In D. H. Norrie & H-W. Six (Eds.), Proceedings of the International Conference on Computer Assisted Learning (ICCAL-90) - Lecture Notes in Computer Science 438, 175-183. Berlin: Springer-Verlag.

COMPUTER ASSISTED CONCEPT LEARNING

Jean-Louis Dessalles
Telecom Paris - Département Informatique
46 rue Barrault, 75013 Paris - France

summary: The present study shows that there is a qualitative difference between concept and skill acquisition, and that it may have some consequences on the design of C.A.I. courseware. We show for instance that concept learning is essentially a logical process, based on rule acquisition or modification, and that conversation (free dialogue) is best suited for concept transmission. This paper describes a mixed-initiative dialogue module which is part of the "SAVANT 3" CAI system.

1. Concepts vs skills

A great deal of effort has been devoted to assisting skill acquisition through the use of training systems, and most CAI systems are designed to help the student acquire new skills or procedural knowledge. But few studies are concerned with searching the best way to present conceptual knowledge. This is surprising, when we observe that many course outlines consist for the most part in concepts.

In the past, the word "concept" did not belong to the vocabulary of programmed instruction, since concepts simply did not exist in behaviorism, on which Skinner's work was grounded [1]. His work and further related developments considered only the student's behavior, as it can be observed and objectively measured when the student is performing a given task. Since what we call concept-acquisition does not automatically make the student behave in a specific way, this is not, according to Skinner, a scientific question. Only behavior acquisition is.

It is crucial to understand why such a conception is too narrow. Let us take two examples: the protein concept, and analog filter design. In classical CAI, it would be nonsense to teach the protein concept as such. One can put the student into a situation in which he or she has to pick out protein formulas among a set of chemical formulas. Just as he or she would be asked to design a particular filter to perform a given function. However two fundamental differences can be outlined between these examples of knowledge acquisition.

2. Irrelevance of performance to concepts

The first difference lies in the student's performance. A student who uses too many components to design a filter can make progress. But one cannot consider that the student performing a discrimination test with fewer errors has made some progress towards a better understanding of what a protein really is. A protein results from the chaining of different amino-acids. The student making mistakes perhaps has problems in identifying amino-acids, which is another story. Or else he or she has no idea of what a protein is, and uses other clues to give plausible answers. But he/she has by no means "half" understood! To measure performance makes sense in case of skill acquisition, but not in case of concept acquisition.

The second difference has to do with the "divisibility" of concepts. In Skinner theory, a behavior which appears too complex to be acquired as a whole can be split into simpler linkable units. In order to design a filter, the student can first learn how to build amplifier sub-units, then he/she learns to connect them, to stabilize the system, to perform simplifications, and so on. Unfortunately, one can hardly see how to split the protein concept: the amino-acid sub-concept is a much more complex concept, as is their link through C--O N--H binding.

Thus we must take note of the fact that the theory underlying classical C.A.I. is not sufficient to deal with concept teaching. In the following analysis, concept acquisition is described as a logical process, when skill acquisition is considered as procedural (cf. [2]). Concept transmission will be modeled by the transmission of a set of logical formulas. This will thus explain why performance, in this case, is necessarily qualitative.

3. Logical representation of a concept

Here a concept is identified with a first order predicate $p(.)$. But how can we represent a new concept? We might mention all its instances $p(x_1),p(x_2),p(x_3),\ldots$, but it is almost never possible (think of the protein concept). The other solution is to define the new concept in terms of already known concepts, through a set of new formulas:

$$\forall x;\ \ p(x) \Longrightarrow (p_1(x) \wedge p_2(x) \wedge \cdots p_n(x))$$

or

$$\forall x;\ \ p(x) \Longleftarrow (p_1(x) \wedge p_2(x) \wedge \cdots p_n(x))$$

This kind of representation has well-known limitations, but which are not relevant to us now. A concept can thus be defined by a set of Horn clauses, as in the following definition of "my_possession":

$$\forall object;\ \ was\_given\_to\_me\,(object) \Longrightarrow my\_possession\,(object)$$

$$\forall object;\ \ I\_have\_bought\,(object) \Longrightarrow my\_possession\,(object)$$

$$\forall object;\ \ I\_have\_earned\,(object) \Longrightarrow my\_possession\,(object)$$

$$\forall object;\ (\ \neg was\_given\_to\_me\,(object) \wedge \neg I\_have\_bought\,(object)$$
$$\wedge \neg I\_have\_earned\,(object)\ ) \Longrightarrow \neg my\_possession\,(object)$$

