(Cahn, J. E., & Brennan, S. E. (1999). A psychological model of grounding and repair in dialog. *Proceedings, AAAI Fall Symposium on Psychological Models of Communication in Collaborative Systems (pp. 25-33)*. North Falmouth, MA: American Association for Artificial Intelligence.)

A Psychological Model of Grounding and Repair in Dialog

Janet E. Cahn **Susan E. Brennan**

Massachusetts Institute of Technology
Cambridge, MA 02139
cahn@media.mit.edu

State University of New York
Stony Brook, NY 11794-2500
susan.brennan@sunysb.edu

Abstract

We formalize and extend the contribution model of Clark and Schaefer (1987, 1989) so that it can be represented computationally; we then present a method for combining the turns of two individual agents into one incrementally determined, coherent representation of the processes of dialog. This representation is intended to approximate what a participant might represent about the dialog *so far*, for the immediate purpose of referring, making contextual inferences, and repairing problems of understanding, as well as for the longer term purpose of storing the products of dialog in memory. Such an approach, we argue, is necessary for enabling a computer-based partner to converse in a way that seems natural to a human partner.

Introduction

Dialog is a collective activity that is managed in real time by agents with limited attentional, computational, and knowledge resources. Even when two agents are rational and cooperative, inhabit the same location, speak the same language, share much of the same knowledge, and use common wording, there is no guarantee that one will understand the other on the first try. For instance, one agent may overestimate another's knowledge, or may not hear part of what was last said. Since neither partner in a dialog has direct access to what the other is thinking, they must coordinate their distinct mental states and get them to converge to some degree in order to communicate successfully. This they do based on the contingent evidence they receive from their partners; H. H. Clark and his colleagues have labeled this process *grounding* (Clark and Brennan 1991; Clark and Marshall 1981; Clark and Schaefer 1987, 1989; Clark and Wilkes-Gibbs 1986; Schober and Clark 1989).

The most formal model to emerge from this framework is Clark and Schaefer's *contribution model* (Clark and Schaefer 1987, 1989), which addresses the detection and repair of communication errors. According to this model, a conversation is made up of *contributions*, and each contribution has two phases -- a *presentation phase*, followed by an *acceptance phase.* In the presentation phase, a speaker presents an utterance to an addressee; in the acceptance phase, evidence of understanding is accrued until it is clear to both parties that the propositions put forth in the original or revised presentation are mutually understood and therefore part of their common ground. The acceptance phase may be as short as one utterance, or longer if it includes a clarification subdialog or repair. An utterance (unless it is discourse-initial) plays two roles: as part of an acceptance, it provides evidence about how its speaker construes a prior utterance; as a presentation, it contributes to the fulfillment of the speaker's ostensible discourse purpose. Contribution sequences are shown as directed graphs whose nodes are labeled C (contribution), Pr (presentation) or Ac (acceptance) (Figure 1).

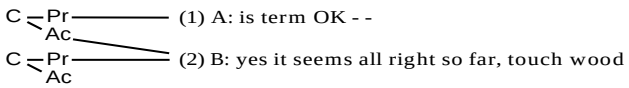

Figure 1: Utterance (2) plays two roles, acceptance and presentation (example from Svartvik and Quirk 1980).

The conversation on which Figure 1 is based had a more complex structure because the acceptance phase for the first contribution included an embedded clarification sequence, as shown in Figure 2, *(2)* and *(3)*.

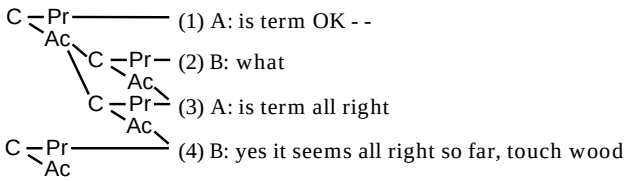

Figure 2: Contribution graph that includes a clarification.

Despite differences in structural complexity, the utterances in these graphs are organized by the same rationale, that each utterance, U_n, is linked to the preceding one, U_{n-1}, according to the *evidence* it provides about how its speaker (the addressee of U_{n-1}) understood or construed U_{n-1}. That is, the addressee of U_n (who produced U_{n-1}) interprets U_n for evidence about how its speaker has understood or construed U_{n-1}. If he interprets the evidence as sufficient (as positive evidence that U_{n-1} is acceptable), he goes on with the conversation by presenting U_{n+1} as the utterance relevant to the next domain task; if he interprets it as insufficient (or as negative evidence), he is likely to use U_{n+1} to initiate a repair (see Brennan 1990, on how speakers seek and provide evidence in grounding). It is important to note that negative evidence alone is not sufficient for grounding, whether the partner is human or

computer. A dialog partner also needs positive evidence, or evidence that the task and the interaction are proceeding on track (Clark and Brennan 1991).

