Dialog Parsing: from Speech Repairs to Speech Acts

by

Mark G. Core

Submitted in Partial Fulfillment

of the

Requirements for the Degree

Doctor of Philosophy

Supervised by

Lenhart K. Schubert

Department of Computer Science
The College
Arts and Sciences

University of Rochester
Rochester, New York

1999

Acknowledgments

I want to start by thanking my advisor, Len Schubert, for asking the question "What is a dialog parser?" and helping me answer it in this thesis. I also want to thank Len for all his support through the Ph.D. program and all the associated papers and presentations we have put together. I want to thank Peter Heeman for all the advice and help he has given me; I also want to thank him for the use of his speech repair identifier. I want to thank my committee members James Allen and Greg Carlson for helping shape the thesis. I want to thank James for giving me the opportunity to work with him and the rest of the Multiparty Discourse Group in researching dialog annotation. I appreciate the help he and Teresa Zollo have given me in various presentations of this work. I also want to thank our annotators Eva Bero, Kristin Guenther, Katrine Shkavrova, Diana Past, and Teresa. Thanks to Teresa for keeping track of all the annotators.

I enjoyed working with the Multiparty Discourse Group and in particular want to thank Masahiro Araki, Barbara Di Eugenio, Johanna Moore, David Novick, and David Traum with whom I worked closely on various aspects of the Backward- and Forward- Looking annotation scheme.

This work has been supported in part by National Science Foundation grants IRI-9503312 and IRI-9623665. This work was also supported by a National Science Foundation grant IRI-9528998 under a subcontract from Columbia University.

I was also fortunate to work with Chris Darken at Siemens and Jennifier Chu-Carroll and Bob Carpenter at Bell Labs. I will have the privilege of working with Johanna Moore at the University in Edinburgh after I leave Rochester.

I have been fortunate to belong to an excellent NL research group here at Rochester. In particular I would like to thank James Allen, Donna Byron, Myrosia Dzikovska, George Ferguson, Lucian Galescu, Peter Heeman, Chung Hee Hwang, Marc Light, Nat Martin, Brad Miller, Massimo Poesio, Eric Ringger, Len Schubert, Amon Seagull, Amanda Stent, Joel Tetreault, David Traum, and Teresa Zollo. Thanks in particular to James, Donna, George, Eric, Len, and Amanda for the seminars and paper meetings that they've run over the years.

I am also grateful to everyone in the department for making it such a supportive and enjoyable place. In particular I would like to thank the support staff - Tim Becker, Liud Bukys, Dave Costello, Ray Frank, Ann Karcich, Brad Miller, and Jim Roche; the administrative staff - Jill Forster, Peggy Franz, Marty Guenther, Pat Marshall, Peggy Meeker, JoMarie Carpenter, and Elaine Heberle; and the members of my class - Umesh Berry, Aaron Kaplan, George Kardaras, Mike Marchetti, Mauricio Marengoni, Wagner Meira, Colm O'Rian, Terry Riopka, Garbis Salgian, Jim Vallino, and Mohammed Zaki.

All the graduate students and faculty over the years have made Rochester an enjoyable place. I've been fortunate to share an office with some great people, in particular, Bob Wisniewski, Donna Byron, and Brandon Sanders. Thanks to Garbis Salgian, David Ahn, and Eduardo

Pinheiro for being great apartment mates. I've also enjoyed some great times with Rahul Bhotika, Rodrigo Carceroni, Melissa Dominguez, Lucian Galescu, Isaac Green, Chris Homan, Andrea Selinger, Greg Sharek, Amanda Stent, Joel Tetreault, and Jim Vallino. I'm particularlly thankful to David Ahn, Aaron Kaplan, Garbis Salgian, Amon Seagull, Rob Stets, Teresa Zollo, Srini Parthasarathy, and Mike Van Wie. I've very grateful for the following people outside the department, my Mother and step family, Jennifer Cuzdey, Gabriela Galescu, and Andreia Ionescu.

Abstract

There are four major dialog-specific challenges in processing natural language: 1) determining an utterance's speech act, 2) finding utterance boundaries, 3) allowing for the possibility that speakers may continue each other's utterances and interrupt each other, and 4) handling speech repairs and editing terms (*uh, I mean*). We worked with the Multiparty Discourse Group to develop the Backward- and Forward-Looking annotation scheme that unlike many current speech act taxonomies allows utterance multi-functionality to be captured. To help with challenge 2, we use a statistical utterance boundary detector. To handle challenges 3 and 4, we developed a unique parsing framework in which metarules specify allowable forms of phrase breakage and interleaving. A stream of words tagged with their speakers are given to the parser. Second speaker continuations are naturally allowed and metarules allow phrase structure to be formed around second speaker interruptions. Similarly, metarules allow phrase structure to be formed around speech repairs and editing terms. The parser can thus include repairs and editing terms in its output, allowing higher-level reasoning processes to make inferences about hesitations and false starts in the input. We have also shown that the parser can use its knowledge of grammar and the syntactic structure of the input to improve pre-parser speech repair identification.

Table of Contents

List of Tables

List of Figures

1 Introduction

Natural language understanding, specifically dialog understanding, plays a crucial role in machine translation and in conversational agents. In machine translation, parsers are often used to translate a speaker's words into a logical form from which words of a second language can be generated. Two notable examples are the VERBMOBIL system [Görz *et al.*, 1996] and the JANUS system [Lavie, 1995]. In both projects, the translator acts as an intermediary in a dialog between two businessmen trying to schedule a meeting. As this technology improves and spreads to more domains, business costs will be reduced since human translators will not be needed as often, and opportunities will be created where previously language was a barrier. As machine translation becomes more general purpose, it will greatly impact travel as machine translators could act as intermediaries between natives of a country and visitors.

Conversational agents are the next generation of human-computer interfaces. Users will be able to ask the agent questions in spoken natural language and state their problems directly. Users will not have to translate their ideas into DOS-like commands or often hard-to-find menu selections but can instead think out aloud. For example, when formating a report, a user might ask *what if we centered the title and made it bigger?* Interaction with a conversational agent forms a dialog; the agent may give the user suggestions, prompt the user for information, or warn the user of potential problems they may face. The user may also ask for clarification or modify what they said previously. Conversational agents often use parsers to translate the user's words into a logical form that the reasoning processes of the system can understand. Three notable examples of such agents are TRIPS [Ferguson and Allen, 1998], Galaxy [Lau *et al.*, 1997], and CommandTalk [Stent *et al.*, 1999].

Early parsing work focused mostly on the structural ambiguity problem as summarized in [Schubert, 1986]. This problem results from the fact that natural language grammars are ambiguous and lead to many interpretations of input; humans can usually select the right syntactic analysis without trouble, but even today, computers cannot disambiguate unconstrained natural language with high accuracy. In early work, the examples discussed were sentences from text such as *Joe bought the book for Susan* versus *Joe bought the book that I had been trying to obtain for Susan.*[1]

However, machine translators and conversational agents must deal with spoken dialog, not simply written text. To get an idea of the added complications of spoken dialog, compare figure 1.1, an excerpt from the book [Kelley and Pohl, 1990], to figure 1.2, an excerpt from the transcript of dialog d92a-1.1 of the TRAINS-93 corpus, a collection of human-human problem solving dialogs in a railway transportation domain [Heeman and Allen, 1995].

[1] The example is taken from [Frazier and Fodor, 1978]. In the first example, most people would say that *for Susan* attaches to *bought*; the beneficiary of the buying was Susan. In the second example, most people would not make this connection.

The first problem seen in figure 1.2 is that the words are not broken into sentences through punctuation. Because of the absence of punctuation, turns form the basic units of the transcription. Turns are defined as a series of uninterrupted words spoken by one speaker.[2]

The next problem seen in figure 1.2 is that words of the two speakers overlap; the plus signs indicate that the words *okay* and *from* overlap as do the words *mm-hm* and *to*. The words of the two speakers cannot be treated totally separately, because as seen in turn 7, one speaker can continue what the other speaker has started to say. On the other hand, speakers can interrupt each other as seen in turns 10 through 12.

To extract meaning from either excerpt, they must be broken into pieces from which the parser's grammar can form syntactic structures and logical forms. The book excerpt has periods marking the end of sentences, and commas set off syntactic constructions such as lists. Here we consider dialog as composed of utterances; utterances can be sentences but they can also be phrasal answers to questions as well as hesitations (*um*) and conventionalized expressions (*hello*, *let's see*). In addition, utterances are not always complete such as when one speaker interrupts another and the first speaker never resumes the interrupted utterance. Some researchers [McKelvie, 1998] claim that dialog has a looser, more phrasal syntax than text and obeys a different grammar. They use the term utterance to make the point that the units of dialog are formed from a different grammar than units of text.

Given that speakers interrupt each other as well as continue each other's utterances, speaker changes are not perfect indicators of utterance endings. Furthermore, speakers may produce several utterances before another speaker takes a turn. The problem under discussion in figure 1.2 is a "warm-up" problem designed to make sure the speakers understand the domain rather than test their problem solving abilities. Because of the simplicity of the problem discussed, turns are short and in this excerpt have at most one main clause. That is not to say there are no multi-utterance turns. A turn such as turn 6, *okay that me- you want to move oranges there*, has two utterances, *okay* and *that me- you want to move oranges there*.[3] Non-trivial examples come up in dialogs with more difficult problems to solve such as d92-1; an excerpt of this dialog is shown in figure 1.3.

Assuming we can break the text of figure 1.2 into appropriate pieces we still have the problem of editing terms (*um*, *uh*) and speech repairs (the restart of turn 6 after the word fragment *me-* and the repetition of *an* on turn 8) making turn 6, turn 8, and turn 9 unparsable.

If the parser produces a syntactic and semantic analysis for a sentence or utterance from one of these excerpts, it then has to determine the speech act (action) behind the utterance/sentence. The sentences in figure 1.1 are intended to inform the reader about the C programming language. The utterances in figure 1.2 involve a variety of speech acts: providing information and acknowledgment, and making requests for information as well as action. Most texts have one central purpose and usually sentences of the text have that same purpose: general texts inform the reader; contracts make commitments; directions give orders or instructions. In some parts of a contract, terms may be defined and one could think of that as a secondary purpose. However, terms are typically defined at the beginning of the contract and followed by various commitments. Thus, we do not see the constant speech act switching that is present in dialogs as speakers switch between responding to other speakers and forming their own requests and statements.

---

[2] Short interruptions such as *mm-hm* are called backchannel interjections and many definitions of turn state that backchannel interjections do not interrupt turns. With such a definition, turns 9, 11, and 13 of figure 1.2 would be combined into one turn, *okay um engine two from Elmira to Corning*. However, disambiguating backchannel interjections from new turns is nontrivial so we do not treat them specially here.

[3] In our analysis, there are actually three utterances with the speech repair splitting the second utterance into *that me-* and *you want to move oranges there*.

"In C, a string is an array of characters, and an array name by itself is a pointer. Because of this, the concepts of arrays, strings, and pointers are intimately related. A pointer is just an address of an object in memory. C, unlike most languages, provides for pointer arithmetic. Since pointer expressions of great utility are possible, pointer arithmetic is one of the strong points of the language.

Arrays are used when many variables all of the same type are desired. For example, the declaration `int a[3];` allocates space for the three-element array `a`. The elements of the array are of type int and are accessed as `a[0]`, `a[1]`, `a[2]`. The index, or subscript, of an array always starts at 0."

