Mixed-Initiative Dialogue Systems for Collaborative Problem-Solving

George Ferguson and James Allen
University of Rochester
{ferguson,james}@cs.rochester.edu

Abstract

This paper describes our motivation for and approach to the design and implementation of mixed-initiative dialogue systems for collaborative problem solving. Our model allows true mixed-initiative dialogue, where goals (and other joint commitments) can come from either the user or the system. The system's behavior with respect to its commitments is driven by a formal model of collaboration based on a theory of joint intention which also allows the system to interpret the utterances and actions of the user in a uniform manner.

Introduction and Motivation

Our goal is the design and implementation of collaborative assistants that help people solve problems and get things done. We are primarily concerned with systems that interact using spoken natural language dialogue since (a) this is a very efficient means of communication; (b) it requires little or no training; and (c) it gives us the greatest insight into the nature of human communication and collaboration. Other modalities such as graphical user interfaces and visualizations such as graphs and tables are also possible, with some caveats that we elaborate towards the end of this paper.

A good assistant is necessarily one that can not only respond to your initiatives but can also take initiative itself. We've all had the bad experience of working with someone who had to be told everything they needed to do. Similarly, a system that rigidly controls the dialogue, such as a telephone menu system, can hardly be considered to be working with you, much less for you. We like Eric Horvitz' characterization of mixed-initiative systems:

> I shall use the phrase to refer broadly to methods that explicitly support an efficient, natural interleaving of contributions by users and automated services aimed at converging on solutions to problems.(Horvitz 2000)

The standard motivations for mixed-initiative systems center on making the most of the different abilities of the user and the system. Who would argue with this?

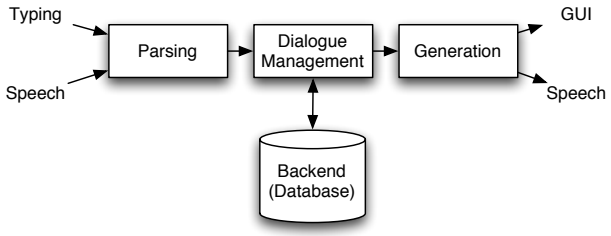

Figure 1: Pipelined dialogue system

We also want to build systems that combine human and machine abilities to solve harder problems faster. But from our dialogue perspective, a crucial motivation for building mixed-initiative systems is to get beyond the "pipeline" architecture of traditional dialogue systems. To explain this, we have to describe a straw man dialogue system architecture briefly. This is shown in Figure 1.

Uninterpreted input arrives at the system through one or more channels and modalities. The primary interpretation of the input is typically some kind of parsing, although this can range from keyword spotting to semantic network activation to full-blown linguistic parsing of various sorts. The interpreted input is used to drive a dialogue through a mystical component that is often labeled "dialogue management", about which more below. The "dialogue manager" interacts with some back-end data sources, typically a database, and constructs the content of the system's response. This is realized through some content-to-language translation, often template-based or embedded in the content itself, and then communicated to the user.

So what's wrong with this? The main problem is that it provides no clear opportunity for mixing initiative. Typically, as the diagram emphasizes, user input triggers the system's execution, requiring that the user be in charge. The system can be in charge to the extent that it somehow gets to ask the first question (perhaps after "connecting" with the user). And of course, since the system might ask a question, to which the user might reply with a clarification question of their own, the "dialogue manager" might be able to track these

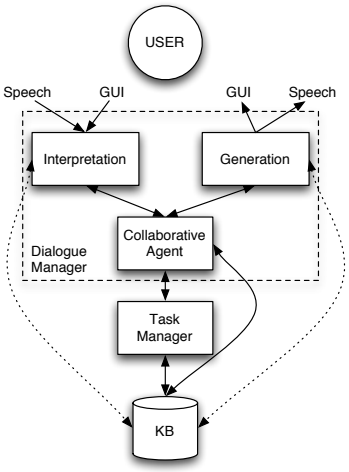

Figure 2: Architecture of a collaborative dialogue agent

short-term transfers of initiative related to turn-taking. But there is no clear notion that the system is doing a number of things, only one of which is the dialogue with the user. Although everything the system does may be in some way related to the user's needs (it is an assistant, after all), a good assistant does things beyond what it was explicitly told to do. In a nutshell, a standard dialogue system is not an *agent*.

Architecture of a Collaborative Dialogue Agent

To address this, we have been developing a model of collaborative dialogue behavior that treats both the user and the system explicitly as agents in a strong BDI sense (Rao & Georgeff 1991). Agents have goals that drive their behavior. In the case of collaborative behavior, agents also make commitments that constrain their behavior. It has been noted some time ago (Pollack 1991; 1992) that joint commitments allow agents to depend on each other and to filter alternative courses of action efficiently. Our model describes how the need to make joint commitments drives the dialogue behavior of a collaborative agent.