Each partner employs his or her own set of standards to evaluate the evidence of understanding provided by the other and to determine whether to actively seek out further evidence. These standards vary according to a partner's current purposes and constitute a *grounding criterion* (Clark and Schaefer 1989; Clark and Wilkes-Gibbs 1986; Clark and Brennan 1991), or threshold at which the evidence that a presentation has been accepted is deemed sufficient. A lax grounding criterion may require only that the addressee display attention to or register hearing an utterance, as in a casual conversation between strangers waiting in line at a checkout counter. A more stringent one would require a response whose semantics are recognizably relevant to the current task. A grounding criterion may shift over the course of a conversation. The requirements for providing feedback might even be legally stipulated, as in conversations between pilots and air traffic controllers. In general, the more stringent the grounding criterion, the more exacting the relevance requirements and the stronger the evidence needed to indicate that things are on track.[1]

When a contribution appears to meet the grounding criteria of both partners, they may each assume that they have mutually understand one another. As this happens, mutually understood propositions are added to each partner's representation of the dialog (Clark and Brennan 1991; Clark and Marshall 1981) and are available for collaborative use. To the extent that these individual representations correspond and partners are mutually aware of this correspondence, the partners have common ground.

There are numerous differences between people and "intelligent" systems that we do not expect will (nor should) disappear anytime soon. But there is also a needless built-in structural asymmetry in human-computer dialog that undermines successful communication, even between partners who are of necessity quite different. The asymmetry is as follows: Systems give error messages when they find a user's input unacceptable, providing *ad hoc* evidence that may be more or less informative to users. But users don't have this power; they have no choice but to accept what the system last presented, and if this is unexpected or problematic, to start over or to figure out how to undo what the system did. Usually, systems do not seek evidence that their last turn was acceptable, and users lack appropriate ways to present such evidence.

To address this asymmetry, we propose that a computational dialog system should be equipped with an architecture that explicitly represents whether previous turns have been grounded, and that it should not represent previous actions or turns in its model of the dialog context unless there is evidence that these were what the user intended. A model of context is necessary because it enables a system to interpret anaphoric expressions and to make inferences about common ground. When it is based on contributions that have been grounded, it is likely to correspond more closely to the user's dialog model than one that simply records dialog history.

Although Clark and Schaefer's contribution model was developed to account for human conversation, it is has also been considered as a representation for human-computer dialog (see, e.g., Brennan 1991; Cahn 1991; Brennan and Hulteen 1995; Heeman and Hirst 1995; Luperfoy and Duff 1996; Novick and Sutton 1994; Traum 1994; Walker 1993). The model provides a good basis for human-computer interaction because it is expressed in the language of computational structures (as a directed graph), its elements are few and context independent, and its representations are built incrementally. However, its original explication contains formal inconsistencies and under-specified operations that prevent its direct incorporation into a dialog system (Cahn 1992). In the following sections, we identify these and propose the structural and operational changes that allow the contribution model to support human-computer interaction.

Adapting the contribution model for HCI

Clark and Schaefer's goal was to model the *process* of conversation. Yet their contribution graphs (Clark and Schaefer 1987; 1989) depict only the final (and presumably shared) *products* of conversation, as might be determined from hindsight. Such representations do not tell the whole story, for they fail to represent the interim products that each agent created moment-by-moment (and perhaps revised or discarded) in order to reach the final product. For this reason, we focus on agents' private models and their consequences. We assume that cooperative dialog partners aim for convergence of their private models so that both are ultimately composed of sufficiently similar parts. However, at every step in its construction, a model represents only the view of the partner who created it, since neither partner is omniscient. Our first addition to Clark and Schaefer's model is to emphasize that *all contribution graphs are private models, and can represent the perspective of only one agent.* Even the final contribution graph represents the state of the dialog as estimated by one partner.

Our second addition is aimed explicitly toward human-computer interaction. We detail the heuristics with which a simple computational agent applies its grounding criterion to the evidence in a turn. To do this, we focus on a database query application for which we originally implemented some of the ideas in this paper. These heuristics are specific to the application and its domain, and depend on: (1) the semantics of the utterance under evaluation; (2) the contents of the agent's private model; (3) the agent's knowledge about the task domain; and (4) the agent's construal of the kind of task (or speech act) intended by its partner.

[1] Conversations in which one partner has a high grounding criterion and the other has a low one (and they are unaware of this difference) are more error-prone than those in which partners have the same grounding criteria and goals (Russell and Schober 1999; Wilkes-Gibbs 1986).

Our third addition is to show that contributions are linked and embedded according to how each model holder evaluates the evidence that the other partner has presented. Because Clark and Schaefer do not work through their examples using private models, it is not evident in their treatment that the structure of the acceptance phase should depend on whether the model holder interprets the evidence as meeting her grounding criterion or not. We claim that a contribution should be embedded only when the evidence it provides fails to meet the grounding criterion of the model holder. As the conversation progresses, partners may revise their models to reflect new evidence.

Finally, we introduce a new structure to the formalism, the *exchange*. The exchange is a pair of contributions linked by their complementary roles: the first *proposes* and the second *executes* a jointly achieved task. As a structure that organizes the verbatim content of a dialog, the exchange explicitly represents the influence of the task on the interaction. It captures the intuition behind *adjacency pairs* (Schegloff and Sacks 1973), or two-part collaboratively-accomplished discourse tasks.