Figure 1.1: Excerpt from page 29 of *A Book on C*, [Kelley and Pohl 1990]

```
turn5  u: so from Corning to Bath by eight a.m. today
turn6  s: okay that me- you want to move oranges there
turn7  u: from corning to bath
turn8  s: okay um what you'll have to do is you'll have to uh pick out
           an uh an engine and schedule a train to do that
turn9  u: okay um engine two
turn10 s: +okay+
turn11 u: +from+ Elmira
turn12 s: +mm-hm+
turn13 u: +to+ Corning
```

Figure 1.2: Excerpt from TRAINS-93 dialog d92a-1.1

```
u: oh that's four hours so wa- we're like screwed as far as tho- those
    two box cars at Bath go like I don- I don't think we s- so to get the
    maximum number of box cars of oranges to bath by seven am I don't think
    we can even use those two at bath because there we because we can't get
    an engine to Bath in time
```

Figure 1.3: Excerpt from TRAINS-93 dialog d92-1

There are four major dialog-specific challenges in processing natural language: 1) finding utterance boundaries, 2) allowing for the possibility that speakers may continue each other's utterances and interrupt each other, 3) handling speech repairs and editing terms, and 4) determining an utterance's speech act. Despite the existence of these unique challenges, most parsers in dialog systems (such as machine translators and conversational agents) are not that different from those used to process written text. How can such parsers be used on dialog? For challenges 1 and 2, interruptions are usually not allowed and often only one utterance can be spoken per turn. In the case of 3, some systems use preprocessing techniques to remove repairs and editing terms; other systems simply treat the input as ungrammatical and may use robust parsing techniques or simply fail to give an analysis. For the fourth challenge, some systems use syntactic information to assign speech acts while there is also much work in statistical speech act prediction/recognition. However, this work all involves sets of speech acts not very different from those proposed in 1975 by Searle [Searle, 1975b]. It is not clear whether this set of speech acts truly captures all the actions an utterance can perform.

In the following subsections, we will discuss in more detail how existing dialog parsers fail to address these dialog-specific challenges and how this limits current dialog systems. We first describe the drawbacks of not allowing multi-utterance turns (section 1.1) and second speaker continuations and interruptions (section 1.2). In section 1.3, we review current parsers and how they deal with speech repairs. The majority of approaches remove reparanda from the parser's input and thus exclude reparanda from further processing. We discuss psycholinguistic studies exploring the different information that reparanda can convey and that is missed by such approaches. In section 1.4, we discuss a major problem with many current speech act taxonomies, their insistence that each utterance be labeled with only one action. In section 1.5, we state our thesis, how utterance boundary detection, parser meterules, and the Backward- and Forward-Looking annotation scheme (henceforth the BF scheme) can solve the problems discussed above. In the Contributions section (section 1.6), we discuss the advantages of our parsing framework and the BF scheme and successes we have had in utterance boundary detection and speech repair identification by including the parser in the process.

1.1 Utterance Boundary Detection

Machine translation systems often allow multiple utterances per turn given that they participate in human-human dialog (although mediated through the machine translator). JANUS [Lavie, 1995] and VERBMOBIL [Görz *et al.*, 1996] are examples of two such machine translators; both use statistical utterance boundary detectors to aid their parsers in processing turns. Conversational agents, on the other hand, often assume users will only speak one utterance per turn and do not address the utterance boundary detection problem. Notable examples of such agents are CommandTalk [Stent *et al.*, 1999], Galaxy [Lau *et al.*, 1997; Seneff, 1992], and TRIPS [Ferguson and Allen, 1998].

Although these agents deal with relatively simple domains such as airline reservations, one could easily imagine complicated scenarios that require users to produce multiple utterances per turn:

```
u: I'd like to go from Rochester to San Francisco
   I need to be in San Francisco August 27th
   I can stay over Saturday night or return anytime
    after 5pm on the 30th
s: okay
```

In fact, in the VERBMOBIL meeting scheduling domain, over 70% of the turns in their corpus of human-human dialogs are more than a single utterance [Kompe *et al.*, 1997], indicating that even in relatively simple domains humans prefer turns longer than one utterance. Thus, conversational agents will need to perform utterance boundary detection.

1.2 Second Speaker Continuations and Interruptions

Current machine translators and conversational agents do not allow one speaker to continue or interrupt another's utterance. This simplifies the utterance boundary detection problem, as changes of speaker can always be considered utterance endings. In a corpus analysis of 31 TRAINS-93 dialogs[4] containing 3,441 utterances and 19,189 words, there were 2521 changes of speaker of which 814 did not end the current utterance.[5] Thus, 67.7% of speaker changes signal utterance ends, rather than 100% as in other corpora. Parsing in general is easier if speakers cannot interrupt each other as the parser does not have to consider whether to ignore a second speaker interruption or include it in the parse of the current speaker's utterance. However, if second speaker interruptions are frequently employed in human-human dialogs, then dialog system designers need to consider supporting them despite the complications they introduce.

In this corpus, 259 utterances overlap with another speaker's utterance and 40 of these 259 cases were continuations or repairs while 153 cases were simple backchannels (*okay*, *yeah*). From this data we can see that 7.5% of utterances were interrupted; 59% of the interruptions were backchannels, 26% were full separate utterances, and 15% were continuations or repairs. If a dialog system is passively listening to a conversation, it will have to deal with backchannel interjections and continuations. Dialog systems that interact with users often have speech recognition failures, and users may want to make direct corrections of system:

```
s: leaving from Cleveland          there are...
u:                        no Chicago
```

Users may also prefer to use continuations for efficiency reasons even when talking to a dialog system.

```
s: you want to book a flight
u:                             from Chicago to Buffalo
```

Thus, architectures for future dialog systems and their parsers should make allowance for the fact that users may want to repair and continue system utterances.

1.3 Speech Repairs

Figure 1.4 shows the form of a speech repair. The *reparandum* is the repaired material which may be followed by one or more editing terms. Editing terms can be filled pauses (*um*, *uh*) or cue phrases (*okay*, *I mean*, *let's see*).[6] Sometimes editing terms appear on their own without

[4] Specifically the dialogs were d92-1 through d92a-5.2 and d93-10.1 through d93-14.1.

[5] Utterances were marked in accordance with the grammar of the parser from the TRIPS dialog system [Ferguson and Allen, 1998] because we later used a modified version of this parser to process the corpus.

[6] Although *um* is a more traditional repair marker, editing terms such as *okay* can also mark repairs as in utterance 46 of trains-93 dialog d92a-3.2: *for okay engine two is at Corning*.

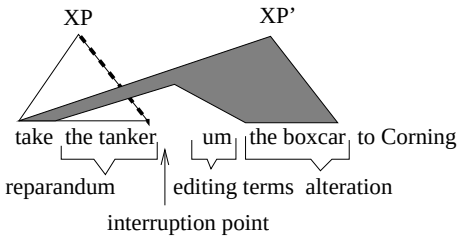


Figure 1.4: Form of a Speech Repair

a repair. These repairs are called abridged since they do not have corrections. When speakers make a non-abridged speech repair, they break off a phrase or sentence, XP and resume it at some previous point resulting in XP'. In some cases, all of XP is corrected and XP and XP' do not share words. The alteration is defined as ending where it completes the smallest phrase straddling the reparandum. These terms are taken from [Heeman, 1997] with modifications to the definition of alteration.

Although different researchers use slightly different terminology, they all specify that repairs consist of repaired material, optional editing terms, and a correction. [Heeman, 1997] notes a possible counterexample, utterance 96 from TRAINS-93 dialog d93-12.4: *we'd be in Elmira at five a.m. five p.m. I m- mean.* However, this may not be a speech repair but instead a more general repair. For example, one could say *oops I meant five p.m.* several utterances after the erroneous one, whereas a speech repair must be an intra-utterance correction. Since the repaired material is topicalized, it would be difficult to utter this correction later in the dialog, but the form of the repair *I mean...* is similar to that of a general repair. In any case such examples are rare and the definition above generally holds. Note, as will be shown in section 3.1, editing terms of a speech repair can themselves be repaired as in *I m- mean.*

Corpus analysis of the TRAINS-93 dialogs in [Heeman, 1997] indicates that 23% of turns contain at least one repair, 54% of turns with at least ten words have at least one repair, and 70% of turns with at least twenty words have at least one repair. Thus, repairs cannot be dismissed as a low-frequency phenomenon that will only disrupt a handful of utterances.

[Heeman and Allen, 1997; Siu and Ostendorf, 1996; Stolcke and Shriberg, 1996b; Nakatani and Hirschberg, 1993; O'Shaughnessy, 1994] describe speech repair detectors that operate either solely on the speech signal or can work with the speech recognizer to use hypothesized words in detecting repairs. Starting speech repair identification at this stage is necessary not only because prosodic features can be extracted directly from the speech signal but because speech repairs can disrupt the word co-occurrences that the speech recognizer's language model depends on.

Should the parser play a role in this process? If repaired material (*reparanda*) and editing terms are simply discarded then the parser does not have to be aware that repairs ever occurred. However, [Heeman, 1997] notes that in the TRAINS-93 corpus 10.1% of the words are part of a reparandum or editing term, so this material is a non-trivial portion of the input. [Smith and Clark, 1993; Brennan and Williams, 1995; Brennan and Schober, 1999; Bortfeld *et al.*, 1999; Fox Tree, 1999; Gibbon and Tseng, 1999] give evidence that this material is useful because it can:

- indicate a heavy cognitive load on the speaker
- indicate speaker uncertainty
- help coordinate the dialog

- cause listeners to judge the speaker as uncomfortable, dishonest, or distant (not familiar)

This suggests that a dialog parser should be aware of editing terms and reparanda and include them in its output to the higher-level reasoning components of a dialog system such as a dialog manager. Editing terms and reparanda could be passed directly to a dialog manager; however, they would be in the form of raw words rather than having a syntactic analysis. A second reason for making the parser aware of reparanda and editing terms is that it can use its knowledge of grammar and the syntactic structure of the input to improve speech repair identification. Below we look at current parsers of dialog to see how they deal with speech repairs and editing terms, and then look further into the role that editing terms and speech repairs play in dialog.

1.3.1 Previous Approaches

Parsers in dialog systems deal with speech repairs and editing terms in various ways. The Galaxy system's [Lau *et al.*, 1997] TINA parser as described in [Seneff, 1992], and the TRIPS system's parser [Ferguson and Allen, 1998] do not address speech repairs at all. [Ferguson and Allen, 1998] does make reference to using robust parsing techniques to handle speech recognition errors and such techniques will help somewhat with the syntactic disruption caused by speech repairs. The JANUS system's parser [Lavie, 1995; Rosè, 1997] also relies on general robust parsing techniques to deal with speech repairs. [Rosè, 1997] states that speech repair identification is planned as future work in the JANUS project, so grouping speech repairs with other syntactic disruptions is not part of their design philosophy. Grouping speech repairs with other syntactic disruptions is inappropriate for several reasons. For robust parsing techniques to handle speech repairs they will have to skip the repair's reparandum; but then they will not know why this material was unparsable and will simply assign a confidence score to the utterance's syntactic analysis based on the number of words skipped. The dialog system may then decide to ask the user to repeat themselves which is exactly what would happen without a robust parser. An additional problem is that the robust parser leaves reparanda out of the syntactic analysis. As will be further explained below, this material can later be referenced and can provide cues to the speaker's mental state. Furthermore, speech repairs are signaled by specific evidence such as pauses and editing terms that a general robust parsing technique could not easily take advantage of.

VERBMOBIL [Ruland *et al.*, 1998] and CommandTalk [Stent *et al.*, 1999] both explicitly handle speech repairs. The input to VERBMOBIL's multiple parsers are word lattices; speech repair detection is performed before parsing by making paths in the lattices that bypass potential reparanda and editing terms. CommandTalk's parser, Gemini [Bear *et al.*, 1992; Dowding *et al.*, 1993], uses pattern matching to identify repairs when ungrammatical input is seen. Neither of these groups discusses what should be done with reparanda other than skipping them to form a corrected utterance.

Some parsers not directly associated with dialog systems have addressed the problem of speech repairs in their input. [Hindle, 1983] assumes that the ends of reparanda are marked by an edit signal. The parser then uses four rules to find the starts of these reparanda. The first rule matches repeated word strings before and after the edit signal. It is applied before the other three rules. The second rule searches for constituents of the same category before and after the edit signal. The third rule also searches for constituents of the same category, but the first may be incomplete. The fourth rule handles reparanda that extend to the start of the utterance (fresh starts). These are assumed to be marked by cue words such as *well, ok, you know*, etc.[7] Hindle states that the reparandum should be available for semantic analysis but

[7] Hindle does not address potential ambiguities where more than one of the last three rules can apply. For example, if the editing term *well* separates two NPs in the middle of an utterance, it is a fresh start?

