its subsequent application, by, for example, generalising incidental aspects of the context. This process is called *explanation-based learning* and is, in our terms, a reflector, as it operates upon the problem-solving process.

Explanation-based learning is a technique for improving the efficiency of problem-solving. It begins with a derivation (an explanation) obtained during problem-solving, generalises it, and records the generalisation as a new 'lemma' for problem-solving. It is intended to improve the speed, not the scope, of problem-solving. As such, it appears to be the kind of reflective activity which users of systems such as AlgebraLand are intended to engage in.

It requires the specification (Mitchell, Keller and Kedar-Cabelli, 1986) of:
 • a goal;
 • the training example (an instance to be explained);
 • a complete domain theory (facts and rules);
 • an operationality criterion (specifying the form in which the generalisation is to be expressed).
The process involves (1) generating a proof tree, (2) backpropagating and substituting variables, where possible, for constants, and (3) expressing the goal as a conjunct of proof tree leaves.

Expressed in these terms, explanation-based learning is very similar to the technique of goal regression in logic programming (Nilsson, 1980). Dejong and Mooney (1986) give a rather less rigorous characterisation of a version of explanation-based learning intended to enable learning from a story (such as the Australian aboriginals' brolga story, in section 1.1). 'Explanatory schemata' are to be developed through a set of mechanisms, such as composition (merging two schemata) and volitionalization (adding an agent to an agent-less schema). Subsequently there has been a huge amount of work on clarifying distinctions between different descriptions of explanation-based learning, relaxing some of the formal requirements, and integrating the method with inductive techniques.

Of particular relevance to computational mathetics is the relation between explanation-based learning and self-explanation. It is a premise, or at least a hope, of writers that students can learn by determining why a single example is an instance of a concept. For example, from Clocksin and Mellish (1981):
   "The goal `var(X)` succeeds if `X` is currently an *uninstantiated* variable. Here is an example ...
```
really_in(W,List) :- var(List), !, fail.
really_in(W,[W|_]).
really_in(W,[_|List]) :- really_in(W,List)."
```
To explain this example to oneself in order to understand the concept of `var`, a considerable domain theory is required, as readers unfamiliar with Prolog will appreciate. On the basis of this one example there is plenty of scope for over- or under-generalising the concept.

It certainly seems plausible that learners will benefit from endeavouring to generate and generalise such explanations. However, from an instructional perspective, the process seems risky, as learners' knowledge of prerequisite concepts (such as uninstantiated, `!`, `fail`, and so on) may be shaky. In practice, of course, such instruction is often integrated with other methods, for example, those which may lead to inductive learning.

Cascade (VanLehn, Jones and Chi, 1992) is a computer simulation of self-explanation using a method for resolving impasses and learning new rules called 'explanation-based learning of correctness'. Given a problem such as that shown in Figure 7.1 and an example solution, Cascade attempts to explain each line of the solution. For example, the line
```
Fax = -Fa cos30
```

is converted into an internal form:
```
        projn(force(knot,st_A),axis(knot,x,0)) = -1*mag(force(knot,st_A))*apply(cos,30)
```
which is then proved using Prolog rules such as:
```
        constraint(projn(V,A)=sign(proj(V,A)*mag(V)*trigfn(proj(V,A))) :-
            instance(V,vector),instance(A,axis),origin(A,O),vertex(V,O).
```
The proof generates a set of 'derivational tuples' which records the values derived and the rules used to derive them in order to facilitate subsequent analogies. If a proof fails, then Cascade tries to use an overly general rule, such as
```
        constraint(sign(P(X,Y))=sign(Q(X,Y))).
```
and, if this overcomes the impasse, creates a specialization of it by instantiating the rule and then substituting variables for problem-specific constants, to create, for example:
```
        constraint(sign(proj(force(K,S),axis(K,x,R))=
                    sign(nearest_half_axis(force(K,S),axis(K,x,R)))).
```

```
    Problem: Figure a shows an object of weight W hung by strings.  Consider tl
    knot at the junction of the three strings to be "the body".  The body rema:
    rest under the action of the three forces shown in figure b.  Suppose we a:
    given the magnitude of one of these forces.  How can we find the magnitude
    the other forces?
```
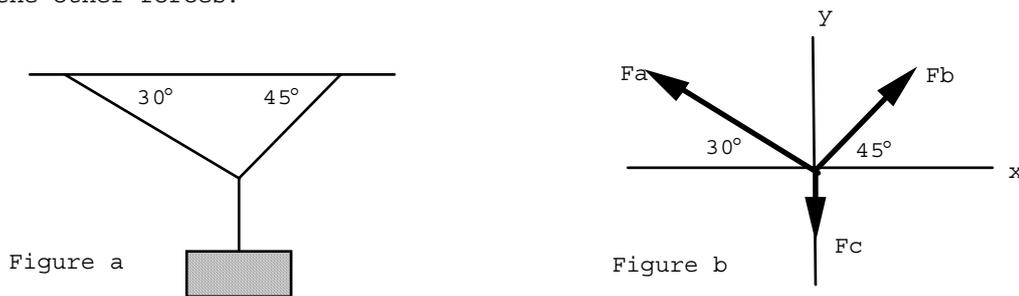

Figure 7.1 A physics example.

7.2.3   Analogy

Computational experiments showed that Cascade's explanation-based learning technique for overcoming impasses was insufficient in itself to account for students' behaviour during problem-solving. If missing knowledge is required to solve a problem then there are simply too many impasses encountered for a student to know which it is profitable to repair (naively, one might predict that, because learning occurs at impasses, the more impasses the better). During learning from an example, however, as the solution path is strongly constrained, only a few impasses will be met and resolving those will probably lead to a solution. Therefore a second mechanism, analogical search control, was introduced to enable the system to use derivations produced while explaining examples to constrain its generation of derivations during problem-solving. If Cascade cannot derive a result (using its domain knowledge or overly general rules) then it just assumes the example is correct and builds a rule which enables analogies to this assumption. For example, if Cascade cannot prove that 'a knot can be a body' it forms a rule that says 'In similar problems choose the knot as a body'. Cascade's mechanism is a particular version of the general method of *derivational analogy* (Carbonell 1986) which is analysed and empirically examined in Blumenthal and Porter (1994).

Anderson, Boyle, Corbett and Lewis (1990) consider that "the major way that students solve problems involving concepts is by analogy to examples of solutions involving those concepts." In fact, their theory of learning underlying the design of the so-called ACT* tutors has no inductive learning mechanisms, unlike ACT* itself. Therefore, the proclaimed success of the tutors is attributed to the principles (section 3.5) derived from ACT*, even though it was not fully adopted, and the necessary changes to the theory can be considered a "profitable flow of influence .. between the theory and the application" (a heads we win, tails we win outcome).

Advocates of case-based reasoning, a form of analogical reasoning where old solutions are mapped to solve new problems, see a clear link to an instructional principle - because students learn from cases, tutors should give them cases which they will recall: "good teaching is good story telling" (Schank, 1990). Case-based teaching software should aim to "place students in situations they find interesting, where the telling of a story would be appreciated". In case-based teaching, "the teacher knows what stories it has to tell and sits waiting, ready to pounce on the situation a student has encountered with an appropriate story".

Kolodner (1991) describes a system which uses a case base of 600 battles to instruct students on battle management. Technically, the difficult questions concern how to index the cases so that appropriate ones may be retrieved and how to specify what an agent might learn from attending to such a case. Also, if the case is not to be presented verbatim, techniques need to be developed to adapt the case to the context. Slade (1991) reviews techniques for indexing, retrieving and adapting cases in case-based reasoning.

Of course, the cases may be organised into a 'curriculum' as in the tutor described by Murray, Schultz, Brown and Clement (1990) which focusses on using analogy to remediate physics misconceptions. The student is led to reflect on a target situation (such as "Does a table exert a force on a book resting on it?") by being presented analogous situations (such as "If you hold a textbook is there a force up on the book?"). The bridging analogies and the paths between them are pre-defined but the particular ones presented depend upon the answers given. The student is asked to explain answers to herself, not to the system.

7.2.4   Conceptual change and belief revision

The design of the analogy-based physics tutor of Murray et al (1990) and, more generally, the existence of the field of *conceptual change* in educational psychology (Caravita and Hallden, 1994; Chinn and Brewer, 1993; Posner, Strike, Hewson and Gertzog, 1982; Vosniadou, 1992; White, 1993) recognises the difficulty of overcoming certain entrenched misconceptions. For example, Bereiter and Scardamalia (1989) report children's attempts to reconcile a study text statement such as "Harmful germs are not trying to be bad when they settle down in your body. They just want to live quietly, eat, and make more germs" with their previous conceptions of germs as aggressors, as depicted in television commercials.

White's ThinkerTools project (White, 1993), discussed in section 6.7, aims to facilitate conceptual change to displace misconceptions which several studies have shown are very resistant to change through normal physics courses. In fact, as most conceptual change studies have been concerned with overcoming naive conceptions of science which are similar to theories of earlier scientists (such as pre-Galilean astronomy and pre-Newtonian dynamics), the learner is often seen as a 'scientist', gathering observations and reconciling conflicts. This is an analogy of questionable utility as the history of science shows that the evolution of scientific theories is a complex and not entirely rational process. Moreover, conceptual change encompasses more than concepts in natural science.

As the representation in our framework shows,

```
reflectors: { ...
        monitors:  { ...,
                conflict(beliefs) ∅ revise(beliefs), ...
            reasoners:  { ...
                    beliefs:   ... } } }
```

the process of conceptual change involves two stages: the detection of a conflict and a subsequent revision. In the simplest case, a conflict arises from the set of beliefs coming to contain both p and ~p. Teachers (and AI-ED systems) have a range of techniques to lead students to such conflicts (for example, the use of entrapment strategies in Socratic dialogues) in the hope that they will be motivated to resolve them. This section is more concerned with the methods used to deal with a set of beliefs considered to be in conflict.

Chinn and Brewer (1993) postulate what they consider to be "close to an exhaustive set" of responses that students may have to anomalous data (i.e. data which contradicts that already believed). It is proposed that students may:
- *Ignore the anomalous data.* The current beliefs are unchanged and the new data is not even explained away.
- *Reject the anomalous data.* In this case, a reason for not accepting the data is given.
- *Exclude the anomalous data.* The data is considered to be outside the domain of the theory corresponding to current beliefs. Beliefs from the new data are compartmentalized from the previous beliefs. Conflicts between beliefs in different 'compartments' are not reconciled.
- *Hold the anomalous data in abeyance.* It is assumed or hoped that the current beliefs will eventually be (re)articulated so that the new data can be explained.
- *Reinterpret the anomalous data.* The data is accepted as being within the scope of the current beliefs but it is reinterpreted. As with all the previous responses, the current beliefs are unchanged.
- *Make peripheral changes to the current beliefs.* If beliefs are organised into a 'core' (which cannot be altered without affecting the entire set of beliefs) and a 'fringe' (which can be altered without necessitating any change to the core), then the new data may be accommodated by relatively minor changes to the fringe beliefs.
- *Change the theory represented by the current beliefs.* In this case, one or more core beliefs are changed.

As this list suggests, students may well appear conservative in their reaction to anomalous data, but justifiably so, as their everyday experience with funfair physics (section 1.1), magic, televisual illusions, and the like tells them that new data is often not what it seems. In this respect, they are not unlike scientists, most of whom immediately dismiss any anomalous data and most of the time are found correct to have done so.

In the event that some conceptual change does ensue, then we may turn to AI work on *belief revision* or *reason maintenance* in the hope of finding some way of describing and performing those changes. The field of belief revision developed in order to deal with the problem of databases which had come to contain contradictory propositions, from which any proposition at all may be derived. Before summarising belief revision techniques, it should be noted that they are somewhat limited in ambition. First, they might better be called 'belief elimination' techniques, as their effect is to remove one or more beliefs from the set of beliefs and not to revise, that is, edit, any of the individual beliefs. More recent attempts to change the structure of beliefs are generally termed 'theory revision' (section 7.3.3). Secondly,

although the techniques take account of the beliefs upon which a belief depends, they do not consider in detail the reasoning processes through which a belief has been derived. Typically, it is assumed that all beliefs are derived by applying a standard rule of inference (such as resolution), whereas we know that students may apply all kinds of non-standard inference. Thirdly, as the term suggests, belief revision is only concerned with changing the contents of the set of beliefs. There are no techniques for similarly detecting conflicts and changing the contents of the other components (such as the set of reasoners or goals) of our framework. Indeed, it is not very clear what would constitute a conflict in such cases.

There are two basic methods of belief revision derived from different philosophies about the nature of a 'justification' for a belief. The *foundational theory* assumes that all derived beliefs are justified by other beliefs and ultimately (as cycles and infinite chains are disallowed) by a set of foundational beliefs or assumptions. The *coherence theory* assumes that a belief is considered valid if it is coherent with all the other beliefs, not that it has an explicit justification. The two theories imply different belief revision techniques: the former leads to attempts to identify one or more assumptions which may be deleted so that a troublesome belief can no longer be derived; the latter leads to the aim of modifying the set of beliefs as little as possible to incorporate a new belief while maintaining coherence.

The ATMS (de Kleer, 1986) and associated systems are the most well-known foundational systems. Its basic algorithm is:
1. If a new belief `p` is to be added to a set of beliefs then find all justifications `j1, j2, ..jk` of `~p`. Each justification will be a set of assumptions `{a1, a2, .. an}` from which `~p` may be derived.
2. Find a minimal set `H` such that at least one member of each justification is in `H`.
3. Remove `H` from the set of beliefs. This will ensure that `~p` can no longer be derived.

Of course, many refinements of this algorithm are possible and necessary. The fate of other beliefs which depend upon assumptions now deleted needs to be considered. As remarked above, we might wish to consider a justification to be more than a set of assumptions. In particular, some of the beliefs may have been derived by default inferences and therefore might be considered more vulnerable to elimination. As the status of a belief may depend in complicated ways on various defaults, it is usual to maintain a record of whether a belief is currently 'in' or 'out', along with its justifications, which are expensive to re-compute. From an AI-ED perspective, we may wonder about the psychological validity of recording or generating justifications. It is likely that students will produce ad-hoc justifications when needed which may have little to do with the original reason for coming to believe a proposition.

The most prominent coherence-based approach, AGM (Gardenförs, 1988), is motivated partly by psychological arguments. Actually, AGM is not a single system but a set of axioms which are intended to capture the notion of rational change of belief and which may be satisfied by many different implementations. If `Expansion(K,p)` denotes the set of beliefs `K` plus a new sentence `p` and its consequences, then `Revision(K,p)` is required to satisfy eight axioms, including the following:

> `p` is a member of `Revision(K,p)`
> `Revision(K,p)` is a subset of `Expansion(K,p)`
> If `~q` is not a member of `Revision(K,p)` then `Expansion(Revision(K,p),q)` is a member of `Revision(K,p & q)`

If revision does occur then, it is postulated, any reasonable revision process should satisfy these axioms (although, as Chinn and Brewer (1993) discuss, the first axiom above is not adhered to when many students encounter anomalous data). According to AGM, belief revision involves finding a minimum set of beliefs to delete which preserves coherence (obviously, if coherence were the only concern we could simply delete all the elements of `K`). In general, there may be more than one way of discarding beliefs consistent with the axioms and in order to choose between them one may specify an *epistemic entrenchment*, which gives an ordering between pairs of sentences. In this case, we prefer to discard the 'less entrenched' beliefs.

As usual, there is a thriving community elucidating the precise relationships between ATMS, AGM, and other systems and related theoretical notions such as default reasoning and nonmonotonic reasoning. So far, belief revision has been seen as an asocial process, but we can anticipate developments towards multi-agent belief revision (Malheiro, Jennings and Oliviera, 1994), where each agent may hold different locally coherent beliefs and have only partial understanding of other agents' beliefs.

As far as computational mathetics is concerned, the potential relevance of belief revision is twofold. First, it provides a set of techniques which an AI-ED system may use for its own purposes. For example, they may be applied to the problem of diagnosis, as discussed in the next chapter. Secondly, it may provide a technical language for clarifying the nature of conceptual change in students. At the moment, as belief revision is orientated towards the efficient management of databases and conceptual change in students appears to be a relatively rare phenomenon, there is large gap between the two.

7.3   Inductive learning

Inductive learning involves generalising from observations. If an agent holds a set of beliefs Δ and has made observations OBS then φ is an inductive conclusion if and only if:

```
1.  ~((Δ ≈ OBS) => ~φ)
```

that is, the conclusion is consistent with the set of beliefs and observations, and

```
2.  Δ ≈ φ => OBS
```

that is, the conclusion explains the observations. Of course, an inductive conclusion is not necessarily a logical conclusion from the set of beliefs and observations.

A large number of inductive learning procedures have been developed and analysed in AI and many have been applied within AI-ED systems (some examples will be given in the next chapter). In order to illustrate the basic methods and the issues that arise, we will consider the following example. Imagine that a student has observed that the elements in Table 7.2 have been described as 'metals' or 'nonmetals' and wishes to develop an hypothesis to explain these observations.

	colour	*sp. gravity*	*state*	*soluble*	
mercury	silver	13.6	liquid	no	metal
tin	silver-white	7.3	solid	no	metal
sulphur	yellow	2.1	solid	no	nonmetal
cadmium	blue-white	8.6	solid	no	metal
boron	brown	2.45	solid	no	nonmetal
phosphorus	white	1.8	solid	no	nonmetal

Table 7.2 Properties of some metals and non-metals

The standard *concept learning* task involves inducing a rule (to define the concept) from a set of examples labelled as positive or negative instances of that concept. The rule should distinguish between positive and negative instances, both observed and as yet unobserved. In general, there are numerous inductive conclusions possible. In order to select between them, various kinds of 'bias' may be applied, as discussed below.

The standard task requires the examples to be both described and labelled. We can imagine trying to remove these requirements. In Table 7.2 the elements are described in terms of a fixed set of features (colour, specific gravity, state and solubility). If the rule to be learned (assuming there is one) is in terms of features not given in the description (for example, the number of free electrons), then it is hard to see how any agent could learn the rule from the specified observations. It is not much of a problem if there are a large number of irrelevant features (for after enough observations they will be seen as such), or even if a 'required' feature is some function of given ones (for we can imagine a concept learning procedure trying to combine apparently relevant features), but if a crucial feature is missing then it will be hard for an agent, computer or human, to 'invent' it.

If the labels are missing, then we have a *concept formation* task. It is then up to the learner to decide which concepts to form and which examples are positive or negative instances of that concept. In this case, the concepts formed should be useful according to the goals of the learner.

7.3.1   Version spaces

A neat method of concept learning is the version space algorithm (Mitchell, 1982). This involves specifying for each feature a hierarchy of possible values and then manipulating upper and lower bounds for those values. For example, Figure 7.3 shows possible hierarchies and bounds for the first two features after viewing the first three examples of Table 7.2. At any moment, we can say that the concept, when it is fully learned, will be the conjunction of terms of the form $feature_i$ = $value_i$ where the latter is a node of the corresponding tree between the bounds. For example, Figure 7.3 indicates that the concept may be one of the following rules:

```
colour = whitish & SG = (any-SG or heavy or very-heavy)
```

After the first three examples, there is also a second version space indicating that the concept may be:

```
colour = (any-colour or whitish) & SG = (heavy or very-heavy)
```

The concept is fully learned when all the bounds enclose a single node.
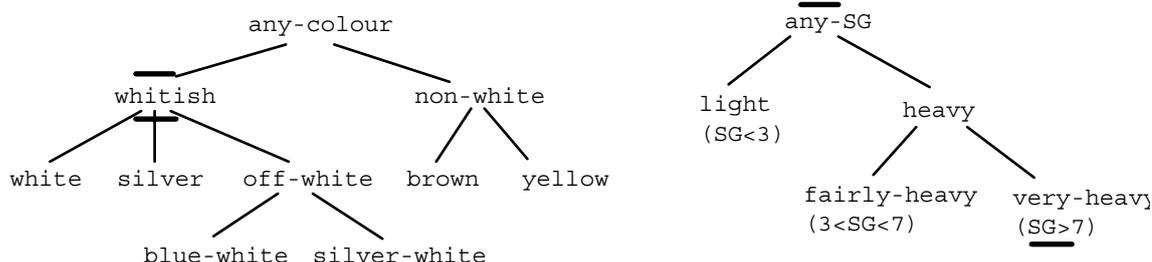
Figure 7.3 A simple version space.

In general, if an example is informative, it moves the lower bound upward if it is positive, and the upper bound downward if it is negative. In outline, the algorithm is:

For positive instances, for each tree, move the lower bound to be above or coincident with the instance value;

For negative instances, select a tree in which the upper bound is above the instance value (if there is more than one way to do this, maintain all selections in parallel) and move the upper bound towards the lower bound to a node above or coincident with the lower bound which is not above the instance value.

If a desired operation cannot be performed then that version space is discarded.

The method has the following properties:

- At an intermediate point, a partially learned concept is available (unlike some methods, which require all the data to be analysed at once).
- The concept learned is independent of the order of the examples.
- Previous examples do not need to be remembered (their contribution is implicit in the positions of the bounds).
- The pre-defined structure of the trees is crucial to what is learned (this is one kind of bias). For example, having decided upon the above trees, the method could never learn a concept of the form

```
colour = (white or yellow) & ...
```

- The method simply fails if the concept to be learned is not conjunctive (which is another bias). For example, it could not learn a concept of the form

```
colour = white or SG = heavy
```

Most everyday or scientific concepts are in fact conjunctive.

- The method is vulnerable to 'noise'. If an example is misdescribed or mislabelled then the bounds may be irretrievably, wrongly adjusted.

7.3.2   Numerically-based methods

The brittleness of the version space algorithm and other symbolic generalisation schemes may be overcome somewhat by using various kinds of statistical or probabilistic analyses. For example, a number of methods use quantitative techniques to build decision trees, for example ID3 (Quinlan, 1986). A decision tree defines a series of questions to ask (colour=white?, state=solid?, and so on) which lead to a decision about whether a particular example is a positive instance of the concept. Such a tree is the outcome of ID3, which is itself concerned with determining the order of the nodes within the tree. This is achieved by information-theoretic computations to find those features which provide optimum splits of the space of instances. Ideally, the tree should test the most relevant feature first, then the next most relevant, and so on. The method has been applied to the diagnosis of student problem-solving performance, as discussed in section 8.2.

One of the simplest learning mechanism involves assuming that the concept can be represented by a linear polynomial

$$f = w_1 \times f_1 + w_2 \times f_2 + ... + w_n \times f_n$$

where w_i is the weight of feature f_i, and using the data to optimise the coefficients of that polynomial. If, for a particular datum, the value of f is >0 (say) then the rule says it is a positive instance, otherwise negative. If it turns out that the function returns a value higher than it should be for a particular example, then we could slightly decrease those weights which add to the value of f and increase those which do not, and vice versa if the value is too low. After many (usually very many) examples, the weights might stabilise.

Such a function is a 'perceptron' or 'one-layer neural network' (Figure 7.4). Perceptrons are very limited in what they can learn, and what is learned (i.e. the weights of the terms) is not declarative, i.e. the values do not correspond to meaningful objects in the world. It is hard, for example, to imagine an AI-ED system being able to discuss with a student what it has learned using such a mechanism. Moreover, the learning mechanism, involving the gradual refinement of coefficients over many iterations, although it is motivated by psychological analogies to 'evolutionary learning', is not the kind of learning with which AI-ED systems are typically concerned.
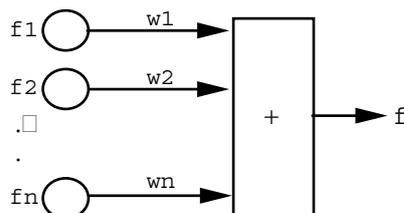
Figure 7.4 A one-layer network

Nonetheless, there is great enthusiasm for overcoming some of the fundamental limitations by using multi-layer neural networks (Figure 7.5), although these are even less declarative. In such networks, we can consider the features to correspond to large numbers of simple processors interconnected in a way similar to neurons in the brain. As the network learns, the associations between the nodes are strengthened or weakened. There has been considerable mathematical analysis of neural networks and much optimism about their efficacy, fired by their apparent simplicity and their opposition to standard symbolic methods. However, as far as AI-ED is concerned, there have been only a few proposals to apply neural network technology and no convincing demonstration of their usefulness in this context.
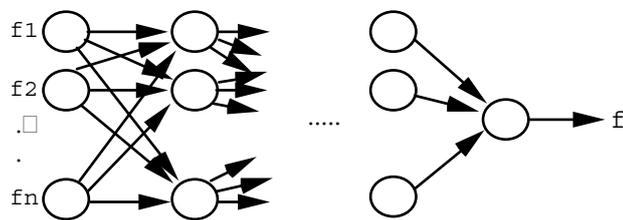
Figure 7.5 A multi-layer neural network

7.3.3   Constructive induction

Most established inductive learning mechanisms use a fixed, pre-defined representation space and are successful only if the original descriptors turn out to be directly relevant to characterising the concepts to be learned. *Constructive induction* attempts to overcome this limitation by including mechanisms for generating new descriptors and modifying or removing old ones.

Inductive logic programming is a field of machine learning concerned with learning logic programs from positive and negative instances in the form of ground literals. For example, given a set of examples:

```
reverse([],[]).
reverse([a],[a]).
reverse([c],[c]).
reverse([a,b],[b,a]).
reverse([b,c],[c,b]).
reverse([a,b,c],[c,b,a]).
```

and a set of defined predicates, such as

```
null, head, tail, assign, append
```

then the inductive logic program FILP (Bergadano and Gunetti, 1993) creates the definition:

```
reverse(X,Y):- null(X), null(Y).
reverse(X,Y):- head(X,H), tail(X,T), reverse(T,W), append(W,[H],Y).
```

Such programs learn theories to explain observations, the theories being expressed in a subset of predicate logic which is more expressive than the decision trees and similar representations used in other methods. Inductive logic programming is based upon solid formal foundations, enabling issues of completeness and relative power to be analysed. The general problem is clearly difficult (imagine the different ways in which the pre-defined predicates may be combined and the subtle ways the variables may inter-relate) and yet the above example has still not tackled the problem of being non-constructive, as the learned concept is expressed only in terms of pre-defined predicates. We can invent a predicate to stand for a commonly occurring conjunction of terms, and use newly learned predicates in further learned concepts, but such constructs can always be replaced by their expansions in terms of the pre-defined predicates (at some cost in brevity and comprehensibility).

However, in the case of recursively-defined predicates it may be necessary to create a new intermediate predicate first. For example, from

```
multiply(0,A,0).
multiply(1,B,B).
multiply(3,4,12).
```

and the definition of the predicate `successor`, any finite definition of `multiply` requires learning the predicate `plus` first. Lapointe, Ling and Matwin (1993) describe a method capable of learning predicates of the form:

```
q(...).
q(...):- q(...), newp(...)
newp(...).
newp(...):- newp(...).
```

provided that certain constraints are satisfied by the arguments of the predicates.

In general, it seems that inductive logic programming involves instantiating standard programming schema which, because of the power of the logic programming notation, are actually quite broad. Many extensions to the basic idea are being investigated. For example, De Raedt, Lavrac and Dzeroski (1993) consider the learning of multiple predicates at the same time, such as the predicates:

```
ancestor(X,Y):- parent(X,Y).
```

```
ancestor(X,Y):- parent(X,Z), ancestor(Z,Y).
father(X,Y):- parent(X,Y), male(X).
mother(X,Y):- parent(X,Y), female(X).
```
given observations in terms of the predicates `parent`, `female` and `male`. Also, Bergadano and Gunetti (1993) consider how the number of examples can be minimised, with the learning program itself asking about any further examples it needs. For example, from the instance
```
partition([1,3,0],2,[1,0],[3])
```
and the six further instances queried from the user, the program learns:
```
partition(L,El,L1,L2):- null(L), null(L1, null(L2).
partition(L,El,L1,L2):-
    head(L,X), tail(L,Y), partition(Y,El,Ls,Bs), cons(X,Bs,L2), assign(Ls,L1),
    X>El.
partition(L,El,L1,L2):-
    head(L,X), tail(L,Y), partition(Y,El,Ls,Bs), cons(X,Ls,L2), assign(Bs,L1),
    X<=El.
```

The state-of-the-art of inductive logic programming is worth bearing in mind when we come to consider (chapter 8) the problem of inducing a 'program' to explain observations which are a student's attempts to solve problems. The student's problem-solution pairs correspond to the set of examples from which we have to learn and the pre-defined predicates correspond to knowledge which we might assume the student already has. Our task might be seen as one of inducing a progam which produces the answers observed.

'Pure' inductive learning, in which an agent attempts to learn, almost from scratch, from a set of observations is rather unrealistic. For humans, at least, most learning is incremental in that we have a partial, provisional theory to explain the observations we have so far and then learn by refining that theory in the light of further observations. In the logic programming context, such an approach is called *theory revision* and can be seen as a hybrid of analytical and inductive methods.

A theory revision system is given
- an initial domain theory (as a logic program) that is incorrect and/or incomplete, and
- a set of positive and negative examples,

and produces a revised theory. Generally, the aim is to minimally modify the theory so that it correctly classifies the examples. This is obviously analogous to the belief revision problem, except that we are now not simply deleting propositions but are carrying out, in principle, any perturbation of the existing theory. Wogulis and Pazzani (1993) describe a system which performs theory revision by the four operations of adding/deleting clauses/literals. It first specialises the theory to exclude all negative examples, and then generalises this theory to include all the positive examples but not the negative ones. Defining what is meant by a minimal modification is not simple because the semantics of two logic programs cannot be easily determined from the text of those programs.

The classic AI program AM (Lenat, 1982) can perhaps be considered to be a constructive theory revision program. It created new concepts from old by co-opting their schema definitions and instantiating their components. It is now conceded that the program worked as well as it did mainly because the Lisp code was so close to the mathematical concepts they defined that most modifications to that code produced mathematically interesting results.

7.4   Active situated learning

The nature of learning as viewed in machine learning may appear limited to educational theorists. First, however, it should be remarked that it is not a view of learning that involves the 'direct transfer of knowledge', as caricatured by some. Or, at least, the simple storing of information transmitted is considered a minor, uninteresting aspect of learning. All the mechanisms we have discussed are 'constructivist' in that they assume that "learning occurs not by recording information but by interpreting it. Effective learning depends on the intentions, self-monitoring, elaborations and representational constructs of the individual learner" (Resnick, 1989).

Maybe learning should be more 'active' than the 'passive' cogitation of examples, but, again, this apparent philosophical difference has been exaggerated Many of the mechanisms involve the active participation of the learner, in, for example, selecting and generating instances to analyse. A sub-field of machine learning, self-directed learning (Goldman and Sachs, 1994), is concerned with learners selecting the presentation order of instances, with the aim of minimising the number of instances seen. In the case of AM, the program determines its own agenda of interesting concepts to study.

The theory of situated learning presents a complementary, not conflicting, view of the nature of learning. The main claim is that "a critical element in fostering learning is to have students carry out tasks and solve problems in an environment that reflects the multiple uses to which their knowledge will be put in the future" (Collins, Brown and Newman, 1989). This is an assertion which may be agreed with or denied, but it has no technical content. Situated learning is said (Collins, Brown and Newman, 1989) to serve the following purposes:
- Students come to understand the purposes or uses of the knowledge they are learning.

- They learn by actively using knowledge rather than passively receiving it.
- They learn the different conditions under which their knowledge can be applied.
- Learning in multiple contexts induces the abstraction of knowledge, so that students acquire knowledge in dual form, both tied to the contexts of its uses and independent of any particular context.

The view that learning experiences should be authentic cannot be categorically agreed with or not, for, as usual, it depends. If we wish to try to teach someone to fly-fish, which is a specific skill only ever practised in one kind of situation, then we would probably do better to do so in a real situation and not through some discussion in a classroom about the abstract principles of fly-fishing. If we wish to teach the concept of a complex number, which is a surprisingly useful concept in many branches of science, then an explanation of its general properties might be better at first. At the time the concept is learned, students are probably unable to appreciate its applications, which may therefore eventually come as a reward for their trust that the concept is indeed useful.

7.5   Social learning

All the above examples, and almost all others from machine learning, are concerned with learning by a single agent (be it computer system or human student). The psychology of learning has predominantly considered learning to be driven by an individual's internal motivations and structures. The fact that "human intelligence develops in the individual as a function of social interaction is too often disregarded" according to Piaget (1967), not least, it seems, by Piaget himself for he is usually considered a prototypical case of focussing on individual learning, in contrast to Vygotsky, who is considered to be the psychologist who first emphasised the essentially social nature of individual cognition. For Vygotsky, individual cognitive processes are the re-enactment by the individual of processes originally experienced in society.

There have been a huge number of experimental studies endeavouring to unravel the conditions and processes through which students may learn with others. Various permutations of cooperative, collaborative, and peer learning situations involving students of different ages, sex, knowledge, interests, etc. in different activities, such as joint problem-solving, group projects, etc., have been studied. The approach of computational mathetics is to complement these studies by attempting to specify the hypothesised learning mechanisms with sufficient precision that one or more of the agents involved in the social learning activity may be a computational agent. As before, such a computational model of social learning may serve as a conceptual resource for understanding the nature of social learning, and maybe as a basis for representing actual social learning processes in an AI-ED system.