Explicitly portraying the system's private model

In any dialog in which two agents take turns, the agents' private models will regularly be out of sync with each other; for example, one agent recognizes that something is amiss before the other one does (Brennan and Hulteen 1995; Luperfoy and Duff 1996). Therefore, it is important to recognize, whether in a theory of communication or an interactive computer application, whose perspective is represented, and why. The consequence of not doing so is confusion about whether the perspective represented belongs to one agent or is shared by both. When the conversation includes a repair, this can result in a graph that appears to display two disjoint perspectives at once (Figure 3).

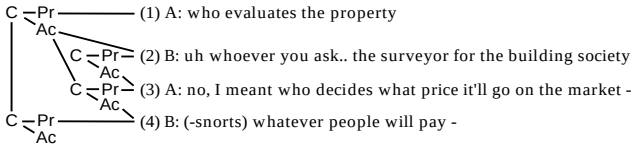

Figure 3: Example with unrooted node, showing disjoint perspectives of two agents in one graph (reproduced from Clark and Schaefer 1989, p. 277).

The graph in Figure 3 shows only the final graph of a conversational interaction. It includes B's early view that *(2)* is a useful and relevant answer to *(1)*, as indicated by a link from the first acceptance node directly to *(2)*. It also includes A's contrasting view that *(2)* is evidence of a misunderstanding that needs repairing, as indicated by its embedding and by the absence of a link between the first acceptance node and the contribution containing it. Using one graph to show two perspectives creates an anomalous structural artifact – an *unrooted contribution node* linking *(2)* and *(3)*. While the embedding of *(2)* is correct because it is the utterance that starts a repair, the contribution to which it belongs is attached to the entire structure only via its leaf nodes. Yet B had introduced *(2)* in an attempt to further the task; it is a relevant part of the acceptance phase. Because of this, its unrooted status is unjustified. Instead, A's and B's final graphs should show that *(2)* is a legitimate part of the acceptance phase of the first contribution. Even though B did not realize it at the time, *(2)* ended up initiating a repair in A's private interim model as well as in both of their final models. The rewrite in Figure 4 shows the divergent views of the two conversants in two separate models.

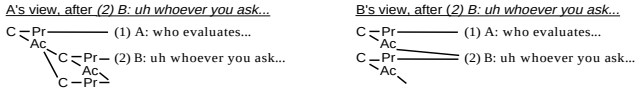

Figure 4: Previous example from CS89, reworked to show the interim, divergent models of the two conversants.

The divergent views of two partners (or of one partner at two different points in time) do not belong in the same graph. By constraining a graph to represent the distinct perspective of only one agent at a single point in time, we emphasize the distinction between interim and final structures.

Using a grounding criterion to evaluate evidence

How an agent evaluates the available evidence and updates its contribution model is determined by its grounding criterion: "speaker and addressees mutually believe that the addressees have understood what the speaker meant to a criterion sufficient for current purposes" (Clark and Schaefer 1987, p. 20). Exactly how do people set their grounding criteria when faced with particular tasks and partners? And how do they determine whether the evidence that partners provide meets their grounding criteria? Clark and colleagues provide no formal answers to these questions, possibly because the answers are specific to speakers, addressees, and situations (Clark and Brennan 1991). To test the evidence in an utterance against a grounding criterion may require using task-specific knowledge, common sense, and metalinguistic awareness.

Unless it has significant reasoning ability, a computational agent's repertoire of response options is very limited, and its evaluation of the relevance of a users' responses is highly domain dependent. Therefore, there may be no general solution to the problem of how such an agent should set and use a grounding criterion; methods would need to be tailored to particular applications. Such limitations actually present an opportunity to observe how grounding criteria can work with respect to a particular application, the approach taken here (and in Brennan and Hulteen 1995). Our proposal depends on simple heuristics for judging whether a user's turn provides positive or negative evidence of acceptance based on, as Clark and Schaefer suggest, the illocutionary act that an utterance appears to propose (Clark and Schaefer 1989).

In our proposal, the user interface is key; it allows mixed initiative, provides explicit response options for the user, and visibly represents relevant discourse context. It includes a small input window in which the user types her query, and a larger window in which the system displays its responses as well as its current supposition about the dialog structure (shown by indenting and nesting utterances). As we noted earlier, a major problem with human-computer dialog is that users typically do not have any choice when it comes to accepting the system's presentations. Our interface addresses this problem by providing four buttons adjacent to the input window, labeled *Ok*, *Huh?*, *No, I meant*, and *Never mind*. Choosing one of these causes its label to appear as the initial text in the input window, followed by any text that the user then types. With these buttons, the user expresses her intentions and understanding to the system. By the same token, the system associates each response button with a task role (either defining or executing a task) for the user's presentation, an embedding for the presentation in the graph of the current exchange, and consequently, a response strategy.