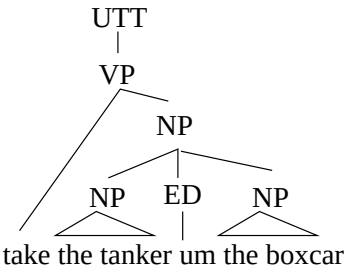


Figure 1.5: McKelvie's Representation of a Speech Repair

removes the material from further syntactic analysis and does not specify how the reparandum might be later analyzed.

The parser described in [Cori *et al.*, 1997] requires repairs to be single syntactic units preceded by an edit signal. Even assuming repairs are marked by an edit signal, there are counterexamples to this requirement such as utterance 11 from TRAINS-93 dialog d93-8.2: *how does the long does that take*. Here *long does that take* is not one syntactic unit, and must be combined with the syntactic structure before the reparandum to form a syntactic unit (an utterance). For each edit signal, Cori *et al.*'s parser searches for a suitable reparandum start where a constituent following the reparandum end can be inserted. Their parser can output a syntactic representation of what the user started to say (a possibly incomplete phrase structure tree) before the repair. However, they do not mention editing terms and furthermore indicate that reparanda are not relevant to a dialog system. It seems unlikely that editing terms would be included in their output.

[McKelvie, 1998] makes rather strong assumptions about allowable repairs and implements repair detection in the grammar rather than the parser itself. Editing terms are assumed to signal cases where one constituent repairs another. Reparanda are constituents in the grammar; the parse tree of the corrected utterance contains both the reparandum and its correction. For a repaired utterance such as *take the tanker um the boxcar*, the repair grammar rule would be the following, `NP -> NP ET NP`, and produce a tree with the reparandum and its replacement as sister NPs. For example, McKelvie would analyze *take the tanker um the boxcar* as shown in figure 1.5. In cases of rules such as `Z -> X|Y`, McKelvie allows an X to be repaired by a Y. Since sometimes speakers break off an erroneous phrase before completing it, McKelvie allows the creation of broken constituents that are missing their rightmost element. For a rule such as `A -> B C`, a broken A can be formed from B if B is followed by an editing term or by another A.[8] Broken constituents can be used in grammar rules just like regular constituents and pass an `abort +` feature value on to constituents of which they are a part.

Although these rules have been used to parse a corpus containing disfluencies, all the parses have not yet been checked to see how many repairs the parser catches and how many repairs posited by the parser are false alarms. McKelvie notes a few of the false alarms he uncovered such as treating the repetition of *you* as a repair in *eh in front of you you should see a stone creek*. McKelvie's restrictions on allowable repairs are more strict than Cori *et al.*'s framework and will prevent some repairs from being analyzed. It will be interesting to see how often McKelvie's and Cori *et al.*'s constraints are broken, to see how many repairs are ill-behaved syntactically.

Cori *et al.* and Hindle both require an external speech repair identifier. McKelvie uses a mix of editing terms and syntactic parallelism to identify reparanda, but without considering prosodic information and word and part-of-speech parallelism he is likely to miss many repairs

[8] In the latter case, an editing term would then not be necessary as a signal of the repair.

and to have a large false alarm rate when he tests his parser. Cori *et al.* and Hindle[9] require the ends of reparanda to be marked, but word and part-of-speech matching might improve their identification of repair beginnings as well.

Neither McKelvie nor Cori *et al.* evaluate their performance so it is difficult to determine the effect of the parser on speech repair identification accuracy. Hindle makes the assumption of a perfectly reliable edit signal so although he gives performance results it is difficult to evaluate them.

[Bear *et al.*, 1992; Dowding *et al.*, 1993] report on two sets of experiments with the Gemini parser. Bear *et al.* report that pattern matching without parser intervention results in a speech repair identification recall rate of 43.6% and precision of 35.4%.[10] Dowding *et al.* report on an experiment where speech repair identification is only applied during parser failure, and repair hypotheses that parse are favored over those that do not. The speech repair identification recall of this experiment is 30.8% and the precision 61.5%. Given that 63.76% recall and 72.54% precision on speech repair identification has been achieved without a parser [Heeman, 1997][11], the case has not yet been made that parsers can improve speech repair identification precision. The fall-back mechanism of only attempting speech repair identification when the parser fails to find a syntactic analysis will neglect repairs that can be included in a parse of the entire utterance. The Gemini parser was tested on airline reservation queries, and switching to a more complex domain will be difficult because syntactic disruptions may be caused by the grammar simply not covering user inputs.

In this section, we have seen two parsers, [McKelvie, 1998] and [Cori *et al.*, 1997], that provide representations of reparanda. However, these parsers make unrealistic assumptions about speech repair structure, use simplified models of speech repair identification (with no use of prosodic information or word or part-of-speech parallelism measures), and do not evaluate the precision and recall of their approaches. [Bear *et al.*, 1992; Dowding *et al.*, 1993] report on evaluating the Gemini parser's ability to eliminate false alarms of a pre-parser speech repair identifier. However, given that state-of-the-art pre-parser speech repairs identifiers outperform the combination of Gemini and its speech repair identifier, it is unclear whether a parser can improve speech repair identification.

1.3.2 Why are Editing Terms and Reparanda Important?

Several studies indicate that editing terms and reparanda can contain useful information. One such study, [Smith and Clark, 1993], gives evidence that "fillers" and "hedges" are correlated with incorrect answers in question-answering experiments, as well as being correlated with low "feeling of knowing" judgments by speakers. Smith and Clark define fillers as cases where speakers "used interjections such as *uh* and *oh*, and sometimes sighed, whistled, or talked to themselves" (p. 34). This definition evidently covers many editing terms, and in fact they say that *uh* and *um* are the most frequent fillers. (However, they do not analyze them separately.) Hedges are brief word sequences expressing uncertainty, such as *I guess*. Though infrequent, these were also associated with incorrect answers. The most frequently observed hedges (*I guess*, 11 examples and *I think*, 4 examples) are again editing terms.[12] In [Clark, 1996], Clark talks

[9] Hindle assumes that reparanda end with a prosodic editing signal. However, a reliable prosodic signal has not been found. Instead the state of the art in speech repair identification such as [Heeman and Allen, 1997] and [Nakatani and Hirschberg, 1993] uses multiple sources of information.

[10] The Gemini parser results are reported on a per sentence basis; i.e., 43.6% of sentences with speech repairs were successfully corrected.

[11] These numbers do not include identification of editing terms. On identifying speech repairs and editing terms, Heeman reports results of 76.79% recall and 65.85% precision.

[12] The hedge, *something* also occurs 4 times in this corpus. It is unclear whether it counts as an editing term.

about other editing terms: *like*, *you know*, and *that is*, stating that they can be used to convey meta-comments. *like* can mean that "I am being approximate"; *you know* can convey upcoming old/mutual information; and *that is* conveys that the current topic is to be further specified.

Smith and Clark also look specifically at the role of *uh* and *um* in their data as well as the London-Lund English conversation corpus. *uh* and *um* often appear after pauses signaling to the listener that the speaker is trying to respond and is not ignoring the question. In Smith and Clark's experimental data, *um* appears after pauses of greater length on the average than *uh*. *uh* and *um* often have following pauses; in the London-Lund corpus *um* is followed more often by a pause than *uh*. In Smith and Clark's experimental data, pauses following *um*s were longer than those following *uh*s. [Brennan and Williams, 1995] repeated this experiment but also measured a passive listener's[13] confidence that the speaker was giving a correct answer. Listeners did not distinguish between *um* and *uh* in their judgments. However, in these experiments, pause length was fixed at 5 seconds for examples with *um* and *uh* (through editing of answer recordings) rather than varying naturally, indicating that the difference between *um* and *uh* is a by-product of their accompanying pauses. Future dialog systems should allow silence to be reasoned about by higher-level processes. Currently it is not, and for the time being systems may need to rely on the difference between *um* and *uh* in measuring speech production difficultly.

In [Brennan and Williams, 1995]'s experiments, examples varied in pause length and in the presence of rising intonation as well as the presence of *um* and *uh*. Subjects marked answers less likely to be correct when following *um* or *uh* than when following a pause of the same length. Since this study also showed that length of pause correlates with perceived uncertainty, it follows that insertion of *um* or *uh* causes listeners to doubt the speaker's answer. Further work [Brennan and Kipp, 1996] explored whether speakers varied their use of *um*s and *uh*s depending on whether they thought the questioner knew the answer or not. This study showed that when the speaker thinks that the questioner knows the answer, the speaker's fillers are more informative about whether they know the answer.

[Fox Tree, 1999] also looked at the role of *um*s in answers to questions. In this case, the questions queried the listener's opinion ("are you here because of affirmative action?" (pg. 16)). Recordings of these questions and their answers were digitally altered so that the interval between the question and answer contained pauses of various lengths and sometimes the word *um*. A set of subjects listened to these questions and answers and rated (1) how well the speakers knew each other, (2) how much speech production difficulty the second speaker had, (3) how deceptive the second speaker was being, and (4) how comfortable the second speaker was with the question topic. For speech production difficulty and comfort with topic, *um*s and long pauses both contributed negatively to listener interpretations and having both was worse than either alone. For familiarity and honesty judgments, an *um* or a pause contributed negatively but having both did not worsen judgments.

In addition to signaling uncertainty, lack of familiarity, dishonesty, anxiety, and speech production difficulty, fillers play an active role in the conversation by signaling that the speaker is trying to form an utterance but is having trouble. [Bortfeld *et al.*, 1999] discusses this turn-taking function of fillers, and notes that after a filler is uttered, listeners can either give the speaker time to finish the utterance or jump in and suggest continuations. Bortfeld *et al.* cite studies showing that disfluencies (speech repairs and editing terms) in general decrease when there is no turn-taking (a monolog) and increase when visual turn taking cues cannot be used, as in telephone conversations.

[Brennan and Schober, 1999; Bortfeld *et al.*, 1999] reports on a study showing that speech repairs and editing terms indicate planning difficulty. In the study, one speaker is the problem solver and the other acts as an assistant. When a speaker switches roles from assistant to

[13] The listener did not hear the question and had to make their judgments based solely on the answer.

problem solver, they generally have a higher rate of speech repairs (repeats and restarts) and editing terms. The problems to be solved involved two participants matching pictures without being able to show each other the pictures. In some trials, the pictures to be matched were photographs of children while other times the pictures were of abstract geometrical shapes called tangrams. Matching tangrams is a more difficult task as judged by average number of words used by subjects in the matching task. The dialogs involving tangrams had more speech repairs and editing terms than the child matching task.

The rate of filler production, though, is actually slightly higher in the child matching task. This distribution suggests that speech repairs and fillers arise from different processes. One hypothesis about these processes is that speech repairs may be somewhat accidental while fillers play an active communicative role (directly signaling processing difficulty and perhaps soliciting help). This hypothesis is supported by the fact that speakers in the problem solver role in these experiments produced more fillers than their assistants.[14]

[Branigan *et al.*, 1999] presents a study of speech repairs in a corpus of Maptask dialogs. In Maptask dialogs, two speakers are given slightly different maps and neither participant can see the other's map. One speaker, the direction Giver, has the job of getting the other speaker, the Follower, to trace a route on his map. In some dialogs, speakers had eye-contact while in other dialogs they did not. Branigan *et al.* measured repair rates as well the percent of words in reparanda. Familarity of speakers did not affect repair rates, but for Givers, familarity coincided with a higher percent of words in reparanda. Branigan *et al.* did not find a direct difference between repair rates between men and women but did not consider fillers in their study. They did find that women had a lower repair rate than men in dialogs where speakers had eye-contact. One interpretation of this result is that some repairs are linked to turn-taking, and that in the eye-contact situation, women pick up visual turn-taking cues alleviating the need for some of these turn-taking related repairs. [Goodwin, 1991] supports such an argument claiming that repairs act to get the listener's attention when eye contact is lost. In Branigan *et al.*'s study, the overall rate of repetition repairs increases in the no eye-contact condition indicating that repetition repairs in particular may used in turn-keeping. Branigan *et al.* also looked at speakers' first attempt at the Giver role for a particular map versus their second attempt. Speakers produced fewer repetitions in their second attempt suggesting that repetitions are a form of hesistation and practice reduces hesistation in this task.