Two studies illustrate the approach. PEOPLE-POWER (Dillenbourg and Self, 1992) is a system with which a student works with a computer-based learner (a co-learner) to develop a joint understanding of the properties of electoral systems. The basic mechanism is a rather literal interpretation of Vygotsky's view that learning involves the internalisation of inter-agent communications. The two learners carry out a dialogue to form an argument for some action, analogous to the self-explanation process, except, of course, that the components of the argument (proposal, rebuttal, agreement, and so on) are contributed by different agents. These argument patterns, which are available to both learners, become the basis for learning mechanisms such as generalisation. Experimental studies showed that human learners appreciated the nature of the collaborative interaction but found the need to communicate at the level and detail of the system's own rules rather irksome. Consequently, they communicated less, and, as learning occurs only through communication, the system learned less, and so the mismatch between the human learner and the co-learner increased, and so on.

A more detailed study of the nature of collaborative learning and the role that communicative processes play in them may be gained by experimenting with two co-learners. Imagine we have two agents, a and b, who are both trying to learn the same concept but do so by studying different examples and by describing the examples in different ways. How can the two agents make sense of what the other agent learns, in order perhaps to integrate the two different learned concepts (in order to develop a richer joint one) or to be able to discuss the differences between the learned concepts?

Brazdil (1992) gives the following illustration. Each agent has a vocabulary V (a list of predicates used to describe the world), a set of observed examples E, and a knowledge base K describing what the agent knows about the examples in terms of its vocabulary. For example,

Agent a	Agent b
V_a = { father,mother }	V_b = { parent,male,female }
E_a = { gfather(oscar,steve),	E_b = { gfather(william,steve),
gfather(paul,louis),	gfather(oscar,peter) }
gfather(oscar,andrew) }	
K_a = { father(paul,oscar),	K_b = { parent(william,sylvia),
father(oscar,louis),	parent(sylvia,steve),

```
       father(louis,steve),                    parent(oscar,helen),
       father(louis,andrew) }                  parent(helen,peter),
                                                male(william), male(oscar),
                                                male(steve), female(sylvia),
                                                male(peter), female(helen) }
```

Applying an inductive logic program, here assumed the same for both agents, to E_i and K_i the agents might induce:

```
       gfather(X,Y):-                          gfather(X,Y):-
         father(X,Z),father(Z,Y).                parent(X,Z),male(X),
                                                  female(Z),parent(Z,Y).
```

Neither agent's rule applies to the other's knowledge base, as the vocabularies are different. Moreover, even if an agent comes to know both rules (say b tells a its rule), then the combined rule

```
       gfather(X,Y):-
         father(X,Z),father(Z,Y); parent(X,Z),male(X),female(Z),parent(Z,Y).
```

is of no use unless an agent has access to the other agent's description of its world. For example, if a uses its own vocabulary to describe the world from which b derived K_b, we have

```
K_a' = { father(william,sylvia),
         mother(sylvia,steve),
         father(oscar,helen),
         mother(helen,peter) }
```

for which the combined rule fails. In order to fully integrate b's knowledge, a needs to learn b's vocabulary as well, i.e. the concepts `parent`, `male`, and `female`. This is possible if b conveys its description of its world to a, so that a can compare the two descriptions K_b and K_a'. As K_b contains examples of the concepts to be learned, a could apply an inductive logic program to K_b and K_a', to learn:

```
       parent(X,Y):- father(X,Y).
       male(X):- father(X,_).
       female(X):- mother(X,_).
```

In this way, using standard machine learning techniques, we can enable an agent to understand another agent's theory. If we regard one of the agents to be an AI-ED system, then we could imagine such a procedure being applied to handle situations where the student has a different viewpoint about the domain to that of the system. If we imagine both agents to be human students, then the above kind of analysis might be adapted to enable a system to help the students work collaboratively to come up with an agreed integrated theory of the domain.

7.6   Simulated students

A successful collaborative learning arrangement in which one of the partners is a computer-based learner demands that this agent embodies an adequate theory of human learning, for otherwise it would become progressively out of step with its partners. The prospect of developing simulated (or artificial or pseudo- or co-) learners has intrigued AI-ED researchers for some time (at least since Goldstein, 1979) but has only recently become a serious proposition (VanLehn, 1991; Ohlsson, 1992; VanLehn, Ohlsson and Nason, 1994).

The general view appears to be that learning is a very complex process, involving cognitive mechanisms and various phases and stages about which we have poor understanding and also affective and social factors about which we have even less, and as a result there is no real hope of building useful simulated students. Moreover, those theories of human learning which are computationally-oriented (SOAR (Laird, Rosenbloom and Newell, 1986) and ACTR (Anderson, 1993)) aim for a comprehensiveness which makes it hard to extract detailed design principles for AI-ED. A better idea of the potential role of simulated students may come from more focussed theories of learning.

Imagine, after VanLehn, Ohlsson and Nason (1994), that one has a theory of learning with the following axioms:
- If in a problem-solving situation with features f1, f2, ... one does action A and it turns out to be successful then form rule: `If f1 & f2 & .. then do A.`
- If one reaches an impasse then try to find a rule which can be generalised by (omitting a condition) to fit the problem situation.

Then the action of borrowing in this example

```
                                          1
          8 5                            7 5
        - 2 7          -->             - 2 7
        ---------                      ---------
```

might result in the rule:

```
       If needs-a-borrow(x) & next-column-left(y) & left-most(y) -> borrow-from(y,x)
```

The problem

```
          7 4 5
        - 1 2 7
        -----------
```

would now cause an impasse because there is no column which is both left-most and to the immediate left of the column needing a borrow. In the case, the second axiom might produce either of the rules:

```
If needs-a-borrow(x) & next-column-left(y) -> borrow-from(y,x)
If needs-a-borrow(x) & left-most(y) -> borrow-from(y,x)
```

In other words, the theory predicts that students may mislearn how to borrow if given two-column examples and indicates that borrowing should be introduced in three (or more) column problems.

This is a simple example but the general idea is, in principle, more widely applicable. The idea is to:

1. Specify a learning theory in computational form. The theory is a program `T(A,O)` which takes a possible instructional action `A` and generates learning outcomes `O` as predicted by the theory.
2. For each available instructional action submit it to the program and select that action which generates the preferred learning outcome. Moreover, a trace of the program's execution gives reasons for the selected action.

Expressed in this way, the idea is hardly novel. The use of computer simulations to help determine courses of action is common in the sciences and even some social sciences, such as economics (where their relative lack of success is a warning that their applicability to education may be problematic). Before considering this further, let us look at a second example.

STEPS (Ur and VanLehn, 1995) is a system able to simulate how a student learns to solve quantitative physics problems. It is based upon the method of explanation-based learning of correctness (section 7.2.2). The simulated student takes as input various kinds of tutorial actions - supplying feedback on past actions, giving hints on future work, and asking questions about the student's knowledge and thought processes. The performance of the simulated student provides data for the effectiveness of putative tutoring actions. STEPS has been used as a discovery learning environment for trainee teachers to learn effective tutoring tactics.

Doubts about the usefulness of simulated students for deriving instructional actions derive from two sources: first, a view that current theories of human learning are inadequate, and secondly, that the computer implementations are not a reliable manifestation of the theories. Computational mathetics has nothing directly to contribute to the first problem. It is a matter for psychology to develop progressively more refined theories of human learning; the demands of computational mathetics may help to determine whether the theories are sufficiently refined.

The second problem is a more difficult methodological issue. When the subtraction example is laid bare as above, the link from theory to instructional prescription seems so straightforward that there appears to be no need for a computer simulation. However, in general, theories of human learning will be so complex that this link will not be so transparent. It is precisely because we cannot determine the implications of a theory of human learning that a computer simulation may be useful. But we will only trust the prescriptions of such a simulation if we believe it to be a faithful rendition of the theory. This is partly why it will be necessary to provide some kind of trace of the simulation's execution. Inevitably, however, because theories of human learning are currently so informally expressed and indeed are so incomplete in their coverage of relevant aspects, much of the code of these simulations will be irrelevant to the theory or difficult to relate to it and yet possibly crucial to their performance. Ultimately, simulated students will only be trusted if we can view them as more than inscrutable, complicated programs allegedly implementing imprecisely-formulated theories of learning. The key theoretical notions will need to be isolated as a set of axioms, to be interpreted by some standard mechanism. If that stage is ever reached, we will have a theory of learning which is not fundamentally different from other axiomatic theories. We could, in principle, express the axioms in some agreed notation such as predicate logic and use standard analytic techniques, such as theorem-proving (computer-based or not), to determine the properties and consequences of such a theory.

8. Diagnosis

We may begin a discussion of the nature of diagnosis in computational mathetics by considering the following comment by two leading researchers in human-computer interaction:

"If the interface is intelligent, then it is not necessary (for it) to know anything about the user, because the interface will be able to interact with the user intelligently" (Newell and Card, 1985)

or, as it might be specialised to AI-ED:

"If the AI-ED system is intelligent, then it is not necessary for it to know anything about the student, because it will be able to interact with the student intelligently"

or, to education, generally:

"If the teacher is intelligent, then it is not necessary for her to know anything about the student, because the teacher will be able to interact with the student intelligently".

The last statement might be considered to betray a view of teaching blinkered from any need or desire to understand individual students, and yet many designers of computer-based learning systems appear to subscribe to the other statements as they aim to design systems which will interact with (or at least react to) the student intelligently without knowing anything about that student. Of course, it depends what is meant by an 'intelligent interaction'. It may be that such an interaction is defined to be one in which each agent takes due account of its knowledge of the other participants. Many designers would not consider themselves to be designing an intelligent system at all: they 'simply' intend the system to respond appropriately to student inputs. This echoes the debate in AI about the extent to which apparently intelligent behaviour can emerge without detailed reasoning, for example, the debate on symbolic versus reactive planning (section 6.4).

The role of diagnosis in computer-based learning systems is controversial (Lajoie and Shute, 1993). The main reasons for denying the need for diagnosis are:
- As discussed above, systems can perform appropriately without it.
- Human teachers appear not to do much diagnosis and therefore systems needn not either. (But the evidence has typically been gathered by protocol analysis of teachers in situations where diagnosis would be difficult and not verbalised. In any case, computer systems should perhaps provide facilities which complement rather than duplicate human teaching.)
- Diagnosis is too difficult and therefore it is better not to attempt it. (But diagnosis does not have to be complete: it needs to be only as detailed as the situation requires and the constraints permit.)
- Diagnosis implies remediation and hence an undesirable style of interaction. (But this is only one interpretation of what diagnosis is for.)

In some cases, the need for diagnosis is implicit in the design, but its need is denied or at least not emphasised. For example, the MIT Media Lab's 'News in the Future' section aims to generate individualised newspapers from the incoming news, which implies that the system knows the individual's interests. The proposed system is described using analogies with an 'English butler', an agent which maintains an unobtrusive, omniscient insight into the needs of its master or mistress (Negroponte, 1994). (It is pleasing that such an anachronistic legacy of the English class system should find a home at the forefront of technological innovation).

The dictionary defines 'diagnosis' as 'the analysis of facts or problems in order to gain understanding' and 'the opinion reached through such an analysis'. In our case, the 'facts or problems' concern the student's interactions with the system and 'the undersanding' is gained not for its own intrinsic interest but to support future interactions. In this chapter, we will first present a general characterisation of the diagnostic process in terms of our framework, then a more limited, but precise, technical description, which will then be relaxed and refined to provide a broader and more realistic picture.

We have hypothesised that an agent's problem-solving performance may be described in terms of a meta-level framework with various components:

```
        reflectors: ...
                monitors:  ...
                        reasoners: ...
                                beliefs:  ...
                                actions:  ...
                                situation: ...
                                goals:    ...
                                plans:    ...
```

In the case of a computer agent designed to solve problems, many of these components are known and maintained by the system itself. In the case of a student agent, none of the components are known. Some, for example, the `actions`, we can be reasonably confident about, if we have no reason to suspect the system's interface is malfunctioning, but most are defeasibly ascribed to the student. Any description of a student created by the system is an ascription (to the student) by the system for the system's purposes. It may or may not have any similarity to any cognitive object

possessed in any sense by the student. We will find that current diagnostic processes focus on the `beliefs` and, to a lesser extent, `reasoners` components but, in principle, diagnosis can involve any component.

Some general points can be made immediately:

- Diagnosis may make use of the system's own descriptions. If this is done simplistically then the ascription to the student will be a variant of the system's description, laying the process open to criticism that it ignores findings on expert-student differences.

- To facilitate diagnosis, systems may have, in addition to problem-solving knowledge, knowledge not of a particular student but of students in general. For example, a system may know that students often confuse momentum and impulse, or that in subtraction they often cannot borrow across a zero.

- As mentioned above, diagnosis is carried out for a purpose. It is important for a system to take account of its purposes, otherwise it may seek more and more detailed diagnoses without any consequent benefit.

- The diagnostic processes through which a system builds a model of the student may or may not bear any comparison to the cognitive processes through which a student actually creates her own cognitive structures. In some cases, techniques are applied for which no claims for any kind of cognitive validity are made; in other cases, the technique may be justified by referring to hypothesised cognitive processes.

- A student is an intelligent agent able to carry out her own diagnoses, to some extent, and it is arguable that she should be encouraged to do so. In any event, it is likely that any successful diagnostic process will involve some shared activity between the diagnoser and the diagnosee.

- Diagnosis is a complex process and it is sometimes hard to imagine that proposed techniques will be usable during an on-going interaction. At this stage, it is too early to make definite statements about the computational tractability of the various techniques, although we can anticipate that many of them will only be usable in an 'off-line' manner.

In this chapter, we will elaborate on these points by first adopting a distinction, similar to that of the previous chapter, between analytical and inductive diagnosis, involving the analysis of one or several pieces of evidence, respectively.

8.1   Analytical diagnosis

As with analytical learning, analytical methods of diagnosis involve the detailed analysis of one (or a few) observations and rely upon prior knowledge of the problem-solving domain and possibly of typical student (mis)conceptions.

8.1.1   Model-based diagnosis

We will first express the AI-ED diagnosis process in the terms used to define model-based diagnosis in general AI (de Kleer, Mackworth and Reiter, 1992). AI-ED texts (such as Wenger, 1987) discuss diagnosis at great length but in entirely descriptive terms, with little attempt to relate the processes to the more rigorous techniques developed in AI (and, conversely, diagnosis in AI is never related to the diagnosis of student behaviour). We will use the terminology and techniques of model-based diagnosis to establish a solid formal base, from which we may explore the considerable differences between diagnosis as it is needed in AI-ED and diagnosis as understood in AI. In this section, we will not pause to discuss those differences.

We must first define what it is we are attempting to provide a diagnosis of. We imagine a student interacting with a learning environment which in some general sense presents the student with problems to solve. We will define an *observation* OBS to be a sequence of <problem, solution> pairs. For example, for a student solving long-division problems, an OBS might be {`<3456/56, 63 rem 38>, <2345/210, 11 rem 35>`}. Problems and solutions may be represented by sentences in predicate logic. Considerable refinement of the notion of a 'problem' and a 'solution' is possible, for example, to enable the use of intermediate steps .

Next it is necessary to define the terms in which a diagnosis is to be expressed. We can say that a student has carried out some 'procedure' to generate a solution for a problem. This procedure may be faulty, incomplete, or inconsistent (indeed, it probably will be if we are carrying out a diagnosis of it). If we wish to argue that there may be no rational, deterministic connection between a problem and the student's solution, then we can allow non-deterministic procedures (such procedures providing diagnoses, in the extreme, for all possible observations).

Unless we want to consider the procedure to be some direct, stimulus-response mapping from problem to solution (which we can, if we wish), the procedure will be considered to have some 'components'. For example, although we might consider that for long-division experts there is some direct mapping 3456/56 -> 61 rem 40, in which case, a diagnosis can say little more than that the procedure is correct or faulty, for students we would imagine that the procedure of 'long-division' involves components such as 'multiplication' and 'subtraction'. We assume that a diagnosis involves some consideration of those component processes, for example, to be able to say that a student has incorrectly subtracted while solving the long-division problem.

A *component* COMP is a procedure which maps a set of input arguments I into a set of output values O. Both I and O may be partially specified, and COMP may be faulty in various ways. A diagnosis will be carried out in terms of a set of components COMPS.

There are three possibilities for COMPS:
- It could be fixed in advance of the diagnosis.
- An appropriate COMPS could be selected by the diagnostic process from a set fixed in advance.
- It could be created dynamically by the diagnostic process.

The third option is essentially automatic programming (and will be considered in section 8.2.3). At the least, it involves the editing of given components. This is likely to be necessary for an adequate diagnostic process but it is technically very difficult: it has not been considered much in diagnosis in AI, where it is usually assumed that the system is diagnosing a manufactured device all components of which are known. In this section we will first assume the first option and then (here and in section 8.4) consider the second option.

If we consider a process such as subtraction then we can see that there are many versions of this process (some correct, many more incorrect). Rather than consider all these versions to be different components (as defined above), we will consider them to be different *modes* of the same component. The rationale is that at any instant a component is in exactly one mode (although we may not be able to determine which). Therefore, a set of modes Mij (j>=1) is associated with each component Ci. (This is a different assumption to that normally made in AI diagnosis. There, all components are independent, with their own modes, so that if a device has, say, two multipliers then one may be faulty and the other may be working correctly. With devices, it is reasonable to assume that components may fail independently. For AI-ED diagnosis, it is more plausible to assume, to begin with, that a student will carry out a process the same way each time. If one is unhappy with this assumption then we can, of course, consider the different manifestations to be different components with their own modes.)

A component is defined by a set of axioms specifying, for each mode, how its input and output arguments are related. Each axiom may be written in the form:

        component(I,O) if mode(C,M) and condition(I,O).

i.e. if the component C is in mode M and some condition between I and O holds then the component maps I into O.

The complete set of such axioms, defining all components for all their modes, is called a *system description* SD. (This is a slightly different definition of SD from that used in device diagnosis. There, a distinction is made between the definitions of the components and of the device or system itself, that is, how the components are put together. In device diagnosis the latter is called the system description. In AI-ED diagnosis, we do not have a system description in this sense: it is part of the diagnostic process to work out how components may have been combined.)

A *diagnostic hypothesis* HYP is a set of assignments of a mode to each component in COMPS, {[Ci,Mij]}. A *candidate* CAND is an hypothesis which is consistent with SD and OBS, i.e.

        SD ≈ OBS ≈ CAND

is satisfiable. A *complete diagnosis* CD is the set of all candidates. A *diagnosis* DIAG is some subset of a complete diagnosis.

This, then, is our starting definition of a diagnosis in AI-ED. A naive diagnostic process would take a given set of components, try all permutations of mode assignments to them, and for each set of assignments check whether it is consistent with the system description and the observations. Trying all permutations would give us a complete diagnosis: often, we will have to settle for less.

Before giving a simple illustration, there are three special cases to consider:
- The *irrelevant* mode. All components are considered to have an implicit 'irrelevant' mode which corresponds to the situation where an explanation for OBS can be found which is independent of a particular component. For example, we might define a 'short-division' component (for division by a single digit), thinking that it may be a useful component for long-division diagnoses. It may then transpire that diagnoses may be obtained without recourse to 'short-division' (i.e. that long-division does not in fact involve any short-division). If a candidate contains [Ci,irrelevant] then, by definition, there will be other candidates where this assignment is replaced by [Ci,Mij] for all the explicit modes Mij of Ci. In such a case, only the [Ci,irrelevant] candidate will be included in CD. If a diagnosis indicates that a component is irrelevant to explain an observation, then clearly the observation provides no evidence about the correctness or otherwise of that component. The need for such a mode has recently been recognised in device diagnosis (de Kleer, Mackworth and Reiter, 1992). Without it (that is, with only 'correct' and 'faulty' modes), not every superset of the faulty components of a diagnosis necessarily provides a diagnosis, which is an undesirable state of affairs.
- The *correct* mode. Sometimes we have a component which is part of the problem-solving process but which is not to be considered during diagnosis, that is, it is considered to be a completely mastered pre-requisite skill. For example, during long-division, it is necessary to compare two numbers to decide which is the greater - therefore, a 'greater' component is a necessary sub-component of 'long-division' - but we may not wish to

develop diagnoses which involve this component. It is as though the component has only one mode (a 'correct' mode). As all candidates would contain the mode assignment `[Ci,correct]`, we omit such an assignment from the candidate and simplify the component definition to:

```
component(I,O) if condition(I,O).
```

- The *expanded* mode. When a diagnosis is performed, COMPS is considered to be an unstructured set. However, the available set of components may well be implicitly structured. For example, 'borrow' may be a component of 'subtract', which may be a component of 'long-division'. In order to permit the hierarchical expansion of the set COMPS (as described below), we introduce the special mode 'expanded'. If a member `Ci` of COMPS is assigned the mode 'expanded' then that component is removed from COMPS and replaced by all its sub-components. These sub-components are determined by the diagnostic process by considering the 'condition' part of axioms of the form:

```
component(I,O) if mode(Ci,expanded) and condition(I,O)
```

Once a component has been expanded, it will no longer form part of a diagnosis but its sub-components will. As the following illustration shows, this is the means by which progressively more detailed diagnoses may be obtained.

The following example is intended only to illustrate the mechanisms, not to imply that long-division is a fascinating problem, nor that our representations are cognitively valid (whatever that may mean). Imagine that COMPS = {ld} and SD (written in Prolog notation) =

```
ld([N,D],[A,R]):- mode(ld,ok), divide([N,D],[A,R]).
ld([N,D],[A,R]):- mode(ld,incorrect), not(divide([N,D],[A,R])).
ld([N,D],[A,R]):- mode(ld,backwards), divide([N,D],[X,R]), reverse(X,A).
ld([N,D],[A,R]):- mode(ld,no_zeroes), divide([N,D],[X,R]), remove_each(0,X,A).
```

This SD defines four modes for `ld`, namely `ok`, `incorrect`, `backwards` and `no_zeroes` (apart from the implicit `irrelevant`). The `backwards` mode corresponds to the possible bug where students write down the digits from right-to-left rather than vice versa, and the `no_zeroes` mode to the bug of not writing down the zeroes at all. Notice that the definitions of these modes make no pretence at cognitive validity - they do not imply that students with the `no_zeroes` bug first solve the problem correctly and then delete the zeroes! We can obtain a diagnosis involving the mode assignment `[ld,no_zeroes]` by seeing whether the specified conditions hold.

For the above COMPS and SD, for OBS = {<3456/56, 63 rem 38>}, we obtain a complete diagnosis of:

```
[[[ld,incorrect]]]
```

For OBS = {<123456/234, 725 rem 138>}, we obtain two candidates:

```
[[[ld,incorrect]], [[ld,backwards]]]
```

Which diagnosis is to be preferred (if there is more than one) and whether such a diagnosis is adequate (or should be more detailed) depends upon the uses to which it may be put, as discussed later. Let us assume we would like a more detailed diagnosis. As there is only one element in COMPS, we can only expand that (i.e. `ld`).

A component is expanded using axioms of the form

```
ld(I,O):- mode(ld,expanded), ....
```

Let us assume there are two such axioms, namely

```
ld([N,D],[A,R]):- mode(ld,expanded), greater([N,D],true),
        left_digit([N,D],[L,Rest,M]), multiply([D,M],X), subtract([L,X],Y),
        right_part([Y,Rest,D],[Z,R]), join([[M],Z],A).
ld([N,D],[[0],[N]]):- mood(ld,expanded), greater([N,D],false).
```

The Prolog details are not important here but it should be said that (unlike the above axioms) these should have some cognitive plausibility. In this case, the axioms specify how long-division may be accomplished by comparing the numerator and divisor, determining a left digit, and then processing the rest of the numerator. The sub-components of `ld` are `greater`, `left_digit`, `multiply`, `subtract`, `right_part`, and `join`.

All of these components now have to be appropriately defined, for example, by:

```
subtract([L,X],Y):- mode(subtract,ok), decimalize(L,DL),
        decimalize(X,DX), minus(DL,DX,Z), decimalize(Y,Z).
subtract([L,X],Y):-                        mode(subtract,does_not_compare),
    sub_does_not_compare([L,X],Y).
subtract([L,X],Y):- mode(subtract,faulty).
subtract([L,X],Y):- mode(subtract,slip), decimalize(L,DL),
        decimalize(X,DX), minus(DL,DX,Z), decimalize(Y1,Z), slip(Y1,Y).
    etc...
```

The new COMPS in terms of which a diagnosis is to be expressed is given by those components which have modes specified (the others being in mode `correct`, as discussed above). In this case, we might obtain COMPS = {multiply, subtract,right_part} and a new diagnosis for OBS = {<3456/56, 63 rem 38>} of:

```
[[[multiply,ok], [subtract,faulty], [right_part,irrelevant]],
[[multiply,no_carry], [subtract,faulty], [right_part,irrelevant]],...]
```

[To emphasise, this is purely illustrative, for we can define whatever components and modes we wish and use whatever programming constructions we wish in our definitions. In particular, the definitions may be as precise or imprecise as we wish to make them, for example, the `faulty` mode for `subtract` simply says that the output `Y` can be any value whatsoever.]

95

We can continue to expand components of `COMPS` as long as there are `expanded` mode axioms defined, and defined modes for sub-components of that component. For example, we might expand `subtract` in terms of `borrow`, etc. to obtain a diagnosis:

```
[[[multiply,ok], [right_part,ok], [borrow,no_decrement], ...], ...]
```

To summarise, the main points of this diagnostic process are:

- There are no restrictions on the computational content of `SD`. `SD` need not contain only simple sequential processes (as it often does in device diagnosis) but may contain any programming construct, including loops, recursion, and subprograms.
- Modes are associated with components, not with particular instantiations of the components. This is not an inherent restriction because we can, if we think it appropriate, distinguish the different instantiations of a component by giving them different names, with their own modes.
- In order to counter the common criticism that component descriptions have to be exorbitantly complicated when we suspect that in many cases students are not executing any procedure at all but merely following some imprecise, 'undefinable' process, component descriptions are allowed to be similarly imprecise. For example, to diagnose that a student is so fundamentally confused about subtraction that she may produce an answer greater than the number subtracted from, we can define an axiom such as:

```
subtract([L,X],Y):- mood(subtract,fundamentally_confused), Y>L.
```

- There is no need to presume to be able to define a single system description in terms of which diagnoses must be made. We can define, for example, two `borrow` processes, corresponding to the two standard ways taught for borrowing. If a borrow is carried out correctly then both modes will appear in a complete diagnosis, but if a mistake is made then the respective expansions may enable the method used to be identified.
- We can define as many 'correct' and 'incorrect' modes as we wish, and as far as the system is concerned there is no difference at all between them. Diagnosis can be carried out without any value judgements by the system concerning the 'worth' of the modes diagnosed.
- The method clarifies the need for cognitive validity. The diagnosis process has been misconstrued as one striving to develop a high-fidelity model of the student's cognitive processes. Others have naturally concluded that such an ambition cannot be realised. However, sometimes cognitive validity is irrelevant, because we can identify modes by detecting specified patterns. Cognitive concerns come to the fore only in the definition of the `expanded` modes, because it is through them that the components in terms of which a diagnoses is developed are identified.
- In principle, there is no reason why the method could not be used to provide diagnoses of processes considered somewhat higher-level than those illustrated above, for example, plans, metacognitive processes, learning processes, and so on. In practice, of course, it is hard to represent such processes (although, as indicated above, we can represent them vaguely, if that is the best we can do).
- The method takes no position about the depth of diagnosis. Hawkes and Derry (1990) say "Deep modelling of procedural bugs is not necessary. In order to derive an effective tutorial intervention, it is not necessary to know the exact nature of the buggy routine that produces a particular error. Rather, it is sufficient to know that a particular error pattern indicates that a particular skill must be targeted for instruction." However, sometimes a 'deep model' is necessary, sometimes it is not, and the decision should be made by the system itself, taking account of what the diagnosis is being carried out for.

8.1.2   *Differential modelling*

For any worthwhile problem the number of potentially relevant components is large, and so it is necessary to limit the search space. One way to do so is to use *differential modelling* as explained by Sherlock Holmes in *The Musgrave Papers*:

"You know my methods in such cases Watson: I put myself in the man's place, and having first gauged his intelligence, I try to imagine how I should have proceeded under the same circumstances. In this case the matter was simplified by Brunton's intelligence being quite first rate."

If the AI-ED system can itself solve the problem using components {Ci} then an initial assumption may be that the student also knows those components. This gives a default diagnosis of {[Ci,correct]}. When observations conflict with this assumption, then we may seek a diagnosis in which one or more components are no longer `correct`. We may consider hypotheses in which a particular component is assigned the mode `irrelevant`, which corresponds to assuming that the student does not use that component at all. Such a diagnosis yields an *overlay model,* in which the student's knowledge is considered to be a subset of the system's knowledge. Or we might have a set of replacement components and modes which are used systematically to change the default diagnosis. This yields what is sometimes called an *extended overlay model*, as the diagnosis includes components and modes which are not part of the system's (assumed correct) knowledge.

We may also limit the number of components to be considered by limiting the size of the 'problem'. In the extreme, we could require that the problem needs only one component to solve it. This is equivalent to insisting that the student inputs to the system all intermediate steps of a problem solution. In this case, the system determines a set of

components {Ci} such that all hypotheses assume that all except one component are in mode `irrelevant`. The one component which is not `irrelevant` is hypothesised to be the one the student actually used. This component may be in mode `correct`, `faulty`, or any other mode. This technique is called *model tracing*. It has been criticised for the constraints it imposes upon the student and for encouraging an overbearing style of interaction in which components diagnosed to be `faulty` are immediately remediated.

8.1.3   Fault-based diagnosis

The `faulty` mode, in which outputs are undefined, can lead to inefficient searches, when it is often known that components tend to 'misbehave' in predictable ways. These tendencies can be used to limit or to order the search space. In AI-ED diagnosis, the study of faulty modes (that is, specific bugs) has been a rich area of research (VanLehn, 1990). Each identified bug corresponds to a mode in the above terminology. The more specific the definition of the mode the more it becomes possible to develop a data-driven search for a diagnosis. This is a kind of abductive method in which a particular observation (for example, the solution has no 0s) triggers a hypothesis (for example, that the student has the `no_zeroes` bug) which may then be subjected to a model-driven analysis.

Fault-based diagnosis has been thoroughly studied in expert systems, where it derives from the maintenance manual approach of listing associations between observations (such as 'engine turns slowly') and causes (such as 'battery is low'). Generally, such searches are organised as troubleshooting hierarchies and make use of probabilistic methods to combine possible diagnoses. The main limitations of fault-based diagnosis for AI-ED purposes are:
- The difficulty of accumulating a comprehensive set of fault modes. This requires an exhaustive analysis of student problem-solving attempts and the development of a precise definition of the modes.
- The fact that, because fault modes are largely domain-specific, one has start afresh when diagnosing in a new domain.
- The problem of dealing with novel fault modes (the simplest solution being to use the catch-all `faulty` mode).
- The fact that mapping student behaviour onto pre-defined faults is a limited view of what diagnosis entails.

Nonetheless, fault-based diagnosis has a role in improving the computational efficiency of more general model-based methods.

The need for a comprehensive set of fault modes can be mitigated, in principle, by developing means of creating new fault modes when required. This would involve the use of impasse-based learning methods such as repair theory (section 7.2.1) whenever the diagnosis system cannot find a diagnosis. Giangrandi and Tasso (1995) describe a technique of this kind. They define the concept of a *meta-bug*, which specifies "a possible way of altering pieces of correct domain knowledge in order to draw out new possible bugs which are not present in the bug library". For example, the meta-bug:

>   During the conjugation of a perfective tense, the auxiliary verb 'to have' could
>   be replaced by the auxiliary verb 'to be'

could be applied to the correct rule:

>   The present perfect is formed with the simple present of the verb 'to have'
>   followed by the past participle of the verb.

Such a meta-bug looks more like a generalisation of bugs which are anticipated but, for some reason, are not in the bug library. It would naturally be more useful, but more difficult, to devise ways to create bugs which have not been anticipated but which nonetheless occur in practice, without creating bugs which in fact never occur.

8.1.4   Explanation-based diagnosis

Rather than search through all components and modes, a data-driven diagnostic process may regard the set of correct components as a domain theory and attempt to derive the student solution, as in the first stage of explanation-based learning (section 7.2.2). Each derivation obtained yields a candidate diagnosis in which those components used are effectively assigned mode `correct` and the others mode `irrelevant`. In other words, a diagnosis is regarded as an explanation or proof of the solution from the correct components.

