Pragmatic Issues in Handling Miscommunication: Observations of a Spoken Natural Language Dialog System

Ronnie W. Smith and **Steven A. Gordon**
Department of Mathematics
East Carolina University
Greenville, NC 27858, USA
{rws|gordon}@math.ecu.edu

Abstract

As with human-human interaction, human-computer dialog will contain situations where there is miscommunication. This paper describes phenomena observed in the handling of miscommunication by an experimental spoken natural language dialog system capable of variable initiative behavior, the Circuit Fix-It Shop. In general, the 141 dialogs obtained from human interaction with this system indicate that in a well-defined problem domain (1) many miscommunications can safely be ignored; and (2) a strategy of repairing miscommunications based on inconsistencies in the machine's domain model can be effective without being redundant and inconvenient. Furthermore, this repair strategy can deal with miscommunication caused by misstatements by the human user. After overviewing the operational environment of the system, we will focus our discussion on a set of examples that illustrate various aspects of miscommunication handling encountered during the formal experimental testing of the system. These examples show that most types of miscommunication encountered during the experiment are handled in an initiative-dependent fashion.

Overcoming Miscommunication in Spoken Human-Computer Dialog

Studies have shown that the most effective general means of communication between humans is spoken natural language (NL). A crucial unresolved issue in extending this communication modality to human-computer interaction is the handling of miscommunication. Human-human speech is full of miscommunication so we should expect to find it in human-computer speech as well. Recent progress has been made in preventing miscommunication at the time a misunderstood utterance is spoken ((Smith 1995) and (Smith & Hipp 1994)). However, not all potential miscommunication can be detected when it happens. Misunderstood utterances that go undetected initially can lead to confusion and in the worst case failure of the interaction. This paper describes phenomena observed in the handling of miscommunication by an experimental spoken natural language dialog system capable of variable initiative behavior, the Circuit Fix-It Shop ((Smith & Hipp 1994) and (Smith, Hipp, & Biermann 1995)). In general, the 141 dialogs obtained from human interaction with this system indicate that in a well-defined problem domain (1) many miscommunications can safely be ignored; and (2) a strategy of repairing miscommunications based on inconsistencies in the machine's domain model can be effective without being redundant and inconvenient. Furthermore, this repair strategy can deal with miscommunication caused by misstatements by the human user. After overviewing the operational environment of the system, we will focus our discussion on a set of examples that illustrate various aspects of miscommunication handling encountered during the formal experimental testing of the system.

The Circuit Fix-It Shop: A Variable Initiative Dialog System

The data used in this study was collected in experimental trials conducted with "The Circuit Fix-It Shop," a spoken NL dialog system constructed in order to test the effectiveness of an integrated dialog processing model that permits variable initiative behavior as described in (Smith & Hipp 1994) and (Smith, Hipp, & Biermann 1995). The implemented dialog system assists users in repairing a Radio Shack 160 in One Electronic Project Kit. The particular circuit being used causes the Light-Emitting Diode (LED) to alternately display a one and seven. The system can detect errors caused by missing wires as well as a dead battery. Speech recognition is performed by a Verbex 6000 running on an IBM PC. To improve speech recognition performance we restrict the vocabulary to 125 words. A DECtalk[1] DTCO1 text-to-speech converter is used to provide spoken output by the computer.

After testing system prototypes with a few volunteers, eight subjects used the system during the formal experimental phase. The subjects attempted a total of 141 dialogs of which 118 or 84% were completed successfully.[2] The system was able to find the correct meaning for 81.5% of the more than 2800 input utterances even though only 50% of these inputs were correctly recognized word for word. After the formal experiment, the system was run on a Sparc-2 workstation where the average response time by the computer was 2.2 seconds.

The most novel contribution of the system is its ability to vary its level of initiative from strongly computer controlled to strongly user controlled or somewhere in between. The need for variable initiative dialog arises because while novice users need detailed assistance, experienced users have sufficient knowledge to take control of the dialog and accomplish several goals without much computer assistance. Thus, user initiative is characterized by giving priority to the user's goals of carrying out steps uninterrupted while computer initiative is characterized by giving priority to the specific goals of the computer. In general, we have observed that the level of initiative that the computer has in the dialog is primarily reflected in the degree to which the computer allows the user to interrupt the current subdialog in order to discuss another topic. When the user has control the interrupt is allowed, but when the computer has control it is not.