The Giver role involves a heavier planning load as Givers must plan referring expressions that the Follower can use to locate points of the path to be followed. Correspondingly, Givers have a higher repair rate. Givers had a higher percentage of insertion and substitution repairs[15] while Followers had a higher percentage of deletion repairs (where the correction and the reparandum have no connection).

Researchers are just starting to understand the role of reparanda in dialog. In extreme cases, the reparandum is critical to dialog interpretation: *take the oranges take them to Bath.* It may also be the case that reparanda can be used to improve user modeling. Speakers may reveal their intentions through reparanda: *take the oranges um send the engine to Corning.* Here, we can hypothesize that the user wants the engine to pick up oranges. If there are oranges at Corning but no boxcar to transport them, a helpful system could warn the user to pick up a boxcar first. In a tutoring domain this type of information can be valuable; an utterance such as *measure the voltage um current between the wires* displays some confusion about the difference between voltage and current. Reparanda can also tell listeners what a speaker does

[14] Half the participants were married couples. Bortfeld *et al.* in presenting this work allude to the fact that men produce more fillers than women, and as problem solvers in child matching may defer to women assistants (especially their wives) in this task.

[15] As the name implies, insertion corrections consist of repeated words with an insertion. Substitutions are word replacements that have syntactic and semantic connections to the words they replace.

not mean: *the box of screws is on the top no look in the tool box.* Here, we can guess that the tool box is not on the top shelf or the top of something. [Brennan and Schober, 1999; Bortfeld *et al.*, 1999] note that repairs and fillers indicate planning difficultly and uncertainty, a signal that a listener should help such as in the hypothetical exchange below:

```
S: attach the aftercooler to the pump
U: um okay
S: the aftercooler should be the object to the right of the pump
```

[Oviatt, 1994] supports this notion with experiments showing that systems with prompts ("where would you like to pick up the rental car?") result in fewer user disfluencies than systems that simply allow users to specify their requests in an open-ended manner.

Editing terms convey information. Fillers can be used during speech production difficultly to signal that the speaker is still engaged in the conversation but in the process of planning their speech [Bortfeld *et al.*, 1999]. Listeners hypothesize (somewhat automatically) about the reasons for these speech production difficulties. They may think that the speaker cannot produce a fast answer because they are fabricating a deceptive answer or they are uncomfortable with the topic (making it hard to think about or to form a non-embarrassing answer about) [Fox Tree, 1999]. In the case of question answering, a speaker may have trouble remembering an answer causing speech production delays marked by fillers and correlating with incorrect answers and speaker uncertainty about answers [Smith and Clark, 1993; Brennan and Williams, 1995]. The speakers have some control over this process and produce more fillers to signal their uncertainty if they feel they will be judged on their answers [Brennan and Kipp, 1996].

Given all the information that reparanda and editing terms provide, they should be included in the representation that the parser provides to the rest of the dialog system. This representation should be very precise and contain positional information so that the dialog system can see exactly where speech production difficulties (and possibly uncertainty) occurred.

1.4 Speech Acts

Searle [Searle, 1975b] described a taxonomy of actions that an utterance (or a sentence in written text) could perform. An utterance could commit, request, make a statement, etc. Dialog system builders took these speech acts to be mutually exclusive categories. Different research groups have adapted Searle's categories in different ways for their research domains, making it difficult to compare the work of different groups. Two examples are the VERBMOBIL research group [Reithinger and Maier, 1995] and the TRIPS group [Ferguson and Allen, 1998]. It is not clear, for example, how VERBMOBIL's DIGRESS speech act relates to any of the TRIPS speech acts, and TRIPS's ACKNOWLEDGE-APOLOGY speech act is similarly confusing for the VERBMOBIL group. As an aid to dialog system development, research groups have hand-tagged corpora with their speech acts. However, given the differences in speech act taxonomies, it is not clear how researchers could use each other's corpora.

A popular area of research is statistical speech act detection/prediction: [Jurafsky *et al.*, 1997; Mast *et al.*, 1996; Reithinger and Maier, 1995; Finke *et al.*, 1997; Samuel *et al.*, 1997; Chu-Carroll, 1997; Nagata and Morimoto, 1994; Taylor *et al.*, 1998]. [Taylor *et al.*, 1998] predicts the upcoming speech act and uses speech-act-specific language models to improve speech recognition performance. Statistical speech act detection/prediction can also be used to study speech act distributions, how speakers interact through speech acts, and how speech acts are marked. The subject of how *indirect speech acts* are marked is especially important. [Searle, 1975a] defines indirect speech acts as utterances whose intended message is not their literal meaning but instead

a meaning gained through convention or inference. Utterances such as *can you pass the salt?* are conventionally requests for action. Other utterances such as *I am busy Thursday* require inference to determine their meaning ("you cannot meet with me on Thursday because you are busy").

However, if current speech act taxonomies are not capturing all the communicative functions of an utterance, such studies will necessarily give an incomplete picture of how speech acts are made and why. Several researchers [Allwood, 1995; Cohen and Levesque, 1990; Hancher, 1979] suggest that multiple action descriptions are required to capture speech acts unlike the one-speech-act-per-utterance model used by many researchers.

Our goal is to develop a speech act taxonomy that captures the multi-functionality of utterances. However, instead of creating another speech act taxonomy specific to one research group, our goal is to create a high-level taxonomy that can be used to help researchers share data. For example, a high-level category such as STATEMENT might encompass VERBMOBIL's DIGRESS speech act and TRIPS's TELL speech act. Corpora from both research groups could at least be combined into a corpus with high-level speech act labels.

1.5 Thesis

In this thesis, we address the following dialog-specific challenges in natural language processing: (1) determining an utterance's speech act, (2) handling speech repairs and editing terms, (3) handling multi-utterance turns, and (4) allowing for the possibility that speakers may continue each other's utterances and interrupt each other.

To learn how to automatically recognize (and produce) speech acts, we study speech-act labeled corpora. We worked with the Multiparty Discourse Group to develop a set of speech acts called the BF scheme.[16] The BF scheme allows multiple communication functions of an utterance to be labeled unlike many current speech act taxonomies. The Multiparty Discourse Group is composed of representatives from various research groups studying dialog, and the BF scheme was constructed to be compatible with a variety of speech act taxonomies to allow sharing of resources. Currently most research groups train and test their systems solely on their own data.

Handling speech repairs and editing terms involves identifying reparanda and editing terms in the input and constructing a representation that includes them. This second step is necessary since reparanda and editing terms can provide useful information. Most work in speech repair identification assumes that all identification takes places before parsing. The one parser [Bear *et al.*, 1992; Dowding *et al.*, 1993] that performs speech repair identification uses the parser's grammatical knowledge to limit false alarms. However, as previously pointed out, given the present state of the art in speech repair identification, the issue is still open whether the parser's knowledge of grammar and the syntactic structure of the input can improve speech repair identification.

Representation of editing terms and reparanda is a major part of the thesis and we take the next few paragraphs to discuss this in some detail. The two parsers [McKelvie, 1998; Cori *et al.*, 1997] capable of outputting a representation that includes speech repairs and editing terms make unrealistic assumptions about the structure of speech repairs. These approaches are correct in claiming that speakers do not randomly inject reparanda into their utterances.

[16] The Multiparty Discourse Group has convened three times at meetings of the Discourse Resource Initiative (DRI). Note, that in previous work the BF scheme was called DAMSL (Dialog Act Markup in Several Layers); the Multiparty Discourse Group adopted the new name at their last meeting. See the DRI home page for more details: http://www.georgetown.edu/luperfoy/Discourse-Treebank/dri-home.html

However, speakers seem to be able to back up to arbitary positions in their utterance and resume from there, so that the only hard constraints we can assume are that the material before the reparandum end must be a valid but possibly broken constituent and that the material following the reparandum end is a syntactically valid continuation of the material ending at the reparandum start.

Modeling this definition of speech repairs with only a context-free grammar seems to be very difficult if possible at all. We would need rules such as `XP -> XP XP` where the first XP is allowed to be broken. However, we must allow the second XP to be missing words from its left side that appear on the left side of the first XP. We will show that such a rule can be implemented in a context-free grammar's feature system if the number of shared words has a fixed bound. However, the resulting grammar is not practical for a real-time parser, and the fixed bound on shared words is an arbitrary restriction.

Thus, we decided to use metarules to augment the grammar. Our notion of metarules is different from the traditional linguistic conception of metarules as rules for generating new PSRs from given PSRs. From a procedural perspective, we can think of metarules as creating new (discontinuous) pathways for the parser's traversal of the input, and this view is readily implementable. From a declarative perspective, we can consider the metarules as specifying allowable patterns of phrase breakage and interleaving. At the end of a possibly broken utterance constituent, a reparandum end point may be posited; the (speech) repair metarule will allow phrase hypotheses ending at the start of this reparandum to be continued by syntactic structures following the reparandum. Only if the repair's correction successfully continues the utterance (resulting in an utterance constituent covering the input) will be the speech repair be allowed.[17]

Given that reparanda are handled through a metarule, editing terms are also accommodated through a metarule.[18] The editing term metarule allows phrase structure to be formed around editing terms.

Current dialog parsers do not allow one speaker to interrupt the other or make mid-utterance corrections or continuations. Many parsers do not even allow multi-utterance turns (point (3) above). Standard chart parsers can, in principle, handle multi-utterance turns but the possibility of utterance endings at almost every position in the input dramatically worsens the structural ambiguity problem. We use a statistical utterance boundary detector that uses boundary tones, syntactic information, silence, discourse markers, and speaker changes to estimate the likelihood of utterance end points.

In accommodating continuations, corrections, and interruptions (point (4) above), we concentrated on the two-speaker case. We can view the word streams of the two speakers as separate and sometimes connecting, or force an ordering on the words and merge the word streams. We take the latter view since it minimizes changes that must be made to the parser. In such a view, utterances may interleave; a few words of utterance 1 may be spoken by speaker A; speaker B then may speak some of utterance 2; speaker A may resume utterance 1; and so on. The non-interference metarule allows this interleaving. Syntactic structures by one speaker may be continued after an interruption by another speaker.

Speakers can interrupt each other forming chains of utterances (such as the one below). At some point (possibly the dialog end), there will be an utterance ending where there are no other utterances in-progress.

---

[17] A correction may be repaired itself but eventually an unrepaired correction must complete the utterance.

[18] Editing terms can be accommodated more easily than repairs in a context-free grammar. However, we prefer to handle speech repairs, editing terms, and second speaker interruptions in a uniform manner. Furthermore, the representation more closely reflects the eventual output of the parser; editing terms are separate utterances and not included in the surrounding phrase structure.

```
u: utterance1  utterance3  utterance5
s:       utterance2  utterance4
```

At this point, the parser can produce an analysis of the input. A series of interleaved utterances stretching from the start of the input to the end must be found. In addition to utterances in this chain, the parser will output utterances corresponding to any skipped material in the chain. This may consist of editing term utterances, or in the case of reparanda, the parser outputs a possibly broken utterance corresponding to what the speaker started to say before the repair. Each syntax tree output by the parser is annotated with the positions of its subconstituents in the input. Thus, the parser provides a representation detailing the types and positions of editing terms in the input as well as the position of speech repairs and a representation of what the user started to say before each repair.

