Exploiting Problem Solving to Select Information to Include in Dialogues between Cooperating Agents

Elise H. Turner
Computer Science Department
University of New Hampshire
Durham, New Hampshire 03824
eht@cs.unh.edu

Abstract

When agents cooperate to solve complex problems in the real-world, they must choose which information to communicate from the mass of information that might affect the problem. A speaker should communicate the information that will be most helpful to the other agent. However, the speaker may not have a great deal of knowledge about the other. In addition, the speaker is also involved in reasoning about the collaborative problem solving task. So, processing that is done solely to select information will be taken from the resources available to work on the primary problem.

In this paper, we present preliminary work on a new approach to selecting information that should be included in a dialogue. Our approach uses the speaker's knowledge of its own problem solving to determine how useful some piece of information might be to other agents. Consequently, the speaker can make its decision to include information in the dialogue using no additional knowledge and few additional computational resources beyond those required to reason about the primary problem solving task. We suggest heuristics which translate problem solving into estimates of how useful information will be for others.

Introduction

Information is one of the most important resources that can be shared by cooperating agents. Since each agent has only partial knowledge of the world, agents can build a more complete representation for problem solving if they exchange this knowledge. By communicating results as they reason separately about the cooperative task, each agent gives the other insight into its problem solving strategy as well as allowing the other to avoid redundant problem solving. In complex, real-world domains, there is a great deal of information that is pertinent to any task — so much information that it cannot all be communicated during problem solving. Therefore, an agent must choose the most helpful information to communicate to others.

We are particularly interested in how agents can select information while working within the constraints of cooperative problem solving dialogues. First, agents have only a partial view of each other. Therefore, they cannot rely on detailed knowledge of another to suggest information that will be helpful. Second, communication is in service of some primary problem solving task. Since resources used solely to process communication will be taken from those available to perform the primary problem solving task, processing used solely to produce and understand messages should be kept to a minimum.

In this paper, we suggest how a speaker can select information by relying solely on knowledge that is *available* and *easily accessible*. We focus on an often-overlooked resource, knowledge of the speaker's own problem solving. If both agents are addressing the same goal and have similar reasoning strategies and capabilities, information that is useful to one of them will be useful to the other. So, if an agent can determine the role of some information in its own problem solving, it gains insight into how useful that information will be to others. Since the agent simply monitors its own reasoning, determining the usefulness of information requires *no additional knowledge* and *little computational effort* beyond that required by the agent to reason about the problem solving task.

We are particularly interested in exploiting problem solving to allow computer systems to select information to include in dialogues. To achieve this, we must create heuristics which translate problem solving into a *usefulness rating* for information. The usefulness rating should indicate both how pertinent the information is to the current problem and how difficult the information will be for others to infer. The usefulness rating can be integrated with existing natural language processing techniques to select information to include in problem solving dialogues.

Using Problem Solving to Select Information: An Example Dialogue

Consider the following fragment of a constructed problem solving dialogue. The discourse takes place between acquaintances planning a party for a mutual friend. Both conversants have only limited knowledge of the guest of honor and of each other. They do not know what the other knows or how the other will approach the problem. We assume that the conversants are pursuing only the goal of planning the party and related subgoals. Under these circumstances, their conversation might include the following:

Angela-1:	When should we have the party?
Bob-2:	Fred and Mary go to dinner every Friday night.
Angela-3:	What about Sunday afternoon?
Bob-4:	That's fine.
	Fred really likes *Joey's*, and they serve a great Sunday brunch.
Angela-5:	We could go there.
Bob-6:	That should cost us each about $150.

Now, let us consider how the problem solving of each individual may have motivated their utterances. From the information that they have considered while reasoning about the

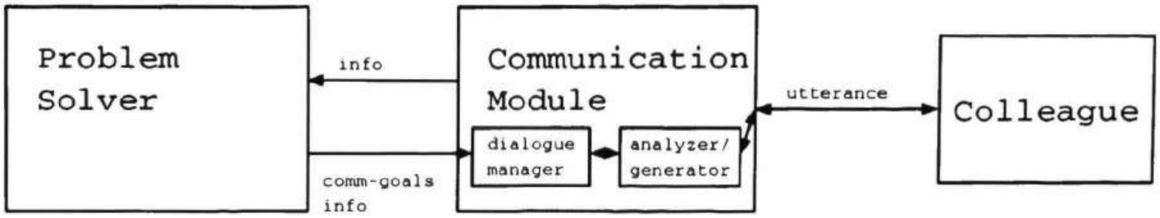

Figure 1: Our architecture for problem solving and communication.

cooperative problem, they will only communicate that which has a high usefulness rating. Speakers must also be able to integrate these usefulness ratings with other available knowledge about language processing, such as, linguistic conventions, hearer's beliefs, and speaker's and hearer's goals. For example, Bob's "that's fine" results from Bob believing that the question "What about Sunday afternoon?" requires a response.

