BREAD TODAY, JAM TOMORROW:
THE IMPACT OF AI ON EDUCATION

Benedict du Boulay and Aaron Sloman[*]

1988

Presented to:
FIFTH INTERNATIONAL CONFERENCE ON TECHNOLOGY AND EDUCATION
Education In The 90s: Challenges Of The New Information Technologies
EDINBURGH, SCOTLAND 28 - 31 MARCH 1988

Cognitive Science Research Papers
Serial No. CSRP 098
School of Cognitive Sciences
University of Sussex
Brighton, BN1 9QN, England

[*]NOTE
Aaron Sloman is now at the University of Birmingham
http://www.cs.bham.ac.uk/~axs

BREAD TODAY, JAM TOMORROW: THE IMPACT OF AI ON EDUCATION
Benedict du Boulay and Aaron Sloman

ABSTRACT

Several factors make it very difficult to automate skilled teacher student interactions, e.g. integrating new material in a way that links effectively to the student's existing knowledge, taking account of the student's goals and beliefs and adjusting the form of presentation as appropriate. These difficulties are illustrated with examples from teaching programming. There are domain-specific and domain-neutral problems in designing ITS. The domain-neutral problems include: encyclopaedic knowledge, combining different kinds of knowledge, knowing how to devise a teaching strategy, knowing how to monitor and modify the strategy, knowing how to motivate intellectual curiosity, understanding the cognitive states and processes involved in needing (wanting) or an explanation, knowing how to cope with social and affective processes, various communicative skills (this includes some of the others), knowing how to use various representational and communicative media, and knowing *when* to use them (an example of strategy).

INTRODUCTION

Attempting to build intelligent teaching systems (ITS) has a large potential pay-off in the long term, both through practical systems produced and because of the theoretical developments in our understanding of teaching, learning and communicating that such systems would necessitate.

However, good research in AI characteristically reveals problems that transform optimists into doubters, and there is at least as much reason for this to happen in the field of ITS as in work on vision, language understanding, learning and so on. Like many other A.I. systems, ITS tend to have a very narrow range of applicability, to degrade badly when working outside their sphere of competence, to be unable to combine different kinds of knowledge, and to lack the wit to know when they are attempting to teach beyond their means. Some of the difficulty is domain specific: explicitly encoding human knowledge in any non-trivial domain is difficult. But many teaching issues cut across domains.

COMMUNICATION

The impoverished repertoire of teaching behaviours of most ITS (ref 1) might best be regarded as an impoverished understanding of communication in the wider sense. In limited cases a teacher can trade-off knowledge of communication against knowledge of the domain and detailed knowledge of the particular student being communicated with. Thus a human tutor can occasionally "teach" by offering a friendly ear to a student in difficulty so that in merely attempting to describe the problem the student solves it without any real help. In other cases a tutor can trigger independent discovery by asking a few leading questions. Such questions need to be chosen to suit both the problem and the student. How? Answering that requires a deep theory of learning, which can explain why certain ways of acquiring information produce more effective learning than others.

Several more kinds of communication used by any good teacher will have to be embodied in truly intelligent teaching systems. For example a good teacher will often get students to feel puzzled about something before presenting the explanation. How? Again an answer presupposes a theory: of the nature of intellectual puzzlement and the cognitive function of explanations - not to be found, for example, in standard text books on explanation in science. (E.g. Nagel, 1961). More generally we need to understand what makes someone *want* to know something. Not all intellectual curiosity is derived from specific practical goals. A good communicator exploits an intuitive understanding of how to make the listener want to know more. Why are we upset when we miss the ending of a story?

It is easy to forget that teaching is a social behaviour that exploits both the teacher's *and the student's* rich human understanding of communication in general. Work is proceeding on unravelling some of the complexities of tutorial dialogues (see e.g. ref 2) but it does not approach a good human teacher's ability to take account of the social and affective aspects of personality with which teaching and learning are intimately bound up. This includes, for example, using cues like tone of voice and facial expression in determining what the real problems in learning are: noticing that the learner is upset may be more important than identifying the concept that has been misunderstood.