First let's look at an alternative to the architecture described previously (see Figure 2). This architecture differs from that of Figure 1 in three important ways. First, dialogue is managed by a subsystem that communicates with a *Task Manager*[1] that is in overall control of the system's agency. The nature of this communication is the key to the approach and will be described shortly. Second, while the components of the dialogue subsystem seem to duplicate the original pipeline, in fact the Interpretation, Collaboration, and Generation components operate autonomously and asynchronously, as we have described elsewhere (Allen, Ferguson, & Stent 2001). Third, the system is backed by a knowledge base shared among components that includes beliefs, desires, and intentions concerning both the system itself and its user. This both provides them with the knowledge they need in order to function, and also provides a form of blackboard-style communication between knowledge-based components. This knowledge base is assumed to be able to represent and reason about the beliefs, desires, and intentions of both the system and the user. While this is itself an interesting and challenging problem, we will not focus on it here, but rather assume some BDI basis for the system's knowledge. In the examples that follow, the modalities `Bel`, `Des`, and `Commit` denote belief, desire, and commitment (intention), respectively.

In addition to the relationships between components, we need to specify what information these are exchanging. As you would expect for a project based on spoken natural language, we have been developing an extensive representational framework for distilling the content of natural language utterances into forms about which the system can reason. Some of this may be evident in the examples that follow, but by and large it is also not the focus of this paper.

Instead, we want to focus on the communication between the Collaboration component of the dialogue subsystem and the Task Manager because this is key to understanding how initiative is handled. We are quite specific about this:

- User initiative is handled as a *suggestion* to the Task Manager that it make certain commitments. Asynchronously, the Task Manager's acceptance or rejection of the suggestion drives the system's collaborative dialogue behavior.
- System initiative is handled by the Task Manager giving the dialogue system a *collaborative goal* to achieve. Achieving this goal drives the system's dialogue behavior until, asynchronously (and possibly incrementally), the goal has been achieved (or abandoned).

We feel that this protocol provides the necessary level of autonomy for the dialogue components while maintaining overall control of the system's behavior (and in particular its commitments) in the Task Manager.[2]

User Initiative

Let's start exploring the model with an example of user initiative, since this is closest to what a traditional dialogue system would support. Suppose the user says: "I want to purchase an LCD projector for my class."

Without dwelling on the internal details of interpretation, which are beyond the scope of this paper, it is worth observing that there are three possible interpretations of this utterance (in all three cases, `PURCHASE123`

[1] A phrase due to Karen Myers

[2] There is no reason in principle why these two levels couldn't be implemented using a common execution framework. The important things are the separation of responsibilities and the restricted form of communication between the two agents.

will be newly-defined to be a task of type *Purchase*, whose object is an LCD projector, `PROJ123`, *etc.*):

1. It could be a direct report of a want or need:

   ```
   (report USER SYS
     (Des USER (Done PURCHASE123)))
   ```

   In this case, a suitable response might be "OK," and the fact about the user's desires would be recorded in the system's KB.

2. It could be a statement of a goal that the user is pursuing independently:

   ```
   (report USER SYS
     (Commit USER (Done PURCHASE123)))
   ```

   A suitable response to this might be "Good luck with that," and again the system might record this information about the user's goals in the KB.

3. Or it could be a proposal that this be adopted as a joint goal:

   ```
   (propose USER SYS
     (Commit (SYS USER) (Done PURCHASE123)))
   ```

   This is the interpretation that drives collaboration.

These interpretations are evaluated using the same model that drives the system's own communicative behavior. That is, it evaluates the user's utterances by considering whether it would have performed a similar act, given the current state. The interpretation subsystem will decide among these, using dialogue context, user model, reasoner support, and heuristics such as preferring the collaborative interpretation whenever possible on the grounds that the system is an assistant.

The mechanism for handling user proposals is, in general, to suggest that the Task Manager commit to them. In the current example, assuming the collaborative interpretation of the user's utterance, this is:

```
(suggest (Commit (SYS USER) (Done PURCHASE123)))
```

Crucially, if the Task Manager adopts the goal, then not only is performing the purchase acceptable from its perspective, but also the system is now committed to performing the joint action. That is, accepting a suggestion is an atomic "test-and-set."