Bob is also trying to find a response to a question, "When should we have a party?", when he considers his knowledge that Fred and Mary are busy on Friday nights. This information rules out Friday as a possible time for the party. By telling Angela, she can also rule out Friday nights. Bob's decision to include this information might result from a heuristic giving high usefulness ratings to information that serves to constrain possible solutions. Specifically, Bob communicates information that leads to a plan failure so that Angela will not waste time considering plans which set Friday night as the time of the party.

In the next utterance, Angela also helps to constrain the solution. Here, Angela communicates a partial solution to the problem by suggesting Sunday afternoon for the time. If the suggestion is accepted, Angela and Bob can concentrate only on plans for the party which can take place at this time.

Next, as Bob continues to think about the party, he begins to think of where the party might be held. Bob's next utterance, "Fred really likes *Joey's*, and they serve a great brunch," might be considered to be a partial solution giving the location. However, Bob may not have yet decided that *Joey's* would be a good place to have the party. He may have included this information because it impacts the problem and is not likely to be known by Angela. For example, if Fred had only gone to this restaurant once, Bob may believe that Fred has never told Angela about it. If a complete model of Angela's beliefs were available, Bob could use it to determine if she already knew about the restaurant. If Bob does not have a good model of Angela's beliefs, he needs a heuristic which can determine when information is unlikely to be known by another agent.

Finally, in Bob-6, Bob tells Angela the cost of the party. This calculation may have been difficult for Bob to make, and, by telling Angela the results, he saves her the trouble of repeating his calculation. This would suggest heuristics which give a high usefulness rating to information that relies on procedures which are difficult for the hearer, or information that is the result of a long chain of inferences.

From Problem Solving to Utterances

The architecture for problem solving and communication that we use is shown in Figure 1. During problem solving, new goals arise which can be achieved through communication. The dialogue manager chooses appropriate discourse schemas for achieving these goals within a coherent, conventional conversation (Turner, 1990; Turner, 1992). These schemas form a template which guides the dialogue manager. In our architecture, usefulness ratings cause the problem solver to activate a communication goal to say some information item when its usefulness rating is higher than some threshold. Usefulness ratings can also be incorporated into other types of discourse systems, as discussed below.

In any system, the most difficult step is finding the usefulness rating for some information item. To do this, heuristics must be found which can convert problem solving knowledge into usefulness ratings. The exact heuristics and their implementation will depend on the reasoners being used by the agents. We are currently implementing heuristics for NONLIN (Ghosh *et al.*, 1992) to test our method with a specific problem solver. The heuristics used in that implementation are derived from our findings about information that is included in dialogues and an initial set of heuristics. We devote the rest of this section to a discussion of these preliminary results.

Information that Should be Communicated.

Before we suggest specific heuristics, we must classify the types of information that should be communicated in terms of the agents' problem solving. We have examined both naturally occurring dialogues and successful communication strategies from distributed artificial intelligence to produce basic categories of uses for information. Information is useful if it helps the agents to do at least one of the following:

Avoid redundant problem solving. Experiments by Durfee, Lesser and Corkill (Durfee *et al.*, 1987) suggest that avoiding redundancy in problem solving reduces the overall time needed to perform the primary problem solving task. Agents can avoid redundant processing by communicating partial solutions. This is particularly useful if many inferences were needed to find the partial solution, or if several potential solutions had to be compared. Agents can also avoid redundant problem solving by including utterances which explicitly assign a subtask to an agent; for example, "I'll figure out how much this will cost."

Pool relevant facts. As agents separately experience the world, they develop unique models for it. When agents join forces to solve a problem, they can exchange this individual information. Information is only useful if it is relevant to the problem at hand. In addition, the less likely that it is known to another agent, the more useful it will be to communicate that information. Agents may also communicate information that another may know but may not be applying to this problem. For example, an agent may remind another of pertinent information that happened in the distant past or that is not directly related to information that has already been communicated. In the latter case, the agent often also communicates the relevance of the information.