If a derivation cannot be obtained (and the student's solution is considered to be incorrect), then the leaves of the unsuccessful proof trees potentially correspond to the student's (incorrect) components (Hoppe, 1994). In general, there are too many such leaves and they must be pruned by domain-specific heuristics before any further analysis. For example, only certain classes of leaf goal may be designated as possible sources of incorrect knowledge. A variant of the explanation-based learning mechanism can then be used to generate incorrect components from which the previously unprovable goal may be derived. For example, in symbolic differentiation if we have the unprovable goal

>   `derivative(sin(x^2),x^2,-cos(x^2))`

i.e. the student believes that `-cos(x^2)` is the derivative of `sin(x^2)` with respect to `x^2`, then the method might generate the generalised (faulty) component:

>   `derivative(sin(X),X,-cos(X)).`

In general, the generated component is a Prolog rule in which the conditions are formed by analysing other failed subgoals. As with any generalisation process, there is no guarantee that the created components are not over-generalisations.

Normally, there is more than one set of components from which a solution may be derived. The diagnosis needs to identify the appropriate set (i.e. context) for carrying out an analysis. For example, adapting an example from Costa, Duchènoy and Kodratoff (1988), imagine that a system has the following beliefs:

```
Beliefs(c) = {Man(Louis-XIV) ,
              Lived(Louis-XIV,17th-century),
         c1: { Man(x) & Noble(x) & Live(x,17th-century) -> Wig(x) },
         c2: { Man(x) & Criminal(x) -> Wig(x) },
         c3: { Man(x) & Bald(x) -> Wig(x) },
         c4: { Man(x) & Joker(x) -> Wig(x) }, ... }
```

where c_1, etc. define possible contexts. If a student asserts that Louis XIV wore a wig (`Wig(Louis-XIV)`) then this cannot be derived in any context. Because the system believes that Louis XIV lived in the 17th century, it might be inclined to context c_1 and so could perhaps ask if (or assume that) the student believed Louis XIV to be a nobleman. If, however, she asserts that Louis XIV was a joker then we would probably adopt context c_4. As this example makes clear, the components in terms of which the diagnosis is conducted correspond to beliefs of the system and also to beliefs (such as `Man(Louis-XIV)`) previously ascribed by the system to the student.

The idea of involving the student in carrying out a diagnosis (by, in this case, providing further information - erroneous or not - to enable a derivation to be completed) is obviously a reasonable, if potentially unreliable, strategy. It is a recognition that entirely system-based diagnosis is likely to be infeasible and also that such an involvement may benefit the student. The nature of interactive diagnosis is considered further in section 8.6.

A student input is only one example of what is referred to as an 'oracle' in the program debugging field. In general, an oracle is an agent external to the diagnostic system which provides information when asked by the system. Because we have viewed components as parts of programs, when a particular desired result cannot be derived from the program, we could interpret that as evidence that the program is faulty and therefore needs debugging. This way of viewing diagnosis will be considered in section 8.2.3.

8.1.5   Diagnosis by metareasoning

Explanation-based learning, and its extensions to handle unsuccessful derivations, is a meta-level process, as it involves analysing an object program. Because in AI-ED the object program may have various 'undesirable' properties, the meta-interpreter needs to be particularly carefully designed. The characterisation of AI-ED diagnosis as a metareasoning task has been investigated by Cialdea (1992), Beller and Hoppe (1993) and Aiello, Cialdea and Nardi (1993).

Aiello et al (1993) first distinguish four sets of beliefs:

E - `Beliefs(c)`, i.e. the set of (correct) beliefs of the system or 'expert knowledge';
S - `Beliefs(c,s)`, i.e. the set of beliefs ascribed to the student by the system, i.e.
   `{ p | B(c,B(s,p)) }`
U - the set of beliefs which "the system knows the student does not know", i.e. (perhaps)
   `{ p | B(c,~B(s,p)) }`
B - a set of incorrect beliefs which the system believes may be held by typical students, i.e. a bug catalogue.

Object-level inferences rules (such as resolution) enable further beliefs to be derived from any set of beliefs. We may ascribe different inference rules (or reasoners) to different agents but in the following we will assume uniform reasoning capabilities. The meta-level predicate

```
Derivable(X,Z,p)
```

is valid if the proposition `p` can be proved in the object-level using exactly the formulae in `Z`, which is a subset of `X`.

If a student indicates that she believes `p` then the system may believe that the student believes all beliefs which contribute to the 'best explanation' `Z` of `p`, or, as a formal diagnostic principle:

```
Answers(p) & Explains(Z,p) & q a Z => Ascribe(q,S)
```

where `Ascribe(x,y)` means that the belief `x` is ascribed to the set `y`.

Several definitions of `Explains` are suggested by Aiello et al (1993):
   • `Derivable(E,Z,p) => Explains(Z,p)`
i.e. `p` is explained by the subset `Z` of `E` (the system's correct beliefs) from which `p` may be derived. This axiom is suitable only if there is just one way of deriving `p`.
   • `Derivable(E≈B,Z,p) & ∀q(q a Z => Askstudent(q)) => Explains(Z,p)`
i.e. if the student answer is incorrect, `p` is explained by a subset `Z` of the expert and buggy knowledge, provided that the student confirms that she believes all members of `Z`, this clause being included presumably because the diagnosis of incorrect knowledge is inherently unreliable.
   • `q a E≈B & ∃Z(q a E & Derivable(S≈{q},Z,p)) => Explains({q},p)`
i.e. `p` is explained by the member `q` of `E` or `B` which when added to beliefs already ascribed to the student enables `p` to be derived. In this case, it is assumed that the set `S` grows incrementally: any answer which cannot be derived from `S` is assumed to be derived from `S` plus one further belief, either correct (in `E`) or incorrect (in `B`).
   • `∃q(q a B & Derivable(E≈{q},Z,p)) => Explains(Z,p)`

i.e. `p` is explained by a subset of `E` plus a single incorrect belief. This assumes that incorrect answers are explained by at most one bug.

Diagnosis tends to concentrate on explaining wrong answers, in terms of what is believed. However, a system may also diagnose what a student does not believe, which can be useful for instructional decisions. For example, if a student answers "yes" to a question `p` then we might infer that she knows no way of deriving `~p`:

```
Answers(p,yes) & ∃Z(Derivable(E≈B,Z,~p) & q a Z) => Ascribe(q,U)
```

Similarly, if she answers "I don't know" then we might infer that she knows no way of deriving `~p` or `p`:

```
Answers(p,dontknow) & ∃C(Derivable(E≈B,Z,p) v Derivable(E≈B,Z,~p) & q a Z)
     => Ascribe(q,U)
```

If the diagnostic process is going to go to the trouble of maintaining `U`, then it could also use `U` within diagnosis itself, as no explanation should use a member of `U`. In other words, the above axioms could be modified to be of the form:

```
… & ~∃r(r a Z & r a U) => Explains(Z,p)
```

As Aiello et al (1993) discuss, these axioms correspond to the diagnostic processes of some classic AI-ED systems and variations can be developed to correspond to others. From a computational mathetics perspective, we can consider what may be gained by expressing diagnostic principles in such a notation, rather than in the more normal, informal, natural language statements which we have briefly summarised in our descriptions of each axiom. The formal axioms are more concise and precise and, to some readers at least, more easy to read. The use of a common notation clarifies the similarities and differences of the various methods. The precision may reveal gaps or inconsistencies in the design of diagnostic processes. For example, in the case of the second axiom for `Explains`, do we really want to ask the student about the correct beliefs used or just the buggy ones? The axioms serve as a specification which may be expressed directly in an appropriate programming language, enabling them to be investigated efficiently and independently in prototypes before being buried in complex, complete AI-ED systems.

However, as the precision also helps make clear, there are many aspects of diagnosis which are not addressed by such formalisms (and if they are addressed within AI-ED systems, are usually done so in an ad-hoc manner). For example, the formalism does not really address the problem of diagnosing a series of observations, some of which may be in error (corresponding to student slips, perhaps) and during which the student may have actually learned something. The above notation does not allow for the fact that students reason differently from experts. It does not allow for the sets `S` and `U` to be inconsistent. It also treats the sets of beliefs as unstructured and does not capture the intuition that members of `B` can be associated with members of `E` which they can be considered to replace. Still, we can begin to imagine how such formalisms might be extended to handle some of these issues.

Van Arragon (1991) combines the ideas of default reasoning and metareasoning in a recognition that system and student reasoning are different. First, beliefs are partitioned into two sets:

> `F` - a set of facts, such as `Novice(s)`;
>
> `A` - a set of defaults, such as `Novice(x) ∅ Understands(x,login)`

A proposition `p` is 'explained' or believed if there is a set `D` of ground instances of elements of `A` such that `F≈D` is consistent and implies `p`. As there is a difference between the (default) reasoning of the system about the learner and the (default) reasoning of the learner about the domain, a meta-level is introduced. The meta-level corresponds to the system reasoning about the learner; the object level to the system's model of the learner's reasoning about the domain. The object level contains facts and defaults ascribed to the learner, for example,

```
F(s) = { ~Error-message("rm t*"), ... }
A(s) = { Side-effects(x) ∅ Error-message(x), ... }
```

The meta-level has the fact that the learner forms explanation as decribed above, that is,

```
F(c) = { F(x)≈D => p & F(x)≈D is consistent & DQA(x) ∅ Explains(F(x),A(x),p) }
```

and the default that the learner's assumptions are consistent, that is,

```
A(c) = { DQA(x) ∅ F(x)≈D is consistent }
```

In this case, using the default reasoning methods of the Theorist system (Poole, 1988) in which the system is implemented, it is possible to derive that

```
Explains(F(s),A(s),~Side-effects("rm t*"))
```

Despite the formidable technicalities, the method really distinguishes between the content of the system and learner reasoning processes. The mechanisms themselves are assumed to be essentially the same, whereas we can be sure that learners' reasoning is not as reliable and thorough as that which we embed in systems.

8.2   Inductive diagnosis

Purely analytical methods of diagnosis are inherently unreliable, rather more so than analytical methods of learning, as a single observation of student problem-solving performance may not be a good indication of underlying competence. Some machine learning methods do take account of noise (i.e. error) in the data, but in the case of AI-ED diagnosis, noise, corresponding to slips, is ubiquitous. Also, when the underlying competence is in fact low then actual performance can be very variable. The early studies on the apparent stability of bugs in subtraction, which were carried out under conditions (such as providing no feedback) which induced stability, have been shown to offer a false

promise for AI-ED. Student bugs are not like program bugs: they do not produce replicated behaviour when re-executed. Typically, when a student does not know how to do something, then she will do it differently.

The model-based method of diagnosis (section 8.1.1) can handle slips, in a way. For example, if we consider that a slip in executing a component means that a student is actually able to execute it correctly but, some reason, the output is 'corrupted', then we could write axioms of the form:

```
C(I,O):- mode(C,slip), correctC(I,Z), corrupt(Z,O).
```

This amounts to considering the `slip` mode as similar to the nondeterministic `faulty` mode, which perhaps it is. One difference is that the `slip` mode is to be assigned to one or more instantiations of the component, not to all of them (because in long division, say, you can make a slip with one subtraction but not necessarily all of them). If this seems a not very profound treatment of slips, then we can attempt to define a more precise axiom for the mode - but the more that we do so, the more the mode seems to define a bug, rather than a slip.

Without care, the `slip` mode will be returned as a candidate diagnosis for any observation whatsoever. Whether or not a diagnoser should accept such a diagnosis is usually resolved by considering the context, that is, the previous problem-solving performance. The model-based method can be extended to diagnose more than one observation. If for observation `OBS1` we obtain candidates `C1` and `C2`, and for observation `OBS2` candidates `C2` and `C3`, then for the series {`OBS1`,`OBS2`} we would expect the diagnosis `C2`, i.e. the intersection of the individual diagnoses. In fact, it is more complicated, because the components are hierarchically organised and therefore candidates may be consistent but not identical. In general, we might seek the least general specialisation of pairs of candidates - except, for slips and maybe other modes,which we might prefer to ignore if we have strong evidence for other candidates. This is leading inexorably towards the use of numerically-based methods.

8.2.1 *Numerically-based methods*

Typical of the shallow, but still possibly useful, numerically-based schemes which can be devised to provide diagnoses in AI-ED is that of *feature-based modelling* (Webb, 1993). The method requires two sets to be defined:
- a set `F` of 'context features', which describe properties of the problem-solving context (for example, for subtraction, 'minuend is larger than subtrahend', 'the subtrahend is zero', and so on).
- a set `A` of 'action features' which describe properties of the student's actions (for example, 'result equals minuend').

A diagnosis is in terms of 'associations' of the form

```
f1 & f2 & ... -> a
```

where `f1`, `f2`, .. are members of `F` and `a` is a member of `A`. An association `f->a` is 'accepted' (i.e. considered part of a diagnosis) if:
1. $N(f{\to}a)$ ε min_evidence
2. $N(f{\to}a)$ ε min_accuracy * ($N(f{\to}a)$ + $N(f{\to}{\sim}a)$)
3. there is no accepted association between a specialisation of `f` and a sibling of `a`.

where `N(f->a)` is the number of observations where all the features of `f` and `a` occur, and min_evidence and min-accuracy are parameters (for example, 3 and 0.8, repectively). The hope is that an association corresponding to a slip will be outweighed by other evidence.

As mentioned in section 5.5, the Sherlock system described the student in terms of a set of 5-tuples, each tuple {p_1,p_2,p_3,p_4,p_5} giving the probabilities of a student having a certain level of knowledge of some concept. For example, `A1`={0,0,.3,.5,.2} might represent that the system is .2 sure that the student has fully developed knowledge of `A1`, .5 that she has partially automated knowledge, .3 unautomated knowledge, 0 limited knowledge and 0 no knowledge. In order to maintain such a representation (i.e. to diagnose student's actions in terms of modifying such probabilities), a set of updating rules of the form:

```
event -> apply the change factor c and range vector {v1,v2,v3,v4,v5} to {p1,p2,p3,p4,p5}
```

are specified where `event` is some action that the student is seen to perform, and the application is defined (for downgrading) by the formulae

for i = 1 to 4, $p_i - c p_i v_i + c p_{i+1} v_{i+1}$ -> new p_i; $p_5 - c p_5 v_5$ -> new p_5

and similarly for upgrading. For example, given initial probabilities of {.2,.2,.2,.2,.2}, a change factor of .1 and a range vector of {0,.3,1,1,1}, after ten downgrades the probabilities are {.29,.33,.18,.14,.06].

The apparent precision of such computations lends a possibly spurious authenticity to them. The system designer has first to define the set of concepts for which probabilities will be specified and then assign them initial probabilities. For each concept, he must then identify the events which influence those probabilities, and for each event he must specify the change factor and range vector. The effects of all these decisions may interrelate in complex ways, or they may turn out to be relatively unimportant. As discussed in section 5.5, the status of such ad-hoc numerically-based schemes is a matter of debate and there is a move to relate them to more precisely-defined methods, such as Bayesian networks, and to integrate them with more logic-based methods.

It is not difficult to add simple probabilistic calculations to the model-based method of section 8.1.1. If we define a priori probabilities for the various modes, for example,

```
prob(subtract,[[ok,.5],[does_not_compare,.1],[faulty,.2],...]).
prob(multiply,[[ok,.4],[no_carry,.2],...]).
```

then the a priori probability of any candidate is the product of the probabilities of the modes assigned (assuming the probabilities are independent). Given such probabilites we can rank the candidates on the basis of their a priori probabilities, and, with a suitable meta-interpreter, prioritise the search for candidates. Poole (1994) develops an extension of such a notation which enables Bayesian networks to be re-expressed in an extended Prolog. For example, the simple network given in Figure 5.1 and repeated here:

```
                              Can-solve 508-253 (ssub)
                                 Prob(ssub|sub)=.95
   Can-subtract (sub)            Prob(ssub|~sub)=.01
    Prob(sub)=.8
                              Can-long-divide (ld)        Can-solve 5108/253 (sld)
                                 Prob(ld|sub,mult)=.6        Prob(sld|ld)=0.9
   Can-multiply(mult)            Prob(ld|sub,~mult)=.1       Prob(sld|~ld)=0.001
    Prob(mult)=.7                Prob(ld|~sub,mult)=.05
                                 Prob(ld|~sub,~mult)=.01
```
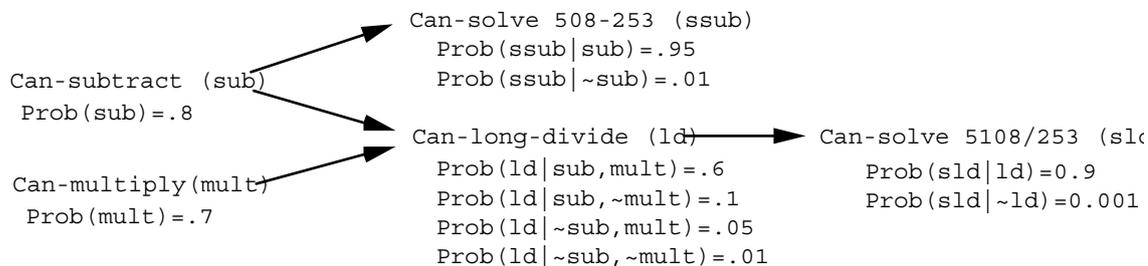
would be represented by:

```
ssub(X):- sub(Y), cpssub(X,Y).
ld(X):- sub(Y), mult(Z), cpld(X,Y,Z).
sld(X):- ld(Y), cpsld(X,Y).
```

where `cpssub(X,Y)` indicates the conditional probability of `ssub(X)` given `Y` (and similarly for `cpld` and `cpsld`) and is defined by a set of 'disjoint' clauses:

```
disjoint([sub(yes):0.8,sub(no):0.2]).
disjoint([mult(yes):0.7,mult(no):0.3]).
disjoint([cpssub(yes,yes):0.95,cpssub(no,yes):0.05]).
disjoint([cpssub(yes,no):0.01,cpssub(no,no):0.99]).
disjoint([cpld(yes,yes,yes):0.6,cpld(no,yes,yes):0.4]).
etc.
```

The definition of the interpretation of `disjoint` enables the probabilities of the various outcomes to be calculated.

However, as discussed in section 5.5, observations can propagate their effects through the probabilites associated with each node and it is not clear how this is achieved in the logic-based formulation. Nonetheless, there is considerable promise in being able to combine the epistemic (logic-based) and heuristic (numerically-based) aspects of diagnosis. A purely model-based approach as sketched in section 8.1.1 is exponentially complex, as each component may be in any of its modes. For diagnostic problems of any interest there will be a very large number of hypotheses to consider, most of which will be very unlikely. Therefore, some modification which takes account of the probabilities of the hypotheses would seem necessary. At the moment, there are only preliminary studies and no convincing demonstrations of the use of Bayesian networks for AI-ED diagnosis, and no attempts to integrate them with model-based methods. Much the same can be said of the use of neural networks (another numerically-oriented method) for diagnosis: there are optimistic proposals but no successful examples.

8.2.2   *Diagnosis using inductive learning methods*

The uses of explanation-based learning, Bayesian networks and other methods outlined above are internal to the diagnosis system and make no claim to be comparable to students' cognitive processes. In general, of course, comparisons may be made - for example, explanation-based learning may be related to self-explanations - but the particular adaptation for diagnosis is not justified by any analogy to hypothesised student processes. If, on the other hand, we had a perfect simulated student (section 7.6) then there would be no need for diagnosis at all: we would just initialise the student model and then the simulation would determine subsequent models. Consequently, most diagnostic methods combine aspects of computational effectiveness and cognitive fidelity.

The interplay between the two is illustrated by the use of inductive learning methods (section 7.3) for diagnosis. If we use model tracing and describe the problem in terms of a set of features and the student's step as an action then we have an association as in feature-based modelling:

```
f1 & f2 & ... -> a
```

Some of these associations may be identified as correct steps and some as incorrect. If we want to determine when a student performs a particular action `a`, then we can consider the correct steps as positive instances and the incorrect ones as negative instances and the task is precisely that of concept learning (section 7.3). Any of the standard methods such as version spaces or ID3 may be applied (but preferably those which can handle noise in the data). The outcome is a diagnosis of when the student considers it appropriate to perform action `a`.

If we have a set of a student's problem solutions, in the form of a sequence of steps, and a corresponding set of correct solutions, then the positive instances can be identified as steps which keep the student on the correct path and negative instances as steps which take the student off the correct path. Ohlsson and Langley (1988) apply this idea in the ACM system for the domain of subtraction. Here, the system does not begin with the student's steps but generates them (as in the first step of explanation-based learning).

ACM performs diagnosis and is not an attempt to model student learning, but cognitive aspects cannot be ignored. The specification of the features and actions in terms of which a student's solution is described is crucial. It is rather implausible that a diagnosis would be useful for AI-ED purposes if it were expressed in terms which bore no relation to those in which the student herself conceptualised the domain, however perfectly that diagnosis explained observations. Once the features and actions are defined (which amounts to defining a domain theory in explanation-based learning terms and is clearly domain-dependent), the generation of a student solution path for problems of any size is difficult, because the search space is so large. In order to reduce this space, Ohlsson and Langley (1988) introduce a set of 'psychological heuristics', including:
- Prefer paths that implicate smaller memory load.
- Assume that there no superfluous steps, so that intermediate results are used subsequently.
- Prefer paths that make minimal assumptions about errors.

The role of such heuristics is difficult to assess. As described, they serve to limit the search space for the student solution path but thereafter play no part in the diagnosis itself, as it might appear that they should. Conversely, a (provisional) diagnosis could itself be used to limit the search space for subsequent problems. More subtle are the implicit assumptions about the nature of a diagnosis, in particular that it involves only the misapplication of operators. It is possible that a more useful diagnosis would concern the very conceptualisations of the domain, which have to be taken as given. The method also assumes, like many methods of diagnosis, that there is a single procedure which it is the aim to identify, whereas empirical studies suggest that students have a range of strategies which they switch between on a problem-by-problem basis (Ohlsson, 1994).

8.2.3 *Diagnosis by automatic programming*

If diagnosis is thought of as a task of determining a procedure that accounts for observed problem-solving performance then this is tantamount to considering it to be a variety of automatic programming, in which programs are created which produce the required outputs for specified inputs. As the discussion of inductive logic programming and theory revision in section 7.3.3 indicates, we cannot be very optimistic that this approach is currently able to solve all our problems of diagnosis.

The THEMIS system (Kono, Ikeda and Mizoguchi, 1994) illustrates some of the issues. First, an extended Prolog notation is adopted to represent the student model: the expression `p(x1,..xn,t)` indicates the system's beliefs about the student's belief regarding the proposition `p(x1,..xn)` according to the value of `t`:
- if `t=true`, then the system believes the student believes `p(x1,..xn)`;
- if `t=false`, then the system believes that the student believes `~p(x1,..xn)`;
- if `t=unknown`, then the system believes that the student neither believes nor disbelieves `p(x1,..xn)`;
- if `t=fail`, then the system does not believe anything about the student's beliefs concerning `p(x1,..xn)`.

The first three correspond to the system predicting that the student will answer 'yes', 'no', 'don't know' respectively, and the fourth to the system itself being unable to make a prediction. We can have facts such as

```
temperate(paris,true)
temperate(tokyo,unknown)
```

and rules such as

```
grow(Plant,Place,T):-
    suitable_temperature(Plant,Place,T1), suitable_soil(Plant,Place,T2).
```

where the value of `T` is determined by truth tables specifying how the four possible values of `T1` and `T2` combine.

Such a rule in a student model is a program which predicts the student's answer. If we imagine that a student asserts that "Rice can grow in Kiev", i.e.

```
grow(rice,kiev,true)
```

and the system has the fact that

```
suitable_temperature(rice,kiev,false)
```

then the rule (i.e. the student model) has to be changed. In THEMIS this is achieved by adding or deleting a clause. Broadly speaking, if the student model predicts `true` instead of `false` then the rule is too general and should be specialised by adding a clause, and vice versa. The task now is to determine which clause to add or delete. Deletion is relatively straightforward as we can work through the clauses of the rule to determine which has failed. Finding a new clause is more complicated, because we must find not only the content of the clause but also where to place it in the rule so that, for example, variables interact as required. In THEMIS this appears to be achieved by some search of a 'refinement graph' which defines relationships between predicates.

8.3 Model maintenance techniques

As the previous discussion indicates, diagnosis in AI-ED necessarily involves a combination of analytical and inductive methods. It would be much too inefficient to carry out a detailed analysis of every single observation (in isolation from other observations) and generally not feasible to wait until we have a sufficient number of observations for reliable induction. Usually, it is necessary to apply analytical or inductive methods, as appropriate, to initialise a student model or whenever an event suggests it may be profitable, and thereafter expect to revise that model in the light of subsequent observations. We might hope that the student model is reasonably accurate and that only minor revisions are required. Such a process is called *student model maintenance*.

A model only needs to be revised if it is not consistent with an observation. The kind of revision required may depend on the kind of inconsistency. As Kono, Ikeda and Mizoguchi (1994) point out, there are four causes of inconsistency:

1. Learning and forgetting: If the student learns or forgets something, then her answers will contradict the model constructed before she learned or forgot.
2. Model inaccuracies: As diagnosis is difficult, the model constructed will be based on various assumptions. The inconsistency arises when these assumptions turn out to be unsound and hence the model is inaccurate.
3. Slips: In this case, the answer will not correspond with the model, which nevertheless may actually represent the student faithfully.
4. Student inconsistency: The student actually has inconsistent beliefs.

Generally, we would expect the student model to need revision in the first two cases but not in the third (assuming that an observation can be identified as a slip). A revision would aim to change the student model so that it restores consistency with the new observation. The fourth case is different: the student model should represent the inconsistency, as this may be useful for tutorial purposes. Therefore,

```
B(c,B(s,p)) & B(c,B(s,~p))
```

may be allowed, but not

```
B(c,B(s,p)) & B(c,~B(s,p))
```

Notwithstanding the fact that it is hard to distinguish these four causes (because their manifestation as an observed inconsistency is the same), we can consider techniques for carrying out the revisions required in the first two cases. These are based on the methods for belief revision discussed in section 7.2.4.

Once student model maintenance is considered to be a problem of revising a belief set so that consistency is recovered, then it falls within the scope of belief revision systems such as ATMS, as recognised in the systems of Huang, McCalla, Greer and Neufeld (1991), Kono, Ikeda and Mizoguchi (1994), Giangrandi and Tasso (1995) and Paiva and Self (1995). However, AI-ED places special demands on the application of ATMS-like methods. For example, Huang et al (1991) emphasise that the contents of belief sets in AI-ED are often derived from various kinds of default assumption and therefore that maintenance techniques need to take account of the different kinds of justification (as used in ATMS). Their method involves an integration of ATMS and model-based diagnosis. Paiva and Self (1995) use the idea of an 'endorsement', that is, an explicit reason or argument for an assumption, and an ordering of endorsements so that the more trustworthy assumptions are more likely to be retained during a revision. As mentioned above, Kono et al (1994) also consider the case where an inconsistency does not lead to a revised consistent model but to a partitioning of the model into two or more 'belief spaces', each internally consistent but possibly inconsistent with others. As discussed in section 7.2.4, students often do not seek a single unified set of beliefs immediately they encounter anomalous data. Giangrandi and Tasso (1995) allow their system to maintain multiple models in parallel, with one model being considered to be the 'current student belief model' but liable to replaced as such if it provides an inadequate prediction and explanation of student behaviour. Whether this is to improve the system's psychological validity or to recognise the difficulty of selecting the best model is not clear.

ATMS-like methods for student model maintenance take only implicit account of the fact that observations are made in sequence. Because a student (unlike a piece of equipment) is (expected to be) changing over time we might prefer to rely on the more recent observations. It is possible to achieve this in a simple numerical fashion. For example, the method of feature-based modelling (section 8.2.1) applies a decay factor of 0.9 when counting observations, so that the impact of past observations progressively lessens. If a more subtle analysis is required then it is likely to involve many of the formal techniques discussed previously. For example, Shanahan (1993) uses the situation calculus and circumscription to generate 'temporal explanations', that is, explanations which involve reasoning backwards in time from observations to causes.

We can illustrate some of the issues with a variant of the benchmark 'stolen car problem', namely: Imagine that on day1 a student is taught concept c_1. On day 2 we might reasonably assume she still knows c_1 (we might try to teach her concept c_2 which has c_1 as prerequisite). On day 3 we find that she does not know c_1. What could be the explanation? The student may have forgotten c_1, but we cannot say whether before or after we tried to teach c_2. This might be represented in the situation calculus as:

```
1.   Holds(believes(a,c1),d1)
2.   ~Holds(believes(a,c1),d3)
3.   Follows(d3,d1)
4.   Follows(t3,t1) ⊃ t3=t1 v ∃action,t2(t3=result(action,t2) & Follows(t2,t1))
5.   ~Holds(believes(x,c),result(forgets(x,c),t))
```

```
    6.    ~Abnormal(action,f,t) ∅ (Holds(f,t) × Holds(f,result(action,t)))
```
Axiom 4 is the definition of the `Follows` predicate, axiom 5 defines the `forgets` action, and axiom 6 is an attempt to address the frame problem.

From axioms 1-4 and 6, we have:
```
    ∃action,t1,t2(Abnormal(action,believes(a,c1),t1) & s2=result(action,t1) &
                    Follows(t1,d1) & Follows(d3,t2))
```
From axiom 5, using circumscription with respect to `Abnormal`, we have:
```
    Abnormal(action,f,t) ×
                (action=forgets(a,c1) & f=believes(a,c1) & Holds(believes(a,c1),t)))
```
and therefore
```
    ∃t1,t2(t2=result(forgets(a,c1),t1) & Follows(t1,d1) & Follows(d3,t2))
```
i.e. the student forgets `c1` some time between `d1` and `d3`.

We can, of course, add further axioms (for example, concerning whether the student was paying attention on day 1, or whether some other student told her something which contradicted `c1`) to enable other explanations to be derived. We may also use the notation in an abductive approach to generating explanations, in which the explanation has the new observation among its consequences (rather than being a deductive consequence). However, the method does not tell us which observations require an explanation, that is, when it is appropriate to bring such complex machinery into play.

Also, like the previous model maintenance methods, we do not seem to have capitalised on the fact that observations of student behaviour do not reach the system as a stream of unpredicted events. Generally, there has been some system action between observations. This action has been determined by the system precisely because it is intended to have some effect on the student. This intended or predicted effect is derived from some theory of instruction and learning. To the extent that the intention is satisfied, the subsequent observation has been predicted. At the least, we might hope that the analysis which determines the system action would serve to guide the subsequent diagnostic process. Therefore, we would expect that diagnosis in AI-ED should be integrated with theories of instruction and learning. So far, this has not been considered in any satisfactory detail. Diagnosis is generally conceived to be an independent activity of an AI-ED system.

8.4   Goal-driven diagnosis

As the illustrations of model-based diagnosis (section 8.1.1) and other methods showed, we can obtain many different diagnoses (each consisting of several candidates) capable of explaining particular observations, by choosing different components end expansions in terms of which to carry out the diagnosis. Which of these diagnoses, and which candidates within them, is best? All of the candidates are sound, that is, they all suggest mode assignments which are consistent with the observations. At the moment, there is no reason at all for preferring one to another.

We can introduce one means of discrimination by assigning a priori probabilites to individual mode assignments. For example, if `[multiply,ok]` is more probable than `[multiply,no_carry]` then in general we would prefer candidates with the first rather than the second assignment. This would enable the candidates within a diagnosis to be ranked and for the most probable candidates to be found first. However, it does not tell us which diagnosis is best. In the following, therefore, we will make the unrealistic, simplifying assumption that all candidates are equally probable and focus only on choosing between diagnoses.

There are two considerations to bear in mind when choosing between diagnoses. One is the computational cost of obtaining the diagnoses, as we are here imagining the diagnosis being carried out on-line. The second is the payoff, that is, the extra pedagogical leverage obtained from a 'deeper' diagnosis. To be precise, it is the 'predicted cost' and 'predicted payoff' which must be considered, as we are not suggesting that we obtain all the diagnoses and then apply our cost-payoff evaluations to select between them. Rather, we imagine a system having to decide whether to seek a better diagnosis, after determining a provisional one.

The payoff cannot be estimated without considering the options for action which the system has available as a result of carrying out a diagnosis. There is no standard notation for expressing such actions so let us adopt, for illustrative purposes only, a production system notation, such as:
```
    [[ld,ok]] -> continue1
    [[multiply,slip], [subtract, ok]] -> continue2
    [[ld,incorrect]] -> action1
    [[borrow,no_decrement]] -> action2
    [[subtract,faulty]] -> action3
```
We are not here concerned with the actions themselves but only with the left-hand sides, which are in terms of diagnosed mode assignments. (There will be other conditions, concerning, for example, the history of interactions, but these are not determined by the diagnosis process and so are omitted here).

Given the diagnosis `[[[ld,incorrect]]]` (as first obtained in section 8.1.1), this 'instructional plan' would recommend only `action1`. However, as we can see, if we obtained a deeper diagnosis then `continue2`, `action2` and `action3` might also be recommended. The system's dilemma is whether to invest the extra computational effort on obtaining the fuller diagnosis in the hope that other actions become possible.