Nevertheless, initiative is not an all or nothing control mechanism. Either the user or computer may have the initiative without having complete control of the dialog. Based on these observations, four dialog *modes* were identified that characterize the level of initiative that the computer can have in a dialog. In *directive mode* the computer has complete dialog control and will not allow interruptions to any other subdialogs. In *suggestive mode* the computer still has dialog control, but will allow minor interruptions to closely related subdialogs. In *declarative mode* the user is given dialog control and can consequently interrupt to any desired subdialog at any time. However, the computer is free to mention relevant facts as a response to the user's statements. Finally, in *passive mode* the user has complete dialog control, and the computer only provides domain information as a direct response to a user question.

Technical details about how dialog mode affects the computer's selection of a response are described in (Smith & Hipp 1994). Previous studies of variable initiative theory include (Kitano & Van Ess-Dykema 1991), (Whittaker & Stenton 1988), and (Walker & Whittaker 1990). In addition to a theory, we provide a mechanism that allows a computer to participate in such dialogs. During the experiment,

[1] DECtalk is a trademark of Digital Equipment Corporation.

[2] Of the 23 dialogs which were not completed, 22 were terminated prematurely due to excessive time being spent on the dialog. Misunderstandings due to misrecognition were the cause in 13 of these failures. Misunderstandings due to inadequate grammar coverage occurred in 3 of the failures. In 4 of the failures the subject misconnected a wire. In one failure there was confusion by the subject about when the circuit was working, and in another failure there were problems with the system software. A hardware failure caused termination of the final dialog.

the system operated in either directive mode or declarative mode. General differences in user behavior depending on the level of computer initiative were observed. When the computer operated in declarative mode, thus yielding the initiative to experienced human users, the dialogs (1) were completed faster (4.5 minutes versus 8.5 minutes); (2) had fewer user utterances per dialog (10.7 versus 27.6); and (3) had users speaking longer utterances (63% of the user utterances were multi-word versus 40% in directive mode).

The Consequences of Word Misrecognition

The primary source of miscommunication during the experiment was the misrecognition of words. Overall, 18.5% of all utterances were misunderstood, and almost all of these misunderstandings were due to misrecognition of one or more words in the user utterance. While speech recognition technology has improved, there will never be 100% accuracy in all situations, especially where there is only voice interaction (e.g. telephone dialogs). We have achieved some measure of success in detecting these types of misunderstandings at the point of origin ((Smith 1995) and (Smith & Hipp 1994)), but these methods are also not 100% successful. Consequently, we can expect the problem of misunderstanding due to word misrecognition to persist.

In order to avoid excessive user frustration, whenever a misrecognition caused the computer to interpret the utterance in a way which contradicted what was meant, the experimenter was allowed to (1) tell the subject that a misrecognition had occurred, and (2) tell the user the interpretation made by the computer, but *could say nothing else*. For example, when one subject said, "the circuit is working", the speech recognizer returned the words "faster it is working." This was interpreted as the phrase "faster." Consequently, the experimenter told the user, "Due to misrecognition, your words came out as 'faster'." It is important to note that the experimenter did not tell the user what to do, but merely described what happened. In this way, the interaction was restricted to being between the computer and user as much as possible, given the quality of commercial, real-time, continuous speech recognition devices at the time of the experiment. Such error messages from the experimenter occurred on average once every 15 user utterances throughout the experiment.

Not all misunderstandings required experimenter interaction. There were a total of 525 misunderstandings, 441 of which met the requirements for experimenter interaction, yet the experimenter notified the user about the misunderstanding only 187 times, or in only 42% of the possible situations where the misunderstanding was severe enough to warrant interaction. In many situations the experimenter felt that the user and computer would be able to work through the miscommunication on their own. The next section will present examples of these situations and illustrate how the dynamics of miscommunication varies depending on the computer's level of initiative in the dialog.

