QUESTION ORDERING IN MIXED INITIATIVE PROGRAM SPECIFICATION DIALOGUE

Louis Steinberg
Department of Computer Science
Rutgers University
New Brunswick, N. J. 08903

ABSTRACT

It would be nice if a computer system could accept a program specification in the form of a mixed initiative dialogue. One capability such a system must have is the ability to ask questions in a coherent order. We will see a number of reasons it is better if such a system produces all the questions it can and has a "dialogue moderator" choose which to ask next, than if the system asks the first question it thinks of. DM [9], the dialogue moderator of PSI [5], chooses questions by searching a network model of a program, under control of a set of heuristic rules. This technique is simple and flexible.

I. Introduction

When you need a computer program, it is usually easier to tell a human being what the program should do than to specify the program directly to a computer system (eg a compiler). There are a number of reasons for this, including the knowledge and reasoning ability that a human has. We will concentrate here, however, on another advantage of communicating with humans, their ability to engage in a mixed initiative dialogue, and on one particular capability required for carrying on such a dialogue, the ability to ask questions in a coherent order.*

A mixed initiative dialogue is one in which either party may take initiative. From the perspective of the work reported here, to "take initiative" in a dialogue is to alter the structure of the dialogue. This definition is essentially equivalent to that of Bobrow, et al [1], who define taking initiative as establishing or violating expectations about what will come next, since it is precisely the structure of a dialogue which gives rise to such expectations. In particular, we will be concerned here with "topic structure", the order and relationships of the topics covered in the dialogue, and with "topic initiative", the ability to affect topic structure.

The work described here [9] been done in the context of the PSI program synthesis system [5]. PSI acquires program specifications via mixed initiative, natural language dialogue.

II. The General Scheme

In order to ask questions, such a system must be able to do two things: it has to decide what aspects of the specification are as yet incomplete, and it has to decide which one of these aspects to ask about next. We will refer to the latter problem, deciding which question to ask next, as the task of "question ordering".

A. Order from the Reasoning Process

One common way to handle question ordering might be summarized as asking the first question the system thinks of. In this scheme, the system goes through its normal reasoning process, and at some point comes across a fact which it wants to know, but cannot deduce. Whenever this happens, the system stops and asks the user. (See, for example, [1] and [4]).

Note that the system stops whenever it finds any question to ask. Thus, the system asks each question as it comes up, and the order is determined by the reasoning process. If a system's reasoning process seems natural to the user, then this scheme produces a question order which seems natural, at least to a first approximation. However, there are some problems.

The basic problem is that this scheme ties the topic structure of the dialogue to the reasoning procedures of the system. This makes topic structure harder to change, since any change in topic structure requires a change in the reasoning procedure. It can also make it hard to transfer the question ordering methods to another system that uses a different reasoning method. Finally, this method of question ordering assumes that there is a single, sequential reasoning process, and is

----

* This work was supported by the Defense Advanced Research Projects Agency at the Department of Defense under contract MDA 903-76-C-0206. The author was also partially supported by an IBM Graduate Fellowship.

The work reported here would not have been possible without the environment provided by Cordell Green and the other members of the PSI project. I would also like to thank N. S. Sridharan for his comments on an earlier draft of this paper.

not possible in a system structure such as that of HEARSAY-II [7].

B. Order from a Dialogue Moderator

A better scheme is to have the reasoning process produce as many questions as it can, and to use some other mechanism to select a single one of them to ask next. This scheme largely avoids the problems of the previous one. Its main drawback is that it requires a reasoning process which is able to produce more than one question at a time. An additional advantage of this scheme is that it allows us to implement question ordering in a separate module, with a clearly defined interface to the rest of the system. I have termed such a module a "dialogue moderator".

Thus, the dialogue moderator is given a list of all the questions currently open, and must choose which one is to be asked next, so as to keep the dialogue well structured. Much recent research (eg [2], [6], [8]) has shown that structure of a dialogue is closely tied to the structure of goals and plans being pursued by the dialogue's participants. One might therefore imagine that the dialogue moderator needs a complete model of goals and plans, both those of the system and those of the user. However, in a program specification dialogue, the goals and plans of both participants are tied very closely to the structure of the program. As will be seen, it has been possible in PSI to use a simple model of the program structure instead of a complex model of goals and plans.

(It might be argued that any system which handles natural language will eventually need the full model of goals and plans anyway, so using a simpler model here is no savings in the long run. It should be noted, however, that mixed initiative does not necessarily imply natural language. A useful system might be constructed which handles mixed initiative dialogue in some formal language.)

III. The Specific Method

DM is the dialogue moderator of the PSI system. As noted above, DM maintains a simplified model of the program being specified. The program is viewed as a structured set of objects. Each object is either a piece of algorithm or a piece of data structure - the pieces of algorithm correspond roughly to the executable statements of a program, and the pieces of data structure correspond roughly to the variable declarations. A specific loop or a specific input operation might be algorithmic objects, while a set or a 5-tuple might be data structure objects.

These objects are structured by two relationships: an object may be a subpart of another (eg an input operation might be a step of a loop, and thus one of its subparts), and an algorithm object may use a data structure object (eg an input operation "uses" the data structure it inputs).

DM represents this structure in a standard network form; nodes represent the objects, and arcs represent the relations subpart/superpart and uses/used-by. Each node also has associated with it a list of questions about the object it represents. (A question asks about some attribute of some specific object. The objects, relations, and questions come from other modules of PSI.)

In order to choose the next question to ask, DM searches the net, starting at the "present topic". The present topic is the object currently being discussed. Determining which object this is is a difficult and important problem in its own right, involving the syntax of the user's sentences as well as the status of the program specification, and has not been seriously dealt with in this work. Instead, some simple heuristics are used, the main one being to assume that most of the time the user will be talking about the object that the system just asked about.