According to our notation, any one action is as good as any other. Therefore, having found one recommendation (such as `action1`) there is no reason to seek another. What is missing is a representation of the intuition that, say, `action2` (which we might imagine to be a piece of remediation directed specifically towards the `no_decrement` bug) is a more useful action, if it is appropriate, than `action1` (which might be a general 'try again' action when an answer is considered to be wrong but we have no idea why). Let us attach weights (on a scale 1-100) to each rule to indicate its usefulness:

```
90: [[ld,ok]] -> continue1
70: [[multiply,slip], [subtract, ok]] -> continue2
10: [[ld,incorrect]] -> action1
50: [[borrow,no_decrement]] -> action2
25: [[subtract,faulty]] -> action3
```

Now we have one recommended action, with weight 10, but other potential actions of greater weight.

If the system decides to expand `ld` and so obtains the diagnosis

```
[[[multiply,ok], [subtract,faulty], [right_part,irrelevant]],
 [[multiply,no_carry], [subtract,faulty], [right_part,irrelevant]],...]
```

then some of the candidates recommend `continue2` and some `action3`. How can we calculate the payoff now? Assuming that all the mode assignments are a priori equally probable (the simplifying assumption made above), then all candidates are equally probable. (Obviously, if we had a priori probabilities then we could do the calculations but this does not affect the point here.) Thus, the probability of a particular mode assignment is given by the proportion of candidates which make that assignment. For example, the probability of `[subtract,faulty]`, according to this diagnosis, is in fact 3/7, or rather 9/13, as the `[right_part,irrelevant]` hides further candidates, as discussed above. We can perhaps estimate the payoff of `action3` by calculating the product of the weight of the rule and the probability that it holds, according to the diagnosis, i.e. 25 x 9/13 = 17.3. In comparison, the payoff of `action1` according to the first diagnosis is 10 x 1. For rules with multiple conditions on the left hand side, such as that for `continue2`, the probability that that rule holds is given by the proportion of candidates in which all the specified mode assignments are made, in this case 2/13. So, the payoff of `continue2` is 70 x 2/13 = 10.8. Assuming that we can carry out only one action, presumably that with the greatest payoff, then the estimated payoff according to the second diagnosis is 17.3, which is better than for the first diagnosis, and hence may have been worth the effort of obtaining.

The details of this example are not important but it does illustrate some general points about what needs to be specified before any decision about the 'best' diagnosis can be made. We must specify an instructional plan defining what a diagnosis is to be used for and attach measures of 'usefulness' to the various actions. We need to consider the probabilities of the candidate diagnoses and use these in some way to determine the overall 'payoff' from a diagnosis. So far, we have still only outlined a method for selecting between diagnoses having obtained them. We have not yet addressed the question of how a system could, having obtained one diagnosis, determine whether a further diagnosis is likely to be a better one. In general, this involves the calculation of conditional probabilities - for example, to determine the probability of `[borrow,no_decrement]` given that `[subtract,faulty]`. This would appear to be suitable for the application of Bayesian network techniques.

In AI generally, it is only relatively recently that it has been recognised that diagnosis should be driven by the goals for which it is being carried out (Freidrich, 1993). Such an obvious fact was overlooked because in the domains in which diagnosis was carried out in AI the goal is implicitly clear - it is to identify and replace faulty components in a device. More globally, however, the goal might be considered to be to minimise the cost in restoring a working system. In AI-ED the short-term goals (determining an instructional action) may be more varied and the longer-term goals (that the student learns) more complex. Without an analysis of such goals any diagnostic method is liable to produce a very large number of candidates, most of which are irrelevant to the goals of the system. A number of investigations of diagnosis-repair systems which tightly couple diagnosis and planning have been carried out (reviewed by Freidrich, 1993) but it is not clear that any of them are suitable for integrated diagnosis-action AI-ED systems.

8.5   Plan diagnosis

So far our consideration of diagnosis has been entirely restricted to the belief component of our framework, whereas (in principle) we might require diagnosis of any other component (monitors, goals, and so on) as well. In practice, the only other component which has been the subject of any detailed diagnostic analysis is that of plans, for *plan recognition* is a significant subfield of user modelling.

Often, an AI-ED system needs to interpret observations in terms of what the student is doing or aiming to do rather than what she believes or knows, because the system may intend to discuss plans and goals directly or because by re-

directing the student's plans the system may lead her more effectively to the desired beliefs. The problem of identifying the student's plans is unfortunately complex for the following reasons:

1. Unlike planning itself, plan recognition is inherently a multi-agent process because it involves one agent (the system) reasoning about the plans of another (the student).
2. It usually involves uncertain reasoning because a set of observed actions rarely uniquely identifies a plan (although definite conclusions may still be drawn even after uncertain reasoning).
3. Students are particularly prone to leave out actions, to insert faulty actions, to interleave actions from some other plan, and often to have no plan anyway.

Many approaches to plan recognition transform it into a parsing problem. A grammar is defined to specify how plans are decomposed into actions and sub-actions, and a particular sequence of observations is regarded as a sentence to be parsed with respect to this grammar. Formally, this is no doubt a sufficient characterisation of the problem, but we will instead first describe a method developed by Kautz (1990) which is closer to our view of diagnosis.

The method requires the specification of three kinds of information:

1. The observations themselves, for example,

```
Occurs(e9,make-pasta)
∃e Occurs(e,make-noodles) & t(e)=17
```

that is, event `e9` is an instance of type `make-pasta`, and an event of type `make-noodles` occurred at time 17. Such a description is based on a general theory of action and time (Allen, 1984) and inherits from it axioms such as

```
∀e,i During(t(sub(i,e)),t(e))
```

that is, the time of the `i`-th subaction of event `e` occurs during the time of event `e`. (We might imagine a student using a simulation to learn how to cook or to perform some similar activity.)

2. An action hierarchy, that is, a complete description of the ways in which an action can be performed and of the ways in which an action can be used as a step of a more complex action. These are specified as axioms of specialisation and decomposition :

```
∀e Occurs(e,make-pasta) Ø Occurs(e,prepare-meal)
∀e Occurs(e,make-fettucini) Ø Occurs(e,make-noodles)
∀e Occurs(e,make-spaghetti) Ø Occurs(e,make-noodles)
...
∀e Occurs(e,make-pasta) Ø
      ∃t Occurs(sub(1,e),make-noodles) & Occurs(sub(2,e),boil)
          & Occurs(sub(3,e),make-sauce)
          & object(sub(2,e))=result(sub(1,e))
          & Holds(noodle(result(sub(1,e)),t) & Overlap(t(sub(1,e)),t)
          & During(t(sub(2,e)),t)
...
```

The decomposition axioms specify the subactions, their preconditions and effects, and constraints on temporal relationships. Of course, subactions may also be decomposed. In addition, the system needs a set of disjointedness axioms, for example,

```
∀e Occurs(e,make-fettucini-alfredo) not-and Occurs(e,make-fettucini-marinara)
```

3. A set of 'simplicity constraints', for example, "minimise the number of top-level actions", to choose between interpretations. These are represented as second order predicate logic sentences, which are instantiated to first-order sentences for any particular case.

Before recognising a plan, the action hierarchy is supplemented by applying circumscription in order to derive axioms which specify the assumptions that (a) the known ways of performing an action are the only ways and that (b) all the possible reasons for performing an action are known:

```
∀e Occurs(e,prepare-meal) Ø Occurs(e,make-pasta) exc-or Occurs(e,make-meat)
...
∀e Occurs(e,make-noodles) Ø ∃a Occurs(a,make-pasta) & e=sub(1,a)
∀e Occurs(e,make-marinara) Ø
      ∃a (Occurs(a,make-fettucini-marinara) & e=sub(3,a)) v
         (Occurs(a,make-chicken-marinara) & e=sub(3,a))
```

Although there is no general method for carrying out circumscription, these axioms are easily derivable by special-purpose algorithms which retain the benefits of having a formal semantics for the process.

Now, given an observation, such as

```
Occurs(e1,make-fettucini) v Occurs(e1,make-spaghetti)
```

that is, that the student is making fettucini or spaghetti (but we're not such which), we may infer

```
∃e Occurs(e,make-pasta)    ..(1)
```

and hence that, for example,

```
∃e Occurs(sub(2,e),boil)
```

So, even though particular actions and plans may not be fully identified, specific predictions may be made. If we now observe:

```
Occurs(e2,make-marinara)
```

then the system can infer, from the specialisation axioms, that

```
        ∃e Occurs(e,make-pasta) v Occurs(e,make-meat)
```
Given the previous inference (1), the simplicity constraint mentioned above would eliminate the second disjunct of this inference.

Thus, the system may monitor the student's actions and attempt to derive the student's plans. The virtues of this approach are that it provides a formal theory with a precise semantics for the plan recognition process by specifying axioms (supplemented by circumscription) from which conclusions are derived deductively. It thus integrates plan recognition with other aspects of student modelling discussed previously, instead of regarding plan recognition as a rather specialist sub-problem for which different techniques are needed.

However, many criticisms are possible (most of them echoing discussions of the diagnosis of belief):
- It may be unreasonable to demand pre-specified axioms of specialisation and decomposition for all possible plans.
- Relying on nonmonotonic reasoning, the method can recognise one or more plans but cannot decide that one plan is more likely than another. This would appear to require the use of probabilistic methods, as advocated by Carberry (1990) and Charniak and Goldman (1993).
- The method takes no explicit account of what a plan is being recognised for. Increasingly detailed and bizarre plans can be hypothesed to explain any set of observations, and the method provides no way of determining when a plan has been diagnosed adequately for whatever purpose the system has.
- The method assumes that the plan to be recognised is indeed one that is implicitly encoded in the definitions. It cannot recognise a plan which is misconceived or malformed.
- What precisely is a 'plan' anyway that it makes sense to try to recognise it? If situationists are right that apparently planned behaviour is actually an emergent property of interactions within contexts, then perhaps plan recognition is a futile ambition.

Many of these points have been considered within the field of plan recognition but we will consider only the problem of diagnosing faulty plans. The field of plan recognition developed from the problem of understanding the goals of characters in narratives in order to enable natural language understanding. In that context, it is reasonable to assume that characters have (correct) plans which it is sensible to try to discover. In AI-ED, however, students often have faulty or vaguely formulated plans and it is in precisely that case that plan diagnosis may be more useful.

As with knowledge diagnosis, we may attempt plan diagnosis with respect to pre-specified incorrect descriptions (buggy plan catalogues) as well as correct ones, and we may try to generate faulty plans by dynamically modifying known correct plans. Calistri-Yeh (1991) claims to present a "complete classification of plan-based misconceptions" which can be used to generate faulty plans. Plans are considered to have four components - parameters, preconditions, steps and temporal constraints - each of which may be violated or have substituted, missing or extra parts. This gives sixteen categories, which may apparently be reduced to ten, as some are indistinguishable in practice. A faulty plan is diagnosed by modifying a standard heuristic search to add, delete or substitute parts of the plan heirarchy graph (somewhat similar to methods for modifying programs (section 8.2.3)). The probability of a misconception is a function of its similarity (closeness to the correct version), obscurity (unfamiliarity to the planner), complexity, the previous discourse and other factors. The system is said to select the best explanation for 98% (i.e. 57) of 58 real-world examples of plan-based misconceptions, 80% within 1 second, the remainder within 2 seconds, despite having only partially implemented heuristics. "Once the remaining heuristic features are added .. the probabilities should improve even more"(!).

However, the method diagnoses the components of a plan, not the plan or goal itself, which is assumed to be known to the system and the user. Plan recognition is usually understood to mean that the user's plans have to be inferred from his interactions. Problems arise because of the imprecision of the interactions, the difficulty of relating interactions to plans, and the possible conflict with the current user model. Eller and Carberry (1992) attempt to recognise faulty plans by defining a set of 'meta-rules' which relax these components so that previously blocked explanations can be considered. For example, one meta-rule states:

```
    If the system believes that acts A1 and A2 are each different ways of doing act
        A3
    & the system believes that the conditions for act A0 to be part of A2 are
        satisfied
    & the system can ascribe to the user both of these beliefs
    & the system cannot ascribe to the user the belief that act A0 is not part of A1
    Then the system may infer A1 from A0.
```

As this example illustrates, the attempt to divorce plan recognition from other kinds of diagnosis is rather futile. It depends upon considering the agent's beliefs about the domain and one another, and arguably its output should be in terms of the agent's beliefs, because it may be more profitable to address the underlying misconceptions which led to faulty plans than the plans themselves. Kass (1991) specifies a set of rules to make default inferences about a user's

beliefs from their interactions with an advisory expert system, many of these rules being derived from principles of conversational constraints. For example, two such rules state (adapting to the student modelling context):

```
Tells(s,c,p) ∅ B(c,B(s,p)) & ∀q (Component(q,p) ∅ B(c,B(s,q)))
```

i.e. if the student states a proposition then the system believes that she believes it and all components of it;

```
Problem(c,p) & Subproblem(q,p) & ~Do(s,q)
        ∅ B(c,(~B(s,Problem(c,p)) v ~B(s,Subproblem(q,p)) v B(s,~Can-do(q))))
```

i.e. if the system sets a problem p for which a subproblem q must be solved but the student does not attempt to solve the subproblem then the system believes that either the student does not believe that p is the problem or she does not believe that q is a subproblem of p or she believes that she cannot solve the subproblem. If (as is likely) the system cannot resolve this indeterminacy then it might form the basis for the subsequent dialogue with the student.

Once plan recognition is seen to be integrated with other diagnostic processes, then all of the previous discussions can be related to the problems of diagnosing plans. For example, the diagnosis of plans, like all diagnosis, should be driven by the purposes for which the diagnosis is needed. Waern (1994) provides a proof-theoretic formulation of the "plan recognition for a purpose" slogan. Instead of viewing an observation OBS as something to be explained by some hypothesis H added to some theory T, i.e.

```
T ≈ H ∅ OBS
```

it is viewed as a condition on a hypothesis or proof, where what is to be proved is the suitability of a possible next action A, i.e.

```
T ≈ H ≈ OBS ∅ A
```

Consider the following situation. A student is given a sequence of steps to solve a problem. The system aims to keep quiet as much as possible while she is solving the problem. Now, the system observes that the student leaves the intended path. The system may either remain quiet or provide advice on how to get back on track. The 'right' advice depends on why the student went off track. If the student still has the original goal, the deviation may be intended or not - she may have thought of a better way, or may have made a slip. Advice may be welcome in the second case but not the first. If the student has changed her goal, then perhaps the system should re-plan and present new advice. If the deviation is small, the system probably does not need to give advice. If the deviation is large and unintentional, maybe advice is needed. All this knowledge would be expressed in the theory T:

```
deviation(large) & unintentional-deviation ∅ give-advice(yes)
new-goal & intentional-deviation ∅ give-advice(yes)
old-goal & intentional-deviation ∅ give-advice(no)
deviation(small) ∅ give-advice(no)
...
```

If we observe a large deviation, then we seek hypotheses to provide proofs of the two possible actions:

```
T ≈ H1 ≈ deviation(large) ∅ give-advice(yes)
T ≈ H2 ≈ deviation(large) ∅ give-advice(no)
```

Waern (1994) derives proofs in the obscure (to me) intuitionistic sequent calculus for Gentzen's LJ, but we can see that proofs are possible if

```
H1 = unintentional-deviation
H2 = intentional-deviation & old-goal
```

A reasonable preference mechanism will lead to the system deciding to give advice. The general point is that the available actions determine the hypotheses that are sought, rather than (as has been the case before) the hypotheses are determined and then the available actions help choose between them.

We may continue our relaxation of the strict demands of the Kautz and Allen formulation, with which we began this section, by considering the method of 'situated plan attribution' (Hill and Johnson, 1995). Here, plans are not regarded as rigid prescriptions for actions but as rather fallible orientations for action. In many cases, the need for an AI-ED system to consider plans is indicated by the environment: if the student's directives are rejected then her plan must be in error; if the directives are not rejected and have the desired effect, then it must be correct or, at least, it requires no tutorial intervention. When a plan is in error, then the analysis could focus on what is required to overcome the problem, not on what caused the problem. Thus, the method does not attempt to build a cognitive model of the student's plan, as the use of the phrase 'plan attribution' rather than 'plan recognition' is intended to indicate. The method's notion of a plan is closer to that of situationists. Students' actions are, to some extent, responses to the evolving situation and "therefore tutoring systems should attend to the student's situation as well and direct less effort at trying to guess what is going on in the student's head".

At this point, it seems necessary to remind ourselves that diagnosis is inherently difficult and we cannot and need not expect or aim for omniscience, and this is especially the case for diagnosing plans and other metacognitive constructs. If reflection is (in the everyday sense) quiet, careful and long contemplation then there is little chance that an AI-ED system would be able to diagnose a student's reflective processes in the way that we have imagined for problem-solving processes (which is difficult enough), as there are simply not enough observations to permit reliable analysis. 'Internal reflection' is not an activity for which moment-to-moment student modelling is possible or appropriate.

Interruptions to "tell me what you are thinking" may well be counter-productive, because they will interfere with on-going cognitive activity.

However, in some situations, it may be beneficial to externalise metacognition. There is plenty of scope for debate on this topic, as it concerns the nature of metacognition and whether it can or should be made explicit. Some systems (for example, Singley, 1990) require a student to express the plan they intend to carry out, before specifying the individual actions (or sub-plans) which constitute the plan. Obviously, this considerably eases the plan recognition task. The difficulties lie in designing a language in which the plans may be expressed (by a student) and in incorporating the specification of the plan into the problem-solving activity in a non-obtrusive manner.

The AI-ED system's role might then be to determine when it is appropriate to externalise planning and other meta-level processes and to share in their execution. For example, if the student model indicates that the student is confused (maybe it contains both `B(c,B(s,p))` and `B(c,B(s,~p))`), then it may be more rational for the system to conclude nothing (let alone embark on a risky reason maintenance exercise) except perhaps that one or both is wrong and to engage the student in a reflective discussion of the issue. Similarly, if the student model indicates that in the current problem-solving situation one of a number of rules could have been applied, rather than attempting to second-guess which (if any) has in fact been applied, a system might do better to enter a meta-level where it is discussed explicitly. Often, such a discussion may bring into the open issues of which a student is only implicitly aware. For example, in algebra problem-solving, performance might lead to the student model containing `B(c,B(s,f))`, i.e. to the system believing that the student 'believes' a particular fault. Whether or not the student 'really believes' such a fault is debatable: perhaps it should be debated with the student. Payne and Squibb (1990) show that students do have sufficient metacognitive awareness that they are able reliably to assign levels of confidence to their answers: in some cases, a wrong answer (which is actually believed to be wrong by the student) is evidence that the student believes that she does not know something (`B(s,∃p~B(s,p))`), not that she genuinely believes something which is in fact incorrect. Different pedagogic interactions are surely needed for such different situations.

8.6   Interactive diagnosis

A picture of diagnosis is evolving which considers two agents, the diagnoser and diagnosee, to be involved in an on-going interaction to develop an explanation of the latter's problem-solving behaviour. AI work on diagnosis obscures a fundamental difference from diagnosis in AI-ED: the system undergoing diagnosis (the student in AI-ED's case) is not, or should not be, a passive agent with no interest in the process of diagnosis. As the student herself has the most to gain from a successful diagnosis, it seems advisable to engage her fully in the process. We may also argue that self-diagnosis is an important metacognitive skill which students should practise. Moreover, the kinds of conceptualisation used in AI-ED diagnoses tend to be of a nature which students would benefit from understanding, unlike, perhaps, the case of medical diagnosis, which is otherwise analogous, where diagnosis may involve the use of specialist medical terms which, generally speaking, patients do not need fully to understand.

Therefore, there appears to be every reason for viewing AI-ED diagnosis as an interactive process involving both system and student, not as an analytical process where the system does all the analysis of student behaviour. However, such a view does not magically solve the problems of diagnosis, for it does not directly tell us what the interaction should be about. In this respect, all the diagnostic techniques discussed earlier are potentially relevant for they may help focus the diagnostic interaction. As we have seen, AI-ED diagnosis is an inherently difficult process because of the ambiguity of the observations. The technical aim might be to determine the right balance between analysis and interaction.

At the moment, there are no formal treatments of this view of diagnosis, only exploratory studies. Wu (1991), Cohen, Schmidt and van Beek (1994) and Elzer, Chu-Carroll and Carberry (1994) consider the nature of clarification dialogues, that is, dialogues initiated to resolve some diagnostic ambiguity. If a diagnosis suggests two candidates $c1$ and $c2$, then it might be better not to attempt a deeper analysis to eliminate one of them, but to engage the student in some interaction about them. The simplest option is to present the student with a menu of candidates from which to select. This is generally unsatisfactory because the candidates may be too detailed or technical for direct presentation, and in any case would lead to boring interactions. Generally, systems attempt to identify some key difference between the candidates and to ask a direct question about that. For example, given the two candidates
```
[[sub,borrow_no_decrement]]
[[sub,smaller_from_larger]]
```
for the observation {<60-45,25>} the system might ask "Did you decrement the 6?". Cohen et al (1994) use a hierarchy of plans to enable the identification of a 'key event', i.e. an event which would be part of one plan but not the other.

In general, there are more than two candidates to discuss. Therefore, it is necessary to devise techniques to enable some optimum partitioning of the set of candidates, as it would be too tedious to discuss them a pair at a time. This is a tricky little exercise involving the consideration of the probabilities of the candidates and the student's likely response to any query. Cohen et al's algorithm selects a question for which the minimal number of candidates predict a

'no' answer, the rationale being that if the answer is indeed 'no' then the system may eliminate the maximum number of candidates and if it 'yes' then the system stays 'on the right track' (or at least gives the student the subjective impression of a more coherent dialogue).

The problem of generating an appropriate question as part of a clarification dialogue merges with that of determining the next problem to pose to the student, which has been studied within AI-ED (for example, Evertsz (1989)). Of the many instructional actions open to a system, one is to present a new problem to aid the system rather than the student, by distinguishing between candidates. So, for example, a system with the above two candidates for subtraction might generate a next problem, such as "What is 61-45?"

This is analogous to the problem of determining the next measurement to make in device diagnosis. De Kleer and Williams (1989) suggest using the problem with the 'minimum entropy', that is, the one which provides the maximum information to discriminate between candidates. However, this does not directly tell us which problems to consider - an exhaustive search of the search space, even for subtraction, would be infeasible. The task is to determine a problem for which (in the case of two-candidate diagnoses) candidate c_1 predicts a student answer a_1 and candidate c_2 predicts a_2, where a_1 and a_2 are different. A complete analytic solution involving the symbolic execution of the hypothetical components corresponding to the candidates would be impractical for dynamic problem generation.

Instead, we may develop a heuristic method (in the sense that it orders the search space but does not eliminate possible solutions) as follows. Given the above two subtraction candidates, we might imagine that good diagnosticians would not generate the next problem by considering new inputs at random or in a fixed sequence but would tweak the previous problem, that is, they would find the minimum change sufficient to discriminate between the candidates. In this case, it is intuitively clear that it is better to change the 0 or 5 rather than the 6 or 4. This intuition can be made more precise by considering which components are affected by which inputs (Self, 1993).

Still, even this interactive view of diagnosis retains an asymmetry which may be disapproved of by those who favour egalitarian educational interactions. We still have one agent essentially being responsible for diagnosis and the other agent being diagnosed. In the case of truly collaborative dialogues, in which two or more agents are working and learning together, then all agents may need to diagnose one another, and (in principle) take account of the diagnoses all other agents have of themselves, and so on. Needless to say, there is little precise description and formulation of such nested diagnoses. But it leads us on to consider the nature of dialogue in AI-ED.

9. Dialogue

We have increasingly come to regard communications between students and AI-ED systems as some form of dialogue between agents, rather than as actions performed by one unto another. To be sure, there may be significant differences in the beliefs, plans and capabilities of the agents concerned, but there are also potential benefits in regarding the interactions as balanced exchanges of opinion aimed towards some learning objective.

It is a pity that criticisms of 'old-fashioned tutoring systems' have come to equate 'communication' with 'transmission', because the former, but not the latter, is necessarily a multi-agent activity. One can transmit a message to the stars seeking intelligent life but one cannot (necessarily) communicate it. Any communication requires an agent to send and an agent to receive. Reception is not a passive process: it requires the receiving agent to recognise the sender's intention to communicate and to understand the meaning of the communication. Communication is very rarely a one-shot, one-way action. All but the very simplest communicative act involves some kind of dialogue, if only to confirm receipt of the communication. For example, if you tell someone a telephone number its receipt is invariably acknowledged or echoed back.

In general, any communication involves the use of an agreed-upon language or set of conventions for communicating and the detailed coordination and monitoring of the components of the communication. Dialogue management will involve many of the concepts previously discussed. For example, it will require consideration of the agents' beliefs and of their beliefs about one another. The following example, adapted from Wilks and Ballim (1987), illustrates some of the issues. Imagine that the system c is mediating an interaction between two medical students a and b and that we have:

```
Beliefs(c) =  {  Type(thalassemia,genetic-disorder),
                 Medically-informed(x) ∅ B(x,Type(thalassemia,genetic-disorder)),
                 Average-person(x) ∅ B(x,Type(thalassemia,disease)),
                 Type(x,genetic-disorder) & Suffers(a1,x) & Suffers(a2,x)
                            & Child(a1,a2,a3) ∅ Suffers(a3,x), ... }
Beliefs(c,a) ={  [Medically-informed]
                 Suffers(fred,thalassemia),
                 Suffers(mary,thalassemia), ... }
Beliefs(c,b) ={  Suffers(fred,thalassemia),
                 Suffers(mary,thalassemia), ... }
```
The system believes a to be medically-informed but not b. From such student models, the system might reason that
```
        B(a,Type(thalassemia,genetic-disorder))
        B(a,Child(fred,mary,x) ∅ Suffers(x,thalassemia))
        B(b,Type(thalassemia,disease))
```
that is, that a will reason that a child of Fred and Mary will suffer from thalassemia but that b will not (on the default assumption that b is an average person). The system could then carry out independent dialogues with the two students but neither such dialogue would be of much interest to the other student. Instead, the system could take account of what one student believes the other student believes. For example, the system might consider that a believes b is also medically-informed (making the default assumption that b is the same as a unless a has evidence otherwise) and thus that
```
        B(a,B(b,Child(fred,mary,x) ∅ Suffers(x,thalassemia)))
```
Then, for example, the system might engage a in discussing with b why b's conclusions differ from those expected by a of b. In general, the point is that in any interaction between two or more agents it may help (or be essential) for an agent to hold beliefs about what may be believed by the other agent(s).

A dialogue between an AI-ED system and a student does not have to be in natural language but we can begin to get some idea of the nature of the problem by first considering the enormous volume of work carried out in computational linguistics, natural language processing and in linguistics, generally. Allen (1995) distinguishes the following kinds of knowledge needed for natural language processing:
- Phonetics - the production, perception and analysis of speech sounds.
- Morphology - the form and structure of words.
- Syntax - the acceptable arrangement of words in sentences.
- Semantics - the meaning of words and sentences.
- Pragmatics - the use of sentences in social contexts.
- Discourse knowledge - the use of sequences of sentences.
- World knowledge - general knowledge of the world and other people, independent of language use.

Texts on computational linguistics are concerned mainly with syntax and, to a lesser extent, semantics, reflecting their relevance to typical computer applications(such as machine translation and question-answering) and their relative tractability. Computational mathetics needs to consider syntax and semantics as well, of course, but does not seem to have any special demands of its own. For example, an AI-ED system might need to be able to handle a student input such as "Rice cannot grow on a hillside unless it is terraced" but will use standard techniques to do so. Therefore, for

these aspects, we can content ourselves with references to standard texts (such as Allen (1995), Gazdar and Mellish (1989)), bearing in mind that as students may not yet be able to use domain terminology correctly there may be more badly-formed inputs than normal.

Of more specific concern to computational mathetics are the more difficult issues of pragmatics and discourse, because we are concerned with the structure of lengthy interactions between systems and students and with the effect that such interactions may have on the participants' beliefs and goals. We can anticipate that a real teacher-student dialogue such as that below (Hull, 1985) will present quite a challenge:

T: Who knows about the structure of the atom? (No reply)
T: No? Well, ..[brief explanation] .. So an atom is mostly ...? (Pause) What's in between?
S: Air.
T: Air is made up of atoms. What's in [indistinct]?
S: How do you split them?
T: We're interested in ...
S: How did they find out about the nucleus and [indistinct]?
T: It would take too long to tell .. All right, then .. there was a chap called Rutherford .. he produced a sheet of atoms ..
S: How?
T: .. and bombarded it with particles.
...

In this chapter, we will first discuss general issues concerned with dialogue and then focus on aspects specifically relevant to AI-ED. When this research relates to dialogue with computer systems, it has usually been with respect to a general user, but we will focus on students in this chapter. The general discussion is necessarily rather superficial, as the topics are very broad but still controversial, with little agreement on technical formulations (Allen discusses these topics only in his last two chapters and Gazdar and Mellish hardly at all).

There is also a perennial debate about the extent to which thinking and learning are inherently related to dialogue. The everyday views that
• language follows from thought, that is, we think and then express thoughts,
• thought follows language, that is, the language we use determines what we think,
• thought and language are inter-dependent,
• language and thought are identical, the latter being only an internalised language,
and so on, provide philosophers, psychologists and linguists with fertile ground, but are outside the scope of computational mathetics. To the extent that such discussions agree that dialogue promotes learning they support our consideration of aspects of dialogue.

9.1   Discourse structure

Because AI-ED systems are concerned with analysing inputs within the context of an on-going interaction, we are concerned with what in natural language understanding is called *discourse analysis*. First, it should be noted that the prefix is dis- (indicating reversal) rather than di- (indicating two) as in dialogue, which indicates that discourse means (a talk) 'running to and fro', typically involving one person, not two. Even so, the techniques developed for (single-agent) discourse will be useful for (multi-agent) dialogues.

Discourse analysis involves the partitioning of a sequence of sentences into segments each of which is locally coherent, in that each sentence in a segment is concerned with the same topic. Segments are considered to be hierarchically-organised so that they may be temporarily suspended during an interrupting subsegment. Within a segment it is assumed that standard techniques of computational linguistics can be applied. The main difficulty lies in identifying transitions between segments. In English, cue phrases ("On the other hand ..", "Incidentally ..,", "Firstly ..", "In particular ..", and so on) often mark the beginning and, less often, the end of segments. One of the early studies of discourse (Grosz, 1977) looked at an expert helping an apprentice assemble a water pump. The discourse followed a plan structure to assemble the pump, with segments corresponding to components of the plan.

Cue phrases signal the intentional structure of the discourse. For example, "In particular" and "For example" indicate a new subsegment which elaborates upon the embedding segment, and "On the other hand" and "Similarly" introduce a parallel construction. Discourse analysis involves the identification of such relationships between sentences and segments.

The Rhetorical Structure Theory (RST) of Mann and Thompson (1986) has been influential in providing a taxonomy of such relations. RST describes text structure in terms of 'nuclei' and 'satellites'. The nuclei refer to the main ideas or goals of the writer. The satellites provide various kinds of support for the nuclei. This support may be an example, an elaboration, a reason, and so on. The top-most schema organises the text as a whole, with subschemata providing a

hierarchical structure to the text. Mann and Thompson postulate that the following relations (which they define fairly precisely) are sufficient to describe normal English text:

```
evidence            justify             antithesis          concession
circumstance        solutionhood        elaboration         background
enablement          motivation          volitional cause    non-volitional cause
volitional result   non-volitional result purpose           condition
otherwise           interpretation      evaluation          restatement
summary             sequence            contrast            joint
```

Moore and Pollack (1992) argue that RST does not distinguish sufficiently between what they call the informational level and the intentional level. Although RST does have both informational (that is, concerned with the subject matter) and intentional (that is, concerned with the intended effect upon the hearer) discourse relations, it requires only a single element to hold between a pair of discourse elements. Moore and Pollack argue that it is necessary to have both (or more) kinds of relation holding simultaneously. For example, in our beloved Socratic dialogues about rice-growing, we might have a dialogue fragment: "(a) The Japanese grow rice. (b) They love saké." If the student knows that saké is made from rice, then she may infer that (b) is an informational element intended to increase her belief in (a). Or she may infer the information that, as (b) is intended to increase her belief in (a), there must be some relation between them, for example, that saké is made from rice. Thus, a satisfactory analysis requires a parallel consideration of both informational and intentional aspects.

RST and similar theories can be used both to describe and to generate discourse. Discourse generation is seen as a problem of generating (linguistic) actions to achieve some goal and therefore as a special case of planning. It differs from general planning in that the acts are directed towards another agent, rather than for one's own benefit. This second agent is liable to interpret the acts in ways similar to oneself. Therefore, language generation involves an interplay between two agents' related abilities to generate and analyse discourse and a consideration of how these abilities take account of the other agents' beliefs, goals, and so on. Moore (1995) provides a detailed account of the use of planning formalisms to generate discourse plans to support human-computer dialogues.