1.6 Contributions

One of the reasons for developing the BF scheme was to capture speech acts that other taxonomies missed because they only assigned one act to each utterance (see chapter 6 for a full discussion of our work developing and working with the BF scheme). Labeling TRAINS-93 dialogs with the BF scheme gave us some insight into what other speech act taxonomies were missing. The syntactic form (declarative, imperative, interrogative) of an utterance's main clause needs to be annotated because all declaratives seem to have a ASSERT function, all imperatives seem to have an ACTION-DIRECTIVE function, and all interrogatives seem to have an INFO-REQUEST function. In this domain, the two speakers are supposed to agree on a plan to meet the problem goals. Thus, ACTION-DIRECTIVEs and OPEN-OPTIONs (weak suggestions) typically involve joint actions. These were marked as having COMMIT labels in the case of ACTION-DIRECTIVEs and OFFER labels in the case of OPEN-OPTION. The acceptances of such utterances also had COMMIT labels. Thus, speech act taxonomies should make a distinction between standard directives and directives involving joint actions. This corpus annotation was surprising in that more combinations of tags did not show up. And in fact, having eight different dimensions (sets of labels) to apply gave us low interannotator reliability for some dimensions. Future work will involve using a lower dimension BF scheme and annotating other corpora since they could exhibit a wider range of utterance multi-functionality than found in TRAINS-93.

Once a proper set of speech acts is developed, researchers can study how people interact through speech acts to achieve conversational goals as well as how people mark their speech acts through lexical choice, prosody, and syntax. The BF labeled corpus shows insight into how people communicate in TRAINS-93 dialogs. Declarative clauses are often used for meta-level planning (management of the planning process) and for action-directives. The latter fact may seem strange but in this domain directives are often expressed declaratively. For example, the utterance, *the train gets there at 10pm and picks up the boxcars*, might be describing a novel action rather than describing an existing plan. Signaling non-understanding and other clarifications are often done through questions. Most responses to ACTION-DIRECTIVEs are ACCEPTs. One speaker has travel-time information and is supposed to keep track of the current plan. However, it is difficult for this speaker to immediately notice a non-optimal choice. Thus they are likely to initially accept proposals by the first user. Examining more complicated sequences of BF tags should provide additional insights.

[Cori *et al.*, 1997] and [McKelvie, 1998] are the first to discuss the idea of representing speech repairs and editing terms through phrase structure but make simplifying assumptions that are not always correct about the structure of speech repairs. Furthermore, they do not test their formalisms through application to a corpus separately annotated with speech repairs. Our formalism (described in chapter 3) is designed to allow any constituent (including a possibly broken

constituent) to be restarted at any point by a following correction. Although many times the reparandum forms a constituent, there are cases where it does not, suggesting that our formalism is correct in not making any stricter claims. In a corpus of 31 TRAINS-93 dialogs[19], comprising 3441 utterances, 19,189 words, and 495 speech repairs, our formalism accommodated all speech repairs.[20] The formalism allows our parser to construct a representation containing what the user started to say before a repair, types and positions of editing terms, repair frequency, and repair positions.

Previous psycholinguistic research has assigned meanings to various editing terms and explored whether reparanda can be used as a source of information by listeners. The metarule representation of editing terms and speech repairs is rich enough to allow these findings to be implemented in a dialog system. If *um* implies uncertainty, then the dialog system will be able to exploit this information, since it will have access to specific data such as how many times *um* appeared in the input and what phrase immediately followed (and thus may have been what the speaker was uncertain about). Reparanda are available to be reasoned about and the system can access repair positions and repair frequency. A high repair rate suggests processing difficultly; the position of repairs can indicate exactly where the difficultly lies. The reparanda can also indicate what concepts are being confused as well as indicate what the speaker does not mean.

No previous work has discussed a representation allowing speakers to interrupt each other to make continuations or corrections. Our framework allows second speaker continuations and corrections while preventing backchannel acknowledgments and other interruptions from disrupting syntax. The representation shows exactly where backchannel acknowledgments occurred, so the parser output allows inference about what the backchannel utterance is acknowledging. When speakers are talking at the same time, the non-interference metarule will be applied several times, indicating that perhaps the speakers are not listening to each other.

[Stolcke and Shriberg, 1996a; Stolcke *et al.*, 1998] discusses the utterance boundary detection problem resulting from multi-utterance turns and mid-utterance interruptions. Stolcke *et al.*'s utterance boundary detector performs comparably to the utterance boundary detector presented in chapter 4.[21] In addition to measuring our utterance boundary detector's overall performance, we also weighed the predictive power of different pieces of evidence against each other. The evidence tested (syntactic information, boundary tones, silence, discourse markers, editing terms, speaker changes, and word fragments) turned out to be partially redundant. Boundary tones are the most predictive evidence but even without them recall only decreased by 4.06%. Discourse markers are the second most predictive evidence term. Syntactic evidence (the "best" constituent ending at the current word) cuts the false alarm rate to a quarter of its original value. Word fragments did not turn out to be predictive; the rest of the evidence had moderate impact on the precision and recall. We also ran experiments performing speech repair detection before utterance boundary detection. The speech repair information reduced the utterance boundary detector's false alarm rate.[22] Of course, utterance boundaries constrain where speech

[19] Specifically the dialogs were d92-1 through d92a-5.2 and d93-10.1 through d93-14.1.

[20] Given that grammar coverage is 39.7% (section 3.4) we did not verify that what people start to say before a repair is grammatical although possibly broken and that corrections always form grammatical continuations. Future work involves looking closer at cases of parser failure to see if the parser failed to construct syntactic analyses because of grammar coverage or whether one of our constraints was violated.

[21] We ran our detector on two slightly different corpora achieving 75.95% recall and 74.17% precision on one corpus and 78.29% recall and 77.11% precision on the other. In [Stolcke and Shriberg, 1996a], Stolcke *et al.* achieve 76.9% recall and 66.9% precision. If they assume perfect part-of-speech tagging, they achieve better results, 79.6% recall and 73.5% precision, but given that we do not assume perfect part-of-speech tagging or perfect parsing, we cannot compare directly to these results. In [Stolcke *et al.*, 1998], Stolcke *et al.* report 76.31% recall and 79.89% precision using word, change of speaker, and pause information. They do not report results on combining this information with their prosodic classifier.

[22] This is not reflected in the results given in the previous footnote. See chapter 4 for more details.

repairs (intra-utterance corrections) can occur. The two processes are intertwined and should be performed together.

Our work is the first to demonstrate that a parser's grammatical knowledge can improve the recall of a state-of-the-art speech repair identifier [Heeman and Allen, 1997] as shown in chapter 5. The grammar's coverage of the input (39.7%, section 3.4) turns out to be crucial. Increases in speech repair recall due to parser intervention vary from 2.7% to 4.8% on different corpora. On a subset of the data where the grammar has 100% coverage, speech repair recall increases to 8.89% due to parser intervention. The reason the last experiment did not have a larger increase in recall was that the parser favors speech repairs with the smallest reparandum. Thus, if part of a reparandum can be included in the main parse, the parser favors such an analysis, sometimes leading to incorrect answers. Future work must take care to weigh repair probabilities, parse probabilities, and reparandum length to avoid such mistakes. The current experiments use grammaticality judgments to rescore speech repair hypotheses. It may be the case that the parser needs to reason directly about phrase-level parallelism to overcome these problems.

[Bear *et al.*, 1992; Dowding *et al.*, 1993] and [Hindle, 1983] describe the only two parsers evaluated on their speech repair identification performance. Hindle assumes reparanda endings are marked perfectly so we cannot compare directly to his results. The Gemini parser [Bear *et al.*, 1992; Dowding *et al.*, 1993] uses syntactic information to eliminate false alarms from the input with a cost to recall: 43.6% recall and 35.4% precision change to 30.8% recall and 61.5% precision with parser intervention. Given that the pre-parser speech repair identifier used here [Heeman, 1997] has recall 63.76% and precision 72.54% without parser intervention we cannot directly compare results.

We chose to concentrate on increasing recall because the metarules give the parser the option of ignoring posited repairs and including reparanda in the main utterance. In addition, we need to be careful to catch repairs where the reparanda can be included in the main utterance. For example, the utterance *take the tanker the boxcar to Corning* could be considered as having the same form as *take the president Bill Clinton to Washington*. Currently, our parser always prefers syntactic analyses without repairs. In future work, the estimated probabilities of parses with and without repairs will be compared in an effort to differentiate between *the tanker the boxcar* and *the president Bill Clinton*. In our current experiments, it is difficult to say how many false alarms are eliminated by our preference for analyses with the fewest missing words. Sometimes, a reparandum shorter than the actual one is chosen and results in a false alarm. Thus, parser intervention actually results in a slight decrease in precision at present.

If we combined the best features of the various parsers discussed above into one dialog parser, we would still be missing several key elements. Many parsers recognize Searle-like [Searle, 1975b] speech acts based on syntactic clues. Although it is known that more than one of Searle's speech acts may apply to an utterance, there has been little discussion about allowing multiple speech act labels on utterances. The BF scheme is the first step toward a taxonomy of speech acts with relaxed mutual exclusivity restrictions and a specification of how to assign speech acts when more than one seems to apply.

Given that we want the ability to process human-human dialog or allow the user to correct the dialog system, the parser must allow second speaker continuations and corrections but prevent backchannel acknowledgments and other true interruptions from disrupting phrase structure. The parser described in this thesis is unique in this ability. The two parsers claiming to represent speech repairs in their output fail on counterexamples from the TRAINS-93 corpus. The metarule representation presented here has proved adequate for all repairs from our corpus of TRAINS-93 dialogs. [Bear *et al.*, 1992; Dowding *et al.*, 1993] demonstrated that false alarms created by their speech repair identifier could be reduced by parser intervention. We performed experiments with a more sophisticated speech repair identifier, getting higher

precision and recall, and show that the parser can increase speech repair recall. Future work will explore tradeoffs between these two approaches: sacrificing recall for precision or vice versa.

The remainder of the thesis is organized as follows. Chapter 2 covers additional background material; the dialogs used in this work, the TRAINS-93 dialogs, are introduced here along with related annotations. This chapter also discusses pre-parser speech repair identifiers as well as relevant psycholinguistic work on speech repair production and recognition. Chapter 3 describes the parser's metarules, their implementation, and the parser's output. In addition, this chapter discusses results of testing the metarules on a subset of TRAINS-93 dialogs. We also explore the effects of adding metarules to the parser: whether this changes the class of languages recognizable by the parser and a context-free grammar; whether the parser's $O(n^3)$ running time is changed; and actual running time comparisons with and without metarules. In chapter 4, we present a statistical utterance boundary detector that achieves a recall of 75.95% with a precision of 74.17%. In chapter 5, we discuss an in-parser speech repair identifier as well as experiments rescoring a pre-parser speech repair identifier. The latter approach was most successful resulting in increases in recall of 2.7% to 4.8% (and 8.9% on a corpus with 100% grammar coverage). In chapter 6, we introduce the BF scheme, discuss its inter-annotator reliability, describe utterance multi-functionality in the TRAINS-93 dialogs, and give a preliminary analysis of the forms of speech acts and their responses. In chapter 7, we conclude and present future work.

2 Background

In this chapter, we describe the dialogs and dialog annotations used to motivate and test the claims made in this thesis. We also provide additional background information on speech repairs. In chapter 1 we discussed previous work on in-parser speech repair identification. Here, we describe pre-parser speech repair identifiers as well as summarizing relevant psycholinguistic work on speech repair production and recognition.