Once the present topic has been chosen, the search proceeds, under control of a set of rules. (The rules are listed in the appendix. See [9] for a discussion of the specific rules.) Each time the search reaches an object, a list of rules is chosen (depending on whether the object is a piece of algorithm or data structure) and these rules are applied in order. Some say to look for a specific kind of question about the current object. Others say to move along some particular kind of arc from the current object, and recursively apply the rules on the object we reach. If no question is found by this recursive application, we come back and continue applying the rules here. If at any point a rule that looks for questions finds one, that question is the one to ask, and the search stops.

This scheme of moving through the net and looking for questions, under control of a set of rules, has proven to be simple and flexible.

A related technique was used in SCHOLAR [3]. SCHOLAR is a CAI system which teaches geography by engaging in a mixed initiative dialogue with the student. Both participants may ask and answer questions. SCHOLAR chooses which question to ask by a random (rather than rule directed) walk on a net which encodes its knowledge about geography. As ultimately envisioned, SCHOLAR would teach in a Socratic manner, that is, by asking a carefully designed sequence of questions. However, the structure of goals and plans in such a dialogue is probably very different from the structure of the net as discussed in [3]. Because of this, a scheme of moving through this net is unlikely to be useful for producing such a sequence of questions.

DM's question ordering behavior has been tested in two ways. First, a log of runs of PSI was surveyed. This log included 42 dialogues which were essentially complete. Each dialogue was checked, both to see if the user complained about the question ordering (there is a comment feature that can be used for such complaints), and also to see if the question order was subjectively acceptable. Except for one instance, later traced to a program bug, DM's behavior was correct. This test was too subjective, however, so a simulated dialogue was recorded, with myself playing the role of PSI and a programmer from outside the PSI group as the user. The inputs DM would have gotten during this dialogue were hand coded and given to DM, and the questions DM chose were compared with those I had chosen. DM had to choose a question at sixteen points, with two to seven questions to choose from. The correct question was chosen at thirteen of these points. An analysis of the errors indicates that they could be removed by some straightforward extensions of the current methodology, particularly by maintaining more history of how the dialogue got to the present topic.

IV. Conclusions

Thus we see that it is advantageous for a system which engages in mixed initiative dialogue to have the reasoning modules produce all the questions they can at each point in the dialogue, and to have a separate dialogue moderator choose which one to ask next. In such a system, the question ordering mechanism is decoupled from the reasoning process, so that either can be modified without changing the other. A given mechanism for selecting one of the proposed questions can be more easily transferred to a system with very different reasoning mechanism. Also, multiple parallel reasoning processes can be used with this scheme.

DM, the dialogue moderator of PSI, represents the program as a simple net of objects and relations. It chooses a question by starting at the node representing the present topic of the dialogue, and searching the net, under control of a set of rules. It is possible to use a simple model of the program, rather than a complex model of goals and plans, because in the program specification task, the participants' goals and plans are so closely tied to the program structure. This general scheme of rule based search is advantageous because it is simple and flexible. These techniques are probably applicable to other settings where the structure of goals and plans can be tied to some simple task related structure.

APPENDIX: Question Choice Rules.

(These are slightly simplified versions of the content of the rules. The actual rules consist of LISP code.)

Rules for Algorithms

A1) Are there questions about the NAME of this object?
A2) Look at all objects that are USED-BY this object.
A3) Are there questions about this object other than EXIT-TEST, PROMPT, or FORMAT?
A4) Are there questions about the PROMPT or FORMAT for this object?
A5) Look at all objects that are SUB-PARTS of this object.
A6) Are there questions about the EXIT-TEST of this object?
A7) Look at all objects that are SUPER-PARTS of this object.

Rules for Data Structures

D1) Look at all objects that are SUB-PARTS of this object.
D2) Are there questions about the STRUCTURE of this object?
D3) Are there OTHER questions about this object?
D4) Look at all objects that are SUPER-PARTS of this object.
D5) Look at all objects that USE this object.

REFERENCES

[1] Bobrow, D., Kaplan, R., Kay, M., Norman, D., Thompson, H., Winograd, T., "GUS, A Frame-Driven Dialogue System." Artificial Intelligence 8 (1977) 155-173.

[2] Brown, G., "A Framework for Processing Dialogue", Technical Report 182, MIT Laboratory for Computer Science, June 1977.

[3] Carbonell, J. R., "AI in CAI: An Artificial Intelligence Approach to Computer-Aided Instruction." IEEE Trans. Man-Machine Syst. 11 (1970) 190-202.

[4] Davis, R., Buchanan, B., Shortliffe, E., Production Rules as a Representation for a Knowledge-Based Consultation Program, Memo AIM-266, Stanford Artificial Intelligence Laboratory, October, 1975.

[5] Green, C., A Summary of the PSI Program Synthesis System, Proceedings of the Fifth International Joint Conference on Artificial Intelligence, Cambridge, Massachusetts, August 1977, 380 - 381.

[6] Grosz, B., The Representation and Use of Focus in a System for Understanding Dialogues, Proceedings of the Fifth International Joint Conference on Artificial Intelligence, Cambridge, Massachusetts, August 1977, 67 - 76.

[7] Lesser, V., Erman, L., A Retrospective View of the HEARSAY-II Architecture, Proceedings of the Fifth International Joint Conference on Artificial Intelligence, Cambridge, Massachusetts, August 1977, 380 - 381.

[8] Mann, W., Man-Machine Communication Research: Final Report, ISI/RR-77-57, USC Information Sciences Institute, February, 1977.

[9] Steinberg, L., A Dialogue Moderator for Program Specification Dialogues in the PSI System, PhD dissertation, Stanford University, in progress.