Hovy (1993) reviews attempts to use RST to generate discourse. Although a parsimonious set of relations may be adequate for descriptive purposes, discourse generation apparently needs a much larger set. Hovy has accumulated relations proposed by other researchers and produced a taxonomy of some 120 relations. With such relations it is possible to generate paragraphs such as

"When making a handoff, the transferring controller relays information to the receiving controller in the following order. He gives the target's position. He gives the aircraft's identification. He gives the assigned altitude and appropriate restrictions."

Of course, we are suitably impressed only after a detailed study of how 'far back' a system has started in generating such a paragraph. We know that any particular output can be produced from more-or-less canned schemata, and indeed, because of the difficulties of generating text of any length, most AI-ED system output is not generated in any profound sense. Nonetheless, Hovy (1993) concludes that "it is not unreasonable to expect the flexible planning and generation of coherent, multi-page texts in limited domains within the next five years".

9.2   Speech acts

Communications between teachers and learners are 'speech acts' like any other communication. In speech act analysis, a given speech act may be described in:
- *locutionary* terms, i.e. in terms of the physical acts of producing a sequence of linguistic signs;
- *illocutionary* terms, i.e. in terms of its content and intended effect;
- *perlocutionary* terms, i.e. in terms of its actual effect upon the hearer(s).

The last two need not coincide, of course. If a doorstep stranger announces that "I am a Jehovah's Witness" then the effect upon the listener may not be that which is intended.

Illocutionary acts may be organised into different types. For example, Searle (1976) proposes five types, based on verb meanings:
- Representatives, where the speaker is committed, in varying degres, to a proposition (e.g. "I believe, confirm, report, ...").
- Directives, where the speaker tries to get the hearer to do something (e.g. "I ask, challenge, insist, ...").
- Commissives, where the speaker is committed to a course of action (e.g. "I promise, guarantee, ...").
- Expressives, where the speaker indicates an attitude to a state of affairs (e.g. "I apologise, deplore, thank, ...").
- Declarations, where the speaker alters the situation by making the utterance (e.g. "I resign, baptise, ...").

Or we could classify according to the mode of communication (Davis, 1990) into:
- Declarative acts, which convey information.
- Interrogative acts, which ask for information.
- Imperative acts, which make a request or issue a command.
- Exclamatory acts, which express an emotion.
- Performative acts, which bring about a condition

Successful speech acts should satisfy their so-called 'felicity conditions'. Some such conditions are preconditions - for example, one cannot resign without being in a position to do so. Some concern the manner in which the act is performed - for example, one cannot welcome with a surly demeanour. Such conditions are met naturally in everyday dialogues, but not necessarily in teacher-student dialogues.

Human teacher-student interactions are particularly rich in subtle speech acts. A student needs to determine what is intended by a surface speech act such as "Do you know if they grow rice in Malaya?". A student who simply responded "Yes" would be considered to have missed the point of the question (or to be bored with the topic or the convoluted way it is being discussed). This is not a real question in many ways - for a start, it may be presumed that the asker knows the answer to the embedded question.

For subjective reasons, many teacher comments are very indirect speech acts. For example, what is a student to make of "That is an interesting answer"? First, perhaps, that it is wrong. Maybe it's an answer that the teacher cannot himself explain. The precise interpretation depends on the context: if a polite computer tutor always used this phrase instead of "Wrong", a student would soon regard them as identical.

Speech acts, like other acts, are presumably carried out for a purpose, that is, to help achieve goals. Therefore they might be defined in ways similar to other acts, namely in terms of the preconditions which must hold before the act may be performed and the effects of the act. The preconditions and effects will refer to the states of the agents involved in the communication. For example, a `convince-by-inform` act might be defined by:

```
convince-by-inform(a,b,p):
    if B(a,p) & At(a,loc(b)) then B(b,p)
```

The idea of speech acts, particularly performatives, has been adopted in the definition of KQML, an agent communication language (Genesereth and Ketchpel, 1994) proposed as a standard for 'knowledge sharing' between distributed information systems. About forty performatives are defined in terms of their components, enabling messages such as:

```
(evaluate :content (val (torque motor1) (sim-time 5)) :ontology motors
    :reply-with q1 :sender a :receiver b)
(reply :content (scalar 12 kgf) :ontology motors :in-reply-to q1 :sender b :receiver
    a)
```

i.e. agent `a` asks agent `b` to evaluate some value and `b` replies.

9.3   Dialogue game theory

Dialogue game theory is a formal device for generating well-formed sequences of locutionary acts. It is semi-empirical in that it is based partly on analyses of discourse and partly on abstract specifications valid processes of reasoning, discussion and argumentation. The theory has three components:
1. A set of 'issue spaces' (Reichman, 1985) or 'commitment stores' describing what each participant believes or is committed to at any given stage of the dialogue;
2. A definition of the set of locutionary events (moves in the dialogue game), with a definition of the changes to the issue spaces when such an event occurs;
3. A set of constraints on the sequence of events, from which is intended to emerge the coherent episodic structure of rational dialogue.

Pilkington, Hartley, Hintze and Moore (1992) use dialogue game theory to design an interface to support argumentation. There are four components to the system:
- The dialogue moves: e.g. `statement`, `question`, `withdraw`, `challenge` and `resolve` (the last being a request to resolve a logical inconsistency).
- A set of 'commitment rules' defining the moves' effects upon the players' commitment stores, e.g. after a `challenge` "why p?" if the next move is a `statement` "q" then both "p" and "if q then p" enter both players' commitment stores.
- A set of 'game rules' defining when the moves are allowed, e.g. after "why p?" your move must be a `withdraw` of p, a `statement` other than p, or a `resolution` demand.
- A definition of when a player wins, e.g. when the opponent cannot legally resolve an inconsistency in his commitment store.

Given these definitions, a computer system can referee a game (an argument) between two players. In itself, this may be useful to help students develop argumentation skills, but it is natural to wonder if the system could take part as one of the players. To do so, the system would need to have (access to) beliefs of its own to argue about and, unless the discussion made explicit every step of the argument, which would be rather tedious, it would need to be able to reason about the content of those beliefs. It would also need to have strategies for playing the game. For example, it might use heuristics such as "If you believe something not in the opponent's commitment store which you intend to use to support your argument, then pose the statement as a question". We may also try to develop strategy rules which are not intended to help the system win the argument but to cause the opponent to come to hold particular beliefs - an

educational objective, which we will reconsider in chapter 10. (If we need to relate this to the earlier framework: the contents of the commitment store are beliefs, the game rules are reasoners, and the strategy rules are monitors.)

9.4   Rational dialogue

In an AI-ED context, dialogue game theory posits a central role for the student model (or issue space, as it is termed there) and considers that student model updating occurs as an on-going part of the instructional dialogue, not as a result of some separate diagnostic process. Dialogue game theory goes some way - maybe far enough to manage AI-ED interactions - towards showing how Gricean maxims of conversation (see below) "fall out from a general characterisation of the aims and means of linguistic exchanges together with obvious assumptions of rationality of the participants" (Carlson, 1983). We may also attempt to make these "obvious assumptions" explicit, to provide a deeper theory of communication.

For example, Cohen and Levesque (1990b) present a four-stage derivation of the basis of a theory of communication. First, they introduce a set of primitive modal operators intended to define the mental states of the participants. These operators are expressed in a modal logic based on a possible world semantics of knowledge and a situation calculus model of action. The four operators defined are:

> `Bel(x,p)` - p follows from x's beliefs (this is therefore an implicit belief);
> `Goal(x,p)` - p follows from x's goals;
> `Bmb(x,y,p)` - p follows from x's beliefs about what is mutually believed by x and y;
> `After(a,p)` - p is true in all courses of events that obtain from act a's happening.

These operators are defined through a set of propositions and lemmas, for example, that of 'shared recognition':

> `Bmb(y,x,Goal(x,p)) & Bmb(y,x,Bel(x,Always(p∅q))) ∅ Bmb(y,x,Goal(x,q))`

Secondly, a theory of rational action is developed by means of a set of propositions defining the properties of ideally rational individual agents with persistent goals, for example, to specify that agents do not knowingly and deliberately make their persistent goals impossible for them to achieve. Theorems may then be derived from such propositions, for example,

> `Persistent-goal(x,p)    &    Always(Competent(x,p))    ∅    Eventually(p   v`
> `    Bel(x,Always(x,~p))`

that is, if an agent has a persistent goal p that it is able to bring about then eventually p becomes true or it believes that nothing can be done to achieve it. The general problem of planning (section 6.4) is concerned with developing such a theory and we can adopt several constraints on rationality, for example, that an agent cannot intend an action it believes it cannot perform, nor an action which it believes mutually excludes another action it intends.

In the third stage, a theory of rational interaction is expressed as a set of definitions and propositions intended to characterise interactions between agents. At this level, we need to take account of an agent's ability to form models of other agents' beliefs, plans and goals. For example, an agent x may be said to be sincere or expert with respect to y and a proposition p under the following conditions:

> `Sincere(x,y,p) × (Goal(x,Bel(y,p)) ∅ Goal(x,Know(y,p)))`
> `Expert(x,y,p) × (Bel(y,Bel(x,p)) ∅ Bel(y,p)))`

Such definitions of cooperative agents provide formal descriptions of the kinds of behaviour summarised by conversational maxims, such as the well-known four maxims of Grice (1975):

- Quality - say what you believe to be true.
- Quantity - say no more and no less than is needed for the purposes of the communication.
- Relevance - say something related to the purposes of the communication.
- Manner - say things clearly.

Of course, such informal maxims provide only general guidelines which are often violated (particularly in teacher-student interactions).

Finally, a theory of communication is presented as descriptions of communicative acts such as questioning and requesting derived from general principles of belief and goal adoption between agents. These descriptions enable a distinction between, for example, real questions, rhetorical questions and teacher-student questions. In principle, multi-act utterances and multi-utterance acts can be handled in the same scheme. The structure of dialogue is defined in terms of various relations, be they called discourse relations, rhetorical relations, dialogue game moves, or whatever.

The definition of the content of the various levels is complex, but the intention is that each level be independently motivated, that is, for example, that the notion of a cooperative agent be developed independent of that of communication, and that of rational action be independent of that of interaction. Thus, the derivation of communicative acts could be based ultimately upon the kinds of representation of agents' cognitive structures that we have previously adopted. Consequently, there is a debate about the need for explicit dialogue relations, as they may be regarded as arising from a more basic model. The extent to which such a deep analysis is necessary to support adequate system-student interactions in practice remains to be seen: at the moment, practical systems manage with explicit dialogue relations.

So far in this chapter, the descriptions of dialogue have derived from considerations of general human-human interactions. We have not specifically considered teacher-student interactions, which are assumed to be just a special case of the general theories. If we use the dialogue game terminology, we can see that certain 'patterns of play' are particularly common in educational interactions, and we will consider three of these in the following sections. We may also anticipate that system-student interactions, especially with the new interfaces, may be rather different and we will discuss this briefly in the final section of this chapter.

9.5   Explanation

In an educational context, the 'explanation' is one of the most common dialogue structures, perhaps corresponding, in the simplest case, to a sequence of `challenge-statement` moves in dialogue game theory, as with a child who keeps asking "Why?". Like a diagnosis, an explanation is both an object and a process for generating that object, and, as with diagnosis, the trend has been towards focussing on the process rather than the object.

The view, previously philosophically dominant, of explanation as an object leads naturally to attempts to classify explanations into different types and to map different kinds of explanation onto different situations and types of student. It is a view implicit in early work on expert systems and indeed in the use of the word in explanation-based learning, where an explanation is simply a proof of some proposition. In most expert system shells, the system generates an 'explanation' by backtracking through its proof. In this view, then, the production of an explanation is intimately tied to a domain model and the main technical problem concerns the selection and textualistaion of parts of the proof.

The next step is to make the nature of an explanation explicit so that a system may dynamically generate an explanation structure instantiated to the specific context. The Explainable Expert System (EES) uses RST-like structures to define about one hundred plans which can be combined to develop a structured explanation (Moore, 1989). For example, text plans such as:

```
persuade-by-motivation(c,s,g):
    if Goal(c,g) & Goal(s,g) & Step(act,g) & Motivation(act,g)
       then Persuaded(s,Goal(s,do(s,act)))
motivate-act-by-means(c,s,g):
    if  Goal(c,g)  &  Goal(s,g)  &  Step(act,g)  &  Inform(c,s,Goal(c,g))  &
    Means(g,act)
       then Motivation(act,g)
```

are used to generate text for a goal , by proving subgoals, some of which produce text as a side-effect:

```
Persuaded(s,Goal(s,do(s,replace)))
-> Motivation(replace,enhance-readability)
   -> Inform(system,s,enhance-readability)
                      -> "I'm trying to enhance the readability of the program"
   &  Means(replace,enhance-readability)
                      -> "by applying transformations that enhance reliability"
      ...
```

Tattersall (1992) describes the use of rhetorical predicates in PORSCHE, an intelligent help system intended to be independent of the application with which the student is being helped. The predicates are divided into content and organisational predicates (mirroring the above distinction between informational and intentional levels). A student query, such as "What is `delete`?", is answered by expanding the core set of rhetorical predicates (`membership`, `attribution`, `constituency`, `replacement`, and `representative`) and then using organisational predicates to structure the content:

> "`delete` is a command which is used to erase messages. It has the effect of setting the deleted and touched flags and updating the current message pointer. `delete` is available in mail mode and takes a mail specification as argument. Its syntax is `delete <mail-specification>`, for example, `delete *` or `delete <message-list>`."

The system supports follow-on questions, which is an implicit recognition that it is difficult to generate explanations at the right level of detail for all students.

The generation of an adapted explanation presumes the system has a model of the student to which the explanation should be adapted. With a good student model, an explanation may be tailored in various ways:
- by using or avoiding particular technical terms (if, for example, the student model indicates that she does not know what a 'touched flag' is);
- by omitting or including material (for example, by omitting detail if the student model indicates it is already understood or would be too complex);
- by using different presentation strategies.

However, this places currently unsatisfiable demands on the student modelling component, and as a result most explanation systems allow follow-on questions through which a student may clarify an initial, provisional explanation. Moore and Paris (1992) discuss how a system can allow for inaccurate or incomplete student models by reacting to student feedback, in the form of follow-on questions or indications of a lack of understanding. The system's reaction

should depend on previous interactions, including the explanations themselves, which must therefore be understood by the system. Their extension of EES (discussed above) uses a student model containing elements of the form:

```
B(c,Goal(s,g));  the system believes the student has goal g.
B(c,B(s,p));  the system believes the student believes some proposition p.
B(c,B(s,Concept(c)));  the system believes the student knows a description of some concept c.
B(c,Competent(s,Do(s,act)));  the system believes the student can perform the action act.
B(c,Competent(s,Achieve(s,g)));  the system believes the student is competent to achieve goal g.
```

and also stereotypes, such as

```
B(c,Novice(s))
```

If a student indicates a lack of understanding of an explanation generated as described above, then the system reflects on its plan for generating the explanation to see what may have gone wrong. If the system finds an assumption that the student knows a concept, for example, B(c,B(s,Concept(generalized-variable))), then it may explain that concept, in case the assumption was unfounded. If it can find no such assumption, then it may re-plan the explanation. If there are no alternative plans, then it may present a menu of follow-on questions, such as "What is a generalized variable?".

Such a reactive explanation capability alleviates the burden on the student modelling component. So, for practical reasons, we come to a view of explanation as not something presented by one agent to another, but as something jointly constructed between the two. An explanation cannot be an object which can be analysed or generated independently of the agents involved. There is, of course, an asymmetry in that the aim is that one agent should come to believe something previously believed by the other agent, but the beliefs, goals, context, etc. of both agents determine the quality of an explanation (consider what constitutes an explanation for Australian aboriginal children, section 1.1).

Not only should explanations be interactive but they should be tightly coupled with the diagnostic process (Cawsey, 1993). In general, the more interaction there is, the more evidence there is to base diagnosis on, and the better the diagnosis, the easier it is to produce a suitable explanation. Therefore, diagnosis and explanation (and other forms of interaction) are mutually reinforcing processes.

The EDGE system (Cawsey, 1993) plans explanations of 'how things work' using about twenty content planning rules, such as

```
plan how-it-works(device)
    if K(s,Constituency(device))
    then Goal(Process(device)) & Goal(Behaviour(device))
```

that is, to explain how it works, if the student knows the structure of the device then describe the device's processes and then its overall behaviour, and

```
plan constituency(device)
    if Device-analogy(device,x) & K(s,Constituency(x))
    then Goal(Compare-constituency(device,x))
```

that is, to explain the structure, if there is an analogous device which the student knows the structure of then compare the two structures. Planning an explanation is beginning to look like a special case of planning a curriculum (section 10.3). The student model used in EDGE, as with most discourse planning systems, is rather simple, being just an overlay model, with no representation of misconceptions of alternative views. The student is modelled as knowing, maybe knowing or not knowing each fact. Direct inferences (i.e. from what the system or student says) about what the student knows are considered more reliable than indirect inferences (i.e. from the domain structure or from stereotype assumptions). During an explanation, the system prefers not to try to make the approximate student model more precise but to ask explicit questions. For example, if the student model indicates that the student maybe knows how an astable multi-vibrator works then rather than assume that she does the system will, if it has that goal, ask the question "Do you know how an astable multi-vibrator works?". The design objective is that such questions be seen by the student as an intrinsic part of the explanatory process, not as an irrelevant question for the system's benefit (similar points were made in section 8.6). The answers to such questions may lead to changes in the student model and to changes in the planned explanation. In principle, such changes could involve general model maintenance techniques (section 8.3), but EDGE actually uses a simple numerical scheme.

The overall conclusion from the work on explanation is that, when broadly construed, the process of giving or constructing an explanation is a central activity of AI-ED systems and therefore it is not surprising that many of the representations and techniques (for planning, diagnosis, dialogue, and so on) that we have discussed need to be integrated in any comprehensive approach.

9.6   Argumentation

In argumentational reasoning a proposition is believed if it cannot be defeated by attacking arguments. This might be expressed as:

```
B(a,p):- ~Defeated(p).
Defeated(p):- Attacks(q,p), B(a,q).
```

This formulation refers only to uni-agent argumentation (that is, reasoning about one's own beliefs by 'arguing with oneself'), in which case argumentation is just a form of nonmonotonic reasoning as discussed earlier. To correspond better to the everyday sense of 'argument' and to relate argumentation to dialogue, we need to imagine that the attacking arguments come from a second agent. The reformulation:

```
B(a,p):- ~Defeated(p).
Defeated(p):- Attacks(q,p), B(b,q).
```

does not, however, capture the to-and-fro, temporal nature of an argument, where agent `b` repeatedly proposes propositions which it believes to attack the proposition believed by agent `a` (and vice versa).

Nonetheless, most studies of the structure and acceptability of arguments in philosophy, logic and AI have not worried overmuch about the source of the propositions which constitute an argument. The general idea in argumentation is that a proposition will be or may be believed only if there is no evidence to the contrary, it being the role of the second agent to present that evidence. As that evidence is also a proposition it too may form the basis for an argument, leading to nested argument structures.

Pollock (1992) relates a philosophical account of argumentation (or defeasible reasoning) to AI's descriptions of nonmonotonic reasoning and reason maintenance. Using our terminology, if an agent has a belief-set `B` and a reasoner-set `R` then it has a *reason* for believing `p` if

```
r1(p1,p2,..,pn) => p
```

where a reasoner `r1` (a member of `R`) when applied to `p1,p2,..,pn` (all members of `B`) derives `p`. A *rebutting defeater* is a reason for believing `~p`:

```
r2(q1,q2,..,qn) => ~p
```

An *undercutting defeater* is a reason for believing that the original reason does not imply `p`:

```
r2(q1,q2,..,qn) => ~(r1(p1,p2,..,pn) => p)
```

These two kinds of defeater are considered to be the only ones there are. An argument is then considered to be a sequence of reasons. An argument `B` is said to defeat an argument `A` if `B` contains a reason which rebuts or undercuts a reason contained in `A`. If an argument `A1` for a conclusion `p` is defeated by an argument `A2` which is in turn defeated by an argument `A3`, then `p` is said to be unwarranted at level 1 and warranted at level 2 (and so on, perhaps). The argument for `p` is *ultimately undefeated* if there is a level `m` such that for every level `n>m`, `p` is warranted.

However it remains to provide a satisfactory account of how the arguments at each alternate level may be provided by different agents, how the steps of the argument may be interleaved, and the effects that such arguments have on the beliefs of the participants involved.

9.7   Negotiation

Explanations, arguments and negotiations overlap in various ways but we may distinguish them simply in the following way:

```
B(a,p) & ~B(b,p)  - Explanation ->  B(b,p)
B(a,p) & B(b,~p)  - Argument ->  (B(a,p) & B(b,p)) v (B(a,~p) & B(b,~p))
B(a,p) & B(b,q)   - Negotiation ->  B(a,r) & B(b,r)
```

In a (successful) negotiation, two agents begin with different views and end with the same view (which may or may not be one of the original views).

In the less dogmatic styles of AI-ED system, the student may have greater scope for following her own goals and developing her own understanding. In such a case, the system may offer (fallible) comment and advice in the role of a cooperative partner rather than a knowledgeable tutor and hence engage in some kind of 'negotiation' with the student (Moyse and Elsom-Cook, 1992). Such a role may be achieved by a disingenuous concealment of its domain knowledge by the system but the role is likely to be more appropriate in situations where computational representations of domain knowledge are unattainable or controversial. If the system's domain knowledge is not complete or necessarily correct, then the system may need reasoning and learning capabilities commensurate with those of the student. With such capabilities, the system may maintain a student model, which together with the system's model, represents some joint understanding of the domain. Naturally, in such a context, the student model is less an internal component of the system but becomes an 'external' focus of discussion.

Concepts such as *negotiation* and *cooperation* have been much studied in distributed AI, where Durfee and Lesser (1989) consider that there is "confusion and misunderstanding among researchers who are studying different aspects of the same phenomenon". They urge that we distinguish carefully between negotiations which are about the shared construction of meaning and those which are about task-sharing or planning. Both are central to the philosophy of AI-ED systems, the former being concerned with the nature of knowledge and the latter with the issue of student control

In most multi-agent problem-solving situations it is not acceptable for a single agent to take sole responsibility for decision-making. Often, it is necessary for the decision-making agent to persuade the other agents to agree to a decision, to enable them to revise their viewpoints so that they may continue to be able to offer contributions and, in the case of human agents, perhaps for subjective reasons, to encourage future cooperation.

Teachers (and ITSs) possess a rich set of techniques for persuading students to adopt a desired viewpoint. The simplest technique is to transmit the required viewpoint, that is, to tell it to the student, and to assume that it will overwrite the student's. The limitations of this technique hardly need elaborating at this point. More subtle are the various 'challenging' techniques in which the student is led, by the use of counter-examples perhaps, to question the basis for her viewpoint and to revise it in the required direction.

In more egalitarian multi-agent systems, where there is no 'distinguished agent' responsible for resolving conflicts and making decisions, it may be possible to apply the technique of *mediation*, in which an independent, external agent, with no vested interest in outcomes, is brought in specifically to find some compromise between viewpoints. Hewitt (1986) proposes such a technique for dealing with conflicts between microtheories (that is, small, consistent sets of beliefs). The outcomes from the microtheories are passed to *metamicrotheories*, which contain axioms about microtheories and engage in extra-deductive techniques such as debate and negotiation to deal with inconsistencies and conflicts between microtheories. The metamicrotheories may themselves be inconsistent with one another. Unfortunately, the decision-making procedures of the metamicrotheories are not formally specified.

If two or more agents have conflicting views which have to be reconciled then, with or without a mediator, some negotiation is necessary. In distributed AI there are many references to techniques which are considered to involve negotiations among nodes of a system. Indeed, the centrality of the idea of negotiation was emphasised by Davis and Smith (1983) in an attempt to establish it as a dominant metaphor for distributed problem-solving. They place negotiation within the mainstream of AI control structure development as a natural extension from the one-way information exchanges of then current AI programming systems.

Technically, their main contribution is the *contract net*, which provides opportunistic, adaptive task allocation among agents using a framework based on task announcements, bids and awarded 'contracts'. An agent can adopt both 'manager' and 'worker' roles and a 'contract net protocol' enables the opportunistic communication between managers and workers. As the contract net works through the mutual selection by both manager and worker processes, it differs from manager-centred invocations (such as procedure calls) and worker-centred invocations (such as data-driven computations). However, there is no mechanism for reasoning about the global effects of local decisions, nor any metalevel control, and consequently any global coherence emerges only incidentally.

Work on negotiation in distributed AI makes only token reference to studies of human-human negotiations, such as union-employer bargaining. Even in work that sets out explicitly to model human-human negotiations, such as that of Sycara (1989), which is concerned with adversarial conflicts in labour relations, it has been found (as usual) that sociological texts do not provide the required precision and it has been necessary to devise new computational representations. In Sycara's system, called Persuader, negotiation is regarded as an iteration through three steps: generation of a compromise proposal by a mediator, generation of a counter-proposal based on feedback from a dissenting agent, and persuasive argumentation based largely on case-based reasoning. Various argument types are defined, based on appeals to universal principle, a theme, authority, 'status quo', 'minor standards', 'prevailing practice', precedents as counterexamples, self-interest, self-protection, and so on - but this is clearly an area where further research is needed.

The different techniques developed in distributed AI address different aspects of negotiation. For example, the contract net is concerned mainly with the allocation of tasks to agents, whereas Persuader emphasises the iterative exchange of counter-proposals leading to compromise. In an attempt to impose some consensual basis, Durfee and Lesser (1989) propose a general definition of negotiation: "the process of improving agreement (reducing inconsistency and uncertainty) on common viewpoints or plans through the structured exchange of relevant information".

This definition has the virtue of distinguishing two aspects which have become intertwined in distributed AI and are in danger of becoming so in AI-ED research, namely, the distinction between negotiations aiming for shared plans and those seeking common viewpoints. Baker (1994) distinguishes the two in the following argument for the importance of negotiation in AI-ED:
Because
- tutors do not possess complete knowledge for the domains which they teach;
- and, for some domains, there is no single correct viewpoint on knowledge;
- and, for some domains, there is no knowledge but only a number of competing sets of justifiable beliefs;
- and, for some problems, there are multiple acceptable solutions
tutors
- should not aim to simply transmit their own knowledge (beliefs) to the student;
- and should not have complete control over the tutor-learner interaction;
- and should jointly construct with the learner representations of the domain to be learned;
- and should use negotiation mechanisms to support tutor-learner interactions.

The conditions are claims about the nature of knowledge and the conclusions are claims about tutor-learner interactions.

Baker (1994) presents a detailed analysis of some aspects of negotiation in tutor-learner dialogues. It is assumed that in a negotiation there is a mutual goal of agreeing on some proposition which is subject to various constraints:

```
Mutual-goal(a,b,Agree(a,b,p))
```

and that both agents have equal rights to propose, accept or reject propositions. Negotiation processes are considered to consist of strategies (i.e. general methods of achieving the negotiation goal), arguments and actions (i.e. communicative acts). In the 'refine' strategy an agent seeks to modify an offer of the other agent. An 'offer' is a different kind of speech act to those considered in section 9.2 in that it is conditional rather than categorical, so that the appropriate epistemic attitude is one of acceptance rather than belief.

The style of the analysis can be illustrated with this very simple dialogue:
1. Why don't we discuss fractions?
2. OK, let's.
3. Fine.

It is assumed that for each communicative act there is a set of 'appropriateness conditions' {c} such that, after x performs the act:

```
K(x,(Mutually-believe(x,y,(Assumes(x,B(y,c))))))
K(y,(B(y,c))
K(y,(Mutually-believe(y,x,(Assumes(x,B(y,c))))))
```

For example, the appropriateness conditions for an offer from x to y might include:

```
B(x,(Accepts(y,p)⊘Accepts(x,p))
```

If p denotes the proposition that "a and b will discuss fractions", then after speech act 1 we have (using obvious abbreviations):

```
1.  K(a,(MB(a,b,(Ass(a,B(b,B(a,(Accepts(b,p)⊘Accepts(a,p)))))))))
2.  K(b,(B(b,B(a,(Accepts(b,p)⊘Accepts(a,p)))))
3.  K(b,(MB(b,a,(Ass(a,B(b,B(a,(Accepts(b,p)⊘Accepts(a,p)))))))))
```

As a made the offer we might also posit:

```
4.  K(a,B(a,(Accepts(b,p)⊘Accepts(a,p)))
```

If the appropriateness condition for act 2 (and 3) is Accepts(x,p) then after act 2 we have

```
5.  K(b,(MB(b,a,(Ass(b,B(a,Accepts(b,p)))))))
6.  K(a,(B(a,Accepts(b,p)))
7.  K(a,(MB(a,b,(Ass(b,B(a,Accepts(b,p)))))))
```

Using the normal modal logic axioms we can infer from 4 and 6 that:

```
8.  K(a,(B(a,Accepts(a,p)))
```

and similarly after speech act 3 that

```
9.  K(b,(B(b,Accepts(a,p)))
10. K(b,(B(b,Accepts(b,p)))
```

From 6, 8, 9 and 10 we might infer (by definition, though a fuller account would consider the mutually believed assumptions):

```
11. Agree(a,b,p)
```

It would be a daunting prospect to attempt such an analysis with more lengthy dialogues, with all the subtle contextual effects and implicit speech acts of normal natural language dialogues. As with other complex analyses, lemmas would be defined to enable macro-inferences, for example,

```
Offers(a,b,p) & Accepts(b,p) & Ratifies(a,p) ⊘ Agrees(a,b,p)
```

The dialogue would be considered to consist of 'stages' linked by dialogue game rules, for example,

```
If Stage(n,Offers(a,b,p1)) & Stage(n+1,Offers(b,a,p2)
   & Mutually-believe(a,b,Incompatible(p1,p2))
then, after stage n+1, Mutually-believe(a,b,~Accepts(b,p1))
   & <standard effects of Offers>
```

Such analyses may or may not turn out to be possible to support the on-line management of negotiative dialogues but they may at least clarify the possible roles of negotiation in teacher-learner or system-learner dialogues and also the nature of negotiation, which is clearly a complex process involving the consideration of mutual goals and understanding and the influence of speech acts upon them.

9.8   Multimedia dialogues

All the theories of dialogue discussed above have been developed by considering natural language dialogues although it is clear that most dialogues in practice involving multiple modes (involving different sensory channels) and media.(involving different technologies for communicating information). With the advent of new multimedia, there is much optimism that some of the difficulties of natural language computer-human dialogue may be circumvented. In principle, the new media should allow communications of better quality (less ambiguity, fewer errors) and greater quantity (as more bits of information may be generated and transmitted per second). Unfortunately, the new media, such as graphical displays, interactive video, CD-ROM, three-dimensional sound, datagloves, virtual reality, and so on, are proliferating faster than it is possible to determine guidelines for their effective use. Even the most

straightforward conventions (such as the greying out of unusable items in menus) turn out to be rather complicated when subjected to empirical study.

Currently, most multimedia presentations are pre-determined and therefore take no advantage of the computer's ability to adapt the presentation to suit the on-going interaction. Pre-determined presentations require conventional skills such as film-making and are not of direct relevance to computational mathetics. Roth and Hefley (1993) discuss why multimedia systems need to make use of AI techniques: to "automate the process of designing presentations and thereby communicate effectively" especially when designers "cannot anticipate all possible combinations of information that will be requested for display." In these terms, the rationale for intelligent multimedia systems is the same as that for AI-ED systems in general.

Attempts to develop analyses of multimedia communication are built directly on the relatively solid ground of natural language theories. Although there is some disagreement that natural language utterances are generated in a two-stage process of planning what one wants to say and then deciding on a form of words, this separation is adopted in most schemes for multimedia presentation. As far as the planning stage is concerned, it is hoped that the mechanisms developed for planning natural language utterances will be directly usable. For example, we can anticipate that rhetorical speech acts such as `compare` or `emphasize` may be achieved by various visual acts, such as overlaying drawings or zooming in.

The outcome of the first stage is generally a tree-like representation of the structure of the imminent multimedia presentation. The second stage, of generating multimedia output, requires descriptions of the various media so that particular outputs can be allocated to appropriate media. It is, of course, the case that different types of information are best presented via different media and therefore a system will need to be able to reason about how to integrate the use of different media to form a coherent communication. As Arens, Hovy and van Mulken (1993) discuss, multimedia presentation rules can be very specific:

```
Ships' locations are presented on maps.
```
or more general:
```
Data duples (e.g. ships' locations) are presented on two-dimensional media (e.g.
    maps).
```
Ideally, such rules should be based upon a set of general features that play a role in multimedia presentation. After reviewing many fields concerned with presentation design, Arens et al conclude that there are four classes of features:
- characteristics of the information to be conveyed;
- characteristics of the media at hand;
- the presenter's goals;
- the perceiver's interests and abilities.

