A Model For Generating Better Explanations

Peter van Beek
Department of Computer Science
University of Waterloo
Waterloo, Ontario
CANADA N2L 3G1

Abstract

Previous work in generating explanations from advice-giving systems has demonstrated that a cooperative system can and should infer the immediate goals and plans of an utterance (or discourse segment) and formulate a response in light of these goals and plans. The claim of this paper is that a cooperative response may also have to address a user's overall goals, plans, and preferences among those goals and plans. An algorithm is introduced that generates user-specific responses by reasoning about the goals, plans and preferences hypothesized about a user.

1. Introduction

What constitutes a good response? There is general agreement that a correct, direct response to a question may, under certain circumstances, be inadequate. Previous work has emphasized that a good response should be formulated in light of the user's immediate goals and plans as inferred from the utterance (or discourse segment). Thus, a good response may also have to (i) assure the user that his underlying goal was considered in arriving at the response (McKeown, Wish, and Matthews 1985); (ii) answer a query that results from an inappropriate plan indirectly by responding to the underlying goal of the query (Pollack 1986); (iii) provide additional information aimed at preventing the user from drawing false conclusions because of violated expectations of how an expert would respond (Joshi, Webber, and Weischedel 1984a, 1984b).

The claim of this paper is that a cooperative response can (and should) also address a user's overall goals, plans, and preferences among those goals and plans. We wish to show that an advice seeker may also expect the expert to respond in light of, not only the immediate goals and plans of the user as expressed in a query, but also in light of (i) previously expressed goals or preferences, (ii) goals that may be inferred or known from the user's background, and (iii) domain goals the user may be expected to hold. If the expert's response does not consider these latter type of goals the result may mislead or confuse the user and, at the least, will not be cooperative.

As one example, consider the following exchange between a student and student-advisor system.

User: Can I enroll in CS 375 (Numerical Analysis)?

System: Yes, but CS 375 does involve a lot of FORTRAN programming. You may find Eng 353 (Technical Writing) and CS 327 (AI) to be useful courses.

The user hopes to enroll in a particular course to help fulfill his elective requirements. But imagine that in the past the student has told the advisor that he has strong feelings about not using FORTRAN as a programming language. If the student-advisor gives the simple response of "Yes" and the student subsequently enrolls in the course and finds out that it involves heavy doses of FORTRAN programming, the student will probably have justifiably bad feelings about the student-advisor. The better response shown takes into account what is known about the user's preferences. Thus the system must check if the user's plan as expressed in his query is compatible with previously expressed goals of the user. The system can be additionally cooperative by offering alternatives that are compatible with the user's preferences and also help towards the user's intended goal of choosing an elective (see response).

Our work should be seen as an extension of the approach of Joshi, Webber, and Weischedel (1984a, 1984b; hereafter referred to as Joshi). Joshi's approach, however, involves only the stated and intended (or underlying) goal of the query, which, as the above example illustrates, can be inadequate for avoiding misleading responses. Further, a major claim of Joshi is that a system must recognize when a user's plan (as expressed in a query) is sub-optimal and provide a better alternative. However, Joshi leaves unspecified how this could be done. We present an algorithm that produces good responses by abstractly reasoning about the *overall* goals and plans hypothesized of a user. An explicit model of the user is maintained to track the goals, plans, and preferences of the user and also to record some of the background of the user pertinent to the domain. Together these provide a more general, extended method of computing non-misleading responses. Along with new cases where a response must be modified to not be misleading, we show how the cases enumerated in (Joshi 1984a) can be effectively computed given the model of the user. We also show how the user model allows us to compare alternatives and select the better one, all with regards to a specific user, and how the algorithm allows the responses to be computed in a domain independent manner. In summary, computing a response requires, among other things, the ability to provide a correct, direct answer to a query; explain the failure of a query; compute better alternatives to a user's plan as expressed in a query; and recognize when a direct response should be modified and make the appropriate modification.