MODELLING GOALS AND BELIEFS

Even though ITS have moved beyond the "bucket" theory of knowledge transfer, they still tend to treat the student as an asocial, disembodied and disinterested seeker after truth or skill, able to communicate effectively through a very narrowly circumscribed set of formalisms.

There is no sense of such systems seeing themselves as part of some wider educational process with its own setting (including other teachers, other students, institutional goals etc), a setting within which the students have chosen to locate themselves with their own complex web of objectives, expectations and pre-conceptions about themselves, the setting, the subject domains and so on.

The student should be modelled as an affective being, with both intellectual and non-intellectual motives and capabilities playing a role in learning. The ITS should take into account the student's beliefs about his or her own knowledge or learning (which may be incorrect) and not just the knowledge of the domain itself: a student's mistaken belief that he understands something is a very effective barrier to learning.

It is important to identify correctly (and possibly adjust) the student's goals in relation to his learning. Learning something deeply can hard work and a student has to be motivated to expend the extra effort. Soloway reported at the Pittsburgh AI in Education Conference 1986 that he had to "bribe" students to learn more deeply by grading their work when he wanted them to produce and *critique multiple* solutions to a programming problem. Is the problem that we do not yet know how to stimulate intrinsic intellectual motivation, or is it inevitable that certain worthwhile forms of learning will not be found intrinsically worthwhile by students?

PRESENTATION

A good teacher communicates using media that are natural for the student, and the problem domain, including speech, diagrams, pictures, etc. The teacher must have the same understanding of these as the learner, and should be able to interpret speech or pictures produced by the learner, make sense of the learner's pointing gestures, etc. The difficulty is not simply an issue about lack of visual, tactile and aural input to the ITS or lack of spoken output. It is about making choices about how to present ideas in different ways for different purposes. Should a diagram be used by the ITS or will a sentential (ref 3,4) representation better facilitate the desired inferences? How? On what basis might such a decision be made? There is also the issue of allowing the student to communicate his or her understanding diagrammatically to the ITS. This needs no hardware revolution but it does need a much richer set of representations for the domain than is common. And, of course, giving computers a human like understanding of a variety of forms of representation is a hard AI task.

COMMON SENSE KNOWLEDGE AND TEACHING PROGRAMMING

Greaterp is one of the best current ITS for LISP programming (ref 5). It is able to take the novice through the first few hours of LISP and monitor and correct mistakes as they build simple programs. The system keeps very close tabs (perhaps too close) on what the student is doing, can recognise many standard errors and will try where possible to prevent the student from getting into a serious muddle. But there are some things that humans can do that it is hard to see yet how to automate within a Greaterp-like framework. Some students require a fair amount of encouragement and emotional support to get to grips with programming and to put up with the many mistakes that naturally occur when one is learning such a complex many- faceted skill. The human teacher, in a sense, can become a confederate sympathising with the student in his or her struggle with the brute machine. It is not clear that the ITS itself could take on this supportive role. An effective teacher needs to be able to identify the particular way the student understands or misunderstands a particular concept, technique or fact, in terms of that student's prior knowledge and concepts (as in Proust, ref 6). But students can misunderstand programming with all the richness of their existing knowledge, so identifying that a student has used the wrong model for some technical concept often requires highly creative insight, involving knowledge of diverse domains. It may not be possible to identify the misunderstanding at all only by reference to the student's problem-solving *within* the domain but may require some kind of conversation with the student at a higher level about the student's beliefs about the domain and its links to other domains.

There are also the difficulties of dealing with some of the other skills (besides program construction) involved in programming. Non trivial learning requires students to be creative about linking knowledge from the new domain with previously known domains.

For many students, the more freedom they are given to choose which bridges to build, the more highly motivated they are likely to be, and therefore the more likely they are to be really engaged in the work, and as a result the more likely they are to acquire deep understanding. Why? Given such freedom human learners are liable to bring in knowledge about almost any topic they know about and thus an intelligent teaching system would need to have encyclopaedic knowledge and powers to grasp good and poor analogies between different domains.

