SOME THOUGHTS ABOUT REPRESENTING KNOWLEDGE IN INSTRUCTIONAL SYSTEMS

Joachim H. Laubsch
Institut fur Informatik, Universitat Stuttqart
Federal Republic of Germany

Abstract

This paper examines the possibilities of applying AI-Methodology developed for natural language question-answering systems to computer-aided instructional systems. Particularly, it focusses on how semantic nets can be extended to handle procedural knowledge, and how an instructional model can be expressed in terms of goal-directed processes separated from subject-matter knowledge.

Introduction

Functionally, question-answering systems and computer-aided instructional systems (CAI-systems) are similar, and it seems appropriate to transfer AI-techniques developed for Q/A-systems (11,13) to CAI-systems. But so far, research on Q/A-systems has not given much thought to the questions a user could or should be asked, or how the system's reply should be expressed as to be easily understandable by the user. These features are not only required in tutoral dialogue, but should also be present in any interactive system, where at least two partners jointly strive to achieve a goal, and each may request use of the capabilities of the other. Clearly, a teacher needs some knowledge of his student's knowledge and goals aside from knowledge about subject matter. That knowledge is referred to as a *user-model.* Other interactive applications like operating systems, debuggers for programming languages, or information-retrieval systems would also benefit from explicitly considering a model of the user in preparing their messages.

We will outline (1) how subject-matter knowledge may be represented in semantic nets, and then (2) how planning of instruction is directed by an instructional model.

Building semantic nets in a subject area

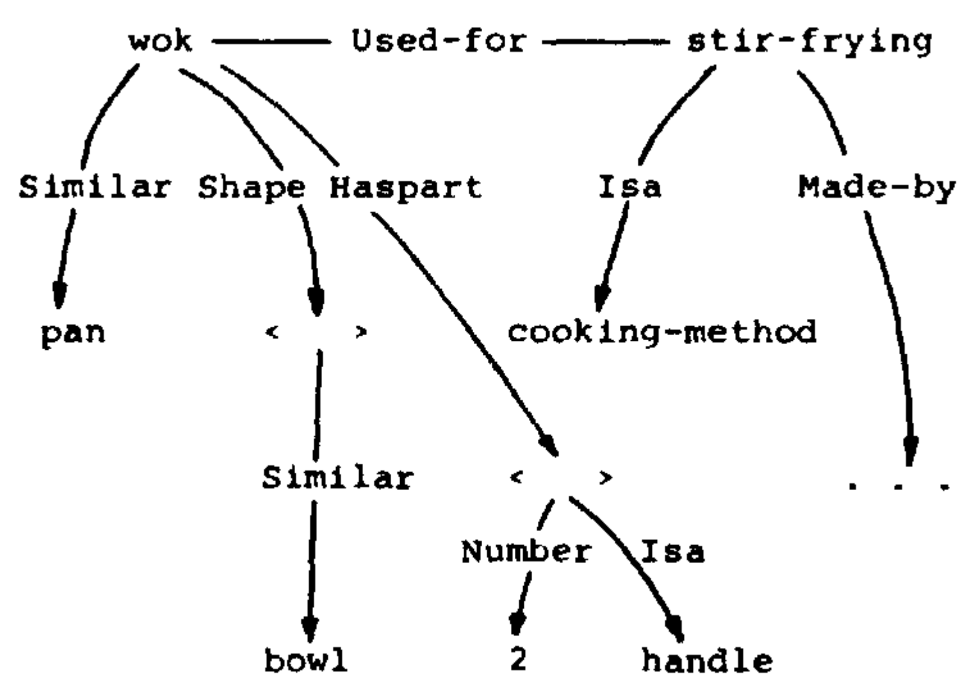
SCHOLAR by Carbonell (3) and SOPHIE by Brown (2) are examples of the use of a semantic net for representing subject-matter knowledge and using it in generating problems, or answering queries. Simmons has hypothesized that semantic nets may serve as the central cognitive structure (11). We will show how a semantic net of a subject-matter area can be built (using the notation of (8)), and how the notion of a semantic net may be expanded in order to achieve qreater flexibility in using it.