2. The User Model

Our model requires a database of domain dependent plans and goals. We assume that the goals of the user in the immediate discourse are available by methods such as specified in (Allen 1983; Carberry 1983; Litman and Allen 1984; Pollack 1984, 1986). The model of a user contains, in addition to the user's immediate discourse goals, his background, higher domain goals, and plans specifying how the higher domain goals will be accomplished. In the student-advisor domain, for example, the user model will initially contain some default goals that the user can be expected to hold, such as avoiding failing marks on his permanent record. It will also contain those goals of the user that can be inferred or known from the system's knowledge of the user's background, such as the attainment of a degree. New goals and plans will be added to the model (e.g. the student's preferences or intentions) as they are derived from the discourse. For example, if the user displays or mentions a predilection for numerical analysis courses this would be installed in the user model as a goal to be achieved.

3. The Algorithm

Explanations and predictions of people's choices in everyday life are often founded on the assumption of human rationality. Allen's (1983) work in recognizing intentions from natural language utterances makes the assumption that "people are rational agents who are capable of forming and executing plans to achieve their goals" (see also Cohen and Levesque 1985). Our algorithm reasons about the user's goals and plans according to some postulated guiding principles of action to which a reasonable agent will try to adhere in deciding between competing goals and methods for achieving those goals. If the user does not "live up" to these principles, the response generated by the algorithm will include how the principles are violated and also some alternatives that are better (if they exist) because they do not violate the principles. Some of these principles will be made explicit in the following description of the algorithm (see van Beek 1986 for a more complete description).

The algorithm begins by checking whether the user's query (e.g. "Can I enroll in CS 375?") is possible or not possible (refer to figure 1). If the query is not possible, the user is informed and the explanation includes the reasons for the failure (step 1.0 of algorithm). Alternative plans that are possible and help achieve the user's intended goal are searched for and presented to the user. But before presenting any alternative, the algorithm, to not mislead the user, ensures that the alternative is compatible with the higher domain goals of the user (step 1.1).

If the query is possible, control passes to step 2.0, where the next step is to determine whether the stated goal does, as the user believes, help achieve the intended goal. Given that the user presents a plan that he believes will accomplish his intended goals, the system must check if the plan succeeds in its intentions (step 2.1 of algorithm). As is shown in the algorithm, if the relationship does not hold or the plan is not executable, the user should be informed. Here it is possible to provide additional unrequested information necessary to achieve the goal (cf. Allen 1983).

In planning a response, the system should ensure that the current goals, as expressed in the user's queries, are compatible with the user's higher domain goals (step 2.2 in algorithm). For example, a plan that leads to the attainment of one goal may cause the non-attainment of another such as when a previously formed plan becomes invalid or a subgoal becomes impossible to achieve. A user may expect to be informed of such consequences, particularly if the goal that cannot now be attained is a goal the user values highly.

The system can be additionally cooperative by suggesting better alternatives if they exist (step 2.3 in algorithm). Furthermore, both the definitions of better and possible alternatives are relative to a particular user. In particular, if a user has several compatible goals, he should adopt the plan that will contribute to the greatest number of his goals. As well, those goals that are valued absolutely higher than other goals, are the goals to be achieved. A user should seek plans of action that will satisfy those goals, and plans to satisfy his other goals should be adopted only if they are compatible with the satisfaction of those goals he values most highly.

```
        Check if original query is possible.

(1.0)   Case 1: { Original query fails }
          Message: No, [query] is not possible because ...
(1.1)     If ( ∃ alternatives that help achieve the intended goal and
             are compatible with the higher domain goals ) then
             Message: However, you can [alternatives]
(1.2)     Else
             Message: No alternatives

(2.0)   Case 2: { Original query succeeds }
          Message: Yes, [query] is possible.
(2.1)     If not ( intended goal ) then
             Message: Warn user that intended goal does not hold and explain why.
             If ( ∃ alternatives that do help achieve the intended goal and
                are also compatible with the higher domain goals ) then
                Message: However, you can [alternatives]
             Else
                Message: No alternatives
(2.2)     Else If ( stated goal of query is incompatible with the higher
             domain goals ) then
             Message: Warn user of incompatibility.
             If ( ∃ alternatives that are compatible with the higher domain
                goals and also help achieve the intended goal ) then
                Message: However, you can [alternatives]
             Else
                Message: No alternatives
(2.3)     Else If ( ∃ alternatives that also meet intended goal but are
             better than the stated goal of the query ) then
             Message: There is a better way ...
          Else
             { No action }
```

Figure 1: Explanation Algorithm

4. An Example

Until now we have discussed a model for generating better, user-specific explanations. A test version of this model has been implemented in a student-advisor domain using Waterloo UNIX Prolog. Below we present an example to illustrate how the algorithm and the model of the user work together to produce these responses and to illustrate some of the details of the implementation.