The most dramatic learning phase in teaching our elementary programming and AI course at Sussex to absolute beginners (some with a non-scientific, non-mathematical, background) usually comes at the "project" stage, after 6 or 7 weeks of "bottom up" learning to use the editor, playing with demonstration programs, and learning a range of programming techniques mostly by imitation and instantiating program schemata. In the "top down" project phase they inv ent their own project, making use of the POP-11 (ref 7) pattern matcher and database package and other programming constructs. They work on the project for 2 to 3 weeks and submit a written report on it.

For many students, even if they have previously done many *tutor-directed* exercises, it is only when they do a project *they* have chosen, which requires them to evaluate, select and use concepts and techniques from among those studied during the previous weeks, that they really begin to understand and internalise those concepts and techniques. The problem for the tutor is that, given this opportunity, the project topics chosen by students vary enormously, including, for example: simple board games, family relationships, household activities, tiny expert systems, adventure games and modelling the actions of simple machines.

In the project specification phase the student identifies the precise nature of the problem domain, its ontology and the kinds of processes that occur either in managing user interaction, or in doing internal inference, simulation, bookkeeping, etc. Advising students at this stage involves knowing both about all the domains and the kinds of things that are likely to be of interest to the students.

In the design and implementation phase the tutor needs to be able to relate statements about the domain to the selection of data-structures and operations on data-structures (e.g. representations and inferences). The tutor also needs to be able to relate descriptions of processes of user interaction or internal inference or simulation to programs that will run on the computer to handle input and output, manipulate data-structures, etc. In other words, even if the tutor is not actually writing the program, in order to advise the student at various stages from design to debugging, s/he needs to have all the knowledge that would be required of a programmer.

So an intelligent tutoring system able to advise students would need encyclopaedic knowledge about the world in which students live, and would have to combine the bulk of an automatic programming system with the ability to infer intentions in another agent from actions and half baked descriptions, and the ability to judge nicely how much advice to offer the student in order to be helpful without taking over the design.

The strategy of teaching by presenting materials then getting students to choose and solve a problem that *they* have invented is appropriate to many teaching situations besides programming. The requirement for essentially encyclopaedic common sense knowledge will therefore be present in intelligent teaching systems in such situations. In teaching mathematics, for example, which in principle can be an entirely self-contained formal discipline, it is in practice essential for most learners that they relate the abstract concepts to concrete examples from non-mathematical domains that they already know about.

CONCLUSION

For all these reasons we expect that it will be a long time before AI teaching systems can be called "intelligent", though we don’t claim it will never happen. The major impact of AI on education in the short term is thus likely to more prosaic: not so much in the design of intelligent tutoring systems, NL front ends, or other systems that can take over some of what teachers do now, but rather in the use of AI languages, environments, and toolkits together with other developments in computing and associated technology. With the help of good teachers, AI languages and environments can already provide powerful teaching tools. But that’s another story.

REFERENCES

1. S. Ohlsson, *Instructional Science*, 14, 3, 293-326 (1986).
2. B. Woolf in *Artificial Intelligence and Instruction* ed G.P. Kearsley, (Addison-Wesley, Reading MA 1987).
3. J.H. Larkin and H.A. Simon, *Cognitive Science* 11, 1, pp. 65-100 1987
4. A. Sloman, in *Research and Development in Expert Systems*, ed. M Bramer (Cambridge University Press 1985)
5. J.R. Anderson and B.J.Reiser *Byte* 10, 4 pp. 159-175 1985
6. W.L. Johnson and E.Solloway *Byte* 10, 4 pp. 179-190
7. R.Barrett, A.Ramsay and A.Sloman *POP-11: A Practical Language for Artificial Intelligence* (Ellis Horwood,
8. Ernest Nagel, *The structure of science* Routledge and Kegan Paul 1961