This mode of logical representation of knowledge may be very useful as far as expert systems are concerned, but it presents a major flaw when we deal with concept handling. If a student knows the concept, then she is supposed to be able to make the following deduction:

$$\forall object;\ my\_possession\,(object) \wedge \neg I\_have\_bought\,(object)$$
$$\wedge \neg I\_have\_earned\,(object) \Longrightarrow was\_given\_to\_me\,(object)$$

But this formulation, which is equivalent to the third clause defining *my_possession*, is not easy to obtain from the Horn clauses representation. That is why we will use **paradoxical clauses** instead:

$$\forall x;\ \ (\,p_1(x) \wedge p_2(x) \wedge p_3(x) \wedge \neg p\,(x)\,) \Longrightarrow \mathbf{F}$$

F stands for a proposition that is always false. For the above example:

$$\forall object;\ \ (\ was\_given\_to\_me\,(object) \wedge \neg my\_possession\,(object)\ ) \Longrightarrow \mathbf{F}$$

means that it would be paradoxical to consider that an object which was given to me doesn't belong to me. We will observe now that this way of representing concepts, besides its technical interest, has some similarity with the way new concepts are naturally introduced.

4. The didactic conversation

In order to design appropriate technical aids to help students acquire new concepts, we can draw our inspiration from a *natural* situation where concepts are transmitted. But is there any natural situation in which we transmit logical formulas? Before answering, let us note that asking the question this way may well represent progress in comparison with the Skinner paradigm which looks for situations where behaviors are transmitted.

The teaching situation, with a teacher and a learner, is not a natural situation in the biological sense. Concepts are exchanged in the most natural way when natural language is used freely: during conversation. It may appear surprising to many, but it will be explained now.

A careful study of natural conversations (cf. [3]) shows that they can be easily classified into three types. Only the first one will be relevant here, namely the type of conversations which are introduced by a paradox, as in the following real example:

- `Tout à l'heure, j'ai vu un village jumelé avec une ville d'Alsace !`
- *`I've just seen the twin village of a town in Alsace !`*

This statement is a paradox if we know that two twin cities must belong to two different countries. This utterance, like the remainder of the conversation, can be translated into a logical representation, something like:

$$\forall town\_A, town\_B\,;\,(\,twins\,(town\_A, town\_B)\,\Lambda\,\neg foreign\,(town\_A, town\_B)\,) \Longrightarrow \mathbf{F}$$

This logical representation is very useful because it allows us to anticipate the possible replies. In this type of conversation, every reply tries to break the paradox. One way to do this is to express doubts about one of its terms (this town is not in Alsace, but in Germany). The other way is to modify the paradox by adding a new fact which is false in the present situation (inland towns with less than 500 inhabitants can be twined)* :

$$\forall town\_A, town\_B\,;\,(\,twin\,(town\_A, town\_B)\,\Lambda\,\neg foreign\,(town\_A, town\_B)$$
$$\Lambda\,\neg village\,(ville\_A)\,) \Longrightarrow \mathbf{F}$$

We have just made three points. First, logic comes forward as the natural representation language of conversation (see [3]), what most people studying human interactions from a sociological point of view do not anticipate. The second interesting point is that a conceptual definition, that we model by means of a set of logical formulas, can be objectively transmitted during a conversation. In the last example, we saw how the concept "town-twinning-threshold" was introduced. Third, thanks to the logical description of a dialogue, we can dramatically limit the universe of possible replies which may follow a given utterance.

It is precisely the prediction power of this logical translation of free conversation that will allow a machine to manage a free dialogue with the learner.

5. The conversational engine

We have just seen how concepts can be translated into logical formulas, and how such "formulas" can be exchanged during conversation. It is therefore natural to try to put the learner and the machine into a situation resembling free conversation. We can hope that learning will thus become easier and more attractive to the learner, and therefore more efficient.

* these last replies are imaginary.