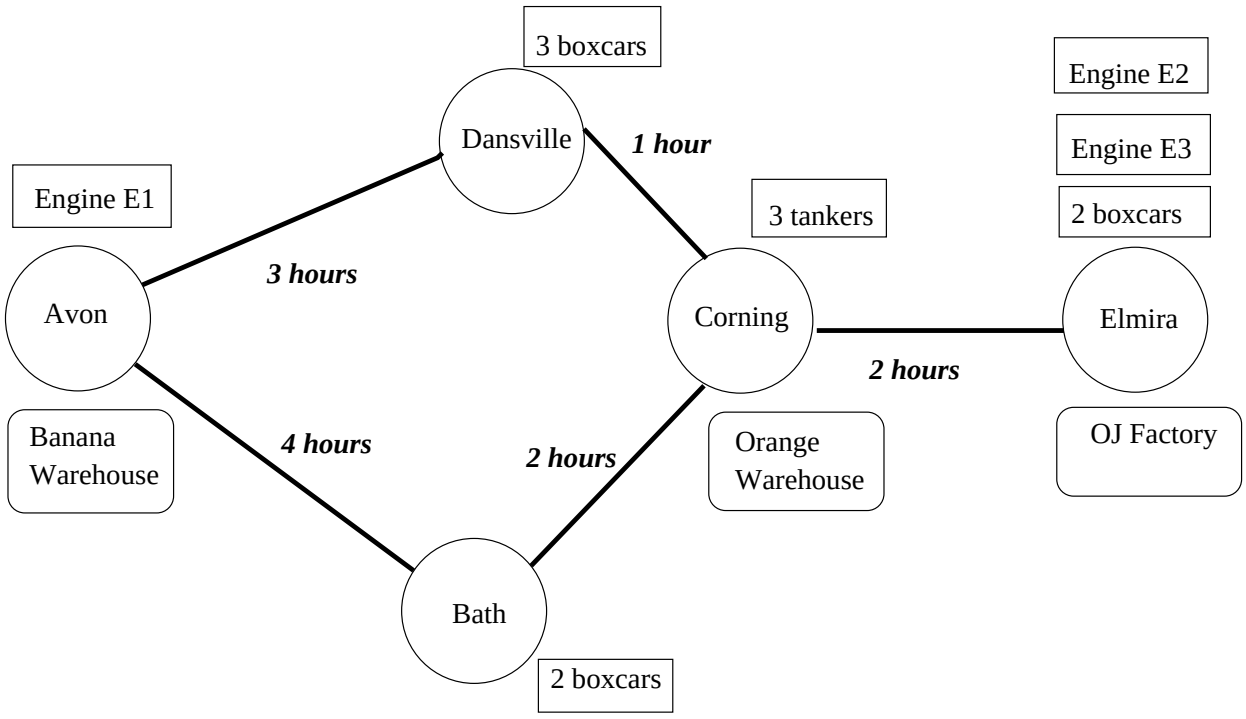
2.1 TRAINS-93 Dialogs and their Annotations

The TRAINS-93 dialogs [Heeman and Allen, 1995] were used to test many aspects of the dialog parser: its basic framework (chapter 3), its utterance boundary detector (chapter 4), and its speech repair identification abilities (chapter 5). In addition, TRAINS-93 dialogs were used to test the BF scheme (chapter 6).

The TRAINS-93 dialogs consist of pairs of subjects who could not see each other, talking over handsets. One subject is labeled the *user* and the other is the *system* although the user knows the system is a person. The user is given the goal of solving a railway transportation problem. The domain of these dialogs is depicted in figure 2.1. Information only known to the system is indicated by bold italics. Typical problems include: "get the maximum number of boxcars of oranges to Bath by seven a.m. given that is now midnight" and "deliver one tanker of orange juice plus four boxcars of oranges to Bath as soon as possible".

All the experiments presented here operate on annotations of the speech signal rather than the speech signal itself. Most of these annotations were done by humans and include word transcriptions and annotations of word fragments, *boundary tones*, utterance boundaries, and speech repairs. Silence and word end points were automatically derived from the speech signal and word transcriptions. Turns are broken into units called intonational phrases defined as stretches of speech with certain intonational patterns. Intonational phrases are required to end with a high or low boundary tone. Given that intonational phrases often end at utterance boundaries or breaks in the utterance (such as speech repairs), these are important evidence in speech repair and utterance boundary detection.

The major simplifying assumption made in this thesis is assuming perfect speech recognition including word fragment recognition. In some of the utterance boundary detection experiments in chapter 4, perfect boundary tone detection is assumed, resulting in 75.95% recall and 74.17% precision. In experiments with no boundary tone information, 72.87% recall and 68.08% precision are achieved. Given the rather low success rates in utterance boundary detection, we used utterance boundary annotations rather than our utterance boundary detector in the speech repair identification experiments of chapter 5. Future work involves using automatic boundary tone and utterance boundary detection as well as using speech recognizer output rather than



More system specific information:

It takes 1 hour to load or unload any amount of cargo on a train.
It takes no time to couple or decouple cars.
Manufacturing OJ: One boxcar oranges converts into one tanker load. Any amount can be made in one hour.
An engine can pull at most three loaded boxcars or tanker cars and any number of unloaded cars.

Figure 2.1: Map of the TRAINS-93 Domain

transcriptions. However, given that current state-of-the-art speech recognizers can have very low recognition rates in free-flowing dialogs, and given that work in this area has just begun, it should not be surprising that we as well as others have had to use word transcriptions. [Shriberg *et al.*, 1997; Nakatani and Hirschberg, 1994; Kikui and Morimoto, 1994; Heeman and Allen, 1997; Siu and Ostendorf, 1996; McKelvie, 1998; Cori *et al.*, 1997; Hindle, 1983; Dowding *et al.*, 1993] all describe speech repair research projects where word transcriptions were used rather than speech recognizer output.

In the TRAINS-93 corpus, the ends of words in transcriptions are aligned with the speech signal. As explained in chapter 3, word ends are used to impose an ordering on the possibly overlapping words of the transcript. If one word ends before another it is counted as coming before the other word. The word stream for a subset of TRAINS-93 dialogs was split into utterances by the author in accordance with the grammar of the TRIPS parser [Ferguson and Allen, 1998]; a modified version of the TRIPS parser was used to parse these dialogs as discussed in chapters 3, 4, and 5. Since utterances may interleave, these dialogs were first broken into dialog units (DUs) with no ongoing utterances at their ends. The first word of a DU starts the main utterance. Other utterances inside the DU are given unique identifiers and words not contributing to the main utterance are tagged with the utterance they belong to. The DU is also annotated with a list of reparanda. Although what the speaker started to say before initiating a repair is treated as an utterance distinct from the speaker's corrected utterance, this is not explicitly annotated but rather is recovered based on the reparandum annotations. We transform the positions of boundary tones and word fragments into word-level annotations; words are marked when they are followed by a boundary tone or word fragment. Words are also annotated with their speaker and the length of any following silence.

A DU annotation is shown in figure 2.2. *right* and *three* are followed by boundary tones and *can* is followed by a silence. *right* is a backchannel acknowledgment, and is labeled as belonging to utterance A rather than the main utterance. The reparandum consisting of the first *we* is marked by the ordered pair, (1 2) in the list of reparanda following the words of the DU.

An additional annotation was added to include the n-best repair hypotheses output for each turn by the speech repair identifier of [Heeman and Allen, 1997] as used in the experiments of section 5.3. The annotation is shown in figure 2.3. Each word is annotated with a list, *repair* that specifies whether the previous word ends a reparandum. Each position in the list corresponds to a different hypothesis. The first element is the most highly ranked hypothesis and the last one is the least favored. Elements labeled n mean the last word does not end a reparandum. A number such as 3 indicates the previous word ends the reparandum of a modification repair and this reparandum is of length 3. The (c 3) element means the previous word ends the reparandum of a fresh start and this reparandum consists of three words.[1]

In experiments 2 and 3 of section 5.3, the speech repair annotations of [Heeman and Allen, 1997] were used. For each DU, this information is encoded in the form of three lists preceding the words of the DU as shown in figure 2.3. These lists correspond to the hits (speech repairs found), misses, and false alarms associated with each of the repair hypotheses in the DU.

The advantage to Heeman and Allen's speech repair annotations is that fresh starts are marked separately from modification repairs. Many times fresh starts can be distinguished from modification repairs by the location of their reparandum beginnings, but this is not always the case. If utterance boundary annotation is done separately from speech repair annotation (as was the case for experiments 2 and 3 of section 5.3), a fresh start may not extend all the way back to the start of the annotated utterance while sometimes modification repairs do extend to the start of annotated utterances. In experiments 2 and 3, we used a speech repair identifier [Heeman and Allen, 1997] that took this difference into account when training and during identification.

[1] These annotations come from Heeman and Allen.

```
(du '(
(make-item
  :speaker 'system
  :word 'so
)
(make-item
  :speaker 'system
  :word 'we
)
(make-item
 :speaker 'system
 :word 'we
)
(make-item
 :speaker 'system
 :word 'can
 :silence '0.106683
)
(make-item
  :speaker 'system
  :word 'do
)
(make-item
  :speaker 'user
  :word 'right
  :tone t
  :utt 'a
)
(make-item
  :speaker 'system
  :word 'three
  :tone t
)
)
'( (1 2) )
)
```

Figure 2.2: Form of Original Annotations

```
(du '(
'(2 2 1 1 1 2)
'(1 1 2 2 2 1)
'(1 1 0 0 1 1)
'(
'(make-item
    :speaker 'user
    :word 'I
    :repair '(n n n n n n)
)
(make-item
    :speaker 'user
    :word 'want
    :repair '(n n n n n n)
)
(make-item
    :speaker 'user
    :word 'ta
    :repair '(n n n n n n)
)
(make-item
    :speaker 'user
    :word 'I
    :repair '(3 3 n n n (c 3))
)
(make-item
    :speaker 'user
    :word 'want
    :repair '(n n n n n n)
)
(make-item
    :speaker 'user
    :word 'to
    :repair '(n n n n n n)
)
...
```

Figure 2.3: Augmented Annotation

During repair identification, the identifier was aware that fresh starts do not necessarily begin at the start of the annotated utterance and that repairs that begin at the start of an annotated utterance may not be fresh starts.

Another issue is the annotation of editing terms. The most frequent editing terms, *um* and *uh*, are not ambiguous given correct speech recognition. The parsing framework presented in this thesis allows the parser the option of forming syntactic structures straddling potential editing terms as well as ones that include them in the main parse. In many cases, only one of these possibilities will result in a syntactic analysis. For example, the two words *you know* cannot be separated out as an editing term in, *you know a good lawyer right?*, but in, *the movie is you know gory*, those two words must be skipped in parsing the main utterance. In future work we will investigate how effective such disambiguation is. In the current project, however, we did not annotate editing terms, and thus did not measure editing term disambiguation accuracy.

[Heeman, 1997] discusses the issue of overlapping repairs, repairs that are correcting some of the same material. We simply annotate the material that each repair removes from the main utterance, although in this case it is not necessarily correct to call this material the reparandum. Heeman notes the example of utterance 4 from d93-16.3: *what's the shortest route from engine from for engine two at Elmira*. The reparandum of the first repair is *from engine* but the correction (*from*) is incomplete. The second repair can be thought of as a second attempt at repairing *from engine* or as a repair of *from*. Our annotations allow either alternative. In speech repair identification, we can ignore previously removed material; in the second repair of utterance 4, we can assume the speaker is making another attempt at repairing *from engine* and ignore the second *from*; or the second *from* can be included in the repair identification context. The context can be *from engine from* or simply *from*. In the speech repair identification experiments of section 5.2, no material is left out of the context. In the experiments of section 5.3 for hypotheses containing multiple repairs, previous reparanda are not used as context for following reparanda.

As detailed in chapter 3, for each repair we create a representation of what the user started to say before the repair. Each of these representations is defined as excluding previous material removed from the main utterance. For utterance 4, one broken utterance would be *what's the shortest route from engine*, a second broken utterance would be *what's the shortest route from*, and the corrected utterance would be *what's the shortest route for engine two at Elmira*.

2.2 Other Speech Repair Related Work

In section 2.2.1, we summarize the corpus analyses of [Levelt, 1983; Shriberg, 1996] to provide background on how speech repairs are structured, when they are made, and how they are distributed. Next we discuss psycholinguistic studies [Lickley *et al.*, 1991; Lickley and Bard, 1996; Lickley and Bard, 1998] that find the earliest point when humans can recognize repairs; these studies also investigate word recognition disruption around repairs and the role of prosody in speech repair detection. In section 2.2.2, we present various pre-parser speech repair detectors (detecting reparanda ends) [Stolcke *et al.*, 1998; Shriberg *et al.*, 1997; Nakatani and Hirschberg, 1994] and correctors (finding reparanda beginnings) [Kikui and Morimoto, 1994]. In section 2.2.3, we discuss speech repair identification in language modeling.