Ignoring Misunderstandings: Initiative Tradeoffs

The impact of misunderstanding tended to be much different depending on the computer's level of initiative in the dialog. In directive mode, misunderstandings were ignored in the situation where the computer's erroneous interpretation led it to conclude that the user's focus was on a task goal that differed from the computer's task goal. In example 1 from figure 1 the computer's misunderstanding led it to conclude the user's focus was on the position of the control knob.[3] The computer ig-

[3] All the examples are excerpts from the experimental dialogs.

```
EXAMPLE 1 -- Computer ignores misunderstanding in directive mode.

Computer: What is the voltage between connector 121 and 120?

User's words :    Voltage is eight
Recognized words: be control six and be
Understood as:    "The control knob is at six."

Computer: What is the voltage between connector 121 and 120?

EXAMPLE 2 -- User-initiated correction in declarative mode

Computer:  Whether there is a wire between connector 111 and connector 120 needs
           to be observed.

User:      There is a wire between one one one and one two zero.
Recognized as: there is do wire it between one one one an one stays zero.
Understood as: "There is a wire between connectors 111 and 1."

Computer:   There is not supposed to be a wire between connectors 111 and 1.

User repeats original assertion, which is now recognized satisfactorily and
interpreted correctly.
```

Figure 1: Miscommunication Excerpts

nored this apparent digression and repeated the question concerning its goal, the voltage measurement between connectors 121 and 120.

In contrast, declarative mode dialogs contained occurrences of interaction where the computer would respond to the misunderstanding, but its response would signal the misinterpretation to the user, leading to a user-initiated correction as shown in example 2 of figure 1. Thus, computer attempts to be helpful by producing utterances relevant to perceived user goals occasionally provided the necessary information to allow users to initiate corrections. However, this was not prevalent, as the experimenter had to notify the user about misunderstandings 118 out of 215 times or 54% of the ones that met the requirements for experimenter interaction. In contrast, the experimenter needed to notify the user for only 30.5% of the ones in directive mode (69 out of 226).

Furthermore, in both modes of interaction there were many occurrences of misunderstandings that did not require experimenter interaction because the system had a mechanism for questioning inconsistencies in its domain model. This is the topic of the next section.

Repairing Domain Inconsistencies

We attempted to resolve misunderstandings by verifying the user's observations *only* when inconsistencies with the domain model arise. Although the source of a domain model inconsistency may have occurred several utterances

```
EXAMPLE 3 -- Computer encounters a dead end.

Computer: The LED is supposed to be displaying alternately flashing one and
          seven.

User's words : LED displaying seven
Recognized as: be led is flashing seven then
Understood as: "The LED is displaying a flashing seven."

Computer:    Is anything else on the LED on?
User:        No.
Computer:    What is the switch at when the LED is displaying only a flashing
             seven?
User:        Switch is up.

Computer:    What is the knob at when the LED is displaying only a flashing
             seven?

(NOTE: At this point, the computer has adopted a goal based on the LED display
       of a flashing seven. The user's one attempt at correction fails, and
       the dialog proceeds.)

User:        One zero.
Computer:    Is there a wire between connector 84 and connector 99?
User:        Yes.
Computer:    Are you sure that the LED is displaying only a flashing seven?

EXAMPLE 4 -- Computer encounters an impossibility.

Computer:    What is the LED displaying?
User:        The LED is displaying a one on the left hand side.
Computer:    Is anything else on the LED on?
User:        No. (NOTE: misstatement--LED displaying both the one and seven.)
Computer:    Is the one on the LED flashing?
User:        The LED is displaying alternately one and seven.
Computer:    Are you sure that the one on the LED is not displaying for a longer
             period of time?
User:        No.
```

Figure 2: Excerpts Illustrating Resolution of Domain Inconsistencies

before the detection of the inconsistency, we believe this approach is more user-friendly and practical than attempting to detect and resolve every potential miscommunication right when they occur.

Our domain model is fairly complex, addressing twelve different features of the LED under three different configurations of the power switch and control knob. There are two different ways that user observations can lead to an inconsistency:

- Dead ends - the computer's interpretation of the user's description of the state of the LED leads to debugging a part of the circuit which turns out to be working correctly.

- Impossibilities - the computer's interpretation of the user's description of the state of the LED cannot possibly occur in the given circuit.

Example 3 of figure 2 is a typical dead end situation. The computer has adopted a goal based on the misunderstood report of "only a flashing seven." When this adopted goal leads to no missing wires in the part of the circuit that would cause "only a flashing seven," the computer attempts to verify the state of the LED. This ends up resolving the specific miscommunication that was inhibiting the correct diagnosis and repair of the circuit.