The first task is to program the machine to manage the logical knowledge which represents the conceptual content to be taught. I began with the design of a paradoxical clauses engine. This program, written in Prolog, works on knowledge which is a (conjunctive) set of paradoxes*:

$$\bigcap_{i=1}^{i=M} \left[\bigcap_{j=1}^{j=N(i)} p_{ij} \implies \mathbf{F} \right]$$

When asked to prove a given proposition p, this program seeks $\neg p$ throughout $\{p_{ij}\}$. If $\neg p$ is in the paradox number i (i.e. $\neg p = p_{ij}$), then the engine tries to prove p_{ik} for every $k \neq j$, without using the ith paradox. As a matter of fact, the ith paradox can be rewritten as:

$$\bigcap_{1 \leq k \leq N(i),\ k \neq j} p_{ij} \implies p$$

Actually the situation is a bit more complicated since this engine is able to deal with first order predicates. A given paradox can be instanciated several ways, and some of these instanciations may be useful to prove another one. But this kind of difficulty can be easily managed in Prolog language. This recursive proving work succeeds when the engine comes to an axiom, i.e. a paradox with a single term:

$$p_{m1} \implies \mathbf{F}$$

The great improvement with this engine is that all the predicates in a given formula of the knowledge base play a symmetrical role. It can therefore prove some predicates that could not be proved if the knowledge were expressed directly in, say, Prolog clauses. You can see that unlike Prolog, the engine does not assign the value "false" to a proposition p which it cannot prove. It does so only if $\neg p$ can be proved. A given proposition can thus have three values at a given moment: "true", "false", and "not yet determined".

Note that conversely what we gain from an increased proving power is traded off against a loss of efficiency. In a knowledge base where premises and conclusions are not *a priori* distinguishable from each other, the large number of combinations makes backward chaining more difficult. However, as far as dialogue management is concerned, the trade off is positive.

6. Learning through dialogue

Another program has been designed to manage a "free" conversation with the student. This program proceeds by looking for inconsistencies in the student's utterances.

* only propositions are considered here for writing simplicity

First, the program looks for a valid paradoxical clause in its knowledge, i.e. a clause in which the terms are either true or not yet determined. It then tries to obtain a validation for non-determined terms by uttering a paradoxical statement or a question.

The student's reply is then analysed through a keyword detection, and the program translates this reply into a logical proposition. When the program succeeds in proving all the terms of a paradox, using the student's statements, it asks her to correct the inconsistency (see the example below).

The following excerpt shows the kind of interaction that can exist between student and machine. Here "ISI" stands for Inter Symbol Interference: interference between successive pulses in digital communication. It is important to notice the following point: the machine knowledge is static. The dynamic aspect of the conversation develops in real time, depending on the student's replies. This has two consequences. First, the dialogue could have been completely different, and could have come to a different conclusion. Second, most of the time there are no "good" or "bad" answers. There may be several possible answers, but each of them has to be well-argued. For instance, the student could have answered "no" to the possibility of cancelling ISI, and she would be right in certain contexts. The machine would have then argued differently.

Machine - *Can the terms causing ISI be simultaneously zero?*
Student - yes
Machine - *We can perhaps modify the pulse duration to cancel ISI...*
Student - yes
Machine - *And then...?*
Student - we can shorten them.
Machine - *But you bypass the problem by modifying the transmitter !*
Student - that's true!
Machine - *In the present situation, the transmitter parameters cannot be changed.*
According to you, we opt for shorter pulses. It's surprising, if we assume that we do not modify the transmitter parameters !
Student - I don't shorten them any more.
Machine - *We assume that pulses overlap at the receiver input. Do you want to eliminate this overlap?*
Student - yes
Machine - *Can we really eliminate ISI by eliminating overlap?*
Student - yes
Machine - *And then...?*
Student - I put a longer gap between pulses
Machine - *According to you, we can slow down pulse emission. It's surprising if we assume that we do not modify the transmitter parameters !*
Student - ok, I don't modify the rate.
Machine - *How do you propose to eliminate ISI ?*
Student - []
Machine - *Can we, for instance, change the pulse shape ?*
Student - yes
Machine - *But you change the transmitter parameters !*
Student - no, we can filter the pulses just ahead of the receiver
Machine - *OK.*

What does the student learn during such a dialogue? We bet that when the student gets caught in a paradox, she does not hold the exact paradoxical clause in her own "knowledge base". The question is then: why not first give her, in order to save time, the total content of the logical knowledge of the system, in natural language? The answer leads us to a crucial point: the **relevance requirement.**

The problem of concept transmission is not a trivial one. Having noticed that concepts can be captured in logical formulas is not sufficient. The student has to recognize these formulas before memorizing them. And the problem stays here. To take an analogy, we have to design the optimal transmitter (the system) which matches a nature-given receiver (the student's brain).