2.2.1 Corpus Analyses and Perception Experiments

[Levelt, 1983] presents a corpus analysis on which much subsequent speech repair work is based. Levelt put forth four claims based on his analysis of task-oriented monologs. His first

claim is that a speech repair's interruption point comes promptly after the speaker detects that words should be corrected. Thus, repairs such as *the man with the umbrella # the woman with the umbrella* will be less frequent than repairs such as *the man # the woman with the umbrella* (# marks the interruption point). This tendency interacts with Levelt's second and third claims. Levelt's second claim is that speakers tend to notice errors in their speech more frequently at constituent boundaries. So in an example such as *over the grey sphere # right of the grey sphere*, the speaker might not have noticed the problem until the end of the PP under construction. Levelt's third claim is that speakers tend to finish words even after detecting an error unless the word itself is incorrect. Levelt's fourth claim is that repairs tend to follow a well-formedness rule: if the reparandum is suitably completed and followed by *and*, then the resulting string is grammatical (assuming the removal of editing terms). Levelt notes counterexamples to the rule and [Heeman, 1997] points out a counterexample in the TRAINS-93 corpus, utterance 81 from d93-10.4: *the two boxcars of orange juice should er of oranges should be made into orange juice.*

Levelt also discussed finding reparandum beginnings based on the first word of the repair's correction. If the word preceding the interruption point is of the same syntactic category as the word starting the correction, then that word is marked as the reparandum. If the word starting the correction is identical to a previous word and has the same syntactic category then the previous word is labeled as the start of the reparandum.[2] These rules apply to 52% of the repairs in Levelt's corpus and produce correct results in 50% of the repairs. In discussing how one might deal with the other 48% of repairs in his corpus, Levelt notes that editing terms can help indicate the extent of repairs. For example, *um* often occurs without a reparandum in what [Heeman, 1997] calls an abridged repair. *sorry* on the other hand occurs more often with reparanda that extend to the start of the utterance.

[Shriberg, 1996] also presents results from a corpus analysis of speech repairs. Shriberg first notes that the probability that an utterance has no repairs decays exponentially with its length. For each of the three domains she studied, Shriberg presents equations approximating the probability of a fluent utterance given its length. Shriberg studied the Switchboard corpus (conversational speech), a corpus of humans talking to travel agents, and the ATIS corpus (humans speaking to an airline travel information system). The probability of a repair decays exponentially with the length of the material the repair edits. The equation $y = cb^k$ (where k is the amount of edited material) can be used to model this decay. There are no significant differences in the constants in this equation between the three domains of study, suggesting that this is a domain-independent property. Another cross-domain finding was that speech repairs were more probable at utterance beginnings. One reason for this could be that in later parts of the utterance less syntactic decisions have to be made and thus fewer errors occur.

In her analysis, Shriberg uses subclasses of repairs called repetitions and deletions; in deletions, there are no correspondences between the correction and reparandum. Shriberg notes that one subgroup of speakers ("deleters") tends to make more deletions than repetitions while another group ("repeaters") has the opposite tendency. [Branigan *et al.*, 1999] found more of a continuum in their corpus with speakers varying in the rates at which they produced repetitions versus deletions. Given these patterns, it may be the case that speaker-specific speech repair models will result in higher accuracy in speech repair identification.

Another way to improve speech repair identification is to use speech recognition difficulties as evidence for a possible repair. In the experiments of [Lickley and Bard, 1998], Lickley and Bard note that human subjects' word recognition rates are lower in disfluent utterances.[3] In

[2] In cases of ambiguity the closest word is chosen.

[3] Lickley and Bard compare subjects' disfluency judgments to their word recognition rates. Word recognition failures do not always coincide with disfluency judgments. Thus, other sources of evidence are needed to model listener's disfluency judgments.

[Lickley and Bard, 1996], Lickley and Bard more closely investigate word recognition errors in disfluent utterances. They performed a word gating study where utterances were presented incrementally (one word was added at each gate). After a gate, subjects wrote down the new word they just heard and possibly made revisions to previous word hypotheses they had written. Late recognition refers to words that are only correctly recognized after one or more subsequent words are heard.

Overall, disfluent utterances had fewer immediate recognitions than fluent utterances and had more word recognition failures. Words in reparanda can be hard to recognize because late recognition will be disrupted by the correction that follows. Indeed, Lickley and Bard found more word recognition failures and fewer late recognitions in reparanda than in similar positions in fluent utterances. In the correction of a speech repair, previous context is disrupted and Lickley and Bard found more failures, more late recognitions, and fewer immediate recognitions in corrections.[4] Words before the reparandum and words following the correction are less affected but still show differences in how they are recognized. The area before the reparandum shows fewer late recognitions because of the disruption in the context following. After the correction there are more late recognitions because of the disruption in previous context.

It is important to note that speech repairs are not always disruptive to human sentence processing. [Fox Tree, 1995] shows that word monitoring latencies are not higher for words following repetition repairs than for the same tokens with the repetitions removed. Although in general other types of repairs can increase word monitoring latencies, this does not seem to hold for repairs whose reparanda extend to the start of the utterance.

[Lickley *et al.*, 1991] shows that prosodic information can be a valuable source of evidence for the presence of a speech repair. Their test materials consisted of fluent and disfluent utterances matched for structure, length, and overall prosody. Following the reparandum end point of the repaired utterances and the equivalent point in fluent utterances, a low pass filter was applied preserving rhythmic and intonational structure but removing segmentational information. Subjects listening to these examples rated the disfluent utterances as more likely to be disfluent despite only being able to hear rhythm and intonation.

[Lickley and Bard, 1998] investigated this claim further by determining when was the earliest point that subjects could reliably detect an interruption point. One experiment Lickley and Bard performed was presenting utterances (with and without speech repairs) incrementally in a word gating study. Subjects were asked at each gate whether they had heard a speech repair and what was the last word that they heard. The word immediately following the interruption point was the earliest location where subjects reliably identified the presence of the interruption point. At this word, subjects made reliably fewer "definitely fluent" responses and more "disfluent" responses indicating they had recognized the interruption point.

To determine this identification point more exactly, Lickley and Bard performed an experiment using gates of fixed duration (35 ms) starting with the word before the interruption point. For each utterance with a speech repair, Lickley and Bard analyzed when the repair was detected and when the word following the interruption point was recognized.[5] The results show that disfluency detection happened before recognition of the word following the interruption point in 66.2% of the cases (failure to recognize the word occurred 12.4% of the time). It is important to realize that subjects did not recognize interruption points with high recall. In the 35 ms gating study, 76.9% of the repairs were found and 6.8% of the controls were falsely labeled as having a repair.

[4] Lickley and Bard define the correction as the direct replacement of the reparandum, or if no such replacement can be found, the correction is the string of words following the reparandum and equal to its length.

[5] Even if the gate was in the middle of a word, subjects were asked to try to guess the word.

	Recall	False alarm rate
Filled Pauses	92.3%	12.9%
Repetition Repairs	83.5%	28.5%
Modification Repairs	77.0%	25.9%
Fresh starts (prosody only)	74.0%	26.0%
Fresh starts (words only)	23.7%	2.3%
Fresh starts (both)	72.8%	17.1 %

Table 2.1: Results from Shriberg *et al.*'s Prosodic Repair Detector

Thus it is likely that humans use the context following the interruption point to increase repair recall. However, a 76.9% recall rate means that prosodic information is an important part of human speech repair detection. The fact that subjects cannot reliably detect repairs before hearing the word following the interruption point presents strong counter-evidence to previous claims of a prosodic repair signal before the word following the interruption point.

2.2.2 Pre-parser Speech Repair Detection and Correction

[Shriberg *et al.*, 1997] investigated whether computational methods could reliably detect speech repairs using only prosodic information as suggested by the corpus analysis of [O'Shaughnessy, 1994] and the perception experiments of [Lickley *et al.*, 1991; Lickley and Bard, 1996; Lickley and Bard, 1998]. Shriberg *et al.* used CART-style decision trees [Breiman *et al.*, 1984] to estimate repair probabilities based on features such as speaker gender, duration (of pauses, vowels, voiced segments), F0 (before and after a boundary, F0 derivative), distance from a pause, and signal to noise ratio. Decision trees were trained to recognize filled pauses, repetition repairs, modification repairs, and fresh starts. The results on filled pauses, repetition repairs, and modification repairs are shown in table 2.1.

For false starts, Shriberg *et al.* also estimated $P(FalseStart|words)$ and combined this with the probability output by the prosody-based decision tree. Results for the prosody model, word-only model, and combined model are shown in table 2.1. The word-only model and combined model use transcribed words, but the false alarm rates are low suggesting that word information could lower false alarm rates even if words are sometimes incorrect.

[Stolcke *et al.*, 1998] continues this work extending its scope to also include utterance boundary detection. Speech repairs are broken into somewhat different categories here: deletions, repetitions, and general speech repairs. In deletion repairs the correction and the reparandum have no correspondences. General speech repairs are those that are not deletions or repetitions. Stolcke *et al.* train a CART-style decision tree that uses prosodic features similar to the ones used in [Shriberg *et al.*, 1997]. Stolcke also constructs two 4-gram word models that include these repair and boundary events. One "word model" has word tokens corresponding to speaker changes and long pauses. This enhanced word model performed the best at recognizing repair and boundary events, and combining it with the prosody-driven decision tree resulted in 93% accuracy on detecting repair and boundary events using transcribed words and 74.9% accuracy using recognizer output.[6]

[Nakatani and Hirschberg, 1994] like Shriberg *et al.* use a CART-style decision tree to estimate repair probabilities. They use as evidence pause duration, presence of word fragments and filled pauses, energy peak, change in energy, F0, change in F0, and whether the current

[6] In these two cases, labeling everything as no-repair/no-boundary results in accuracies of 81.8% and 69.2% respectively.

word was accented or de-accented. Some lexical information is also considered: distance from the start and end of the utterance, total number of words in the utterance, whether the current word reoccurs as one of the three words following, a part-of-speech window of four words around and including the current word, and whether the next word shares the same part of speech as the current word (only to be used if they are both function words). Assuming perfect speech recognition and part-of-speech tagging, Nakatani and Hirschberg obtain a recall of 86.1% and precision of 91.2%. Earlier work [Nakatani and Hirschberg, 1993] excluded part-of-speech and word fragment information resulting in 78.1% recall and 89.2% precision. Given that 73.3% of the reparanda in their corpus were marked by word fragments and that word fragments are hard to detect, both results should be considered in evaluating this work. Note that each of the test utterances in this corpus had a repair, so precision results are bound to be higher than if the model had been tested on full dialogs.

Once a speech repair is detected, its reparandum beginning must be found. [Kikui and Morimoto, 1994] focus on this problem assuming perfect speech recognition, part-of-speech tagging, and speech repair detection. They use an algorithm originally designed for analyzing coordinate structures to find a reparandum start, S defining a reparandum that is most similar to the words following the repair. The actual reparandum start they choose is the one closest to S (with shorter reparanda favored) that results in a well-formed utterance after the reparandum is removed. They test well-formedness using a language model called an adjacency matrix that specifies what lexical categories can follow one another. This technique allows a 92% correction rate given part-of-speech annotations and perfect speech repair detection.

2.2.3 Language Models Allowing for Speech Repairs

[Stolcke and Shriberg, 1996b] use speech recognizer output in their speech repair detector (rather than word transcriptions) and test the hypothesis that language modeling will be improved if repairs are detected. A trigram language model was straightforwardly extended to include *um* and *uh* as well as tokens for one- and two- word repetitions, sentence initial deletions, and other one- and two- word deletions. Tokens such as REP1 (one word repetition) are introduced into the word stream so that trigrams may exist such as $P(REP1|BECAUSE\ I)$. If a filled pause or repair token is hypothesized then appropriate editing occurs before words after the token are considered. In the example *because I I want*, if a REP1 is recognized between the occurrences of *I* then the context for recognizing *want* is *because I* not *because I I*. This language model, however, does not get significantly higher word recognition rates than a standard language model. One problem is that some filled pauses occur at utterance beginnings and are better predictors than words from the previous utterance. Future work will see if detecting utterance boundaries and treating filled pauses differently in utterance-initial position will allow better speech recognition accuracy.