Although the user initiates most domain tasks without using these buttons (by simply typing a query), at any point in the subsequent exchange, both user and system can ask for clarification or propose alternatives. The system requests clarification by presenting either text or a clickable menu of choices; it indicates that the user's input is acceptable by attempting to proceed with the domain level task (typically, executing a database query). The user, on the other hand, requests clarification using *Huh?*, following a proposal by Moore (1989) that systems should support vague clarification requests from users. In our prototype, *Huh?* may either precede a clarification request that the user then types in, or else serve as the entire turn. It evokes whatever the system is able to provide by way of an explanation or else a paraphrase when one is available, (e.g., information about the choice of one of several parses, or an expansion of an earlier error message; see Creary and Pollard 1985). The button labeled *No, I meant* prefaces input by which the user revises or replaces her previous query, initiating a third turn repair (Schegloff 1992). The button labeled *Never mind* simply aborts the current domain-level exchange and resets the exchange graph to the point of the last previously grounded exchange.

These heuristics recognize that evidence of acceptance may be either explicit or implicit. The system presents positive evidence implicitly when it answers the user's domain level query and negative evidence explicitly when it asks a clarification question or presents an error message. Similarly, the user's negative evidence is always explicitly labeled (*Huh?*, *No, I meant* or *Never mind)*. Her positive evidence need not be; it can be inferred from her actions, such as when she responds to the system's answer by sending the next domain level query. Alternatively, she may choose the *Ok*. button, explicitly accepting the system's answer.[2]

Not only does this approach give the user options for providing the system with evidence about whether she finds its presentations acceptable, but it also displays the system's evaluation of this evidence in the dialog window. For instance, when the user implicitly accepts the system's answer by inputting the next relevant query, her query appears in the dialog window automatically prefaced by *Ok.* This indicates that the system construes her new query as implicitly accepting its last answer. The system's current view of the overall dialog structure appears as indented turns in the dialog window, corresponding to embedded presentations in its private model)[3].

It is only when the evidence is positive, that is, the system's last turn has been accepted either implicitly or explicitly by the user, that it becomes part of the system's dialog model and licenses the system to add a summary of the previous exchange to its representation of common ground.

Using the evidence to structure contributions

To support human-computer interaction, our model distinguishes domain tasks and conversational tasks. Domain tasks include such joint activities as getting the answer to a database query and delegating an action to an agent; conversational tasks involve detecting and clearing up problems of understanding. Which domain tasks need to be supported depends on the particular application. Conversational tasks, on the other hand, are domain-independent and are about communicating; in our prototype, these consist of clarifications and third turn repairs.

In our algorithm, structural embedding (as in Figure 2) is used only to reflect a model holder's evaluation of the evidence in an utterance as negative (insufficient to merit acceptance, or else indicating a likely misunderstanding). This differs from Clark and Schaefer's additional use of embedding for explicit acknowledgements (such as *uh huh*) in the parts of an installment utterance[4] (Clark and Schaefer 1989). Our rationale for embedding only utterances that provide negative evidence is that they introduce a structurally subordinate task: a repair. Explicitly stated acknowledgments, on the other hand, count as positive evidence of understanding and do not introduce a new task. They should be represented at the same structural level as the presentations they ground (see the discussion of installment presentations in Cahn 1992; Cahn and Brennan 1999).

---

[2] In our application, implicit and explicit *Ok*s have the same consequences for the system's model, but in another kind of application explicit *Ok* could provide a way of reaching explicit closure before changing a topic or ceding initiative back to the system.

[3] Representing U_n at the same level as U_{n-1} signals its acceptability; indenting it indicates that it was problematic for the addressee.

[4] In an installment utterance, the speaker adopts a relatively high grounding criterion and presents information in small parts that can be grounded individually, such as the parts of a telephone number.

The inconsistent embedding in Clark and Schaefer's examples appears to confound *task subordination* with *dialog initiative*. It is important to distinguish these. Dialog initiative is a concept that captures which individual agent in a dialog is responsible for initiating the current domain task (such as asking a domain-relevant question or issuing a command) or conversational task (such as initiating a repair or requesting clarification). At any particular moment in a dialog, one partner can be said to have taken the initiative (Whittaker and Walker 1990). In spontaneous conversation, initiative alternates freely between partners as joint purposes evolve. In others, particularly in those oriented to predefined tasks (such as an interview), as well as in most human-computer applications (such as database query or automated teller machine dialogs), initiative is less flexible. However, the contributions of the partner who follows are no less important than the contributions of the one who leads; both are needed for successful collaboration. Our algorithm rules out embedding when the evidence is positive, representing presentations from both partners on the same level. Embedding occurs only when closure does not; it represents the interim work toward grounding the presentation that initiated the exchange.

In our prototype system, each of the response buttons on the interface is associated with a task role and embedding that the system uses to construct its graph and choose its response (see Table 1).

Exchanges: Mapping contributions onto tasks

Contributions do not easily map interaction onto domain tasks. Therefore, we propose that the *exchange*, rather than the contribution, should be the minimal jointly determined dialog unit. An exchange consists of two contributions, each initiated by different partners, and each playing a unique role in accomplishing a collaborative domain task. With the first contribution, the task is initiated or *defined*; with the second, it is completed or *executed*. Either partner may take the initiative in defining a task. In both the definition and execution phases of a task, the other partner may attempt to modify what the first has begun.