A 'media allocation rule' refers to such features to determine appropriate media, for example,
```
If relation = elaboration or relation = contrast
   & type(contents(leaf1) = type(contents(leaf2))
   & medium(leaf1) = medium(leaf2)
Then create presentation-node with contents(leaf1,leaf2) and medium(leaf1)
```
i.e. merge elaboration or contrast relations of the same type into one presentation. The media allocation rules traverse the discourse structure tree bottom-up to generate the entire presentation. The system described (WIP) is said to have only twelve media allocation rules and the examples given do not seem to make much use of some of the features mentioned above.

Three of the four classes of feature (that is, all but the characteristics of the media) are the same as for natural language dialogue and we can anticipate that when attention is focussed on those features the same concerns as raised earlier in this chapter will re-emerge. For example, multimedia explanations should take account of the prior discourse and the student's knowledge and goals, by, for example, omitting or emphasising aspects of the explanation. Multimedia explanations do, however, raise issues which do not need to be considered with purely linguistic explanations, such as, the integration of components of the explanation presented via different media and the effective use of temporal aspects, as it is possible to coordinate or overlap different components and to use unreal timing effects.

As with linguistic explanations, we can expect an increased recognition that multimedia explanations are not objects to be created by the system and simply presented to the student. A successful explanation is likely to be interactive and coupled with on-going diagnostic processes. For example, PPP (André et al, 1993), an extension of WIP, mentioned above, allows the user to ask follow-on questions about the domain and the presentation and monitors the effectiveness of a presentation so that instructional strategies may be continuously adapted. However, such a point of view presents particular problems for multimedia explanations, because there is an inevitable asymmetry in the participants' contributions. Whereas students can be encouraged to provide an 'equal' linguistic input, it is not possible for students to create and input drawings and films on-line. Of course, certain kinds of non-linguistic input are possible, for example, pointing and rough sketches, but the effective incorporation of these in truly interactive multimedia interactions has yet to be developed.

The quality and effectiveness of students' interactions with computer-based learning systems has been improved by the new media and there appears to be great potential for further developments. These improvements have not been underpinned by the kinds of theoretical analysis which computational mathetics aims to provide for AI-ED systems. However, the new media have led to something of a reversion to previously discredited display modes of instruction. As the limitations of such methods are (re)discovered and there is more emphasis on the need for meaningful interactions, taking fuller account of the beliefs and goals of all participants, so we may anticipate an increased need for analyses in the style of computational mathetics.

10. Instruction

The first sentence of Gagné, Briggs and Wager (1992) defines instruction to be "a human undertaking whose purpose is to help people learn". Even with the limitation to humanity, this is a broad definition which surely avoids the negative connotations attributed to the activity of instruction, for example, in slogans such as "instructivism versus constructivism". Gagné et al stress that the term 'instruction' is not meant to imply a particularly limited kind of undertaking, such as teaching in a classroom in a didactic manner. For them, teaching is only one form of instruction. Instruction also includes the activity of structuring an environment (a book, a video, a computer simulation, and so on) and also activities internal to the learner (which they consider to be 'self-instruction'). Therefore, instruction, despite the term's pejorative overtones, must be considered within the scope of computational mathetics.

However, as we will see, there is not a lot left to say about instruction. One of the aims of computational mathetics is that instruction become a deductive consequence of formalisations of its constituent processes such as diagnosis, dialogue, interaction, and learning. Of course, we are far from being able to carry out this deduction fully analytically but we can at least sketch the objectives and put them within the context of the current status of instructional systems design.

To establish a foundation, here are a set of instructional design principles for optimal learning presented by Spector (1993) - originally written by Gagné - which are intended to crystallise the common conclusions of established theories of instruction (such as the Gagné-Briggs theory, algo-heuristic theory, structural learning theory, component display theory, elaboration theory, and motivational theory):
1. Different learning objectives require different instructional strategies.
2. There are five different types of learning objective, each with a distinctive instructional strategy:
    verbal knowledge - relate to known knowledge, use spaced review;
    concepts - provide definition, examples and non-examples;
    procedural rules - assure component skills are mastered before the total skill is tried;
    motor skills - practice with reinforcement;
    attitudes - demonstrate using human models.
3. Begin with an event that arouses and sustains learner interest.
4. Communicate clearly what the learner must learn to do.
5. Stimulate recollection of previously learned relevant knowledge.
6. Make the stimulus aspect of the task readily perceptible.
7. State rule-then-example or example-then-rule before learner performance.
8. Guide the learner through elaborations.
9. Verify initial learning by learner performance.
10. Provide varied practice with corrective feedback.
11. Communicate the relation between what is being learned and how it will be used.
12. Arrange occasions that require retrieval.

We may echo some of the comments made in section 3.5 with respect to principles proposed for AI-ED systems design. The principles listed above are not all of the same kind - some are statements of methodological preference (for example, the opinion that there are five types of learning objective) and some are prescriptions. The principles are quite vaguely stated, even allowing for the fact that they may have been watered down to make them acceptable to all instructional theorists. It is not clear how the principles interact. For example, when should one 'guide through elaborations' or 'provide varied practice'? The derivation of the principles from the instructional theories is only loosely argued.

However, criticism is not our purpose. The above list is intended to be illustrative of the results and methodologies of instructional theorists, who have made a concerted effort for some decades to establish useful instructional principles. It enables us to put the aims of computational mathetics in a realistic context.

10.1   Theories of instruction

Bruner (1966) attempted to establish the nature of a theory of instruction by contrasting it with a theory of learning. The latter was considered to be descriptive (concerned with describing learning events that are observed); the former was considered to be prescriptive (concerned with prescribing activities more likely to cause learning to occur). However, a theory of instruction is clearly derivative of a theory of learning, as it should not prescribe activities which a theory of learning indicates would be unproductive. Reigeluth (1993) comments that "learning theory can provide a basis for the (deductive) development of instructional theory, but instructional theory can just as readily be developed (inductively) through trial and error". At the moment, trial and error development is often necessary because learning theories are too vague or incomplete to permit many instructional design decisions to be made. In this section, however, we will be concerned with the possible deductive development of instructional theory, because, this, if it is successful, would provide analytical reasons for instructional prescriptions.

In some cases, a theory of learning seems so straightforward that instructional prescriptions seem to follow directly. For example, the concepts of behaviourism led directly to the design of programmed learning texts. For more complex theories, the instructional implications are not so easy to determine. For example, if we consider the two main candidates for a unified theory of cognition, ACT* and SOAR, it is not obvious that the twelve principles listed above follow from them. The theories are not comprehensive (for example, metacognitive mechanisms are not explained in detail) so that it is necessary to engage in some liberal interpretation or extrapolation of the theories. It is probably the case that all the principles could be argued to follow to some extent but that cognitive theorists would prefer to re-express the principles in ways more compatible with their own theoretical constructs. For example, if we compare Anderson's eight principles (section 3.5) we can see some superficial similarity but many differences of expression and emphasis.

At the moment, proponents of a particular theory of learning will work out its instructional implications and contrast them with those said to follow from other theories. It is quite difficult for a proponent of one particular theory to determine if a set of instructional principles does, in fact, follow from a different theory. It is also hard to show that a particular principle attributed to a theory does not follow from it. Therefore, it is relatively easy to build arguments by denigrating principles alleged to follow or not follow from other theories. For example, Sack, Soloway and Weingrad (1994) quote Hirsch (1988):

> "Good reading, like good chess, requires the rapid deployment of schemata that have already been acquired and do not have to be worked out on the spot. Good readers, like good chess players, quickly recognize typical patterns..."

and then proceed to refer to "the transmission paradigm implicit in Hirsch's proposal". It is not clear that any such paradigm is implicit in the quote. Many studies have shown that good chess players can indeed recognize typical patterns. One may dispute, as situationists do, that such recognition depends on the use of internal schemata, but even if one believes in their existence it does not follow at all that one is advocating teaching chess learners by 'transmitting' such schemata to them. If I believed that it were an objective fact that it is bad idea, in general, to have two pawns in the same column then it would not follow that I believed that I should try to transmit (that is, tell, presumably) this fact to a chess learner. Our hope is that the approach of computational mathetics will eventually lead to a tightening up of such argumentation.

If one has a theory of learning, then instructional prescriptions must be consistent with it but they cannot be derived from it alone. The derivation of specific instructional prescriptions or general instructional principles requires the specification of the following five components:
1. A theory of learning, as discussed above.
2. A theory of knowledge, if the learning theory refers to relationships between knowledge items.
3. The set of possible instructional actions (in terms of the possible interactions with the learner and the conventions of educational practice).
4. The relevant characteristics of the learner (in terms of prior knowledge and individual differences).
5. The instructional goals (in terms of the knowledge and attitudes to be acquired, usually taking account of the resources available).

We may consider these components in turn, with a simple illustration. A theory of learning is a set of statements describing how the cognitive state of a learner changes as a consequence of instructional actions (bearing in mind that we have a broad notion of what may constitute an instructional action). In general, then, it might be defined by a set of axioms of the form:

```
State(s,s1,t) & I-Action(ia,x) ∅ State(s,s2,ia(x,t))
```
i.e. if the student `s` is in cognitive state `s1` in situation `t` and `ia` is an instructional action with parameters `x` then she will be in state `s2` in the situation reached by applying `ia`, that is, `ia(x,t)`. We do not, at this stage, need to define what is meant by a 'cognitive state' but may assume that it is represented by ascriptions of the form we have described.

For example, an impasse-based theory of learning from examples might have the following axioms (simplifying, of course):

```
1.    State(s,[],t) & I-Action(present-positive,[p,f,a]) ∅
          State(s,B(s,f->a),present-positive([p,f,a],t))
```
i.e. if the student knows nothing then if we present a positive example `p` which has feature `f` and where action `a` was used to solve the problem, then the student will believe that any problem with feature `f` must be tackled by applying action `a`.

```
2.    State(s,[],t) & I-Action(present-positive,[p,[f,g],a]) ∅
          State(s,B(s,f&g->a),present-positive([p,[f,g],a],t))
```
i.e. similarly, if the positive example has two features `f` and `g` then the student will believe that action `a` is appropriate for problems with both features.

```
3.    State(s,s1,t) & I-Action(question,[q,fs]) & Impasse(s1,q,fs) ∅
          State(s,generalisation(s1),question([q,fs],t))
```
i.e. if the student is presented a question `q` with features `fs` which leads to an impasse then she generalises her cognitive state.

```
4.    State(s,s1,t) & I-Action(question,[q,fs]) & Solves(s1,q,fs) ∅
              State(s,s1,question([q,fs],t))
```
i.e. if the question is solved then the student's state is unchanged.

We also need a definition of the predicates `Impasse` and `Solves` and the function `generalisation`, which might include, for example, the axioms:
```
5.    Impasse(B(s,f&g->a),q,f)
```
i.e. if the student believes the rule `f&g->a` but the question has only feature `f` then she reaches an impasse.
```
6.    generalisation(B(s,f&g->a)) = B(s,f->a) v B(s,g->a)
```
i.e. by generalising, the student will believe that one or other feature alone is sufficient for the action.
```
7.    Solves(B(s,f->a),q,f)
```
i.e. if the student believes `f->a` and the question has only that feature then she will solve the problem. A set of such axioms and associated definitions constitutes, in our terms, a theory of learning.

A theory of knowledge would define all the relationships between knowledge items to which a theory of knowledge refers. For example, if a theory of learning has an axiom to say that an analogous, more simple concept should be introduced before the more complex concept, then a theory of knowledge would need to define what is meant by 'analogous'. In general, of course, this would involve formalisations used in AI knowledge representation, but for this illustration this component is not needed.

The third component involves the definition of the set of available instructional actions. A theory of learning defines the effects of general instructional actions, but does not say which actions are actually available. Let us imagine that we have two example problem solutions `p1` and `p2`, the first with feature `f1` and the second with features `f1` and `f2`, where action `a1` was applied. (To be concrete, `p1` could be 745-127 with `f1=borrow-next-left`, `p2` could be 85-27 with `f1=borrow-next-left` and `f2 = borrow-leftmost`, and `a1 = borrow`). If we assume the two problems could also be presented as questions, we have four possible instructional actions:
```
8.    I-Action(present-positive,[p1,f1,a1])
9.    I-Action(present-positive,[p2,[f1,f2],a1])
10.   I-Action(question,[q1,f1])
11.   I-Action(question,[q2,[f1,f2]])
```

Next is a description of the individual learner. In this case, we will assume no prior knowledge:
```
12.   State(s,[],t0)
```
and no individual differences to affect the learning theory posited above.

Finally, we must define the instructional goal or goals. In general, this could be expressed in terms of the desired contents of any component of our framework (beliefs, goals, attitudes, reasoners, monitors, etc.) or in terms of the properties of such components (for example, that they be comprehensive or consistent) without regard to any comparison to any desired content. Normally, the goal would also require that this state be reached in some optimal manner. Normally, also, we would not necessarily expect to reach the stated goal but would aim to minimise the distance from it. In our case, let us imagine that we would like the student to believe the rule `f1->a1` after only two instructional actions `ia1` and `ia2`, that is, we wish to determine an `ia1` and `ia2` with arguments `x1` and `x2` respectively such that:
```
        State(s,B(f1->a1),ia2(x2,ia1(x1,t0)))
```

In general, the number of instructional actions will not be specified, but the goal may be to minimise the number of them, or perhaps to minimise the 'cost' associated with those actions, for example, the total time required. As in AI in general, the difficulty of determining optimum multi-step solutions means that often the goal is expressed in terms of determining one step at a time, each step minimising the distance from some target. Usually, this will not lead to globally optimum solutions (and sometimes will lead to no solution at all, even though one may exist). This is considered further below.

Once the five components are defined, we may attempt to prove that the goal may be reached, such a proof yielding the requisite instructional actions. From the twelve premises above, we may derive an instance of the goal:
```
        State(s,B(f1->a1),question([q1,f1],present-positive([p1,f1,a],t0)))
```
i.e. if we present the positive example `p1` followed by question `q1` then the student will believe `f1->a1`. This, then, is the derived instructional prescription, since, as it happens, the only other conclusions from two-action sequences, according to the above premises, are:
```
        State(s,B(f1&f2->a1),question([q2,[f1,f2]],present-
            positive([p2,[f1,f2],a1],t0)))
        State(s,B(f1->a1)              v          B(f2->a1),question([q1,f1],present-
            positive([p2,[f1,f2],a1],t0)))
```
i.e. the student may have an over-special rule, or may have generalised as required or not.

This simple formulation re-expresses the argument described in section 7.6 for presenting three-column subtraction before two-column subtraction. The proof is general (in the sense that it could be applied to any similar problem-solving situation) and the conclusion could be presented as an 'instructional guideline'. In other words, not only may

```

we try to derive specific instructional actions but we may try to prove general instructional theorems. The derivation of such a proof is a general theorem-proving process and could, in principle, be carried out by automatic means. The general aim, then, is to express the theory of learning, theory of knowledge, instructional actions, learner attributes and instructional goals formally so that an instructional theory may be presented as a set of conclusions derived rigorously from assumptions, not as a set of vaguely expressed guidelines . The premises of the learning theory must, of course, be independently verified (if possible) by psychological experimentation.

The above illustration is much too simple to convince anyone that the methodology is feasible. Premises to describe the effect of some instructional actions are missing; the description of the learner and goals are too simple; we should represent some kind of probabilistic analysis, as the effects of instructional actions are not so deterministic; ideally, we should include considerations of many of the other aspects considered, for example, the use of monitors and reasoners (implicit above) during problem-solving, the effect of limited reasoning abilities, non-monotonicity, and so on.

However, simplified though it is, the specific conclusion drawn is one which is counter to established educational practice and the argument was originally presented as justification for developing a detailed learning theory to be implemented as a computer program so that its properties may be studied. When the premises become more realistically complex, derivations will require computational assistance and thus there is no fundamental difference between an axiomatic analysis such as the above and a computational simulation. The issue is which mode of description is most clear, concise, precise, convincing, and ultimately successful in leading to the specification of the instructional component of AI-ED systems design.

## 10.2    Instructional systems design

When educationalists talk of 'automating instruction' (few of them do, of course) they generally mean the designing of a structured environment to support learning by following a systematic development process, this process to be carried out by humans and to lead to a product delivered to students. The instructional systems design (ISD) process is typically considered to have five phases - analysis, design, development, implementation and maintenance - which is reminiscent of standard software engineering aiming to deliver products to clients (which includes those who commission the product and those who use it). Such an approach is now somewhat outmoded, as it does not take adequate account of the clients' involvement in the process and, as a result, does not enable adequate adaptation of the delivered product. Nonetheless, a brief review of ISD may help put our aims in perspective.

ISD has always been based on contemporary theories of the learning process, or rather has lagged behind them. The influence of cognitive psychology has almost obscured ISD's roots in behaviourism but the more recent social and situational views of learning have yet to change ISD. Traditional instructional design sees the learner as essentially inactive, receiving instruction.

ISD considers in detail four of the five components mentioned in the previous section - learning theories, knowledge theories (in the form of curricula), instructional goals (or objectives) and instructional actions. It rather ignores the fifth - the individual learner - for the reasons mentioned above. ISD does not say anything new about learning theories as such but aims to extract their implications for instruction. There is great scope for such discussion because the theories are so varied and their implications so imprecise. Gagné, Briggs and Wager (1992) attempt to provide a brief discussion of the relation between the basic processes of learning and instruction.

Assuming that the derivation of instructional implications is sound (which is debatable), the validity of ISD is dependent upon that of the psychological theory of learning on which it is based. If ISD is based upon a theory which posits the existence of memory schemata for motivation, say, then if that premise is shown to be invalid then the ISD process falls. In computational mathetics, the ascriptions we make to learners are not justified by psychology but by their utility to us as observers and designers. Confusion arises because it is assumed that, because we and learners are similar, any representation which we find useful will also be found useful by a learner, and therefore the learner must possess it as well, in some sense. In other contexts, such an assumption does not arise: if we observe the motion of planets, then we may ascribe a formula to explain and predict the motion but we would not assume that the planet possessed (that is, had an 'internal representation' of) that formula. If we wished to affect the behaviour of the planets (somehow), such ascriptions might be adequate. The way of deriving a theory of instruction (using the methods of computational mathetics) does not depend on psychological soundness. The results of such a derivation depend on psychological soundness only because the premises (of learning, etc.) do.

ISD has invested most effort on considering the nature of objectives, on the grounds that the best way to design instruction is to work back from the intended outcomes. This implies a commitment to a view that objectives can be divorced from the means of achieving them and that general principles (that is, principles that are not domain-based) can be applied to relate means to objectives. ISD aims to describe objectives so precisely that an independent person can tell whether it has been accomplished by observing what the learner does.

As objectives usually cannot be achieved in one step, it is necessary to specify intermediate objectives, leading to a focus on *task analysis*, which aims to identify which objectives are prerequisite to others. The standard representation is a 'learning hierarchy', which is a directed acyclic graph in which if node `i` points to node `j` then whatever `i` represents is prerequisite to whatever `j` represents. The definition of a prerequisite is part of a theory of knowledge. The role of a prerequisite is expressable as an axiom of a learning theory, perhaps:

```
State(s,K(s,i),t)   &   Prerequisite(i,j)   &   I-Action(ia,x)   ∅
    State(s,K(s,j),ia(x,t))
```

or maybe, as there is no presumption that one action will be sufficient:

```
State(s,K(s,i),t)      &      Prerequisite(i,j)      &      I-Actions(ias,x)      ∅
    State(s,K(s,j),ias(x,t))
```

where this axiom is a shorthand for a series similar to the one above. Maybe a prerequisite is not a definition of what may be learned but what may not be - if `i` is not known then `j` will never be:

```
State(s,~K(s,i),t1) & Prerequisite(i,j) & After(t2,t1) ∅ ~State(s,K(s,j),t2)
```

A prerequisite may be in terms of any component of our framework, not just knowledge. For example, we might require the student to be 'aware' or paying attention to learn anything:

```
State(s,~Aware(s),t1)      &      Prerequisite(Aware(s),j)      &      After(t2,t1)      ∅
    ~State(s,K(s,j),t2)
```

Presumably such an axiom justifies the third instructional principle ("Begin with an event that arouses and sustains learner interest") in the list at the beginning of this chapter.

ISD's analysis of instructional actions considers the kinds of events and how they relate to the various media. Gagné, Briggs and Wager (1992, p203) consider that "the information-processing (or cognitive) model of learning and memory" implies that there are nine kinds of instructional event:

1. gaining attention
2. informing the learner of the objective
3. stimulating recall of prerequisite learning
4. presenting the stimulus material
5. providing learning guidance
6. eliciting the performance
7. providing feedback about performance correctness
8. assessing the performance
9. enhancing retention and transfer

Unsurprisingly, these events map almost one-to-one onto the afore-mentioned list of instructional design principles, if we ignore the methodological preferences. So, the cognitive theory implies certain kinds of event and instructional theory just says that these events should be performed well.

Just as there are many varieties of learning theory, we can anticipate that there will be different forms of instructional theory. Halff (1993) distinguishes the forms of instruction according to the degree of interactivity:

- non-interactive instruction, e.g. uninterruptable lectures.
- self-paced instruction, e.g. reading a book.
- instruction with local feedback, where the situation automatically provides some feedback, e.g. practising skills such as swimming.
- instruction with context-free interactive control, e.g. branching in standard computer-based training.
- instruction with context-sensitive interactivity, e.g. computer simulations.
- conversational instruction with some form of planning and abstraction of the instructional context, e.g. tutorial discussions.

ISD has been mainly concerned with instruction at the top end of this list: AI-ED systems are towards the bottom of the list. ISD has concentrated on presentation methods: AI-ED complements this with more consideration of interaction, dialogue and diagnosis. These differences help explain some of the limitations of ISD for AI-ED purposes:

- ISD is more suitable for organising courses and curricula than for managing moment-to-moment interactions with an AI-ED system (or between human teachers and learners, for that matter).
- Task analysis emphasises the static compartmentalisation and fractionation of knowledge, whereas current conceptions see knowledge as dynamic, integrated and holistic.
- There is no direct link between the phases of the ISD process, so that, for example, task analysis yields only guidelines and not precise prescriptions.
- Because there is no formal derivation of instructional designs, there is no immediate way in which new understanding of learning can influence ISD. Therefore, ISD will lag behind theoretical conceptions and will play little role in their development.
- ISD is expensive, even with its limitations, and of little demonstrated effectiveness.

As a result, it seems reasonable to seek alternative methodologies.

10.3 Instructional planning

ISD provides a form of instructional planning but one not precise enough for computer implementation (although there are expert system tools to aid its practitioners). In this section we will consider the prospects for on-line planning in AI-ED systems. As the planning of a detailed course on complex topics is obviously a major exercise, we can anticipate (as for general planning in AI) the need for plan decomposition, leading to the idea of curricula and lessons, and the use of interaction and opportunism to modify on-going plan execution.

Many AI-ED systems have no need to form instructional plans at all. For example, simple simulations and learning environments need only to react to student inputs. Coaches and problem-solving monitors which are intended to respond to students' solutions may need to plan their response, for example, to present an explanation or example, but they may not engage in any extended dialogue with a student and they have no need to plan the content and delivery of instruction. If the student is expected to do more than solve problems, for example, to discuss with the system possible misconceptions, then dialogue management becomes necessary. This involves all the issues of dialogue and diagnosis discussed in earlier chapters and is where the bulk of AI-ED research has focussed.

Often dialogue is managed through a set of rules which can be thought of as pre-specified plan components to be assembled by the system. For example, GUIDON (Clancey, 1987) used twenty-six 'discourse procedures' defined in terms of some 200 teaching rules ('t-rules'), the sequencing of the procedures being controlled by a 'dialogue transition network'. For example, the discourse procedure for changing the current goal after a student request to do so is:

```
1. T-RULE 22.01
   If    Discussion of the new topic is complete
   Then  Say: donetopic
         Discuss the final value of the new topic and wrap up the discussion
   before
           returning to previous topics (Proc006)
         Say that the dialogue is returning to discussion of the goal currently
                          being discussed
         Exit this procedure
2. T-RULE 22.02
   If    The new topic is not a subgoal of (or the same as) the goal currently
   being                   discussed
   Then  Say: will-proceed
3. Discuss the goal with the student in a goal-directed mode (Proc001)
4. Say that the dialogue is returning to discussion of the goal currently being
      discussed.
```

The set of discourse procedures and t-rules is intended to be domain-independent and provides a very elaborate structure for dialogue control. The content of the procedures and rules was presumably developed after detailed study of tutor-student and GUIDON-student interactions. The content is said to be guided by the following set of principles, although there is, of course, no formal derivation and the principles themselves are only implicitly distributed over the discourse procedures and t-rules:

• Be perspicuous.
• Provide orientation to new tasks by top-down refinement.
• Strictly guide the dialogue.
• Account for behaviour in terms of missing expertise.
• Probe the student's understanding when you are not sure what he knows.
• Provide assistance by methodically introducing small steps.
• Examine the student's understanding and introduce new information.

Similarly, the MENO-TUTOR (Woolf, 1988) uses a 'discourse management network' of forty nodes organised as a tree of three levels - pedagogic states, strategic states and tactical states. Transitions through the tree can be pre-empted by 'meta-rules' which enable opportunistic transitions between any pair of nodes. For example, a meta-rule to 'move the tutor to begin a series of shallow questions about a variety of topics' is:

```
If the present topic is complete and the tutor has little confidence in its
   assessment of the student's knowledge
Then generate an expository shift from detailed examination of a single topic to
   a shallow examination of a variety of topics on the threshold of the student's
   knowledge.
```

In the terms of section 6.4, such an approach provides reactive planning, in that apparently planned behaviour is an emergent phenomenon arising from using compiled responses to the evolving situation. There is no deliberative planning in which explicit reasoning is carried out to determine actions to reach some goal state. Peachey and McCalla (1986) were the first to express the instructional planning task in standard AI planning terms. The goal is considered to be to reach some specified student description, using teaching operations defined in terms of preconditions and effects, selecting appropriate operators by a means-end strategy.

Unfortunately, the multitude of different kinds of knowledge which needs to be brought to bear leads to the adoption of powerful but undisciplined mechanisms such as blackboard systems and a plethora of semi-technical terminology which it is hard to translate from one system to another. In a blackboard system, knowledge is expressed as a set of independent modules any one of which may contribute at any time to a problem solution, and any conflict between

modules may be resolved by a meta-level, also organised as a blackboard (and so on). The precise properties of such democratic disorder are hard to determine but blackboards do, at least, provide a framework within which designers may try to incorporate relevant knowledge.

The system described by Murray (1990) uses thirteen blackboards, the four most relevant to planning being concerned with:
- The instructional plan - to deal with the tutor's goals, intended activities and intended procedures.
- Planner control - to monitor plan execution, diagnose plan problems, and edit plans.
- Knowledge sources - to define, for example, the plan executor, plan refinement operators, and the plan repairer.
- History - to record executed activities, student questions and requests, and assessments.

There are 43 knowledge sources (that is, independent modules which access the blackboards) to generate, monitor, diagnose and edit plans. Fernández-Castro, Verdejo and Díaz-Ilarraza (1993) use two blackboards, for the control and the domain. They describe their implementation by presenting a 'conceptual schemata network' which has 42 nodes, with labels such as 'pedagogic-decision', 'current-focus', 'control-ks', and so on. There is no formal way to compare such system designs. It is difficult to determine the behaviour of the systems from descriptions of their components and, as a result, to compare design proposals on the basis of the performance of implementations. Perhaps instructional planning is so inherently complex that relatively undisciplined methods are necessary.

10.3.1   Lessons

A 'lesson' is a social concept rather than a theoretical one, unless it is considered that it arose because of assumptions about the optimum attention or endurance span of learners. AI-ED systems generally do not have a notion of a lesson, that is, a limited period of instruction on a single topic. Instructional guidelines developed by studying teachers in classroom lessons have relevance to AI-ED only if they have lesson-independent merit or if students may assume they transfer to AI-ED systems.

Classroom interactions are structured by complex social conventions. What students learn is influenced by their reasoning backwards from such conventions. For example, VanLehn (1987) argues that lessons are assumed to encapsulate a single learning episode and that that assumption may help disambiguate potentially confusing instructional actions. To be specific, he suggests that the confusion of the two-column subtraction example (discussed in section 10.1) might be overcome by an assumption that only one new element is introduced at a time. So, the second axiom above might be re-expressed as:

```
2.    State(s,[],t) & I-Action(present-positive,[p,[f,g],a]) Ø
            State(s,B(s,f->a) v B(s,g->a),present-positive([p,[f,g],a],t))
```
i.e. the student believes that only one or other of the features is relevant to the action.

Although there may be only one thing to learn in a lesson, several instructional actions may be necessary to cause that to happen. Therefore, it is necessary to consider the planning of sequences of actions in lessons. The principles identified by Leinhardt and Ohlsson (1990), given in section 3.5, are relevant to this, and Gagné, Briggs and Wager (1992) suggest that a lesson should be based upon the sequence of nine instructional events they identify (given in section 10.2). AI-ED systems generally do not have explicit rules for planning lesson segments, such structures emerging from their dialogue management networks, blackboards, and so on.

10.3.2   Curricula

Educationalists have a strong proprietary interest in the subject of curricula and AI-ED system designers would perhaps be unwise to attempt to redefine or to over-simplify the concept. Nonetheless, if we regard a curriculum to be an organised sequence of lessons (which we have taken to be instructional sessions on a single topic), then the main new problem concerns how a sequence of topics may form a coherent curriculum, or, conversely, how a complex topic may be decomposed into a sequence of lesson-sized sub-topics.

In those terms, planning a curriculum is just an instance of the general planning problem, and, as Lesgold (1988) discusses, we can expect standard difficulties, such as the interdependence of subgoals and the simultaneous achievement of multiple objectives. A learning hierarchy (discussed in section 10.2) is a pre-specified plan for a curriculum, although it is under-specified because a hierarchy does not determine a unique sequence of lessons. A valid curriculum is one which is, in technical tems, a topological sort of the nodes of the hierarchy. Moreover, any particular target concept can be addressed by a number of learning hierarchies. For example, a curriculum on car maintenance might be organised by components (e.g. engine, clutch, exhaust, etc.), by mechanisms (e.g. electrical, mechanical, etc.), by symptoms (e.g. lights off, engine stalls, etc.), and so on. Planning a curriculum to satisfy multiple hierarchies, which might specify conflicting orderings, is not straightforward.

AI-ED systems tend not to plan curricula in detail for obvious practical reasons. If a curriculum is a sequence of lessons, which are sequences of instructional actions, then curriculum planning would involve the consideration of states reachable after such actions, i.e. the states s in expressions such as