Prune the search space. For complex problems, the search space may be huge, including many plans for solving the problem and many constraints on each of those plans. Agents can help each other to prune this tree in three important ways. First, they can communicate partial solutions so that only a single solution for some subgoal needs to be considered. Second, they can communicate failures so that the other agents can avoid solutions which include those failures. Third, the agent may indicate its problem solving strategy by explicitly suggesting an approach to the problem or by involving the other agents in a discussion of a subgoal. For example, if two agents are planning an evening out, one may indicate its overall-strategy with "How about dinner and a movie?" and later involve the other in the discussion of a subgoal with "Which movie should we see?"

The Heuristics

In order for the heuristics to be useful, they must rely on specific, easily monitored aspects of problem solving. They also must capture the difference between the speaker and a hearer who might share general problem solving strategies, but not specific information. In addition, since information is often used in a variety of ways during problem solving, we must be able to combine heuristics. How the information was used in problem solving, knowledge structures where it was found, and the differences between problem solvers may all influence the usefulness of information. To allow heuristics to be easily combined, we state the heuristics in terms of increasing or decreasing a usefulness rating.

Problem solving information needed Our heuristics draw from three specific types of problem solving information:

- the *knowledge structure* that a given piece of information came from
- the *type and difficulty of reasoning* used to reach or infer that information
- the *effect of that information* on performing the task

The knowledge structure that contains a piece of information, and the knowledge about the information contained in that structure, is important because it may indicate how likely it is to already be known by the hearer. If the knowledge is tagged as an instance, or if the knowledge was learned from a private source, the hearer is unlikely to know this information. Heuristics based on the knowledge structure may be in opposition to ones based on reasoning. For example, position in the knowledge structure indicates that the less general information is likely to be more useful to communicate, despite the fact that it is easier to infer if known.

The problem solving which leads an agent to some piece of information also provides clues to the usefulness of that information to others. By considering only information that an agent has come upon while problem solving, we are guaranteed that the information will in some way be pertinent to the current task. The mere fact that a piece of information was used during problem solving is especially helpful in "brainstorming" situations where agents have not yet organized the problem into assigned subproblems.

It will also be important to know the details of the problem solving. If the new information will be "difficult" for another to infer, communicating it will allow the receiver to avoid that work. If agents process information in the same way, the sender can find how difficult an inference will be for a receiver simply by examining its own processing. The number of inferences made and the alternatives on the decision path, for example, would indicate how difficult it would be for another to infer the same information.

Another important insight from the senders' problem solving is the role that the information played. For example, did the information warn of a failure that should be avoided, provide a constraint that could help focus problem solving, or contribute a possible partial solution? If so, the information is likely to be more valuable to another agent working on the same problem.

An initial set of heuristics

1. **The length of the chain of reasoning from shared knowledge to the information being considered should be directly proportional to the usefulness rating of the information.** This heuristic helps to reduce redundant problem solving by measuring the difficulty of inferring information. When agents have similar reasoning capabilities, if information is difficult for one agent to infer, it will be difficult for another to infer. By communicating the information, the receiver is saved that effort.

   This heuristic is implemented by simply counting the number of inferences made by the speaker from some information item that is known to both speakers. An information item is assumed to be known to both speakers if it has been communicated. In addition, if an agent has a model of another that includes information known to that agent, that information can be considered to be shared.

2. **The more choices made in the course of finding a partial solution, the higher its usefulness rating.** This heuristic prunes the search space by focusing the collaborators on the same subproblem. If reasoning can proceed in only one way, it is easy for one agent to re-create the reasoning of another. However, each time an agent makes a choice, the other agent may have to try all possible choices before arriving at the same conclusion. Therefore, to allow other agents to follow its reasoning, an agent should communicate the results of its choices. This heuristic can be implemented simply by counting the number of choices that an agent makes, counting both the choice points and the number of possible values at each of these points. As the number of choices increases, the usefulness rating also increases. This means the usefulness rating is low when there are few choices and the hearer could try all possibilities. The rating gets higher as the hearer's work to reproduce the speaker's results increases.

3. **The more often a given information item has been used for problem solving, the higher its usefulness rating.** This heuristic helps the agents to pool relevant facts. The agent can implement this heuristic by simply increasing a counter each time the information is used in problem solving. This heuristic is meant to point to salient features of an entity. If the information is usually important, it is likely to be important for this specific task. This heuristic is also applicable to listing attributes in a description or events in a narrative.