The exchange captures the observation that utterances tend to occur in meaningful pairs that, together, accomplish a single collaborative task (Schegloff and Sacks 1973). Figure 5 shows a hypothetical dialog fragment depicted first as adjacency pairs and then as exchanges. Both adjacency pairs and exchanges contain utterances that address a common purpose. However, the reasons that one adjacency pair follows another (or else is embedded within another) are not obvious from their structural components. In contrast, the exchange graph includes connections at the leaf nodes to explicitly illustrate the dependencies between successive contributions. Exchanges are more general than adjacency pairs because what is critical is mainly whether a task is being defined or executed. The exchange is a structural primitive that make a coherent link between the turns from two agents; we propose it as an analytical and structural bridge to higher level discourse models, such as the focus space model developed by Grosz and Sidner (1986).

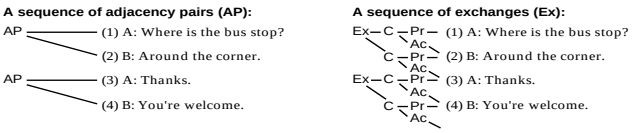

Figure 5: Structural comparison of a dialog represented by adjacency pair and exchange graphs.

Table 1: Interpreting the evidence in a user's turn, U_n, for the acceptability of U_{n-1}.

User's presentation, U_n	Relation of U_n to task	Embedding of U_n in the system's graph	How the system construes evidence in U_n about the user's beliefs about the system's prior presentation, U_{n-1}.
Ok	Confirms end of previous exchange; may be followed by a task definition in the same turn.	Top level (domain task)	Explicit acceptance: The user accepts the system's presentation, U_{n-1}, as an acceptable execution of the task initiated by U_{n-2} (typically, as a relevant answer to her question).
Domain-level query	Task definition.	Top level (domain task)	Implicit acceptance: The user accepts the system's presentation, U_{n-1}, as an acceptable execution of the task initiated by U_{n-2} (if U_n is dialog-initial, it simply initiates a domain-level task.)
No, I meant	Task definition - attempts to repair the system's misconstrual of the initial query.	In the acceptance phase of an embedded exchange.	Contradiction or revision: The user rejects the task definition the system proposed in U_{n-1}, believing that the system misunderstood U_{n-2}, and therefore re-interprets U_{n-1} as having begun a repair task.
Huh?	Task definition - defines a clarification task for the system.	Embedded within another exchange.	Request for clarification: The user agrees with the system about the role of U_{n-1} in the exchange, but requests clarification on other aspects of the utterance.
Selecting a response on a menu	Task execution.	Embedded: Second contribution in a clarification exchange.	Response to a request for clarification: The user accepts U_{n-1} as having initiated a clarification sequence, just as the system intended.
Never mind	Task termination - ends exchange.	Irrelevant – cancels current exchange and removes it from the dialog model.	Abort: The user rejects the dialog segment represented by the current domain-level exchange, which renders moot any questions about U_{n-1}.

The use of evidence by an interactive system: An example

The application we used to develop the exchange model is a database query system that processes textual natural language queries, maps them onto logical queries, and provides answers from an employee information database (described further in Brennan 1988; Creary and Pollard 1985; Nerbonne and Proudian 1988). In this section we will discuss the system's private model (its best estimate of the state of the dialog) at each turn, using the example:

(1) User: Where does Dan work?
(2) System: In the natural language group.
(3) User: ***No, I meant*** where is his cubicle?
(4) System: Near post H33.
(5) User: Where is Jill's cubicle?

We will consider what happens at turn (3), *No, I meant where is his cubicle*. The user produced this turn by choosing the No, I meant button, typing a revised query, and hitting carriage return. After this turn, the system evaluates the evidence in (3), updates or revises its exchange graph concerning the role of (2) in the dialog, computes the gist of a revised query, sends it to the database, constructs a response, and updates a model of the dialog so far that estimates common ground with the user.

Evaluating the evidence presented by the user

The system can respond cooperatively only if it construes the evidence in U_n about the user's acceptance of U_{n-1} as the user intended. It relies on three sources of information: (1) publicly available evidence in the form of the response option the user has chosen; (2) privately held evidence, including the structure of its graph of the dialog so far, whether U_n (and U_{n-1} before it) appear to be attempts to *define* or to *execute* a task,[5], and whether it experienced any internal errors in response to the user's input in U_n; and (3) domain knowledge, including expected kinds of turns that the user might have produced at this point.

The system must first determine what speech act the user is proposing. Many approaches to dialog systems depend on identifying speech acts (e.g., Litman and Allen 1987; Pollack 1990; McRoy and Hirst 1995; Heeman and Hirst 1995; Traum and Hinkelman 1992). In our prototype, the mapping is straightforward; speech acts are limited to the conversation-level tasks defined by the four-button interface and domain-level ones consisting of queries and responses to clarification questions by the system. So the identification of what task the user intends to initiate is relatively simple and unambiguous; most of the time, the user implicitly or explicitly identifies the act she intends by virtue of typing an utterance into the input window, selecting a button, or both.[6] It is also relatively unambiguous (compared to human conversation) as to whether the user intends U_n to initiate a new task or to execute one proposed by the user.