Given a query by the user, the system determines whether the stated goal of the query is possible or not possible and whether the stated goal will help achieve the intended goal. In the hypothetical situation shown in figure 2, the stated goal of enrolling in CS572 is possible and the intended goal of taking a numerical analysis course is satisfied[1]. The system then considers the background of the user (e.g. the courses taken), the background of the domain (e.g. what courses are offered) and a query from the user (e.g. "Can I enroll in CS572?"), and ensures that the goal of the query is compatible with the attainment of the overall domain goal.

In this example, the user's stated goal of enrolling in a particular course is incompatible with the user's higher

[1] Recall that we are assuming the stated and intended goals are supplied to our model. This particular intended goal, hypothetically inferred from the stated goal and previous discourse, was chosen to illustrate the use of the stated, intended, and domain goals in forming a best response. The case of a conflict between stated and intended goal would be handled in a similar fashion to the conflict between stated and domain goal, shown in this example.

Scenario:

The user asks about enrolling in a 500 level course. Only a certain number of 500 level courses can be credited towards a degree and the user has already taken that number of 500 level courses.

Stated goal:	Enroll in the course.
Intended goal:	Take a numerical analysis course.
Domain goal:	Get a degree.

User:

Can I enroll in CS 572 (Linear Algebra)?

System:

Yes, but it will not get you further towards your degree since you have already met your 500 level requirement. Some useful courses would be CS 673 (Linear Programming) and CS 674 (Approximation).

Figure 2: Example from student advisor domain

domain goal of achieving a degree because several preconditions fail. That is, given the background of the user the goal of the query to enroll in CS572 will not help achieve the domain goal. Knowledge of the incompatibility and the failed preconditions are used to form the first sentence of the system's response.

To suggest better alternatives, the system goes into a planning stage. There is stored in the system a general plan for accomplishing the higher domain goal of the user. This plan is necessarily incomplete and is used by the system to track the user by instantiating the plan according to the user's particular case. The system considers alternative plans to achieve the user's intended goal that are compatible with the domain goal. For this particular example, the system discovers other courses the user can add that will help achieve the higher goal.

To actually generate better alternatives and to check whether the user's stated goal is compatible with the user's domain goal, a module of the implemented system is a Horn clause theorem prover, built on top of Waterloo Unix Prolog, with the feature that it records a history of the deduction. The theorem prover generates possible alternative plans by performing deduction on the goal at the level of the user's query. That is, the goal is "proven" given the "actions" (e.g. enroll in a course) and the "constraints" (e.g. prerequisites of the course were taken) of the domain. In the example of figure 2, the expert system has the following Horn clauses in its knowledge base:

course (cs673, numerical)
course (cs674, numerical)

Figure 3 shows a portion of the simplified domain plan for getting a degree. Consider the first clause of the *counts_for_credit* predicate. This clause states that a course will count for credit if it is a 500 level course and fewer than two 500 level course have already been counted for credit (since in our hypothetical world, at most two 500 level courses can be counted for credit towards a degree). The second clause is similar. It states the conditions under which a a 600 level course can be counted for credit.

```
get_degree(Student, Action) <-
        receive_credit(Student, Course, Action);
get_degree(Student, []);

receive_credit (Student, Course, Action) <-
        counts_for_credit (Student, Course),
        enrolled (Student, Course, credit, Action),
        do_work (Student, Course),
        passing_grade (Student, Course);
receive_credit (Student, Course, Action) <-
        enrolled (Student, Course, credit, []),
        enrolled (Student, Course, incomplete, Action),
        complete_work (Student, Course),
        passing_grade (Student, Course);

counts_for_credit (Student, Course) <-
        is_500_level (Course),
        500_level_taken (Student, N), lt (N, 2);
counts_for_credit (Student, Course) <-
        is_600_level (Course),
        600_level_taken (Student, N), lt (N, 5);
```

Figure 3: Simplified domain plan for course domain.