4. **Information retrieved from a case, or a specific instance in a ISA-hierarchy should be given a high usefulness rating.** This heuristic also helps to pool relevant facts. Information in a case or instance is known only to agents who have experienced the case or instance or who have been told about it. The information cannot be inferred simply by following links to more general knowledge. To implement this heuristic, the agent increases the usefulness rating on any information that is retrieved from a case or instance during problem solving.

Initial Implementation in NONLIN

We are currently testing our approach by implementing heuristics for NONLIN (Tate, 1977). When developing the heuristics for a specific planner, we must adapt the general heuristics described above to the specific data structures and reasoning used by that planner. We are just beginning this process for NONLIN. In this section, we describe the original heuristics that we are developing for NONLIN. We have made several assumptions for this initial implementation that we will loosen as we gain experience in this testbed.

The Implementation

To implement our heuristics we are modifying the implementation of NONLIN available from the University of Maryland (Ghosh *et al.*, 1992). Each NONLIN agent runs in its own Common LISP environment. The agents communicate through UNIX ports and can communicate across the internet.

Our domain is planning activities for an evening out. Currently, the plans are fairly simple and both agents share the same plan library. The agents also share knowledge about sources of entertainment, such as restaurants and movie theaters. The domain is rich enough, however, that we will be able to expand the plans and differentiate the agents knowledge as we continue to experiment.

In NONLIN, an agent uses *schemas* to represent methods for achieving goals. Schemas are associated with plan *nodes* as the agent builds its plan. The agent has a *table of multiple effects (TOME)* which shows how plan steps will interact with each other. The agent imposes an order on some plan steps, or *linearizes* these steps, so that harmful interactions will be avoided. NONLIN agents solve problems hierarchically. A single schema may set the entire (abstract) course of actions. The agent selects a step at random to expand. This way, more and more detail is added until all steps have been planned to the level of primitive actions.

Heuristics for NONLIN

The general heuristics are meant to point to pertinent information that may be difficult to infer. For NONLIN we are implementing the following heuristics which reflect the same philosophy. Currently, we are implementing only three heuristics. For simplicity, two of the heuristics are stated in terms of creating an information item and giving it an initial usefulness rating. This is possible because the heuristics do not currently interact. If these heuristics prove beneficial, they will be rewritten to operate on the usefulness rating when the heuristic set is extended. These heuristics can be seen as combining the more general heuristics so that they can be more easily be applied in NONLIN.

1. **A schema's usefulness rating is directly proportional to the number of schemas which may be used to satisfy a goal.** This heuristic serves two purposes in NONLIN. when agents are working together on a goal, they can prune the search space by focusing on the schema that has been communicated. When agents are working on separate subproblems, the agent is communicating a partial solution by communicating the chosen schema. This heuristic is a specialization of Heuristic #2 above. NONLIN finds all schemas which could be used to achieve a goal before selecting one to use in this attempt at planning. This heuristic, then, is easily implemented by simply counting the schemas that were found.
2. **An information item is created and given a high usefulness rating when two actions that interfere with each other are linearized.** This heuristic helps agents to avoid redundant problem solving. These kinds of linearizations are found by reasoning about the TOME. Since that reasoning is likely to be computationally expensive, we can think of this heuristic as a specialization of Heuristic #1 of the general heuristics. Instead of counting each inference, we simply give the information a high usefulness rating.
3. **An information item is created and given a high usefulness rating when a plan variable is bound. This usefulness rating is increased in direct proportion to the number of possible values.** The first part of this heuristic combines Heuristic #1, Heuristic #3, and Heuristic #4. Values of a variable are often derived from many inferences that have been made throughout the planning process, so the actual chain of inferences used to select a value may be difficult to trace as a side-effect of problem solving. Also, variable bindings are important because they may constrain many steps of the plan. Furthermore, when agents have different knowledge about the world, some possible values for variables may not be known to all agents. The second part of this heuristic is a specialization of Heuristic #2.

Other Approaches to Selecting Information

In this section we discuss two widely-accepted approaches to natural language processing and their implications for selecting information to include in a dialogue. A third method of selecting information, pruning redundant information, is also discussed. For each technique, we suggest how usefulness ratings can be used to select information.

The plan-based approach. Allen and Perrault's (1980) approach to generating "helpful responses," following the *plan-based theory of speech acts* (Cohen & Perrault, 1979), suggests that a speaker must infer a hearer's plan in order to identify helpful information. Unfortunately, the need to infer the questioner's plan keeps this approach from being extended to the general case of selecting information for cooperative problem solving dialogues. First, the speaker may not be able to infer another agent's plan. Without knowledge of a detailed plan, the speaker cannot determine how a specific piece of information will fit into that plan. Second, even if the information is available, inferring the hearer's plan takes a great deal of valuable computational effort. Since this reasoning is redundant with the hearer's planning, the effort is wasted.