In our example, the user's selection of ***No, I meant*** enables the system to identify her turn as an attempt to correct the system's understanding of the task she proposed two turns earlier in U_{n-2}. This third-turn repair tells the system that the user did not find the evidence in its turn, (U_{n-1}) acceptable. This guides its next actions: updating its exchange graph and responding to the user.

Updating the system's exchange graph

Each conclusion the system draws about whether the user has accepted (in U_n) its last presentation (in U_{n-1}) leads to a different operation on its exchange graph. The operation is determined by whether the evidence of acceptance is positive (meets the system's grounding criterion) or negative (fails to do so), as well as by the current state of the graph, that is, by what role (task definition or execution) U_n proposes to play in the current exchange.

Table 2 shows the operations on the exchange that

Table 2: Updating the system's exchange graph to include the system's construal of the user's evaluation of the system's previous utterance, U_{n-1}.

If system concludes that ↓ and →	U_n, the user's current utterance, does not propose a task	U_n proposes a task
U_{n-1}, its previous utterance, is acceptable to the user	***A.*** *Begin a new task:* (1) If U_n, the pending contribution, is acceptable, close the pending exchange, whose final contribution is U_n; (2) Or, if U_n is not acceptable, initiate a clarification: create an exchange embedded below U_n, whose first presentation will be U_{n+1}.	***B.*** *Execute the task defined by U_n:* The acceptance phase for U_{n-1} is complete, so create a new exchange whose first presentation is U_n, and construct U_{n+1} to execute the task defined by U_n.
U_{n-1} is not acceptable to the user	***C****. Revise the task definition (third turn repair) and in the fourth turn, execute the re-defined task:* (1) Unlink U_{n-1}; it is no longer the second presentation in an exchange. (2) Create a new exchange, initiated by U_{n-1}, and embed it beneath U_{n-2}. (3) Construct U_{n+1} to execute the task defined via the sequence U_{n-2}, U_{n-1} and U_n.	***D****. Provide clarification:* Embed an exchange below U_{n-1}, whose first presentation is U_n, and construct U_{n+1}.

[5] When a partner attempts to define a task, she takes the initiative and presents what she believes is the first contribution in an exchange. When she attempts to execute a task, she follows the lead of her partner and responds with the second contribution in an exchange.

[6] In a mixed initiative dialog with a more capable agent, the problem of identifying what task a speaker intends to initiate would be much more difficult. Whether a system can identify what speech acts users intend would need to be determined empirically for a particular application.

correspond to different combinations of the acceptability status of U_{n-1} (in Table 2's rows) and the task role of U_n (in Table 2's columns). In our example, U_{n-1} is the system's first attempt to answer the user's query, and U_n is the repair beginning *No, I meant*. Here, U_{n-1} is unacceptable to the user, and U_n attempts to execute a task (the repair). These are the conditions for selecting cell C from the table: the system revises its hypothesis that U_{n-1} executed the task that the user proposed in U_{n-2} and re-defines it as having (inadvertently) introduced a subordinate task (requiring a repair). Consequently, the exchange graph is revised so that an exchange composed of U_n and U_{n-1} is embedded beneath U_{n-2}. Repairs like this one demonstrate that an exchange graph is only tentative--it expresses a hypothesis about the state of the discourse that must be tested and revised to incorporate new evidence (Brennan 1990).

Representing and using the gist of an exchange

In our example, the question that the system eventually answered was one that was never actually uttered by the user: *Where is Dan's cubicle?* This query cannot be reconstructed from any single presentation. Instead, the system computed the gist of the (repaired) task definition in order to send a relevant version query to the database (Cahn 1991). At the point where any one exchange ends successfully, the system summarizes the gist of the exchange into the propositions that it estimates have been grounded (e.g., *Dan's cubicle is near post H33)*. It then adds these propositions to its representation of the dialog so far (which is meant to estimate the common ground it shares with the user). This it does after turn (5) in our example, upon recognizing the user's intention to go on to another domain-level task. The system's interim graphs, which emerge from the strategies detailed in Tables 1 and 2, are shown in Figure 6.