Basic objects and relations

First, certain concepts are decided to be pre-requisite and known to any student. Some objects and relations are also used as primitives in the problem-solving component of the system. The relation "north-of" may either be assumed basic or explained to the student in terms of latitude, as well as computed in terms of lat itude. In choosing basic relations and objects, it has to be considered that natural language sentences may be conveniently generated.

Concept nodes

Secondly, the concepts to be taught are defined in terms of relation-object tuples (like a LISP property list). E.g. if we teach, say "Chinese Cookery", some basic objects may be: rice, soy-sauce, pan etc., and some basic relations are: inqredient-of, made-by, shape etc.. An object like a "wok" is not basic because the student might need to know that it is like an iron pan, having a bowl-like shape with two handles, and is used mainly in a cooking method called "stir-frying":



Several questions about a "wok" may be asked by the student, and it is possible to retrieve respective answers. Generally, it is a problem to decide, how many nodes deep we shall go into the net when retrieving the subnet used for the answer. Carbonell (3) used irrelevancy tags as weights to determine the subnet included in the answer. We feel that this is a weak point in SCHOLAR, and that the relevancy should rather be determined on the basis of an instrutional model.

Situation nodes or frames

Finally, concept no serve as building blocks for larger structures capable of expressing situations, events and compositions of events. A situation can be thought of approximately as the description of part of the (hypothetical) world at an instance. Again, teaching "Chinese Cookery", we may describe the initial situation before preparing a course by listing each ingredient, respective amount, state of preparation, tools, relative position etc.. Anything altering a given situation is called an event. The description of an event should express the changes it will make, if applied to the network structure representing that situation. As a consequence it is natural to allow compositions of events (nested, sequential or parallel concatenations). In order to include such knowledge, semantic nets require procedures within them.

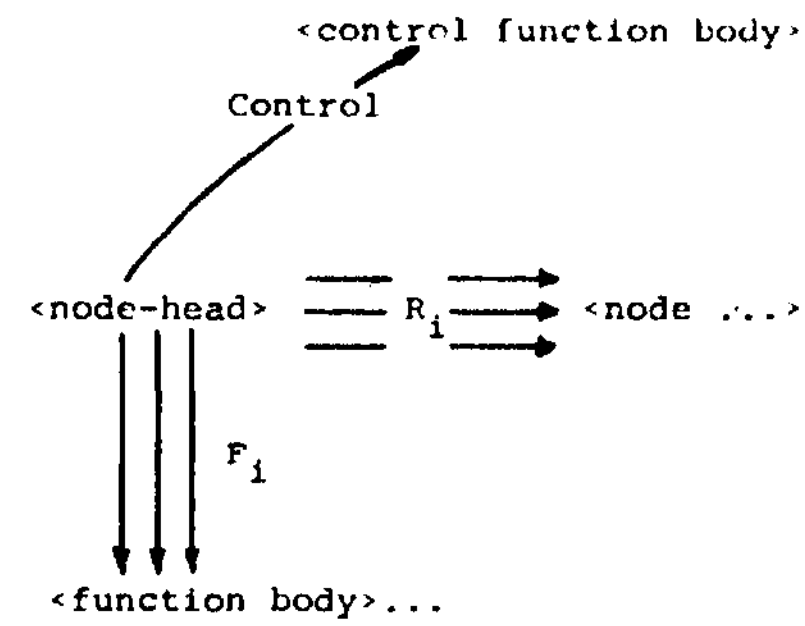
We follow Minsky's notion of a frame "as a data structure for representing a stereotyped situation"(10), and interpret it for representing subject-matter in an instructional system: "A frame is a collection of questions to be asked about a hypothetical situation; it specifies issues raised and methods to be used in dealing with them." (Minsky (10)). Organizing a curriculum for a CAI-course should then start by identifying such situations, applicable questions, and the ways they are connected, rather than building a "hierarchy of principles" (Gagne (4)). Some of the questions associated with a situation might be, what concepts are prerequisite, how a similar situation would look, which solutions are possible, or methods could be tried, etc.. We shall not dig further here now, but look for a way to extend the computational aspect of a semantic net in order to give it some of the capabilities of a frame-system. A frame differs from a conventional node in a semantic net in the sense that:

(a) Before any node being pointed to (terminal) is accessed from a frame head, an applicability test is made. If it fails, a list of alternative frames is made available. If it succeeds, it may determine which arcs to access and in what order.

(b) If a frame is accessed, it may take over control and not return to the calling frame.

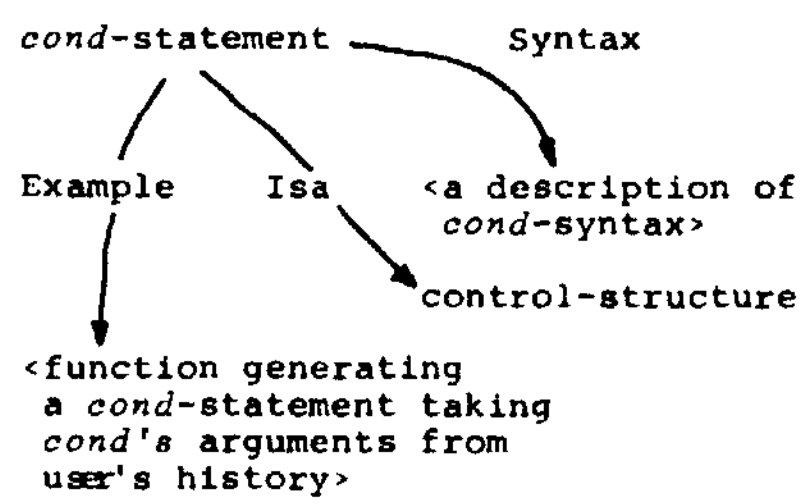
(c) The terminals may specify conditions that have to be satisfied. These conditions are functions that may have as arguments values of other terminals of this frame, or a frame through which the current frame has been previously activated. If the condition fails, a default value is given.

Schematically, we visualize a node in the net now as:



The R_i are relations in the usual sense, while the F_i are atoms with a function property. In case a terminal is accessed through his arc, the function body is *eval'*ed. An F_i may test conditions, and specify default values. The Control arc is optional, tests applicability of its frame, and determines access of terminals and return of control.

We give some examples of extended nets where the subject-matter taught is "*Programming in LISP*":



If an example for a *cond*-statement is required by the student, the arc labelled "example" is accessed. Since it has a function property, the body is *eval'*ed making it possible to generate an example applicable in the current programming problem. The advantage is, that examples for *cond*-statements can be adapted to the student's current goal.

The frame itself may take over control, and evaluates itself. Whatever is supposed to be done with control is specified in its body.

Take the previous example:

```
                      Syntax ——→ ...
                      Isa ———→ ...
cond-statement  ————— Example ——→ ...
      |               Semantics
      |                  ↓
      |          <description of what cond
Control           does to its arguments with
      |           annotations>
      ↓
<if  user wants to know what it means
     in general:
     extract annotation of semantics
 else find currently focussed cond-
     statement
     (in case of failure make example)
     singlestep semantics.
```

When control is regained, return to caller.>

The basic operations of creating, storing, retrieving, and removing concept nodes are handled by standard LISP-functions. For frames such functions must be redefined. Also, nodes should be retrievable by pattern-directed invocation (5). The development of such tools promises to be valuable for implementing a teaching system more versatile than SCHOLAR. Nevertheless it was SCHOLAR which motivated these reflections.