The domain plan is then employed to generate an appropriate response. The clauses can be used in two ways: (i) to return an action that will help achieve a goal and (ii) to check whether a particular action is a possible step in a plan to achieve a goal. In the first use, the *Action* parameter is uninstantiated (a variable), the theorem prover is applied to the clause, and, as a result, the *Action* parameter is instantiated with an action the user could perform towards achieving his goal. In the second case, the *Action* parameter is bound to a particular action and then the theorem prover is applied. If the proof succeeds, the particular action is a valid step in a plan; if the proof fails, it is not valid and the history of the deduction will show why. In this example, enrolling in CS673 is a valid step in a plan for achieving a degree.

Recall that the system will generate alternative plans even if the user's query is a valid plan in an attempt to find a better solution for the user. The (possibly) multiple alternative plans are then potential candidates for presenting to the user. These candidates are pruned by ranking them according to the heuristic of "which plan would get the user further towards his goals". Thus, the better alternatives are the ones that help satisfy multiple goals or multiple subgoals[2]. One way in which the system can reduce alternatives is to employ previously derived goals of the user such as those that indicate certain preferences or interests. In the course domain, for instance, the user may prefer taking numerical analysis courses. For the example in figure 2, the suggested alternatives of CS673 and CS674 help towards the user's goal of getting a degree and the user's goal of taking numerical analysis courses and so are preferable[3].

5. Joshi Revisited

The discussion in the previous section showed how our model can recognize when a user's plan is incompatible with his domain goals and present better alternative plans that are user-specific. Here we present examples of how our model can generate the responses enumerated by Joshi. The examples further illustrate how the addition of the user's overall goals allows us to compare and select better alternatives to a user's plan.

Figure 4 shows two different responses to the same question: "Can I drop CS 577?" The student asking the question is doing poorly in the course and wishes to drop it to avoid failing it. The goals of the query are passed to the Prolog implementation and the response generated depends on these goals, the information in the model of the user, and on external conditions such as deadlines for changing status in a course. For example purposes, the domain information is read in from a file (e.g. consult(example_1)). Figure 3 shows the clausal representation of the domain goals and plans used in this example (the representations for the goal of avoiding a failing mark are not shown but are similar).

[2] Part of our purpose is to characterize domain independent criteria for "betterness". Domain dependent knowledge could also be used to further reduce the alternatives displayed to the user. For example, in the course domain a rule of the form: "A mandatory course is preferable to a non-mandatory course", may help eliminate presentation of certain options.

[3] Note that in this example the user's intended goal also indicates a preference. Other user preferences may have been previously specificed; these would be used to influence the response in a similar fashion.

```
%
% Can Ariadne drop CS 577?
%
?consult(example_1);

?       query(change_status(ariadne, 577, credit, nil),
              not_fail(ariadne, 577, Action));

Yes, change_status(ariadne, 577, credit, nil) is possible.
But, not_fail(ariadne, 577, _461) is not achieved since...
        is_failing(ariadne, 577)
However, you can ...
        change_status(ariadne, 577, credit, incomplete)
This will also help towards receive_credit

%
% Can Andrew drop CS 577?
%
?consult(example_2);

?       query(change_status(andrew, 577, credit, nil),
              not_fail(andrew, 577, Action));

Yes, change_status(andrew, 577, credit, nil) is possible.
But, there is a better way ...
        change_status(andrew, 577, credit, incomplete)
Because this will also help towards receive_credit
```

Figure 4: Sample responses

Example 1: In this example, the stated goal is possible, but it fails in its intention (dropping the course doesn't enable the student to avoid failing the course). This is case 2.1 of the algorithm. The system now looks for alternatives that will help achieve the student's intended goal and determines that two alternative plans are possible: the student could either change to audit status or take an incomplete in the course. The plan to take an incomplete is presented to the user because it is considered the best of the two alternatives; it will allow the student to still achieve another of his goals: receiving credit for the course.

Example 2: Here the query is possible (the student can drop the course) and is successful in its intention (dropping the course does enable the student to avoid failing the course). The system now looks for a better alternative to the student's plan of dropping the course (case 2.3 of algorithm) and determines an alternative that achieves the intended goal of not failing the course but also achieves another of the student's domain goals: receiving credit for the course. This better alternative is then presented to the student.