In deciding whether to adopt the commitment (goal), the Task Manager needs to be able to reason about the suitability of what the user has proposed to determine whether or not it makes sense before taking it on as a goal. So, for example, suppose that the user requests the LCD projector at 9:00 a.m. Then at 5:00 p.m. she requests one again. Given that LCD projectors are purchased very rarely, it may be that the system should double-check with the user as to whether she wants to buy a second projector, or whether she simply forgot that she already asked the system to get it (this would be the sort of proactive help that one would expect from a good assistant that understands your intentions). Our architecture explicitly supports this type of system-initiated interaction by separating the process of interpreting the user's utterance (as a proposal in this case) from the decision about adopting it and responding. This flow of information between agents is both more flexible and more natural than the standard dialogue system pipeline.

While the Task Manager considers its options regarding the proposed purchase, the dialogue components are not necessarily idle. For example, the Generation subsystem knows from purely dialogue principles that the system has an obligation to respond to the user's proposal. It can therefore generate appropriate communicative behaviors (for example, taking and holding the turn with aural or visual gestures) even in the absence of the content of the response. And of course, crucial to supporting natural user initiative, if the user continues and offers further information or an additional proposal, the Interpretation components can start processing it asynchronously. This may even result in the addition of information that would affect the Task Manager's decision.

To wrap up the example, eventually the Task Manager accepts the suggestion, and the dialogue subsystem can generate an acceptance of the user's proposal (*e.g.*, "Ok, let's do that"). The system's overall behavior is now driven by the need to achieve the joint goal of (eventually) purchasing an LCD projector.

System Initiative

If the user initiative case seems simple, the system initiative case shows off our model more completely. Let's suppose that the system has the goal of purchasing an LCD projector, whether because the user proposed doing so or by some other means (perhaps our user and her system are responding to a projector order from elsewhere).

Needing to Collaborate

First, the Task Manager needs to know that in order to (jointly) purchase the projector, certain aspects of the purchase need to be jointly agreed (committed to). This is not as far-fetched as it might sound. The general idea is that in jointly solving certain problems (achieving certain goals), some aspects of the solution need to be agreed jointly.[3] One approach might have the Task Manager incrementally deciding what needs to be joint as it works on achieving the goal of purchasing the projector. Another approach would use general principles and ontological knowledge to "pre-compile" the decisions about collaboration. Or in some domains, it might even make sense to simply encode what are the "necessarily joint" elements of the solution.

Regardless of the details, let's assume that as part achieving the performance of `PURCHASE123`, the system needs to agree with the user about the budget for the purchase. This requirement is represented as:

[3]Indeed, it may be that what needs to be agreed is specifically what needs to be agreed, a possibility that we do consider and are implementing.

```
(Commit-What-Is (SYS USER)
  (the BUDGET of PURCHASE123))
```

That is, the sub-goal or precondition is to have a joint commitment regarding the identity of the budget of our previously-committed-to purchasing.[4]

The Task Manager hands off this goal to the dialogue subsystem to achieve. When the dialogue system accepts the Task Manager's request it starts working towards achieving the goal.

Collaborative Behavior

The Collaborative component of the dialogue subsystem reasons about its (collaborative) goals and how to achieve them via dialogue. To do this, it relies on the BDI state of the system as represented in its KB and a set of reactive procedures. Given the above goal, the Collaborative component would perform the following procedure:

1. If there is a jointly agreed value for the budget, then the goal has been achieved.
2. Otherwise, if the system believes it has already committed to a value for the budget, it will **report** that.
3. Otherwise, if the system desires some value for the itself, then it will **propose** that.
4. Otherwise, if it believes that the user desires some value, then it will **check** that.
5. Otherwise, so far as the system believes, neither party has a preference, so it will **ask** the user (other strategies are possible).

This procedure is essentially a compiled version of the definitions of the speech acts such as in (Cohen & Levesque 1990a; 1990b). Interestingly, we also use these procedures in reverse for recognition of user intent during interpretation.

In this example, let's assume that the first three queries fail, and that the system decides to ask the user about the budget. This will be realized by the generation sub-system as something like "what is the budget of the purchase?". The Generation components are smart about using context to realize communicative acts, but that is not the focus of this paper.

The Collaborative agent has now done all that it can do towards the goal it was given by the Task Manager (although it could be otherwise), and so it suspends work on that goal pending new circumstances.

User Proposals

Suppose the user responds to the system's question with: "Fifteen hundred dollars."

Skipping the details of interpretation, which would include, for example, using the fact that we just asked a question about the budget and checking that $1500 is a valid value for the budget property of `PURCHASE123`, we arrive at the following interpretation (glossing the representational details):

```
(propose USER SYS (PURCHASE123 budget $1500))
```

Now, the standard semantics of the propose communicative act are twofold:

1. The speaker desires the content of the act, in this case that the budget be $1500.
2. The speaker will commit to this if the hearer will also.