Planning instruction

Assume, a semantic net representing knowledge in a subject-area has been built along these lines. How could an algorithm to *teach* it be designed? One approach is to wait for a user query, understand it, invoke the frame(s) relevant to it, and call a procedure which generates a natural language reply. A different way is to let the system decide what the current goal of instruction should be, and proceed as before. Since the initiative of the dialogue is either on the system's or on the user's side, Carbonell used the term mixed-initiative dialogue to express that SCHOLAR enabled both modes (3). But it is important to notice, that *both* modes need a specification of the dialoaue's goal. The user's query must be understood as a further specification of the (possibly vaguely expressed or implied) instructional goal. Its meaning depends upon the context of previous dialogue, assumptions about the user's knowledge, and the purpose of instruction.

A human instructor (hopefully) knows what he wants to instruct, and also takes note of what he believes his student has already understood. Sometimes, before starting to teach a new concept, he makes sure that the student has grasped alle pre-requisite concepts by asking questions. The student's answer is then used for updating his notes. This illustrates some of a human teacher's intelligence. How can it be made artificial?

Let us define an *instructional model* as a representation of (1) goals of instruction, (2) methods to achieve their, and (3) a user model. The goals may be formulated in terms of a set of concepts the student can apply to a set of situations. These goals may be modified by particular user preferences. The methods include general, hence transferable knowledge a teacher has. The user model contains the history of understood concepts and solved problems, a description of situation features the user is likely to be motivated by, and possibly some characteristics of his ability (like reading speed, preferences for inductive or deductive reasoning, forgetting factor etc.). The function of an instructional model within a CAI-system would be to search the semantic net, retrieve the frames to be taught, match their pre-requisites with the prior knowledge of the student and interpret user queries and answers as updates of the instructional model.

A CAI-system should explicitly incorporate such model, but presently only few attempts have been made to do so (6). The traditional approaches to this problem using decision theory and stochastic learning models (1,7,12) have reached a dead end due to their oversimplified representation of learning. The reason for stochastic learning models failing as models for instruction is their lack of representing the *content* to be taught. The teacher's decision can in general only be made, if he knows *what* is understood, but not the probability that a specific response will be given to a specific stimulus.

A new approach would be to express the instructional model as a structure of goal-directed processes (in a language such as CONNIVER (9) or PLANNER (5)). We can sketch the idea for a (overly simplistic) first approximation to such algorithm:

Let the goal of instruction be that the student can solve problems generatable from a list of situation nodes of the network. This list could be a tree implying a default ordering of concepts (e.g. in terms of the pre-requisite relation). In planning his next move, the instructor takes a node from this list, and tries to find out whether it is answerable by the user model.

A problem generated from a situation node is *answerable* if its listed pre-requisites are marked as known or if a problem-solving procedure containing only steps previously demonstrated in other situations can be successfully applied. The function that evaluates whether a problem is answerable may in case of failure return a list of unknown pre-requisites, and also recommend other frames to be examined first.

To find whether a pre-requisite is met, or a substep in a solution sequence is available to the user, the model may yield insufficient information such that a *teat* becomes necessary. A special logic for retrieving tests can ' constructed: For example, knowing that a given relation holds for the superset node of a concept implies that it is known for any node that is connected via the "Isa"-relation. Similar theorems hold for other relations like part-of or temporal ordering. They should also be used when analyzing answers to problems not given for test purpose in order to mark the probable knowledge-state of concepts.

If it turns out that a pre-requisite concept is necessary, we search for a node designed to *examplify* it. The net is built to facilitate such search by having "Example"-arcs from concepts to frames. Also, special procedures for generating examples may be constructed: Find a previous situation where the solution would have been simplified through the use of that concept, or a situation which in case a terminal is bound differently requires that concept in order to be answerable.