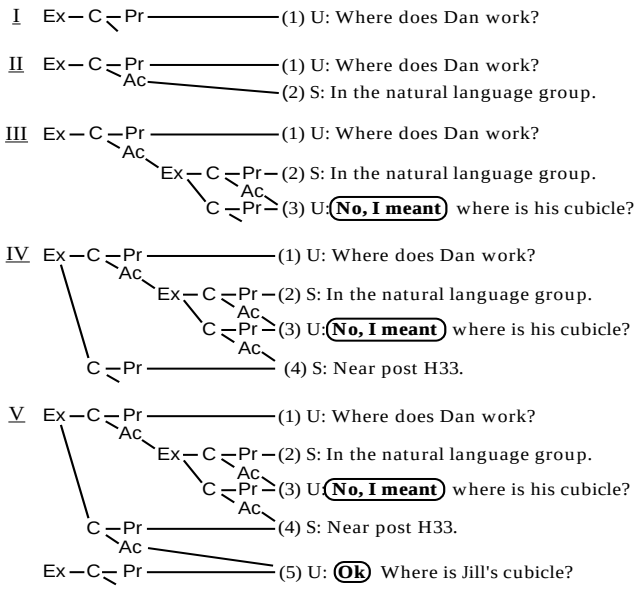

Figure 6: The system's revised graph, step by step

Conclusions and future work

Exchange graphs are detailed and coherent pictures of the interim products of dialog; they provide a basis for a system to estimate what a user represents about the dialog context. We assume that the system should represent successfully grounded utterances rather than dead ends, and the gist of a query rather than the verbatim form of any incremental attempts to formulate it. But exactly what people represent and remember about a dialog is an empirical question. In many situations, people have poor memory for the exact surface form of spoken or written text, but good memory for its meaning or gist (see, e.g., Sachs 1967).

What information should be represented in a dialog model is an open and interesting question. For instance, the rates and types of repairs could be tracked. Brennan and Hulteen (1995) proposed that such information indicates how smoothly a dialog is going, which in turn enables an agent to adjust its grounding criterion and determine both how much detail to provide and how much evidence to seek out. Additional information that could be represented in a dialog context includes given and new entities and their salience, to help the system choose and resolve referring expressions (see, e.g., Sidner 1979). The model could also keep a record of the surface forms of referring expressions and vocabulary used previously in the dialog, to enable the system to produce and expect the same terms that it has converged upon previously with the user in constructing a shared perspective (see Brennan 1996; Brennan and Clark 1996).

Many interesting questions arise when a psychological model is formalized for use in a prototype system. For instance, how should an agent calculate and adjust its grounding criterion when the mapping of evidence onto consequences is not so simple (Traum and Dillenbourg 1996)? What does it mean to package contributions as many short turns versus fewer, longer turns (Brennan 1990)? How is this granularity affected by the communication medium (Brennan and Ohaeri 1999; Clark and Brennan 1991)? Note that the need for grounding in HCI is not confined to language-based interfaces (Brennan 1998); how should multi-modal turns be represented in an exchange graph? We present additional questions and details about our prototype, the exchange algorithm, and its context-free notation in Cahn 1992 and Cahn and Brennan 1999. Finally, we have not conducted any extensive user testing of the kind of interface that we propose here, and such testing, we believe, is important.

We predict that human-computer interaction will be significantly improved by enabling systems to estimate shared dialog context and by enabling users to evaluate and express the relevance of a system's actions. Not only should this reduce frustration: it should better exploit the intelligence already present in abundance in human-computer dialog--the intelligence of the human.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grants No IRI9202458, IRI9402167, and IRI9711974, and by the News in the Future Consortium at the M.I.T. Media Laboratory.

References

Brennan, S. E. 1988. The Multimedia Articulation of Answers in a Natural Language Database Query System. In Second Conference on Applied Natural Language Processing, 1-8. Austin, TX: Association for Computational Linguistics.

Brennan, S. E. 1990. *Seeking and Providing Evidence for Mutual Understanding*. Ph.D. dissertation, Dept. of Psychology, Stanford University.

Brennan, S. E. 1991. A Cognitive Architecture for Dialog and Repair. In *Working Notes of the AAAI Fall Symposium Series: Discourse Structure in Natural Language Understanding and Generation*, 3-5. Asilomar, CA: American Association for Artificial Intelligence.

Brennan, S. E. 1996. Lexical Entrainment in Spontaneous Dialog. In *Proceedings, 1996 International Symposium on Spoken Dialogue* (ISSD-96), 41-44. Philadelphia, PA.

Brennan, S. E. 1998. The Grounding Problem in Conversations With and Through Computers. In S. R. Fussell and R. J. Kreuz, eds., *Social and Cognitive Psychological Approaches to Interpersonal Communication*, 201-225. Mahwah, NJ: Erlbaum.

Brennan, S. E., and Clark, H. H. 1996. Conceptual pacts and lexical choice in conversation. *Journal of Experimental Psychology: Learning, Memory and Cognition,* 6:1482-1493.

Brennan, S. E., and Hulteen, E. 1995. Interaction and Feedback in a Spoken Language System: A Theoretical Framework. *Knowledge-Based Systems*, 8(2-3):143-151.

Brennan, S. E., and Ohaeri, J. O. 1999. Why do Electronic Conversations Seem Less Polite? The Costs and Benefits of Hedging. In *Proceedings, International Joint Conference on Work Activities, Coordination, and Collaboration (WACC '99)*, 227-235. San Francisco, CA.

Cahn, J. E. 1991. A Computational Architecture for Dialog and Repair. In *Working Notes of the AAAI Fall Symposium Series: Discourse Structure in Natural Language Understanding and Generation*, 5-7. Asilomar, CA: American Association for Artificial Intelligence.

Cahn, J. E. 1992. Towards a Computational Architecture for the Progression of Mutual Understanding in Dialog. Technical Report, 92-4, Media Laboratory, MIT.

Cahn, J. E., and Brennan, S. E. 1999. A Psychological Model of Grounding and Repair in Human-Computer Dialog. Manuscript.