As with the user-initiative case described previously, the dialogue subsystem does not make commitments on behalf of the system. Only the Task Manager does that. The Collaborative agent therefore suggests to the Task Manager that it make the commitment:

```
(suggest
  (Commit (SYS USER)
    (PURCHASE123 budget $1500))
```

In this example, the content of the suggestion is that the system commit to making the budget of its goal `PURCHASE123` be $1500. And recall from before that the important thing about the semantics of suggest is that it is an atomic test-and-set. If the answer is yes, then not only is it acceptable to the Task Manager that the budget be $1500, but furthermore it is now the case that, according to the system's KB, the budget is $1500.

Reaching Agreement

Assume for purposes of the example that the Task Manager adopts the suggestion and agrees for its part to make the budget $1500. The Dialogue Manager receives notification of this fact, which causes it to accept the user's proposal by generating an **accept** act, that would be realized as something like "OK."

The crucial next step is to observe that when the Collaboration component executes, it will now notice that there is joint agreement as to the identity of the budget. That is:

```
(Commit-What-Is (SYS USER)
  (the BUDGET of PURCHASE123))
```

It will therefore report to the Task Manager that the goal of knowing what the budget is has been achieved. Furthermore, additional knowledge may have been asserted to the KB during the interaction, either because of extended interactions or during the interpretation process itself. But the important thing about this exchange is that it notifies the Task Manager that the subgoal of reaching agreement about the budget has been completed. If the Task Manager was waiting to continue in pursuit of `PURCHASE123`, then it can now proceed (of course, other models are possible, it might not have been waiting, *etc.*).

[4] We follow (Morgenstern 1991) on "knowing what is," generalized to the BDI modalities like `Commit`, but the details are not relevant here. For this example we have also used a syntax similar to that of KM (Clark & Porter 1997) for this paper, although the real thing is more complex.

Additional Issues

In this section we touch very briefly on some issues that arise in thinking about collaborative assistants and mixed-initiative dialogue systems.

A first question is whether this model applies only to natural language dialogue. The answer is no. Although we are primarily interested in NL dialogue for the reasons we stated at the outset, in fact we think that this model of interaction in terms of collaboration is broadly applicable. However, for our approach to be useful, the interface must meet two requirements. First, to support interpretation, the context displayed or implied by the interface must be made explicit and available for use by the Interpretation and Collaboration components. For example, for a graphical interface, the interface must explicitly represent what is visually salient (rather than simply rendering it), what information is being communicated (rather than just having it in a data structure associated with a widget), and what are the ontological relationships between various elements (rather than their being simply tokens or labels). Second, the actions permitted by the interface must be expressed in terms of communicative acts with semantically meaningful content (rather than simply being tied to programmed callbacks). These two requirements taken together allow non-NL interfaces to be used for collaboration and allow language to be used in place of the interface if desired or required.

Another question that comes up can be paraphrased as "isn't there more to collaboration than dialogue?" The answer is obviously yes. A collaborative system has to be able to reason about its (joint) intentions and how to achieve them. This may sometimes involve dialogue but it might also involve taking other forms of action, such as performing actions on the web or invoking robots to perform physical actions. By basing our approach on a general theory of joint action and commitment, we are ready to work within a system that can do those things in addition to dialogue. Meanwhile, we will keep concentrating on the (many) problems remaining in handling collaboration via dialogue.

Related Work

The semantics of speech acts and the relation to intentions is derived from (Cohen & Perrault 1979; Allen & Perrault 1980). The logic of intentions and commitment is loosely based on (Cohen & Levesque 1990a). The challenge for us has been to apply these principles in a practical system that supports natural language dialogue.

Basing inter-agent collaboration on joint commitments is key to the Shared Plans formalism(Grosz & Sidner 1986; 1990). Collagen (Rich & Sidner 1998) builds on Shared Plans and implements a collaborative assistant that performs actions with and for a user of an on-screen computer application. Rich and Sidner refer to Collagen as an application-independent "collaboration manager", which corresponds to our view of the separate Collaboration component of the mixed-initiative dialogue sub-system. They also emphasize that it is left to the underlying "black box" agent to actually make decisions, corresponding to our separation between collaborative dialogue manager and Task Manager, although it is somewhat unclear exactly what is communicated between the levels in Collagen. There are some differences between our approaches. We have concentrated on the problems of interpreting natural language in practical dialogue, and in particular how the same knowledge that drives collaboration can be used to interpret the user's input. The Collagen approach (based on (Lochbaum 1991)) to "discourse interpretation" is something that we separate into BDI reasoning (which may involve domain- or task-specific reasoning).