We take from this approach the notion that information about another's problem solving should be taken into consideration when selecting information to communicate to that agent, when it is available. Our research addresses the limitations of this approach because it does not attempt to simulate the other's reasoning for the sole purpose of selecting information.

The schema-based approach Another method for determining what should be included in a dialogue is to rely on schemas which capture the conventions of conversation (e.g., (Hovy, 1989; McKeown, 1985)). These schemas cannot fully specify the conversation. Instead, the speaker must make decisions to choose between alternatives in the schemas, to include or omit an optional step, and to select the schemas that will guide the dialogue.

These decisions can be made in two ways. Linguistic knowledge can be used (McKeown, 1985). However, decisions which reflect general usage may not be appropriate for the specific problem solving situation. Another approach is to select schemas based on the problem solving goals of the speaker (Hovy, 1989; Turner, 1990). Our work suggests specific heuristics which can be used to set the priorities of these goals.

Pruning redundant information. In an effort to avoid unwanted implicature (Grice, 1975), a standard technique is to prune information that is known or can be inferred by the hearer. We cannot rely solely on this approach to select information. First, the technique can only be used to prune information from a dialogue, not to include it. Second, this approach seeks to eliminate information that is represented either explicitly or implicitly in the model of an agent. To find the implicitly represented knowledge, the sender must try all reasoning which could potentially lead to the user inferring the information. This is time-consuming and forces an agent to be distracted from the domain task. Finally, there are times when information that the hearer already knows or can infer *is* included in the conversation. Including already-known or inferable information can make problem solving more efficient by serving as a reminder, indicating the speaker's reasoning, or saving the hearer the trouble of inferring the information. In our approach, if information is already known by the hearer, the speaker can decide if its usefulness rating is high enough to warrant reminding the speaker of that information.

Conclusion

In this paper, we have presented preliminary work on a new approach to selecting information that should be included in a dialogue. Our approach uses the speaker's knowledge of its own problem solving to determine how useful some piece of information might be to other agents. This addresses some limitations of previous approaches because it does not call for the speaker to have any knowledge of the other agent and requires only minimal computational effort beyond that needed for the primary problem solving task.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. IRI-9309703. Thanks to Laura Donia for many helpful discussions. Thanks also to Martin Dempsey for the software to support communication through UNIX ports, to Laura Donia for help in implementing the heuristics in NONLIN, and to Collette Matthias and Scott Shaw for creating the original plans for a night out.

References

Allen, J. F. & Perrault, C. R. (1980). Analyzing intention in utterances. *Artificial Intelligence*, 15(3):143–178.

Cohen, P. R. & Perrault, C. R. (1979). Elements of a plan-based theory of speech acts. *Cognitive Science*, 3:177–212.

Durfee, E. H., Lesser, V. R., & Corkill, D. D. (1987). Cooperation through communication in a distributed problem solving network. In Huhns, M., editor, *Distributed Artificial Intelligence*, pages 29–58. Morgan Kaufmann Publishers, Inc., Los Altos, CA.

Ghosh, S., Hendler, J., Kambhampati, S., & Kettler, B. (1992). NONLIN Common Lisp implementation (v1.2): User manual. Computer Science Dept., University of Maryland

Grice, H. P. (1975). Logic and conversation. In Cole, P. & Morgan, J. L., editors, *Syntax and Semantics*, volume 3. Academic Press, New York.

Hovy, E. H. (1989). Approaches to the planning of coherent text. Technical Report ISI/RS-89-245, Information Sciences Institute. Presented at the 4th International Workshop on Text Generation

McKeown, K. R. (1985). *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text.* Cambridge University Press, New York.

Tate, A. (1977). Generating project networks. In *Proceedings of IJCAI–77*, pages 888–893.

Turner, E. H. (1990). Integrating intention and convention to organize problem solving dialogues. Technical Report GIT–ICS–90/02, School of Information and Computer Science, Georgia Institute of Technology. Ph.D. thesis

Turner, E. H. (1992). Organizing dialogue from an incoherent stream of goals. In *Proceedings of the Fifteenth International Conference on Computational Linguistics (COLING-92)*, pages 338–344, Nantes, France.