```
State(s,S,ian(xn,..ia2(x2,ia1(x1,t0))..))
```

where n may be a very large number. Given the indeterminacy and cost of the learning and diagnostic processes and the large number of potential events, n is usually kept small (giving the opportunism often considered characteristic of instructional planning).

10.4   Modes of interaction

Instructional systems design separates the organisation of content from the consideration of the means of delivery to the student. This is a broad subject but we will restrict ourselves to considering possible arrangements between learners and teachers (or computer-based systems) from the point of view of instructional theory. In particular, we can distinguish individualised instruction (one learner, no teacher), tutoring (one learner, one teacher), and group instruction (several learners, one teacher).

10.4.1   Individualised instruction

Individualised instruction is an undertaking whose purpose is to help people learn on their own and therefore one which most teachers would consider desirable. Individualised instruction takes various forms, such as independent study plans, self-directed study, learner-centred programmes, self-pacing, student-determined instruction and computer-adaptive instruction (Gagné, Briggs and Wager, 1992). The last of these is, of course, of most relevance to us and fundamentally different to the others, because it is the only form of individualised instruction where the delivery system is not passive. With the other forms, "experience has indicated that satisfactory performance of individualized systems depends upon maintenance of appropriate procedures. It is owing to a lack of continuing support for these routines that individualized systems typically fail" (Gagné, Briggs and Wager, 1992). With computer-adaptive instruction, the system itself may be proactive in adapting itself to the needs of the learner, and that, obviously, is one of the main motivations for developing AI-ED systems. It is not clear that studies of non-computer-based individualised instruction provide many lessons for designers of AI-ED systems, except to warn that learning environments without "maintenance of appropriate procedures" typically fail, although as the AI-ED systems take more of an active role then studies of human tutoring may be informative, as we may now consider.

10.4.2   Tutoring

The result that students receiving one-to-one tutoring achieve two standard deviations more than students receiving conventional classroom instruction (Bloom, 1984) is often quoted (but never duplicated - researchers seem content to settle for this). It leads some to regard tutoring as the optimum teaching-learning arrangement and, as a result, to empirical studies to try to determine the nature of successful human tutoring. Such studies should inform the design of computer-based tutors.

However, Merrill, Reiser, Merrill and Landes (1993) conclude, after reviewing several such studies (Fox, 1991; Graesser, Person and Huber, 1993; Lepper and Chabay, 1988; Littman, Pinto and Soloway, 1990; McArthur, Stasz and Zmuidzinas, 1990), that "rather than a consistent view of tutorial strategies, [the] empirical studies of tutors have suggested a number of competing accounts of tutorial guidance". It seems that researchers identify different aspects of human tutoring as the key process. For example, Fox (1991) considers it most important that tutors offer frequent confirmatory feedback to keep students on productive paths, the absence or delay of such feedback being interpreted as a signal that an error has occurred. Lepper and Chabay (1988) concentrate on the motivational aspects, believing that the main goal of tutors is to keep students from becoming discouraged. So good tutors help students attribute failure to the nature of the problem, not their own inadequacy, and help them repair errors themselves by leading questions, not by direct telling. Littman, Pinto and Soloway (1990), however, consider that tutors structure the entire interaction around quite direct feedback after errors. McArthur, Stasz and Zmuidzinas (1990) describe their successful tutors as using 'microplans' (that is, particular sets of behaviour) in certain problem-solving situations. Yet again, Graesser, Person and Huber (1993) conclude that tutoring is successful because it gives students the opportunity to learn through asking their own questions. Even allowing for the academic need to find a different point of view, it is disconcerting that the nature of human tutoring should appear so elusive.

The methodology of such studies is typically to establish a set of categories of tutor and student action, to transcribe protocols to establish which sequences of actions occur, and to attempt to explain why observed sequences are successful. For example, Merrill, Reiser, Merrill and Landes (1993) define 36 categories, which are worth listing in full in order to indicate the kind of analysis attempted:

Student actions	Tutor actions
(Student constructs a problem solution)	(Tutor performs part of the problem-solving)
Student correction	Tutor example
Student elaboration	Tutor focus attention
Student example	Tutor read
Student focus attention	Tutor refer

Student indicate difficulty	Tutor type
Student indicate lack of understanding	(Tutor offers guidance for student's problem-solving)
Student read	Tutor confidence builder
Student refer	Tutor hint
Student set goal	Tutor indicate difficulty
Student type	Tutor set goal
(Student asks for help from tutor)	Tutor supportive statement
Assist plan assertion	(Tutor confirms a student step)
Assist plan question	Tutor confirmation
Assist understanding	Tutor elaboration
Student informational request	(Tutor gives error feedback after an incorrect step)
(Student understands tutor's utterances)	Tutor correction
Student confirmation	Tutor plan based feedback
(Student checks the current answer)	Tutor surface feature feedback
Student simulate process	(Tutor attempts to assess the student's understanding)
(Miscellaneous non-task-related utterances)	Tutor probe
Student comment	Tutor prompt
	(Tutor helps student check the correct answer)
	Tutor simulate process
	(Miscellaneous non-task-related utterances)
	Tutor comment

They then count how often one type of event is followed by another type of event, for example, a tutor confirmation or elaboration follows a student problem-solving action on 44% of occasions, or 66% of correct such actions. Tutors give error feedback after 53% of errors, usually as the next action. On the basis of a battery of such statistics, Merrill, Reiser, Merrill and Landes (1993) develop "a theory of tutorial guidance", the main principles of which are quite hard to extract from an entirely verbal description. It seems that "tutors assist students' problem-solving with careful guidance, in which the tutor keeps the student's problem-solving on track via ongoing confirmatory guidance and new goals to achieve after correct steps and error feedback after errors". The content and timing of error feedback depends upon an assessment of the costs of floundering and the potential benefits of self-repair.

In their study, McArthur, Stasz and Zmuidzinas (1990) categorise tutor actions into 44 types under eight headings: multipurpose (9 types), task management (11 types), performance assessment (4), knowledge assessment (3), remedial - simple techniques (7), remedial - complex techniques (8), local clarification (1), and motivation (1). It is reasonable to wonder how these various categorisations relate to one another. At this stage, the most useful conceptualisations to develop a theory of tutoring are not clear, but it seems rather redundant for each empirical study to begin by inventing a new categorisation. We might also wonder how these categories differ from, say, the 120 relations listed by Hovy (1993), discussed in section 9.1, for general discourse. Tutorial discourse is only a special case of general discourse, and it seems that more general techniques may be adapted.

As dialogue game theory shows, and as the references to microplans and tutoring schemata indicate, we need to consider sequences of interactions, rather than action-reaction pairs, in order to develop an informative theory of succesful tutoring. We need also to take account of the evolving context. A premise of cognitive apprenticeship is that interactions move through four stages - modelling, coaching, fading and reflecting - and clearly tutorial actions differ in these stages.

A computational mathetics view would be that tutorial actions would be determined by an analysis of the many relevant factors (ultimately - it is not possible to perform such an analysis now). If, for example, the system believes p and believes that the student believes f, an incorrect version of p, then the system might select one of a number of 'emendation procedures'. For example, McCoy (1989) suggests that misconceptions may be emended by a three-part system output: (i) a denial of f, (ii) an assertion of p, and (iii) a 'justification', often based on a refutation of the user's support for f. Various frames are specified for addressing certain kinds of misconception - for example, for a 'misattribution':