Clark, H. H., and Brennan, S. E. 1991. Grounding in Communication. In L.B. Resnick, J. Levine, and S.D. Teasley, eds., *Perspectives on Socially Shard Cognition*, 127-149. Washington DC: APA.

Clark, H. H., and Marshall, C. R 1981. Definite Reference and Mutual Knowledge. In A. K. Joshi, B. L. Webber, and I. A. Sag, eds., *Elements of Discourse Understanding*, 10-63. Cambridge: Cambridge University Press.

Clark, H. H., and Schaefer, E. F. 1987. Collaborating on Contributions to Conversations. *Language and Cognitive Processes*, 2:1-23.

Clark, H. H., and Schaefer, E. F. 1989. Contributing to Discourse. *Cognitive Science*, 13:259-294.

Clark, H. H., and Wilkes-Gibbs, D. 1986. Referring as a Collaborative Process. *Cognition*, 22:1-39.

Creary, L., and Pollard, C. J. 1985. A Computational Semantics for Natural Language. In *Proceedings of the 23rd Conference of the Association for Computational Linguistics*, 172-179, Chicago, IL: ACL.

Grosz, B. J., and Sidner, C. L. 1986. Attention, Intentions, and the Structure of Discourse. *Computational Linguistics*, 12(3):175-204.

Heeman, P. A., and Hirst, G. 1995. Collaborating on Referring Expressions. *Computational Linguistics*, 21(3):351-382.

Litman, D. J., and Allen, J. F. 1987. A Plan Recognition Model for Subdialogues in Conversation. *Cognitive Science*, 11:163-200.

Luperfoy, S., and Duff, D. 1996. A Centralized Troubleshooting Mechanism for a Spoken Dialogue Interface to a Simulation Application. In *Proceedings of the International Symposium on Spoken Dialogue* (ISSD-96), 77-80, Philadelphia PA.

McRoy, S. W., and Hirst, G. 1995. The Repair of Speech Act Misunderstanding by Abductive Inference. *Computational Linguistics*, 21(4):435-478.

Moore, J. D. 1989. Responding to "Huh?": Answering Vaguely Articulated Follow-up Questions. In *Proceedings, CHI '89, Human Factors in Computing Systems*, 91-96. Austin TX: ACM Press.

Nerbonne, J., and Proudian, D. 1988. The HP-NL System. Technical Report, STL-88-11, Palo Alto, CA: Hewlett-Packard Company.

Novick, D. G., and Sutton, S. 1994. An Empirical Model of Acknowledgment for Spoken-Language Systems. In *Proceedings of the 32nd Conference of the Association for Computational Linguistics*, 96-101, Las Cruces, NM: ACL.

Pollack, M. E. 1990. Plans as Complex Mental Attitudes. In P. R. Cohen, J. Morgan, and M. E. Pollack, eds., *Intentions in Communication*. Cambridge, MA: M.I.T. Press.

Russell, A., and Schober, M. 1999. How Beliefs About a Partner's Goals Affect Referring in Goal-Discrepant Conversation. Discourse Processes, 27, 1-33.

Sachs, J. D. 1967. Recognition Memory for Syntactic and Semantic Aspects of Connected Discourse. *Perception and Psychophysics*, 2:437-442.

Schegloff, E. A. 1992. Repair after Next Turn: The Last Structurally Provided Defense of Intersubjectivity in Conversation. *American Journal of Sociology*, 97(5):1295-1345.

Schegloff, E. A., and Sacks, H 1973. Opening up Closings. *Semiotica*, 7:289-327.

Schober, M. F., and Clark, H. H. 1989. Understanding by Addressees and Overhearers. *Cognitive Psychology*, 21:211-232.

Sidner, C. L. 1979. A Computational Model of Co-Reference Comprehension in English. Ph.D. dissertation, MIT.

Svartvik, J., and Quirk, R. 1980. *A corpus of English conversation*. Lund, Sweden: Gleerup.

Traum. D. R. 1994. *A Computational Theory of Grounding in Natural Language Conversation*. Ph.D. dissertation, Dept. of Computer Science, University of Rochester.

Traum, D. and Dillenbourg, P. 1996. Miscommunication in Multi-modal Collaboration. In Working Notes of the AAAI Workshop on Detecting, Repairing, and Preventing Human-Machine Miscommunication, 37-46.

Traum, D. R., and Hinkleman, E. A. 1992. Conversation Acts in Task-Oriented Spoken Dialogue. *Computational Intelligence*, 8(3):575-599.

Walker, M. A. 1993. *Informational Redundancy and Resource Bounds in Dialogue*. Ph.D. dissertation, Dept. of Computer Science, University of Pennsylvania (Technical report IRCS-93-45).

Whittaker, S. J., and Walker, M. A. 1990. Mixed Initiative in Dialogue: An Investigation into Discourse Segmentation. In *Proceedings of the 30th Meeting of the Association for Computational Linguistic*s, 1-9. ACL.

Wilkes-Gibbs, D. 1986. *Collaborative Processes of Language Use in Conversati*on. Ph.D. dissertation, Dept. of Psychology, Stanford University.