Driving dialogue behavior from models of rational behavior is also proposed by Sadek (Bretier & Sadek 1996; Sadek *et al.* 1996; Sadek, Bretier, & Panaget 1997). The specific application that is described involves very simple question-answering dialogue on specific topics, so it's hard to know exactly how practical the ideas really are. We imagine that in less constrained situations there would difficulties similar to those we face in trying to handle true mixed-initiative problem-solving dialogue.

Conclusions

The model we have described allows true mixed-initiative dialogue, where goals (and other joint commitments) can come from either the user or the system. The system's behavior with respect to its commitments is driven by a formal model of collaboration based on a theory of joint intention. This model not only drives the agent's own dialogue behavior, but also allows it to interpret the utterances and actions of the user in a uniform manner. The model of collaboration and its role in the dialogue system is entirely application- and domain-independent (although it depends on reasoning that may be specific to the problems at hand). We believe that the associated architecture is a practical way to develop collaborative assistants based on knowledge-based systems. We are currently trying to do just that in several domains including personal health care, command and control of agent teams, office assistants, and several artificial domains used to further explore the use of mixed-initiative dialogue systems for collaborative problem solving.

Acknowledgements

This material is based upon work supported by a DARPA/SRI International subcontract #03-000223 and from the National Science Foundation grant #IIS-0328811. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of above named organizations.

References

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

Allen, J.; Ferguson, G.; and Stent, A. 2001. An architecture for more realistic conversational systems. In *Proceedings of Intelligent User Interfaces 2001 (IUI-01)*, 1–8.

Bretier, P., and Sadek, D. 1996. A rational agent as the kernel of a cooperative spoken dialogue system: Implementing a logical theory of interaction. In *Proceedings of the ECAI-96 Workshop on Agent Theories, Architectures, and Languages*.

Clark, P., and Porter, B. 1997. Building concept representations from reusable components. In *Proceedings of AAAI-97*.

Cohen, P. R., and Levesque, H. J. 1990a. Intention is choice with commitment. *Artificial Intelligence* 42(2–3):213–361.

Cohen, P. R., and Levesque, H. J. 1990b. Performatives in a rationally based speech act theory. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, 79–88.

Cohen, P. R., and Perrault, C. R. 1979. Elements of a plan-based theory of speech acts. *Cognitive Science* 3:177–212. Also in *Readings in Artifical Intelligence*, B. L. Webber and N. J. Nilsson (eds.), 1981, pp. 478–495.

Grosz, B. J., and Sidner, C. L. 1986. Attention, intention, and the structure of discourse. *Computational Linguistics* 12(3):175–204.

Grosz, B. J., and Sidner, C. L. 1990. Plans for discourse. In Cohen, P. R.; Morgan, J.; and Pollack, M. E., eds., *Intentions in Communication*. MIT Press.

Horvitz, E. 2000. Uncertainty, action, and interaction: In pursuit of mixed-initiative computing. *IEEE Intelligent Systems*.

Lochbaum, K. E. 1991. An algorithm for plan recognition in collaborative discourse. In *Proceedings of the Twenty-Ninth Annual Meeting of the Association for Computational Linguistics (ACL-91)*, 33–38. Berkeley, CA: University of California.

Morgenstern, L. 1991. Knowledge and the frame problem. *International Journal of Expert Systems* 3(4).

Pollack, M. E. 1991. Plans as complex mental attitudes. In Cohen, P. R.; Morgan, J.; and Pollack, M. E., eds., *Intentions in Communication*. Cambridge, MA: MIT Press.

Pollack, M. E. 1992. The uses of plans. *Artificial Intelligence* 57(1):43–69.

Rao, A., and Georgeff, M. 1991. Modeling rational agents within a BDI-architecture. In Allen, J.; Fikes, R.; and Sandewall, E., eds., *Proceedings of KR-91*, 473–484.

Rich, C., and Sidner, C. L. 1998. COLLAGEN: A collaboration manager for software interface agents. *User Modeling and User-Adapted Interaction* 8(3/4):315–350.

Sadek, M.; Ferrieux, A.; Cozannet, A.; Bretier, P.; Panaget, F.; and Simonin, J. 1996. Effective human-computer cooperative spoken dialogue: The AGS demonstrator. In *Proceedings of the Fourth International Conference on Spoken Language Processing (ICSLP'96)*, volume 1, 546–549.

Sadek, M.; Bretier, B.; and Panaget, F. 1997. Artimis: Natural dialogue meets rational agency. In *Proceedings of IJCAI-97*, 1030–1035.