A careful study of the natural conceptual information exchanges which take place in spontaneous conversation shows that speakers pay great attention to relevance (cf. [3]). A speaker speaks only if her utterance anchors on the logical structures built during previous replies. For instance, a relevant reply will modify the original paradox to prevent it from being applied to the present situation. In case of irrelevance, we observe reactions of incomprehension ("`Why do you say that?`"), which can even be aggressive.

This relevance requirement can be understood if we become aware that it makes the management of our logical knowledge much easier. During conversation, we carefully assemble logical terms into what appear to be logical formulas*, and so terms appear only in context. Speakers need not memorize context-free logical items, unlike what is required from students during traditional courses. This point is crucial, and that is why a great deal of effort is devoted to make the machine able to utter *relevant* replies.

7. The SAVANT 3 dialogue module

SAVANT 3 is a complete CAI system that we developed at Telecom-Paris. The dialogue module is not yet very sophisticated: the input from the learner is processed with a keyword detection which allows the system to recognize the logical proposition(s) expressed by the student. The system then answers by mixing logical propositions translated into natural language with pre-established articulations. However, it is striking to observe that the students spontaneously enter a high-level conversation with the machine. This is not an Eliza-like effect (cf.[4]), since here the students perceive the logical relevance of the system's replies and feel compelled to argue.

Some systems are able to sustain elaborate dialogues with a human being. They need two kinds of knowledge: contingent, which changes with the topic, and fixed, which enables them to conduct the dialogue. In a conventional CAI system, contingent knowledge is made of keywords and numerical domains which the system uses to classify the student's answer among expected good and bad responses. It also includes the action

* actually, pure first order logic is insufficient to represent the logic of conversation (cf. [3]), but this is not relevant here.

to be performed after this classification.

More sophisticated programs like "GUIDON" [5] have a significant amount of contingent knowledge available to them. In the case of GUIDON: all the Mycin rules!

But contingent knowledge is expensive. It is one of the main obstacles to a wide-range development of learning aids. A ground principle in SAVANT 3 is to make contingent knowledge *local.* The knowledge relevant to each dialogue is separately stored as a set of paradoxical clauses. We must add questions that the system may ask to determine missing predicate values, and keywords for answer detection. This represents a rather limited amount of knowledge, about two pages of listing for a dialogue of about ten replies, on the average.

This small amount of contingent knowledge necessary for the system to discuss a topic makes course development no more cumbersome than conventional answer analysis. However, a lot of attention must be paid how logical formulas are written. I have written a paradoxical clause editor, and we are currently working on an interactive program which will translate course content expressed by a non logician author into logic.

8. Further work

The first results are stimulating, because they prove the feasibility of this approach to the didactic dialogue. Our next task will be to improve dramatically the flexibility of interaction. We will then try to amplify the knowledge management possibilities, horizontally (topic shift) and vertically (several detail levels). Finally, since the engine can deal with predicates, we will try to detect not mere propositions, but predicates in the student's answers.

Expert systems owe their success, not to a technical breakthrough, but to the idea (audacious in the past, but well accepted now) of representing human expertise in logical form. Paradoxically, while concepts are essentially logical entities, skills were the first to be transcripted into rules in expert systems. I have tried to show that CAI can take advantage of logical concept representation and use it through student-machine conversation.

- References -

[1]: B.F. Skinner: La révolution scientifique de l'enseignement - Dessart, Bruxelles 1969 -

[2]: Kurt VanLehn: Toward a Theory of Impasse-Driven Learning, in H. Mandl, A. Lesgold: Learning Issues for Intelligent Tutoring Systems - Springer Verlag, New York 1988 -

[3]: J-L Dessalles: Stratégies naturelles d'acquisition des concepts et EAO - Cognitiva 85 - CESTA, Paris 1985

[4]: J. Weizenbaum: Computer Power and Human Reason - W.H. Freeman & Co, San Francisco 1976 -

[5]: W.J. Clancey: Methodology for Building an Intelligent Tutoring System, in G. Kearsley: Artificial Intelligence and Instruction - Addison-Wesley Publishing Company, 1987-