Rather than looking for problems with answerable pre-requisites, a different plan would present a situation node first in "general terms" by binding some terminals to simple cases, and proceed interaction with the user, specifying terminals in the direction indicated by the student's knowledge and interest. In SCHOLAR it is a problem, how deep one should follow arcs to retrieve relevant information at a given instant. If the instructor marks the arcs already retrieved, he can avoid being repetitious. Furthermore, it seems good pedagogic practice to connect new concepts with ones already known. This leads to finding connections from the presented node to already marked ones, retrieving only differences or similarities (to enhance discrimination or generalization). All this may not yet involve any restrictions by user goals. But his history may indicate that he is more interested in say political facts about South-America than its main rivers. The dialogue should be used to extract special procedures that evaluate the student's motivation when retrieving a node and filling its terminals.

The system has to *have* the *problem-solving ability* it tries to teach (at least within the topic of discourse). A student query concerning a concept, may force the instructor to explain the concept in the context of the current problem. Or, if the student has correctly solved a problem and indicated his steps, some of the used concepts may be inferred (or guessed) since each step is solvable by the instructor too. These used concepts can then be marked as "known"(or "probably known"). What does the instructor do about a bug in a user answer? He can try to produce the same bug by using heuristics about preferred errors detected in users (like leaving out a ")" or a terminating condition for recursion in LISP). Another hypothesis to test is, whether the user's answer was a correct reply to a similar problem, indicating that the situation was not correctly understood. Finally, if the instructor is not intelligent enough to understand why the proposed solution is wrong, he can tell the user facts contradictory to his solution or ask for intermediate results.

It appears within reach of AI-methodology to develop CAI-systems that act more like human teachers. A crucial step in this development is the separation of subject-area knowledge from instructional planning knowledge. What has been said is provisional, and may serve as a guideline that will need to be further extended and specified.

References

(1) Atkinson,R.C., & Paulson,J.A. An Approach to the Psychology of Instruction. *Psych.Bull.*,1972, *78,* 40-61.

(2) Brown,J.S.,Burton,R.R.,ft Bell,A.G. SOPHIE a Sophisticated Instructional Environment for Teaching Electronic Troubleshooting.BBN Rep.No.2790,Cambr.Mass.,1974.

(3) Carbonell.J.R. Mixed-initiative Man-Computer Dialogues. BBN Rep.No.1971, Cambr., Mass. 1970.

(4) Gagne,R.M. *The Conditions of Learning,* Academic Press, N.Y., 1965.

(5) Hewitt,C. PLANNER: A Language for Proving Theorems in Robots. D.E. Walker ft L.M. Norton (Eds.) *Proc.IJCAI1,*1969, 295-301.

(6) Koffman,E.B. Blount,S.,ft Wei,M. CAI in Digital Logic Design, Debugging and Programming. *Compute Elect.Engng,* 1973,299-320.

(7) Laubsch,J.H.,& Chiang,A. Application of Mathematical Models of Learning in the Decision Structure of Adaptive CAI-systems. A. GUnther et al.(Eds.) *Int.Comp.Symp.*1973, Amsterdam, 1974, 481-487.

(8) Lindsay,P.H.ft Norman,D.A. *Human Information Processing.* Acad.Press,N.Y.,1972.

(9) McDermott,D.,ft Sussman,G.J. The CONNIVER Reference Manual AI-Memo 259, MIT *Artificial Intelligence Lab.,*Cambr.,Mass. 1972.

(10) Minsky,M. A Framework for Representing Knowledge. AI Memo 306,MIT AI-Lab. Cambridge, Mass., 1974.

(11) Simmons,P.F. Semantic Networks: Their Computation and Use for Understanding English Sentences. R.Schank ft Colby (Eds.) *Computer Models of Thought and Language.* Freeman & Co.,San Francisco,1973,63-113.

(12) Smallwood,R.D.,Optimal Policy Regions for Computer-directed Teaching Systems. W. Holtzman (Ed.) *Computer-assisted In~ struction, Testing and Guidance.* Harper ft Row, New York, 101-121.

(13) Winograd,T. *Understanding Natural Language.* Edinburqh Univ. Press, Edinburgh, 1972.