The computer's ability to detect and repair dead ends is a function of its level of initiative. When it has the initiative (i.e. directive or suggestive mode), it is able to complete its preferred set of debugging steps and determine that a dead end has occurred. Conversely, when it is operating in declarative or passive mode, the current subcircuit under investigation may continually change depending on the user's desires. Consequently, the computer may never gain sufficient information to logically conclude that a dead end has been encountered.

Example 4 of figure 2 is a typical impossibility. The user's erroneous report that the LED is displaying a one and nothing else with the knob at zero is impossible according to the computer's domain model. The computer resorts to verifying this observation. After the user corrects the misstatement, the computer must still verify the last uncertain piece of information (the duration of display for the one). Note that in declarative mode, the computer would have behaved in exactly the same fashion.

Using consistency checking of the domain model generally detects just those miscommunications which threaten to inhibit success. The strategy of repairing these miscommunications by verifying just the observations involved with the inconsistency generally allows the user to correct the computer's interpretation of the state of the LED.

An important observation is that it does not matter whether the miscommunication was due to the computer misunderstanding the user (as in example 3) or the user misspeaking (as in example 4). While this system is not always robust enough for the user to retract misstatements, if a misstatement leads to a dead end or inconsistency, the user often gets an opportunity to make a correction.

Lessons Learned

1. Other than for impossibilities, behavior varies as a function of initiative. While computer-controlled dialogs avoid many dead ends by ignoring unestablished task goals, once erroneous task goals are established, user-initiated correction of the misunderstanding will not succeed. In one experimental interaction, it took the computer 28 utterances before it adopted a new goal! Consequently, the ability to change initiative to user control during the dialog becomes crucial for refocusing the computer on the correct task goal.

2. In a noisy environment where word misrecognition can frequently lead to misunderstanding, the computer should be as explicit as reasonable in establishing its context when responding to the user. For example, instead of the question, "Is anything else on the LED on?" from example 3, the computer could ask, "Is anything else on the LED other than a flashing seven?" This technique must of course be balanced with the inherent problems of utterances that are too verbose and redundant. Studies in conversational grounding (Traum 1994) may prove helpful in this area.

3. The close coupling of task and dialog facilitates dialog repair based on detection of domain model inconsistencies. We believe this technique can be effective in most task-oriented domains, particularly those where the graph of problem-solving steps is relatively *shallow* (i.e. completion of only a small number of task goals is needed to move from the problematic observation to a solution). Alternative techniques are probably needed when the problem-solving graph requires deeper search or when the domain is less task-oriented (e.g. information-gathering dialogs).

In general, we have discussed aspects of miscommunication handling that were observed during experimental interaction with a spoken natural language dialog system that provides assistance in an electronic circuit repair task. Work continues on dealing with these and other issues concerning miscommunication handling. Our ultimate goal is a theory of dialog processing that enables a spoken natural language dialog system to successfully complete dialogs with human users *without* the need for occasional experimental intervention. Only then can we claim to have an effective model of robust dialog interaction.

Acknowledgments

Other researchers who contributed to the development of the experimental system include Alan W. Biermann, Robert D. Rodman, Ruth S. Day, D. Richard Hipp, Dania Egedi, and Robin Gambill. In particular, we credit Dania Egedi with the development of the LED domain model that enabled the system to check for inconsistencies and engage in repair dialogs. The writing of this paper has been supported by National Science Foundation Grant NSF-IRI-9501571 and the Research and Creative Activity Committee of East Carolina University.

References

Kitano, H., and Van Ess-Dykema, C. 1991. Toward a plan-based understanding model for mixed-initiative dialogues. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, 25–32.

Smith, R., and Hipp, D. 1994. *Spoken Natural Language Dialog Systems: A Practical Approach.* New York: Oxford University Press.

Smith, R.; Hipp, D.; and Biermann, A. 1995. An architecture for voice dialog systems based on Prolog-style theorem-proving. *Computational Linguistics* 281–320.

Smith, R. 1995. Resolving miscommunication during collaborative spoken natural language dialog. Manuscript under Review.

Traum, D. 1994. *A Computational Theory of Grounding in Natural Language Conversation.* Ph.D. Dissertation, University of Rochester.

Walker, M., and Whittaker, S. 1990. Mixed initiative in dialogue: An investigation into discourse segmentation. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, 70–78.

Whittaker, S., and Stenton, P. 1988. Cues and control in expert-client dialogues. In

Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics, 123–130.