```
B(c,B(s,f: x has attribute y with value v)) &
B(c,p: x has attribute y with value w) &
∃z B(c,z has attribute y with value v) &
B(c,Similar(x,z))
    ∅  Deny(f) & Assert(p) & Comment("Have you confused x and z", etc)
```

The appropriate procedure will, in principle, depend upon many factors (in practice, for real-time interactions, the analysis might be quite shallow). It should take account of previous interactions, motivational factors, the student's and tutor's goals, the kind of misconception identified, the degree of confidence the system has in its diagnosis, and so on. The action decided upon may be simple (for example, to state the correct conception) or complex. For example, if the system considers that the student has developed the misconception through some impasse-repair mechanism then it

may decide to address this mechanism, rather than the specific misconception itself, that is, to point out that 'syntactic patches' to overcome local difficulties are not always productive, with the intention that this leads to longer-term learning benefits.

Complex though such descriptions will be, they will avoid over-simplistic interpretations of empirical data, for example, to conclude that remediation is no better than re-teaching, without qualification as to when these actions are carried out. Re-teaching is always a possible response when an error is detected. However, if a problem-solving process involves many steps and the system confidently identifies one step as in error, then remediation focussing on that one step seems likely to be more effective than re-teaching the whole procedure; if, on the other hand, the system believes that there are several error steps or that it cannot reliably attribute failure to only one step, then re-teaching may be a better option. As discussed in the previous chapter, tutorial dialogues are exceptionally subtle. Nonetheless, a theory of instructional communication should follow (as discussed in section 9.4) from theories of rational interaction and action.

The various so-called Socratic strategies should be followed by AI-ED systems not because they have been observed in teacher-student interactions but if an analysis predicts beneficial effects. For example, a counter-example (that is, a problem p such that, in a simple case, the predicted student's answer differs from the system's (correct) answer) might be presented if the predicted student's answer violates any beliefs ascribed to the student about answers in general (for example, a fraction problem for which the predicted answer is 0). Different instructional interactions may then ensue depending on whether or not the student realises that the example is in fact a counter-example. Among many more subtle remediations, we might imagine the use of 'garden-path problems', that is, problems which (according to the student model) the student can solve but only in such a tortuous fashion that she may realise that her current knowledge, although not actually incorrect, is inadequate. Methods for the automatic generation of such problems have yet to be developed. Formally, it would appear to be related to the conditions which promote reflection and to the results of any reflective process.

10.4.3   Group instruction

The gap between empirically-based analyses of tutoring and the kinds of descriptions we are seeking in computational mathetics is even more apparent when we consider the many forms of group instruction, which (from our definition of instruction) includes teacher-led and non-teacher-led activities. There have, of course, been very many studies of group-based classroom activities but it is hard to pinpoint their relevance for AI-ED system design. If we consider the nine kinds of instructional event identified by Gagné, Briggs and Wager (1992), given in section 10.2, then it is to be anticipated that most if not all of them are harder to perform satisfactorily with a large group of students than with a single student. For example, stimulating recall of relevant previous knowledge becomes difficult if the members of the group have different previous experiences, and providing learning guidance and assessing performance becomes much harder to manage as learners take different paths.

All in all, one might conclude that group instruction is an inferior form of instruction made necessary only by the constraints of educational budgets and therefore one which AI-ED systems need not aim to support. However, as we have seen, there has been increased enthusiasm recently for systems embedding instruction in a group setting. The reasons given are the following (Brown and Palinscar, 1989; Resnick, 1989):
- Cooperative learning provides a supportive means of promoting the use of self-regulatory strategies, with reduced anxiety and increased motivation.
- More complex problems can be tackled in a collaborative group than an individual could accomplish.
- The collaborative group provides support for learners at different levels of expertise, enabling them to use skills that are just emerging.
- Conflict between opposing viewpoints plays an important and positive role in cooperative instructional settings as a means of building and elaborating knowledge.
- Cooperative learning provides opportunities to experience knowledge in use, in contexts where the meaningfulness of individual elements of an activity to the whole is apparent to the learner.

It is for educational research to confirm these hypotheses, or rather to work out more precise descriptions of when they apply. Computational mathetics aims to clarify some of the concepts (such as cooperation, collaboration, self-regulatory strategies, motivation, conflict, and viewpoints) and mechanisms, and although we have described some work in this direction there is obviously much more to do.

10.5   Evaluation

Discussions of instructional design invariably end with a consideration of the problem of evaluation. The computational mathetics view of evaluation is rather radical - it aims to eliminate it. As evaluation processes are so deeply ingrained in any educational enterprise, this startling aim needs some clarification.

There are three different kinds of evaluation involved: of the learner or group of learners, of the AI-ED system, and of the AI-ED system design process. At the moment, these three kinds of evaluation are conflated. It seems that the only

way to evaluate AI-ED systems (the process of designing them and the products themselves) is to see if they deliver what they promise, that is, that students learn more, faster, better, and so on. Of course, students will always need to be evaluated for other purposes but their evaluation is not necessarily a reliable guide to the qualities of an AI-ED system. The evaluation of an AI-ED system design process (such as one based on a computational mathetics type of analysis) must be made by comparing it with alternatives.

Imagine the following situation: we have a comprehensive, reliable theory of learning, instruction, diagnosis, dialogue, motivation, metacognition, and everything else needed to design an AI-ED system (it is quite a feat of imagination). Those theories lead inexorably, analytically to the design of a specific AI-ED system which, according to the theories, is optimum for that learning problem. Imagine, now, that those theories actually predict that learners will only learn 2% more (or even 2% less) than by some other method, such as reading a book. If, nonetheless, the system is implemented, it will, because our perfect theories predict so, yield only a 2% change in learner performance. However, the AI-ED system design process is flawless: it has produced the optimum design for the circumstances.

The converse situation may more easily be imagined: we have scanty, unreliable theories, which we interpret loosely to design an AI-ED system, which, lo-and-behold, produces significant learning benefits (or not, as the case may be), although we are not sure why. In this situation, the design process is poor, despite the positive result.

The aim of computational mathetics is to move towards a situation where we have reliable, precise theories from which systems may be formally derived which will produce the learning benefits predicted by the theories. If the theories are sufficiently reliable, then, by definition, we will not need to evaluate student performance.

References

Aiello, L.C., Cialdea, M. and Nardi, D. (1993). Reasoning about student knowledge and reasoning, *Journal of Artificial Intelligence in Education*, 5, 199-254.

Aiello, L. and Micarelli, A. (1990). SEDAF: an intelligent educational system for mathematics, *Applied Artificial Intelligence*, 4, 15-37.

Allen, J.F. (1984). Towards a general theory of action and time, *Artificial Intelligence*, 23, 123-154.

Allen, J.F. (1995). *Natural Language Understanding*, Redwood City: Benjamin/Cummings.

Anderson, A. and Belnap, N. (1975). *Entailment: the Logic of Relevance and Necessity*, Princeton: Princeton University Press.

Anderson, J.R. (1983). *The Architecture of Cognition*, Cambridge, Mass.: Harvard University Press.

Anderson, J.R. (1993). *Rules of the Mind*, Hillsdale, N.J.: Erlbaum.

Anderson, J.R., Boyle, C.F., Corbett, A.T. and Lewis, M.W. (1990). Cognitive modelling and intelligent tutoring, *Artificial Intelligence*, 42, 7-49.

Anderson, J.R., Boyle, C.F., Farrell, R. and Reiser, B.J. (1989). Cognitive principles in the design of computer tutors, in P. Morris (ed.), *Modelling Cognition*, New York: Wiley.

Anderson, J.R., Boyle, C.F. and Yost, G. (1985). The geometry tutor, *Proc. of the Int. Joint Conf. on Artificial Intelligence*, Los Angeles.

Anderson, J.R., Conrad, F.G. and Corbett, A.T. (1989). Skill acquisition and the Lisp tutor. *Cognitive Science*, 13, 467-505.

Anderson, J.R. and Reiser, B.J. (1985). The Lisp tutor, *Byte*, 10, 4, 159-175.

André, E., Graf, W., Heinsohn, J., Nebel, B., Profitlich, H.-J., Rist, T. and Wahlster, W. (1993). PPP - Personalized Plan-based Presenter, DFKI Report D-93-5, Saarbrucken, Germany.

Arens, Y., Hovy, E. and van Mulken, S. (1993). Structure and rules in automated multimedia presentation planning, *Proc. of Int. Joint Conf. on Artificial Intelligence*, Chambery.

Baker, M.J. (1994). A model for negotiation in teaching-learning dialogues, *Journal of Artificial Intelligence in Education*, 5, 199-254.

Bann, S., ed. (1974). *The Tradition of Constructivism*, New York: Viking.

Barrett, A. and Weld, D.S. (1994). Partial-order planning, *Artificial Intelligence*, 67, 71-112.

Barwise, J. and Perry, J. (1983). *Situations and Attitudes*, Cambridge, MA: MIT Press.

Bauer, M., Biundo, S., Dengler, D., Koehler, J. and Paul, G. (1993). PHI - a logic-based tool for intelligent help systems, *Proc. of the Int. Joint Conference on Artificial Intelligence*, Chambery.

Bell, B.L. and Bareiss, R. (1993). Sickle Cell Counselor: using a goal-based scenario to motivate exporation of knowledge in a museum context, *Proc. of the World Conference on Artficial Intelligence in Education*, Edinburgh: AACE.

Beller, S. and Hoppe, H.U. (1993). Deductive error reconstruction and classification in a logic programming framework, *Proc. of the World Conference on Artficial Intelligence in Education*, Edinburgh: AACE.

Bereiter, C. and Scardamalia, M. (1989). Intentional learning as a goal of instruction, in L.B. Resnick (ed.), *Knowing, Learning and Instruction*, Hillsdale, N.J.: Lawrence Erlbaum.

Bergadano, F. and Gunetti, D. (1993). An interactive system to learn functional logic programs, *Proceedings of the International Joint Conference on Artificial Intelligence*, Chambery.

Bickhard, M.H. and Terveen, L. (1995). *Foundational Issues in Artificial Intelligence and Cognitive Science*, Amsterdam: Elsevier.

Bielaczyc, K., Pirolli, P. and Brown, A.L. (1993). Strategy training in self-explanations and self-regulation strategies for learning computer programming, Technical Report CSM-5, UC Berkeley.

Bierman, D., Breuker, J. and Sandberg, J., eds. (1989). *Artificial Intelligence and Education: Synthesis and Reflection*, Springfield, VA: IOS.

Birnbaum, L., ed. (1991a). *Proceedings of the Conference on the Learning Sciences*, Charlottesville: AACE.

Birnbaum, L. (1991b). Rigor mortis, *Artificial Intelligence*, 47, 57-77.

Bhuiyan, S.H. (1992). Supporting students in the use of mental models of recursion, ARIES Technical Report 93-4, Dept. of Computational Science, University of Saskatchewan,

Bloch, G. and Farrell, R. (1988). Promoting creativity through argumentation, *Proc. of Intelligent Tutoring Systems 88*, Montreal.

Bloom, B.S. (1984). The 2 sigma problem: the search for methods of group instruction as effective as one-to-one tutoring, *Educational Researcher*, 13, 4-16.

Blumenthal, B. and Porter, B.W. (1994). Analysis and empirical studies of derivational analogy, *Artificial Intelligence*, 67, 287-327.

Bonar, J. and Cunningham, R. (1988). BRIDGE: an intelligent tutor for thinking about programming, in J. Self (ed.), *Artificial Intelligence and Human Learning*, London: Chapman and Hall.

Bond, A.H. and Gasser, L. (1988). *Readings in Distributed Artificial Intelligence*, San Mateo: Morgan Kaufmann.

Bos, E. and van de Plassche, J. (1994). A knowledge-based English verb form tutor, *Journal of Artificial Intelligence in Education*, 5, 107-129.

Brazdil, P.B. (1992). Integration of knowledge in multi-agent environments, in E. Costa (ed.), *New Directions for Intelligent Tutoring Systems*, Berlin: Springer-Verlag.

Bredeweg, B. and Winkels, R. (1994). Student modelling through qualitative reasoning, in J.E. Greer and G.I. McCalla (eds.), *Student Modelling: the Key to Individualized Knowledge-Based Instruction*, Berlin: Springer-Verlag.

Breuker, J. and Wielinga, B.J. (1989). Model driven knowledge acquisition, in P. Guida and C. Tasso (eds.), *Topics in the Design of Expert Systems*, Amsterdam: North-Holland.

Brna, P., Ohlsson, S. and Pain, H., eds. (1993). *Proceedings of AI-ED 93*, Charlottesville: AACE.

Brooks, R.A. (1991). Intelligence without representation, *Artificial Intelligence*, 47, 139-159.

Brown, A. (1987). Metacognition, executive control, self-regulation and other more mysterious mechanisms, in F.E. Weinert and R.H. Kluwe (eds.), *Metacognition, Motivation and Understanding*, Hillsdale, N.J.: Lawrence Erlbaum.

Brown, A.L. and Palinscar, A.S. (1989). Guided, cooperative learning and individual knowledge acquisition, in L.B. Resnick (ed.), *Knowing, Learning, and Instruction*, Hillsdale, NJ: Erlbaum.

Brown, J.S. (1990). Towards a new epistemology for learning, in C. Frasson and G. Gauthier (eds.), *Intelligent Tutoring Systems: at the Crossroads of Artificial Intelligence and Education*, Norwood, N.J.: Ablex.

Brown, J.S., Burton, R.R. and de Kleer, J. (1982). Pedagogical, natural language, and knowledge engineering techniques in SOPHIE I, II and III, in D.H. Sleeman and J.S. Brown (eds.), *Intelligent Tutoring Systems*, London: Academic Press.

Brown, J.S., Collins, A. and Duguid, P. (1989). Debating the situation, *Educational Researcher*, 18, 4, 10-12.

Brown, J.S. and VanLehn, K. (1980). Repair theory: a generative theory of bugs in procedural skills, *Cognitive Science*, 4, 379-426.

Bruner, J.S. (1961). The act of discovery ???

Bruner, J.S. (1966). *Toward a theory of instruction*, New York: W.W. Norton.

Burton, R.R. and Brown, J.S. (1979). An investigation of computer coaching for informal learning activities, *Int. Journal of Man-Machine Studies*, 11, 5-24.

Calistri-Yeh, R.J. (1991). Utilizing user models to handle ambiguity and misconceptions in robust plan recognition, *User Modeling and User-Adapted Interaction*, 1, 289-322.

Caravita, S. and Hallden, O. (1994). Re-framing the problem of conceptual change, *Learning and Instruction*, 4, 89-111.

Carberry, S. (1990). Incorporating default inferences into plan recognition, *Proc. AAAI-90*, Boston.

Carbonell, J.G. (1986). Derivational analogy: a theory of reconstructive problem solving and expertise acquisition, in R.S. Michalski, J.G. Carbonell and T.M. Mitchell (eds.), *Machine Learning: An AI Approach, Vol. 2*, Los Altos: Morgan Kaufmann.

Carbonell, J.R. (1970). AI in CAI: an artificial intelligence approach to computer-assisted instruction, *IEEE Trans. on Man-Machine Systems*, 11, 190-202.

Carlson, L. (1983). *Dialogue Games: An Approach to Discourse Analysis*, Dordrecht: Reidel.

Cawsey, A. (1993). *Explanation and Interaction: The Computer Generation of Explanatory Dialogues*, Cambridge, MA.: MIT Press.

Charniak, E. (1991). Bayesian networks without tears, *AI Magazine*, 12, 4, 50-63.

Charniak, E. and Goldman, R.P. (1993). A Bayesian model of plan recognition, *Artificial Intelligence*, 64, 53-79.

Chi, M.T.H., Bassok, M., Lewis, M.W., Reimann, P. and Glaser, R. (1989). Self-explanations: how students study and use examples in learning to solve problems, *Cognitive Science*, 13, 145-182.

Chi, M., Glaser, R. and Rees, E. (1982). Expertise in problem solving, in R. Sternberg (ed.), *Advances in the Psychology of Human Intelligence*, Hillsdale, N.J.: Lawrence Erlbaum.

Chinn, C.A. and Brewer, W.F. (1993). The role of anomalous data in knowledge acquisition: a theoretical framework and implications for science instruction, *Review of Educational Research*, 63, 1-49.

Cialdea, M. (1992). Meta-reasoning and student modelling, in E. Costa (ed.), *New Directions for Intelligent Tutoring Systems*, Berlin: Springer-Verlag.

Clancey, W.J. (1979). Transfer of rule-based expertise through a tutorial dialogue, PhD thesis, Stanford University.

Clancey, W.J. (1984). Methodology for building an intelligent tutoring system, in W. Kintsch, P.G. Polson and J.R. Miller (eds.), *Methods and Tactics in Cognitive Science*, Hillsdale: Erlbaum.

Clancey, W.J. (1987). *Knowledge-Based Tutoring: the GUIDON program.* Cambridge, Mass.: MIT Press.

Clancey, W.J. (1992a). Representations of knowing - in defense of cognitive apprenticeship, *Journal of Artificial Intelligence in Education*, 3, 139-168.

Clancey, W.J. (1992b). New perspectives on cognition and instructional technology, in E. Costa (ed.), *New Directions for Intelligent Tutoring Systems*, Berlin: Springer-Verlag.

Clancey, W.J. (1993). GUIDON-MANAGE revisited: a socio-technical systems approach, *Journal of Artificial Intelligence in Education*, 4, 5-34.

Clancey, W.J. (1995). A tutorial on situated learning, in T.-W. Chan and J.A. Self (eds.), *Emerging Computer Technologies in Education*, Charlottesville: AACE.

Clocksin, W.F. and Mellish, C.S. (1981). *Programming in Prolog*, Berlin: Springer-Verlag.

Cohen, D.K. (1989). Teaching practice: plus ça change, in P. Jackson (ed.), *Contributing to Educational Change: Perspectives on Research and Practice*, Berkeley, CA.: McCutchan.

Cohen, L.J. (1992). *An Essay on Belief and Acceptance*, Oxford: Clarendon Press.

Cohen, P.R. and Levesque, H.J. (1990a). Intention is choice with commitment, *Artificial Intelligence*, 42, 213-261.

Cohen, P.R. and Levesque, H.J. (1990b). Rational interaction as the basis for communication, in P.R. Cohen, J. Morgan and M.E. Pollack (eds.), *Intentions in Communication*, Cambridge, MA.: MIT Press.

Cohen, R., Schmidt, K. and van Beek, P. (1994). A framework for soliciting clarification from users during plan recognition, *Proc. of the 4th Int. Conf. on User Modeling*, Cape Cod.

Collins, A. (1988). Cognitive apprenticeship and instructional technology, in B.F.Jones and L. Idol (eds.), *Dimensions of Thinking and Cognitive Instruction*, Hillsdale, NJ: Lawrence Erlbaum.

Collins, A. and Brown, J.S. (1988). The computer as a tool for learning through reflection, in H. Mandl and A. Lesgold (eds.), *Learning Issues for Intelligent Tutoring Systems*, New York: Springer-Verlag.

Collins, A., Brown, J.S. and Newman, S. (1989). Cognitive apprenticeship: teaching the crafts of reading, writing and mathematics, in L.B. Resnick (ed.), *Knowing, Learning and Instruction*, Hillsdale, N.J.: Lawrence Erlbaum.

Collins, A. and Stevens, A.L. (1982). Goals and strategies for inquiry teachers, in R. Glaser (ed.), *Advances in Instructional Psychology II*, Hillsdale, N.J.: Erlbaum.

Corno, L. and Snow, R. (1986). Adapting teaching to individual differences among learners, in M. Wittrock (ed.), *Handbook of Research on Teaching*, New York: Macmillan.

Costa, E., ed. (1992). *New Directions in Intelligent Tutoring Systems*, Berlin: Springer-Verlag.

Costa, E., Duchènoy, S. and Kodratoff, Y. (1988). A resolution based method for discovering students' misconceptions, in J.A. Self (ed.), *Artificial Intelligence and Human Learning,* London: Chapman and Hall.

Cox, R. and Brna, P. (1995). Supporting the use of external representations in problem solving: the need for flexible learning environments, to appear in *Journal of Artficial Intelligence in Education*.

Crews, T. and Biswas, G. (1993). A tutor for trip planning: combining planning and mathematics problem solving, *Proc. of the World Conference on Artficial Intelligence in Education*, Edinburgh: AACE.

Crystal, D. (1987). *The Cambridge Encyclopedia of Language*, London: Guild Publishing.

Davis, E. (1990). *Representations of Commonsense Knowledge*, Palo Alto: Morgan Kaufmann.

Davis, R. (1980). Meta-rules: reasoning about control, *Artificial Intelligence*, 15, 179-222.

Davis, R. and Smith, R.G. (1983). Negotiation as a metaphor for distributed problem solving, *Artificial Intelligence*, 20, 63-109.

de Corte, E., Linn, M., Mandl, H. and Verschaffel, L., eds. (1991). *Computer-Based Learning Environments and Problem-Solving*, New York: Springer-Verlag.

Dejong, G. and Mooney, R. (1986). Explanation-based learning: an alternative view, *Machine Learning*, 1, 145-176.

de Jong, T. and Ferguson-Hessler, M.G.M. (1986). Cognitive structures of good and poor novice problem solvers in physics, *Journal of Educational Psychology*, 78, 279-288.

de Kleer, J. (1986). An assumption-based truth maintenance system, *Artificial Intelligence*, 28, 127-162.

de Kleer, J. and Brown, J.S. (1981). Mental models of physical systems and their acquisition, in J.R. Anderson (ed.), *Cognitive Skills and their Acquisition*, Hillsdale NJ: Erlbaum.

de Kleer, J. and Brown, J.S. (1984). A physics based on confluences, *Artificial Intelligence*, 24, 7-83.

de Kleer, J., Mackworth, A.K. and Reiter, R. (1992). Characterizing diagnoses and systems, *Artificial Intelligence*, 56, 197-222.

de Kleer, J. and Williams, B.C. (1989). Diagnosing with behavioral modes, *Proceedings of the International Joint Conference on Artificial Intelligence*, Detroit.

Del Soldata, T. and du Boulay, B. (1995). Motivational tactics in tutoring systems, to appear in *Journal of Artificial Intelligence in Education*.

De Raedt, L., Lavrac, N. and Dzeroski, S. (1993). Multiple predicate learning, *Proceedings of the International Joint Conference on Artificial Intelligence*, Chambery.

Derry, S. and Hawkes, L.W. (1993a). Local cognitive modeling of problem-solving behavior: an application of fuzzy theory, in S. Lajoie and S. Derry (eds.), *Computers as Cognitive Tools*, Hillsdale, N.J.: Erlbaum.

Detterman, D.K. (1993). The case for the prosecution: transfer as an epiphenomenon, in D.K. Detterman and R.J. Sternberg (eds.), *Transfer on Trial: Intelligence, Cognition, and Instruction*, Norwood, NJ: Ablex.

Detterman, D.K. and Sternberg, R.J., eds. (1993). *Transfer on Trial: Intelligence, Cognition, and Instruction*, Norwood, NJ: Ablex.

Dewey, J. (1938). *Experience and Education*, New York: Collier.

Dillenbourg, P. and Self, J.A. (1992). A computational approach to socially distributed cognition, *European Journal of Psychology and Education*, 7, 353-372.

Di Sessa, A.A. (1987). The third revolution in computers and education, *Journal of Research in Science Education*, 24, 343-367.

Donini, F.M., Lenzerini, M., Nardi, D., Pirri, F. and Schaerf, M. (1990). Non-monotonic reasoning, *Artificial Intelligence Review*, 4, 163-210.

Duncan, P.C. (1992). The Space Shuttle Fuel Cell tutor: a simulation-based intelligent tutoring system with yoked expert systems, *Journal of Artificial Intelligence in Education*, 3, 297-313.

Durfee, H.H. and Lesser, V.R. (1989). Negotiating task decomposition and allocation using partial global planning, in L. Gasser and M.N. Huhns (eds.), *Distributed Artificial Intelligence II*, San Mateo: Morgan Kaufmann.

Elliott, C. (1993). Using the Affective Reasoner to support social simulations, *Proc. of the Int. Joint Conf. on Artificial Intelligence*, Chambery.

Eller, R. and Carberry, S. (1992). A meta-rule approach to flexible plan recognition in dialogue, *User Modeling and User-Adapted Interaction*, 2, 27-53.

Elsom-Cook, M., ed. (1990). *Guided Discovery Tutoring Systems*, London: Paul Chapman.

Elzer, S., Chu-Carroll, J. and Carberry, S. (1994). Recognizing and utilizing user preferences in collaborative consultation dialogues, *Proc. of the 4th Int. Conf. on User Modeling*, Cape Cod.

Evertsz, R. (1989). Refining the student's procedural knowledge through abstract interpretations, in D. Bierman, J. Breuker and J. Sandberg (eds.)*, Artificial Intelligence and Education,* Amsterdam: IOS.

Fagin, R. and Halpern, J.Y. (1987). Belief, awareness, and limited reasoning, *Artificial Intelligence*, 34, 39-76.

Farr, M.J. and Psotka, J., eds. (1992). *Intelligent Instruction by Computer*, Washington, DC: Taylor and Francis.

Fernández-Castro, I., Verdejo, F. and Díaz-Ilarraza, A. (1993). Architectural and planning issues in intelligent tutoring systems, *Journal of Artificial Intelligence in Education*, 4, 357-395.

Flavell, J.H. (1976). Metacognitive aspects of problem solving, in L. Resnick (ed.), *The Nature of Intelligence*, Hillsdale, NJ: Erlbaum.

Forbus, K. (1984). Qualitative process theory, *Artificial Intelligence*, 24, 85-168.

Forbus, K. (1991). Towards tutor compilers: self-explanatory simulations as an enabling technology, in L. Birnbaum (ed.), *The Int. Conf. on the Learning Sciences*, Evanston.

Ford, K. and Hayes, P, eds. (1991). *Reasoning Agents in a Dynamic World: The Frame Problem*, Greenwich: JAI Press.

Foss, C.L. (1987). Learning from errors in Algebraland, Technical Report IRL-87-003, Institute for Research on Learning, Palo Alto.

Fox, B.A. (1991). Cognitive and interactional aspects of correction in tutoring, in P. Goodyear (ed.), *Teaching Knowledge and Intelligent Tutoring*, Hillsdale, NJ: Ablex.

Frasson, C. and Gauthier, G., eds. (1990). *Intelligent Tutoring Systems: at the Crossroads of Artificial Intelligence and Education*, Norwood, N.J.: Ablex.

Friedrich, G. (1993). Model-based diagnosis and repair, *AI Communications*, 6, 187-206.

Gagné, R.M., Briggs, L.J. and Wager, W.W. (1992). *Principles of Instructional Design*, Fort Worth: Harcourt Brace Jovanovich.

Gan, A. (1922). *Konstruktivisma.* Kalinin: Tver.

Gardenförs, P. (1988). *Knowledge in Flux*, Cambridge, Mass.: MIT Press.

Gazdar, G. and Mellish, C. (1989). *Natural Language Processing in Lisp: an Introduction to Computational Linguistics*, Reading, Mass.: Addison-Wesley.

Genesereth, M.R. and Ketchpel, S.P. (1994). Software agents, *Communications of the ACM*, 37, 48-53.

Genesereth, M.R. and Nilsson, N.J. (1987). *Logical Foundations of Artificial Intelligence*, Los Altos: Morgan Kaufmann.

Giangrandi, P. and Tasso, C. (1995). Truth maintenance techniques for modelling students' behaviour, to appear in *Journal of Artificial Intelligence in Education.*

Ginsburg, M.L. (1993). *Essentials of Artificial Intelligence*, San Mateo: Morgan Kaufmann.

Goldman, S.A. and Sloan, R.H. (1994). The power of self-directed learning, *Machine Learning*, 14, 271-294.

Goldstein, I.P. (1979). The genetic graph: a representation for the evolution of procedural knowledge, *Int. J. of Man-Machine Studies*, 11, 51-77.

Goodyear, P. (1991). *Teaching Knowledge and Intelligent Tutoring*, Norwood, NJ: Ablex.

Graesser, A.C., Person, N.K. and Huber, J.D. (1993). Question asking during tutoring and in the design of educational software, in M. Rabinowitz (ed.), *Cognition, Instruction, and Educational Assessment*, Hillsdale, NJ: Erlbaum.

Greeno, J.G. (1989). Situations, mental models and generative knowledge, in D. Klahr and K. Kotovsky (eds.), *Complex Information Processing*, Hillsdale, NJ: Erlbaum.

Greeno, J.G., Smith, D.R. and Moore, J.L. (1993). Transfer of situated learning, in D.K. Detterman and R.J. Sternberg (eds.), *Transfer on Trial: Intelligence, Cognition, and Instruction*, Norwood, NJ: Ablex.

Greer, J.E. and McCalla, G.I., eds. (1994). *Student Modelling: The Key to Individualised Knowledge-Based Instruction*, Berlin: Springer.

Grice, H.P. (1975). Logic and conversation, in P. Cole and J.L. Morgan (eds.), *Speech Acts*, New York: Academic Press.

Grosz, B. (1977). The representation and use of focus in dialogue understanding, Stanford Reasearch Institute Technical Notes 5.

Grosz, B. and Kraus, S. (1993). Collaborative plans for group activities, *Proc. of Int. Joint Conference on Artificial Intelligence*, Chambery.

Halff, H.M. (1993). Prospects for automating instructional design, in J.M. Spector, M.C. Polson and D.J. Muriada (eds.), *Automating Instructional Design*, Englewood Cliffs, NJ: Educational Technology Publications.

Halpern, J.Y. and Moses, Y. (1992). A guide to completeness and complexity of modal logics of knowledge and belief, *Artificial Intelligence*, 54, 319-379.

Hammond, K.J. (1990). Explaining and repairing plans that fail, *Artificial Intelligence*, 45, 173-228.

Hanks, W.F. (1991). Preface to Lave, J. and Wenger, E., *Situated Learning: Legitimate Peripheral Participation*, Cambridge: Cambridge Univ. Press.

Harel, D. (1979). *First-Order Dynamic Logic.* New York: Springer.

Hawkes, L.W. and Derry, S.J. (1990). Error diagnosis and fuzzy reasoning techniques for intelligent tutoring systems, *Journal of Artificial Intelligence in Education*, 1, 43-56.

Hayes, P.J., Ford, K.M. and Agnew, N. (1994). On babies and bathwater, *AI Magazine*, 15, 4, 15-26.

Hays, D.G. (1967). *Introduction to Computational Linguistics*, New York: Elsevier.

Hewitt, C. (1986). Offices are open systems, *ACM Transactions on Office Information Systems*, 4, 271-286.

Hill, R.W. and Johnson, L.J. (1995). Situated plan attribution, to appear in *Journal of Artificial Intelligence in Education.*

HIntikka, J. (1962). *Knowledge and Belief*, Ithaca: Cornell University Press.

Hirsch, E.D. (1988). *Cultural Literacy*, New York: Vintage.

Hofstadter, R. (1962). *Anti-intellectualism in American life*, New York: Vintage.

Holland, J.H., Holyoak, K.J., Nisbett, R.E. and Thagard, P.R. (1986). *Induction: Processes of Inference, Learning and Discovery*, Cambridge, Mass.: MIT Press.

Hoppe, H.U. (1993). Cognitive apprenticeship - the emperor's new method?, *Journal of Artificial Intelligence in Education*, 4, 49-54.

Hoppe, H.U. (1994). Deductive error diagnosis and inductive error generalization, *Journal of Artificial Intelligence in Education*, 5, 27-49.

Hovy, E.H. (1993). Automated discourse generation using discourse structure relations, *Artificial Intelligence*, 63, 34-385.

Huang, X. (1994). Modelling a student's inconsistent beliefs and awareness, in J.E. Greer and G.I. McCalla (eds.), *Student Modelling: the Key to Individualized Knowledge-Based Instruction*, Berlin: Springer-Verlag.

Huang, X., McCalla, G.I., Greer, J.E. and Neufeld, E. (1991). Revising deductive knowledge and stereotypical knowledge in a student model, *User Modeling and User-Adapted Interaction,* 1, 87-115.

Hustadt, U. (1994). A multi-modal logic for stereotyping, *Proc. of the 4th Int. Conf. on User Modeling*, Hyannis.

Hull, R. (1985). *The Language Gap: How Classroom Dialogue Fails.* London: Methuen.

Hutchins, E. (1991). The social organization of distributed cognition, in L. Resnick, J. Levine and S. Teasley (eds.), *Perspectives on Socially Shared Cognition*, Hyattsville, MD: American Psychological Association.

Ikeda, M. and Mizoguchi, R. (1994). FITS: a framework for ITS - a computational model of tutoring, *Journal of Artificial Intelligence in Education*, 5, 319-348.

Jones, R. and VanLehn, K. (1992). A fine-grained model of skill acquisition, *Proc. of 14th annual meeting of the Cognitive Science Society*, Hillsdale: Erlbaum.

Kaelbling, L.P. and Rosenschein, S.J. (1990). Action and planning in embedded systems, in P. Maes (ed.), *Designing Autonomous Agents*, Cambridge, Ma.: MIT Press.

Kamsteeg, P.A. (1994). *Teaching Problem Solving by Computer*, Den Haag: CIP

Kass, R. (1991). Building a user model implicitly from a cooperative advisory dialog, *User Modeling and User-Adapted Interaction*, 1, 203-258.

Katz, S. and Lesgold, A. (1993). The role of the tutor in computer-based collaborative learning situations, in S.P. Lajoie and S.J. Derry (eds.), *Computers as Cognitive Tools*, Hillsdale: Erlbaum.

Katz, S., Lesgold, A., Eggan, G. and Gordin, M. (1994). Modelling the student in SHERLOCK II, in J.E. Greer and G.I. McCalla (eds.), *Student Modelling: the Key to Individualized Knowledge-Based Instruction*, Berlin: Springer-Verlag.

Kautz, H.A. (1990). A circumscriptive theory of plan recognition, in P.R. Cohen, J. Morgan and M.E. Pollack (eds.), *Intentions in Communications*, Cambridge: MIT Press.

Kay, J. (1994). Lies, damned lies and steretypes: pragmatic approximations of users, *Proc. of the 4th Int. Conf. on User Modeling*, Hyannis.

Kolodner, J.L. (1991). Improving human decision making through case-based decision aiding, *AI Magazine*, 12, 2, 52-68.

Kolodner, J.L. (1993). *Case-Based Reasoning*, San Mateo: Morgan Kaufmann.

Kommers, P., Jonassen, D. and Mayes, T. (1992). *Cognitive Tools for Learning,* Berlin: Springer.

Kono, Y., Ikeda, M. and Mizoguchi, R. (1994). THEMIS: a nonmonotonic inductive student modeling system, *Journal of Artificial Intelligence in Education*, 5, 371-413.

Konolige, K. (1988). Reasoning by introspection, in P. Maes and D. Nardi (eds.), *Meta-Level Architectures and Reflection*, Amsterdam: North-Holland.

Kuipers, B. (1986). Qualitative simulation, *Artificial Intelligence*, 29, 289-388.

Laird, J., Rosenbloom, P. and Newell, A. (1986). *Universal Subgoaling and Chunking: the Automatic Generation and Learning of Goal Hierarchies*, Hingham: Kluwer.

Lajoie, S.P. (1993). Computer environments as cognitive tools for enhancing learning, in S. Lajoie and S. Derry (eds.), *Computers as Cognitive Tools*, Hillsdale, N.J.: Erlbaum.

Lajoie, S.P. and Derry, S.J., eds. (1993). *Computers as Cognitive Tools*, Hillsdale: Erlbaum.

Lakemeyer, G. (1993). All they know: a study in multi-agent autoepistemic reasoning, *Proc. of Int. Joint Conference on Artificial Intelligence*, Chambery.

Lakoff, G. (1987). *Women, Fire and Dangerous Things*, Chicago: University of Chicago Press.

Lapointe, S., Ling, C. and Matwin, S. (1993). Constructive inductive logic programming, *Proceedings of the International Joint Conference on Artificial Intelligence*, Chambery.

Larkin, J., Chabay, R. and Sheftic, C., eds. (1992). *Computer-Assisted Instruction and Intelligent Tutoring Systems: Establishing Communication and Collaboration*, Hillsdale, N.J.: Lawrence Erlbaum.

Lave, J. and Wenger, E. (1991). *Situated Learning: Legitimate Peripheral Participation*, Cambridge: Cambridge Univ. Press.

Leinhardt, G. and Ohlsson, S. (1990). Tutorials on the structure of tutoring from teaching, *Journal of Artificial Intelligence in Education*, 2, 1, 21-46.

Lemmon, E.J. (1965). *Beginnng Logic*, London: Nelson.

Lenat, D.B. (1982). AM: discovery in mathematics as heuristic search, in R. Davis and D.B. Lenat (eds.), *Knowledge-Based Systems in Artificial Intelligence*, San Francisco: McGraw-Hill.

Lepper, M.R. and Chabay, R.W. (1988). Socializing the intelligent tutor: bringing empathy to computer tutors, in H. Mandl and A.M.Lesgold (eds.), *Learning Issues for Intelligent Tutoring Systems*, New York: Springer-Verlag.

Lepper, M.R., Woolverton, M., Mumme, D.L. and Gurtner, J-L. (1993). Motivational techniques of expert human tutors: lessons for the design of computer-based tutors, in S.P. Lajoie and S.J. Derry (eds.), *Computers as Cognitive Tools*, Hillsdale: Erlbaum.

Lesgold, A.M. (1988). Toward a theory of curriculum for use in designing intelligent instructional systems, in H. Mandl and A.M.Lesgold (eds.), *Learning Issues for Intelligent Tutoring Systems*, New York: Springer-Verlag.

Lesgold, A.M., Lajoie, S.P., Bunzo, M. and Eggan, G. (1992). Sherlock: a coached practice environment for an electronics troubleshooting job, in J.H. Larkin and R.W. Chabay (eds.), *Computer-Assisted Instruction and Intelligent Tutoring Systems*, Hillsdale, NJ: Erlbaum.

Levesque, H. (1984). A logic of implicit and explicit belief, *Proceedings AAAI-84*, Austin.

Levesque, H. (1990). All I know: a study in autoepistemic logic, *Artificial Intelligence*, 42, 263-309.

Littman, D., Pinto, J. and Soloway, E. (1990). The knowledge required for tutorial planning: an empirical analysis, *Interactive Learning Environments*, 1, 124-151.

Lin, F. and Shoham, Y. (1992). A logic of knowledge and justified assumptions, *Artificial Intelligence*, 57, 271-289.

Locke, J. (1690). *An Essay Concerning Human Understanding*.

Maes, P. and Nardi, D., eds. (1988). *Meta-Level Architectures and Reflection*, Amsterdam: North-Holland.

Mandl, H. and Lesgold, A., eds., (1988), *Learning Issues for Intelligent Tutoring Systems*, New York: Springer-Verlag.

Mann, W.C. and Thompson, S.A. (1987). Rhetorical structure theory: A theory of text organization, in C. Polanyi (ed.), *Discourse Structure*, Norwood, NJ: Ablex.

Mark, M.A. and Greer, J.E. (1993). Evaluation methodologies for intelligent tutoring systems, *Journal of Artificial Intelligence in Education*, 4, 129-153.

Martin, J.D. and VanLehn, K. (1993). OLAE: progress toward a multi-activity, Bayesian student modeler, *Proc. of the World Conference on Artificial Intelligence in Education*, Edinburgh: AACE.

Maybury, M.T. (1994). Research in multimedia and multimodal parsing and generation, to appear in *Artificial Intelligence Review*.

McArthur, D., Stasz, C., and Zmuidzinas, M. (1990). Tutoring techniques in algebra, *Cognition and Instruction*, 7, 197-244.

McCarthy, J. and Hayes, P. (1969). Some philosophical poblems from the standpoint of artificial intelligence, in B. Meltzer and D. Michie (eds.), *Machine Intelligence 4*, Edinburgh: Edinburgh Univ. Press.

McCoy, K.F. (1989). Generating context-sensitive responses to object-related misconceptions, *Artificial Intelligence*, 41, 157-195.

McDermott, D. (1987). A critique of pure reason, *Computational Intelligence*, 3, 151-160.

Mengel, S. and Lively, W. (1991). On the use of neural networks in intelligent tutoring systems, *Journal of Artificial Intelligence in Education,* 2, 43-56.

Merrill, D.C. and Reiser, B.J. (1994). Scaffolding effective problem solving strategies in interactive learning environments, *Proc. of the 16th Annual Conference of the Cognitive Science Society*, Atlanta.

Merrill, D.C., Reiser, B.J., Merrill, S.K. and Landes, S. (1993). Tutoring: guided learning by doing, Technical Report 45, The Institute for the Learning Sciences, Northwestern University.

Mitchell, T.M. (1982). Generalization as search, *Artificial Intelligence*, 18, 203-226.

Mitchell, T.M., Keller, R.M. and Kedar-Cabelli, S.T. (1986). Explanation-based generalization: a unifying view, *Machine Learning*, 1, 47-80.

Moore, J.D. (1989). A reactive approach to explanation in expert and advice-giving systems, PhD thesis, University of California, Los Angeles.

Moore, J.D. (1995). *Participating in Explanatory Dialogues: Interpreting and Responding to Questions in Context*, Cmabridge, Mass.: MIT Press.

Moore, J.D. and Paris, C.L. (1992). Exploiting user feedback to compensate for the unreliability of user models, *User Modeling and User-Adapted Interaction*, 2, 287-330.

Moore, J.D. and Pollack, M.E. (1992). A problem for RST: the need for multi-level discourse analysis, *Computational Linguistics*, 18, 537-544.

Moyse, R. and Elsom-Cook, M., eds. (1992). *Knowledge Negotiation*, London: Academic Press.

Murray, T. (1993). Formative qualitative evaluation for 'exploratory' ITS research, *Journal of Artificial Intelligence in Education*, 4, 179-208.

Murray, T., Schultz, K., Brown, D. and Clement, J. (1990). An analogy-based tutor for remediating physics misconceptions, *Interactive Learning Environments*, 1, 79-101.

Murray, T. and Woolf, B.P. (1992). Tools for teacher participation in ITS design, *Proc. of Intelligent Tutoring Systems 92*, Montreal.

Murray, W.R. (1990). A blackboard-based dynamic instructional planner, Report No. R-6376, FMC Corporation.

Musto, D. and Konolige, K. (1993). Reasoning about perception, *AI Communications*, 6, 207-212.

Negroponte, N. (1994). *Being Digital*, New York: Hodder & Stoughton.

Newell, A. (1982). The knowledge level, *Artificial Intelligence*, 18, 87-127.

Newell, A. and Simon, H.A. (1972). *Human Problem Solving*, Englewood Cliffs: Prentice-Hall.

Newell, A. and Card, S.K. (1985). The prospects for psychological science in human-computer interaction, *Human Computer Interaction*, 1, 209-242.

Nichols, P., Pokorny, R., Jones, G., Gott, S.P. and Alley, W.E. (1993). Evaluation of an avionics troubleshooting tutoring system, Technical Report, Armstrong Laboratory, Brooks Air Force Base, Texas.

Nilsson, N.J. (1980). *Principles of Artificial Intelligence*, Palo Alto: Tioga.

Nilsson, N.J. (1986). Probabilistic reasoning, *Artificial Intelligence*, 28, 71-87.

Ohlsson, S. (1991). System hacking meets learning theory: reflections on the goals and standards of research in artificial intelligence and education, *Journal of Artificial Intelligence in Education*, 2, 3, 5-18.

Ohlsson, S. (1992). Artificial instruction: a method for relating learning theory to instructional design, in M. Jones and P.H. Winne (eds.), *Adaptive Learning Environments*, Berlin: Springer-Verlag.

Ohlsson, S. (1994). Constraint-based student modelling, in J.E. Greer and G.I. McCalla (eds.), *Student Modelling: the Key to Individualized Knowledge-Based Instruction*, Berlin: Springer-Verlag.

Ohlsson, S. and Langley, P. (1988). Psychological evaluation of path hypothese in cognitive diagnosis, in H. Mandl and A. Lesgold (eds.), *Learning Issues for Intelligent Tutoring Systems*, New York: Springer-Verlag.

O'Shea, T. and Self, J.A. (1983). *Learning and Teaching with Computers: Artificial Intelligence in Education*, Brighton: Harvester.

Paiva, A.M. and Self, J.A. (1995). TAGUS: a user and learner modeling workbench, to appear in *User Modeling and User-Adapted Interaction*.

Palinscar, A.M. (1989). Less chartered waters, *Educational Researcher*, 18, 4, 5-7.

Papert, S. (1980). *Mindstorms: Children, Computers and Powerful Ideas*, New York: Basic Books.

Papert, S. (1990). Introduction, in I. Harel (ed.), *Constructionist Learning*, MIT Media laboratory, Cambridge, Mass.

Papert, S. (1993). *The Children's Machine*, New York: Basic Books.

Payne, S.J. and Squibb, H.R. (1990). Algebra mal-rules and cognitive accounts of error, *Cognitive Science*, 14, 445-481.

Peachey, D.R. and McCalla, G.I. (1986). Using planning techniques in intelligent tutoring systems, *International Journal of Man-Machine Studies*, 24, 77-98.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*, San Mateo: Morgan Kaufmann.

Perkins, D.N. and Salomon, G. (1989). Are cognitive skills context-bound?, *Educational Researcher*, 18, 1, 16-25.

Piaget, J. (1967). *Biology and Knowledge*, Paris: Gallimard.

Piaget, J. (1976). *The Grasp of Consciousness: Action and Concept in the Young Child*, Cambridge, MA: Harvard University Press.

Pilkington, R.M., Hartley, J.R., Hintze, D. and Moore, D. (1992). Learning to argue and arguing to learn, *Journal of Artificial Intelligence in Education*, 3, 275-295.

Ploetzner, R. (1995). Problem-oriented support of coordinated knowledge use in physics.

Pollack, M.E. (1990). Plans as complex mental attitudes, in P.R. Cohen, J. Morgan and Pollack, M.E. (eds.), *Intentions in Communication*, Cambridge, Ma.: MIT Press.

Pollack, M.E. (1992). The uses of plans, *Artificial Intelligence*, 47, 43-68.

Pollock, J.L. (1992). How to reason defeasibly, *Artificial Intelligence*, 57, 1-42.

Polson, M.C. and Richardson, J.J., eds. (1988). *Foundations of Intelligent Tutoring Systems*, Hillsdale, N.J.: Erlbaum.

Poole, D.L. (1988). A logical framework for default reasoning, *Artificial Intelligence*, 36, 27-47.

Poole, D.L. (1994). Probabilistic Horn abduction and Bayesian networks, *Artificial Intelligence*, 64, 81-129.

Posner, G., Strike, K., Hewson, P. and Gertzog, W. (1982). Accommodation of a scientific conception: toward a theory of conceptual change, *Science Education*, 66, 211-227.

Pressley, M., Snyder, B.L. and Cariglia-Bull, T. (1987). How can good strategy use be taught to children? Evaluation of six alternative approaches, in S.M. Cormier and J.D. Hagman (eds.), *Transfer of Learning*, New York: Academic.

Quinlan, J.R. (1986). Induction of decision trees, *Machine Learning*, 1, 81-106.

Reed, S.K. (1993). A schema-based theory of transfer, in D.K. Detterman and R.J. Sternberg (eds.), *Transfer on Trial: Intelligence, Cognition, and Instruction*, Norwood, NJ: Ablex.

Regian, W. and Shute, V.J., eds. (1992). *Cognitive Approaches to Automated Instruction*, Hillsdale, N.J.: Lawrence Erlbaum.

Reichman, P. (1985). *Getting Computers to Talk Like You and Me*, Cambridge, MA.: MIT Press.

Reigeluth, C.M. (1993). Functions of an automated instructional design system, in J.M. Spector, M.C. Polson and D.J. Muriada (eds.), *Automating Instructional Design*, Englewood Cliffs, NJ: Educational Technology Publications.

Reiter, R. (1987a). Nonmonotonic reasoning, *Annual Review of Computer Science*, 2, 147-186.

Resnick, L.B. (1989). Introduction, in L.B. Resnick (ed.), *Knowing, Learning, and Instruction*, Hillsdale, N.J.: Lawrence Erlbaum.

Reusser, K. (1993). Tutoring systems and pedagogical theory: representational tools for understanding, planning, and reflection in problem-solving, in S. Lajoie and S. Derry (eds)., *Computers as Cognitive Tools*, Hillsdale, N.J.: Erlbaum.

Rich, E. (1989). Stereotypes and user modeling, in A. Kobsa and W. Wahlster (eds.), *User Models in Dialog Systems*, Berlin: Springer-Verlag.

Rich, E. and Knight, K. (1991). *Artificial Intelligence*, New York: McGraw-Hill.

Roberts, A. and Mountford, C.P. (1969). *The Dawn of Time*, Adelaide: Rigby Ltd.

Roos, N. (1992). A logic for reasoning with inconsistent knowledge, *Artificial Intelligence*, 57, 69-103.

Roth, S.F. and Hefley, W.E. (1993). Intelligent multimedia presentatio systems: research and principles, in M.T. Maybury (ed.), *Intelligent Multimedia Interfaces*, Menlo Park: AAAI Press.

Russell, S.J., Subramanian, D. and Parr, R. (1993). Provably bounded optimal agents, *Proc. of the Int. Joint Conference on Artificial Intelligence*, Chambery.

Ryle, G. (1949). *The Concept of Mind*, New York: Barnes & Noble, Inc.

Sack, W., Soloway, E. and Weingrad, P. (1994). Re-writing Cartesian student models, in J.E. Greer and G.I. McCalla (eds.), *Student Modelling: the Key to Individualized Knowledge-Based Instruction*, Berlin: Springer-Verlag.

Sandberg, J. and Weilinga, B. (1992). Situated cognition: a paradigm shift?, *Journal of Artificial Intelligence in Education*, 3, 129-138.

Schank, R.C. (1990). Case-based teaching: four experiences in educational software design, *Interactive Learning Environments*, 1, 231-254.

Schank, R.C. and Cleary, C. (1995). *Engines for Education*, San Mateo: Morgan Kaufman.

Schank, R.C. and Edelson, D.J. (1989). Discovery systems, in D. Bierman, J. Breuker and J. Sandberg (eds.), *Artificial Intelligence and Education*, Amsterdam: IOS.

Schoenfeld, A.H., ed. (1987). *Cognitive Science and Mathematics Education*, Hillsdale, N.J.: Lawrence Erlbaum.

Schofield, J.W., Evans-Rhodes, D. and Huber, B.R. (1990). Artificial intelligence in the classroom: the impact of a computer-based tutor on teachers and students, *Social Science Computer Review*, 8, 24-41.

Searle, J.R. (1976). A classification of illocutionary acts, *Language in Society*, 5, 1-23.

Self, J.A., ed. (1988). *Artificial Intelligence and Human Learning*, London: Chapman and Hall.

Self, J.A. (1990). Theoretical foundations of intelligent tutoring systems, *Journal of Artificial Intelligence in Education*, 1, 4, 3-14.

Self, J.A. (1992). Computational mathetics: the missing link in intelligent tutoring systems research, in E. Costa (ed.), *New Directions for Intelligent Tutoring Systems*, Berlin: Springer-Verlag, 295-352.

Self, J.A. (1993). Model-based cognitive diagnosis, *User Modeling and User-Adapted Interaction*, 3, 89-106.

Self, J.A. (1995). Dormorbile: a vehicle for metacognition, in T.-W. Chan and J.A. Self (eds.), *Emerging Computer Technologies in Education*, Charlottesville: AACE.

Shanahan, M. (1993). Explanation in the situation calculus, *Proc. of the Int. Joint Conference on Artificial Intelligence*, Chambery.

Shevell, R.S. (1983). *Fundamentals of Flight*, Englewood Cliffs: Prentice-Hall.

Shoham, Y. (1993). Agent-oriented programming, *Artificial Intelligence*, 60, 51-92..

Shortliffe, E.H. (1976). *Computer-based Medical Consultation: MYCIN*, New York: Elsevier.

Shute, V.J. (1993). A macroadaptive approach to tutoring, *Journal of Artificial Intelligence in Education*, 4, 61-93.

Shute, V.J. and Bonar, J. (1986). An intelligent tutoring systems for scientific inquiry skills, *Proc. of the 8th Cognitive Science Society Conference*, Amherst.

Shute, V.J. and Glaser, R. (1990). A large-scale evaluation of an intelligent discovery world: Smithtown, *Interactive Learning Environments*, 1, 51-78.

Shute, V.J. and Psotka, J. (1994). Intelligent tutoring systems: past, present and future, to appear in D. Jonassen (ed.), *Handbook of Research on Educational Communications and Technology*.

Sime, J. (1993). Modelling a learner's multiple models with Bayesian belief networks, *Proc. of Artificial Intelligence in Education 1993*, Edinburgh.

Singley, M.K. (1990). The reification of goal structures in a calculus tutor: effects on problem-solving performance, *Interactive Learning Environments*, 1, 102-123.

Singley. M.K. and Anderson, J.R. (1989). *The Transfer of Cognitive Skill*, Cambridge, MA: Harvard Univ. Press.

Slade, S. (1991). Case-based reasoning: a research paradigm, *AI Magazine*, 12, 1, 42-55.

Sleeman, D.H., Kelly, A.E., Martinak, R., Ward, R.D. and Moore, J.L. (1989). Studies of diagnosis and remediation with high school algebra students, *Cognitive Science*, 13, 551-568.

Sleeman, D.H. and Smith, M.J. (1981). Modelling students' problem solving, *Artificial Intelligence*, 16, 171-187.

Smith, D.A., Greeno, J.G. and Vitolo, T.M. (1989). A model for competence for counting, *Cognitive Science*, 13, 183-211.

Snow, R.E. (1990). Toward assessment of cognitive and conative structures in learning, *Educational Researcher*, 18, 9, 8-14.

Soloway, E., Guzdial, M., Brade, K., Hohmann, L., Tabak, I., Weingrad, P. and Blumenfeld, P. (1992). Technological support for the learning and doing of design, in M. Jones and P.H. Winne (eds.), *Adaptive Learning Environments*, Berlin: Springer-Verlag.

Spada, H. (1993). How the role of cognitive modeling for computerized instruction is changing, *Proc. of the World Conference on Artficial Intelligence in Education,* Edinburgh: AACE.

Spector, J.M. (1993). Introduction, in J.M. Spector, M.C. Polson and D.J. Muriada (eds.), *Automating Instructional Design*, Englewood Cliffs, NJ: Educational Technology Publications.

Strawson, P.E. (1971). Intention and convention in speech acts, in J.R. Searle (ed.), *The Philosophy of Language*, London: Oxford Univ. Press.

Sycara, K. (1989). Multiagent compromise via negotiation, in L. Gasser and M.N. Huhns (eds.), *Distributed Artificial Intelligence II*, San Mateo: Morgan Kaufmann.

Tattersall, C. (1992). Generating help for users of application software, *User Modeling and User-Adapted Interaction*, 2, 211-248.

Teasley, S.D. and Roschelle, J. (1993). Constructing a joint problem space: the computer as a tool for sharing knowledge, in S.P. Lajoie and S.J. Derry (eds.), *Computers as Cognitive Tools*, Hillsdale: Erlbaum.

Twidale, M.B. (1989). Intermediate representations for student error diagnosis and support, in D. Bierman, J. Breuker and J. Sandberg (eds.)*, Artificial Intelligence and Education,* Amsterdam: IOS.

Ur, S. and VanLehn, K. (1995). STEPS: a simulated, tutorable physics student, to appear in *Journal of Artificial Intelligence in Education*, 6.

Van Arragon, P. (1991). Modeling default reasoning using defaults, *User Modeling and User-Adapted Interaction*, 1, 259-288.

Van Joolingen, W. (1994). QMaps: Qualitative reasoning for simulation learning environments, *Journal of Artificial Intelligence in Education*, 5, 177-198.

VanLehn, K. (1982). Bugs are not enough: empirical studies of bugs, impasses and repairs in procedural skill, *Journal of Mathematical Behaviour*, 3, 3-71.

VanLehn, K. (1987). Learning one subprocedure per lesson, *Artificial Intelligence*, 31, 1-40.

VanLehn, K. (1988). Toward a theory of impasse-driven learning, in H. Mandl and A. Lesgold (eds.), *Learning Issues for Intelligent Tutoring Systems*, New York: Springer.

VanLehn, K. (1990). *Mind Bugs: the Origins of Procedural Misconceptions*, Cambridge, Mass.: MIT Press.

VanLehn, K. (1991). Two pseudo-students: applications of machine learning to formative evaluation, in R. Lewis and S. Otsuki (eds.), *Advanced Research on Computers in Education*, Amsterdam: Elsevier.

VanLehn, K. (1993). Cascade: a simulation of human learning and its applications, *Proc. of the World Conference on Artficial Intelligence in Education*, Edinburgh: AACE.

VanLehn, K., Jones, R.M. and Chi, M.T.H. (1992). A model of the self-explanation effect, *Journal of the Learning Sciences*, 2, 1-59.

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

Vera, A.H. and Simon, H.A. (1993). Situated action: a symbolic interpretation, *Cognitive Science*, 17, 7-48.

Vila, L. (1994). A survey of temporal reasoning in artificial intelligence, *AI Communications*, 7, 1, 4-28.

Villano, M. (1992). Probabilistic student models: Bayesian belief networks and knowledge space theory, in C. Frasson, G. Gauthier and G.I. McCalla (eds.), *Intelligent Tutoring Systems*, Berlin: Springer-Verlag.

Vosniadou, S. (1992). Fostering conceptual change: the role of computer based environments, in E. De Corte, H.C. Linn, H. Mandl and L. Verschaflel (eds.), *Computer-Based Learning Environments and Problem Solving*, Berlin: Springer-Verlag.

Vygotsky, L.S. (1978). *Mind in Society*, Cambridge, Mass.: Harvard University Press.

Waern, A. (1994). Plan inference for a purpose, *Proc. of the 4th Int. Conf. on User Modeling*, Cape Cod.

Wahlster, W., André, E., Finkler, W., Profitlich, H-J., and Rist, T. (1993). Plan-based integration of natural language and graphics generation, *Artificial Intelligence*m 63, 387-427.

Webb, G.I. (1993). Feature based modelling, *Proceedings on the World Conference on Artificial Intelligence in Education*, Edinburgh.

Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems*, Los Altos: Morgan Kaufmann.

Wexler, J.D. (1970). Information networks in generative computer-assisted instruction, *IEEE Trans. on Man-Machine Systems*, 11, 181-190.

Weyhrauch, R. (1980). Prolegomena to a theory of mechanized formal reasoning, *Artificial Intelligence*, 13, 133-170.

White, B.Y. (1993). Intermediate causal models: a missing link for successful science education?, in R. Glaser (ed.), *Advances in Instructional Psychology*, 4, 177-250.

White, B. and Frederiksen, J. (1990). Causal model progressions as a foundation for intelligent learning environments, *Artificial Intelligence*, 24, 99-157.

Whitehead, A.N. (1932). *The Aims of Education*, London: Benn.

Wilkin, B. (1994). The self-explanation effect with self-generated diagrams, Technical Report CSM-9, UC Berkeley,

Wilks, Y. and Ballim, A. (1987). Multiple agents and the heuristic ascription of beliefs, *Proc. of the Int. Joint Conf. on Artificial Intelligence*, 118-124.

Wineburg, S.S. (1989). Remembrance of theories past, *Educational Researcher*, 18, 4, 7-10.

Winne, P.H. (1993). A landscape of issues in evaluating adaptive learning systems, *Journal of Artificial Intelligence in Education*, 4, 309-332.

Winograd, T. (1975). Frame representations and the declarative/procedural controversy, in D. Bobrow and A. Collins (eds.), *Representation and Understanding*, New York: Academic Press.

Winston, P.H. (1992). *Artificial Intelligence*, Reading, Mass: Addison-Wesley.

Wogulis, J. and Pazzani, M.J. (1993). A methodology for evaluating theory revision systems, *Proceedings of the International Joint Conference on Artificial Intelligence*, Chambery.

Wooldridge, M.J. and Jennings, N.R. (1994a). Intelligent agents: theory and practice, *Knowledge Engineering Review*.

Wooldridge, M.J. and Jennings, N.R. (1994b). Formalizing the cooperative problem solving process, *Proc. of the 13th Int. Workshop on Distributed Artificial Intelligence*, Lake Quinalt, WA.

Woolf, B. (1988). Representing complex knowledge in an intelligent machine tutor, in J.A. Self (ed.), *Artificial Intelligence and Human Learning,* Chapman and Hall.

Wu, D. (1991). Active acquisition of user models: implications for decision-theoretic dialog planning and plan recognition, *User Modeling and User-Adapted Interaction*, 1, 149-172.

Zadeh, L. (1987). Commonsense and fuzzy logic, in N. Cercone and G. McCalla (eds.), *The Knowledge Frontier: Essays in the Representation of Knowledge*, New York: Springer-Verlag.