6. Future Work and Conclusion

Future work should include incorporation of existing methods for inferring the user's goals from an utterance and also should include a component for mapping between the Horn clause representation used by the program and the English surface form.

An interesting next step would be to investigate combining the present work with methods for varying an explanation from an expert system according to the user's knowledge of the domain. In some domains it is desirable for an expert system to support explanations for users with widely diverse backgrounds. To provide this support an expert system should also tailor the content of its explanations according to the user's knowledge of the domain. An expert system currently being developed for the diagnosis of a child's learning disabilities and the recommendation of a remedial program provides a good example (Jones and Poole 1985). Psychologists, administrators, teachers, and parents are all potential audiences for explanations. As well, members within each of these groups will have varying levels of expertise in educational diagnosis. Cohen and Jones (1986; see also van Beek and Cohen) suggest that the user model begin with default assumptions based on the user's group and be updated as information is exchanged in the dialogue. In formulating a response, the system determines the information relevant to answering the query and includes that portion of the information believed to be outside of the user's knowledge.

We have argued that, in generating explanations, we can and should consider the user's goals, plans for achieving goals, and preferences among these goals and plans. Our implementation has supported the claim that this approach is useful in an expert advice-giving environment where the user and the system work cooperatively towards common goals through the dialogue and the user's utterances may be viewed as actions in plans for achieving those goals. We believe the present work is a small but nevertheless worthwhile step towards better and user-specific explanations from expert systems.

7. Acknowledgements

This paper is based on thesis work done under the supervision of Robin Cohen, to whom I offer my thanks for her guidance and encouragement. Financial support is acknowledged from the Natural Sciences and Engineering Research Council of Canada and the University of Waterloo.

8. References

Allen, J. F., 1983, "Recognizing Intentions from Natural Language Utterances," in **Computational Models of Discourse**, Ed. M. Brady and R. C. Berwick, Cambridge: MIT Press.

Carberry, S., 1983, "Tracking User Goals in an Information-Seeking Environment," *Proceedings of National Conference on Artificial Intelligence*, Washington, D.C.

Cohen, P. R. and Levesque, H. J., 1985, "Speech Acts and Rationality," *Proceedings of ACL-85*, Chicago, Ill.

Cohen, R. and Jones, M., 1986, "Incorporating User Models into Expert Systems for Educational Diagnosis," Department of Computer Science Research Report CS-86-37, University of Waterloo, Waterloo, Ont.

Jones, M. and Poole, D., 1985, "An Expert System for Educational Diagnosis Based on Default Logic," *Proceedings of the Fifth International Conference on Expert Systems and Their Applications*, Avignon, France.

Joshi, A., Webber, B., and Weischedel, R., 1984a, "Living up to Expectations: Computing Expert Responses," *Proceedings of AAAI-84*, Austin, Tex.

Joshi, A., Webber, B., and Weischedel, R., 1984b, "Preventing False Inferences," *Proceedings of COLING-84, 10th International Conference on Computational Linguistics*, Stanford, Calif.

Litman, D. J. and Allen, J. F., 1984, "A Plan Recognition Model for Subdialogue in Conversations," University of Rochester Technical Report 141, Rochester, N.Y.

McKeown, K. R., Wish, M., and Matthews K., 1985, "Tailoring Explanations for the User," *Proceedings of IJCAI-85*, Los Angeles, Calif.

Pollack, M. E., 1984, "Good Answers to Bad Questions: Goal Inference in Expert Advice-Giving," *Proceedings of CSCSI-84*, London, Ont.

Pollack, M. E., 1986, "A Model of Plan Inference that Distinguishes Between the Beliefs of Actors and Observers," *Proceedings of ACL-86*, New York, N.Y.

van Beek, P., 1986, "A Model for User-Specific Explanations from Expert Systems," M. Math thesis, published as Department of Computer Science Research Report CS-86-42, University of Waterloo, Waterloo, Ont.

van Beek, P. and Cohen, R., 1986, "Towards User-Specific Explanations from Expert Systems," *Proceedings of CSCSI-86*, Montreal, Que.