Shriberg and Stolcke [1996] discusses another way that filled pauses can increase speech recognition accuracy. They show that words after a HES event (filled pause or repetition) are harder to predict even after normalizing word history distributions for HES and non-HES examples.[7] Additionally, utterances with HES events are more likely to have n-grams not seen in training. Thus when evaluating word hypotheses that follow a HES, a language model could favor words that are unlikely given the context and disfavor likely continuations. Given that all possible n-grams will not be seen during training, language models assign unseen n-grams non-zero probabilities using various techniques. Given the presence of a HES, a language model could be more liberal in the probability it assigns to these unseen n-grams. However, word recognition experiments will be necessary to see if such changes actually improve speech recognition performance.

[7] The difficulty in word prediction is not all due to word history disruptions caused by the HES.

[Siu and Ostendorf, 1996] extend this work by looking at conversational markers (coordinating conjunctions, discourse markers, the editing phrase *you know*, and filled pauses). For each conversational marker, they studied the perplexity (difficulty of prediction) of words following the marker in sentence initial, mid-repair, and mid-utterance (but not mid-repair) positions. If words following markers in different positions have different perplexities, then markers are split into different words for training. For example *um* would be split into UM-B, UM-R, and UM-M corresponding to *um*'s at the beginning of the utterance, *um*'s in a repair, and all other *um*'s. The best language model for each type of marker depends on the position of the marker. For example, in the middle of repairs a trigram model is better since it captures the correspondence between reparandum and correction in one word repetitions. In the middle of utterances, bigram models perform better for conversational markers than trigrams perhaps because speakers hesitate when what they say next is not obvious from the previous context. Siu and Ostendorf can treat utterance boundaries and repairs as hidden events and test on unmarked data. Using a function-specific n-gram language model (that can vary the amount of context in cases such as when UM-R is the last word in the context), Siu and Ostendorf reduce the perplexity of a standard trigram language model from 82.9 to 81.1 on their test corpus.

[Heeman and Allen, 1997] also achieves perplexity reduction using a language model that allows for speech repairs. As in [Siu and Ostendorf, 1996], hidden events are used to model repairs. In this case there are hidden events for boundary tones (T), editing terms (E), repair type (R), reparandum start (O), word correspondence (L), and word correspondence type (C). The nontrivial values for these event variables (e.g., not no-boundary-tone or no-repair) are defined as follows. E=Push means that the current word starts an editing term, E=ET means that the editing term continues, and E=Pop means that the previous word ends the current editing term. Repair type (R) can be a modification repair (M) or fresh start (C) where fresh starts are restarts of the utterance. R=M means that the previous word ends a modification repair reparandum and correspondingly R=C means that the previous word ends a fresh start reparandum. T is true when a boundary tone precedes the current word. O equals the start of the reparandum (if any) ending after the previous word. L equals the position of the word (if any) that the current word matches or replaces (where both have the same POS tag) in a repair. C=m means that the current word matches a word in a reparandum. C=x means that the current word replaces a word in a reparandum. If we include part-of-speech tags (P), the language modeling problem thus becomes that of determining if there is a boundary tone:

$P(T_i|W_{1,i-1}P_{1,i-1}C_{1,i-1}L_{1,i-1}O_{1,i-1}R_{1,i-1}E_{1,i-1}T_{1,i-1})$

determining the editing term tag:

$P(E_i|W_{1,i-1}P_{1,i-1}C_{1,i-1}L_{1,i-1}O_{1,i-1}R_{1,i-1}E_{1,i-1}T_{1,i})$

determining the repair tag:

$P(R_i|W_{1,i-1}P_{1,i-1}C_{1,i-1}L_{1,i-1}O_{1,i-1}R_{1,i-1}E_{1,i}T_{1,i})$

finding the start of the reparandum:

$P(O_i|W_{1,i-1}P_{1,i-1}C_{1,i-1}L_{1,i-1}O_{1,i-1}R_{1,i}E_{1,i}T_{1,i})$

finding the position of a reparandum word corresponding to the current word:

$P(L_i|W_{1,i-1}P_{1,i-1}C_{1,i-1}L_{1,i-1}O_{1,i}R_{1,i}E_{1,i}T_{1,i})$

determining if this correspondence is a match or replacement:

$P(C_i|W_{1,i-1}P_{1,i-1}C_{1,i-1}L_{1,i}O_{1,i}R_{1,i}E_{1,i}T_{1,i})$

determining the current part of speech (POS):

$P(P_i|W_{1,i-1}P_{1,i-1}C_{1,i}L_{1,i}O_{1,i}R_{1,i}E_{1,i}T_{1,i})$

and determining the current word:

$P(W_i|W_{1,i-1}P_{1,i}C_{1,i}L_{1,i}O_{1,i}R_{1,i}E_{1,i}T_{1,i})$

After editing terms and reparanda are identified they are removed from the context of these equations. The probabilities of these equations are estimated by decision trees. In addition to the context represented in the above equations, the trees can access whether a non-filled pause editing term was seen and whether a proposed reparandum overlaps with any previous repair. POS tags were clustered into a binary tree. Furthermore each POS tag is the root of word tree whose leaves are all the possible expansions of that POS. These trees assign a binary code to all words and parts of speech. The decision tree can ask questions about words and parts of speech that relate to this encoding: "is the second bit of the POS tag equal to 1?".

Because of data sparseness, Heeman and Allen could not simply add silence as another conditioning term in the above equations. If we represent the context in the above equations by $CONTEXT_i$, Heeman and Allen had to assume that silence affects the probability of each tag, T_i independently of $CONTEXT_i$. This assumption allows Heeman and Allen to use the term $\frac{P(T_i|S_i)}{P(T_i)}$ to adjust the above probabilities based on length of the preceding silence.

We can evaluate this model on several levels: boundary tone detection, speech repair identification, and language model perplexity. In detecting mid-turn boundary tones, Heeman and Allen achieve 70.5% recall and 69.4% precision with this model. In detecting all boundary tones, they achieve 83.9% recall and 81.8% precision. In identifying speech repairs, they achieve 63.76% recall and 72.54% precision as reported in [Heeman, 1997]. The reduction in perplexity from a standard trigram language model is 14.1%. The next step in this work is to see whether the perplexity reductions of Heeman and Allen and Siu and Ostendorf actually lead to better speech recognition results. In the case of [Heeman and Allen, 1997] the assumption that word fragments can be perfectly recognized needs to be relaxed by using actual speech recognition output to identify speech repairs.[8]

We use the speech repair identifier of [Heeman and Allen, 1997] in the experiments of chapter 5. Heeman and Allen's results cannot be compared to many of the studies presented here because these studies do not perform speech repair identification. Rather they only perform speech repair detection (i.e., they do not find reparandum beginnings) or only perform speech repair correction (they assume a reparandum end has been identified). Some studies such as [Stolcke and Shriberg, 1996b; Siu and Ostendorf, 1996] identify reparandum intervals as part of language modeling but do not present separate results on this process.

[8] 32% of reparandum ends in the TRAINS corpus are marked with word fragments.

3 Dialog Parsing

Parsers in a dialog system are often not too different from parsers designed for analyzing written text such as Wall Street Journal articles. However, there are some dramatic differences between human-human dialog and text; three of these differences are that (1) speakers may hesitate (*um*) in mid-utterance; (2) speakers may continue each other's utterances as well as interrupt each other; and (3) intra-utterance corrections may be made by the original speaker or by another dialog participant.

The first step in modifying a parser to better accommodate dialog is to allow it to accept words from more than one speaker. Consider the example below. Here the user interrupts the system just after the word *Cleveland* to make a correction.

```
USER:   I want to go to Chicago from New York City
SYSTEM: You want to go to Cleveland              from New York City
USER:                                 No, to Chicago
```

To enter such an example into our parser, we merge the two speakers' word streams into one. Thus, second speaker repair and continuation happen naturally. In the example above, the parser sees *You want to go to Cleveland no to Chicago from New York City.* Words are tagged with their speaker and this example can be treated in the same manner as a self-correction (see below). Sometimes, a second speaker interrupts not to repair or continue but to acknowledge (*mm-hm*) or start a separate utterance. In these cases, the parser will have to be able to form phrase structure around these disruptions.

We will use the term *speech repair* to encompass intra-utterance corrections by either speaker. Hesitations fall under the broader category of *editing terms*, word sequences (*um*, *I mean*) that follow the interruption point of a speech repair. Normally, the interruption point occurs after the reparandum, the corrected material of the repair. However, sometimes editing terms occur in an abridged repair where no correction occurs: *take the um first one.*[1]

For the parser to allow speech repairs and editing terms in its input, it must have a method to form phrase structure around these disruptions. The parser also needs to be able to form phrase structure around second speaker interruptions that are not continuations or corrections. We use metarules to allow the parser to skip over this material (parsing the material separately) or in the case of speech repairs, construct alternative representations (one for the erroneous material and one for the correction). From a declarative perspective, these metarules can be thought of as specifying allowable patterns of phrase breakage and interleaving. From a procedural perspective, we can think of metarules as creating new (discontinuous) pathways for the parser's traversal of the input, and this view is readily implementable. This notion of metarule is different

[1] These definitions are taken from [Heeman, 1997].

from the traditional linguistic conception of metarules as rules for generating new PSRs from given PSRs.[2]

Section 3.1 describes the parser metarules in more detail. Implementation issues are discussed in section 3.2 including questions such as "how much context is necessary for finding speech repairs?" and "how do the metarules interact?". Section 3.3 specifies the output of the parser on examples involving speech repairs, editing terms, and second speaker continuations and interruptions. Section 3.4 discusses the results of running the parser on a test corpus of TRAINS-93 dialogs, and includes sample outputs. In some cases, the parser's grammar did not cover examples from the corpus; in these cases, the metarules were tested by hand. Section 3.5 describes a context-free grammar allowing for speech repairs assuming a fixed bound on the number of words shared between a phrase later restarted and the smallest phrase straddling the reparandum. Future work will involve determining the generative power of a grammar with no such restriction. Section 3.6 discusses extending a $O(n^3)$ chart parsing algorithm to implement the editing term and repair metarules. However the extension presented here has a running time of $O(n^2c^n)$. Future work involves determining if a better algorithm exists. Section 3.7 compares the running time of the parser with and without metarules.

The following point should be kept in mind during this chapter. The TRAINS-93 dialogs are two-person dialogs, and we restrict ourselves to discussing the framework with respect to this case. However, the definitions of the metarules are independent of the number of speakers in the dialog and future work will involve testing them on multi-party dialogs.

The implementation of the dialog parser discussed in this chapter is a modified version of the chart parser in the TRIPS dialog system [Ferguson and Allen, 1998]. Users of this system participate in a simulated evacuation scenario where people must be transported along various routes to safety. Interactions of users with TRIPS were not investigated in detail because they contain few speech repairs and virtually no interruptions.[3] But, the domains of TRIPS and TRAINS-93 are similar enough to allow us to run TRAINS-93 examples on the TRIPS parser.

3.1 Parser Metarules

The corpus to be processed, the TRAINS-93 dialogs, consists of a transcription of words aligned with speech files. These alignments provide the location of word ends in the input. Speech recognizers should also be able to provide such information. The first step in processing input is to merge the two speakers' words into one stream, and tag each word with its speaker. To merge the two word streams, word ends are used to impose an ordering. If word A ends before word B, then word A is counted as occurring before word B. Clearly, this could be inaccurate given that words may overlap. Moreover, speakers may be slow to interrupt or may anticipate the first speaker and interrupt early. However, this approximation works fairly well as discussed in section 3.4.

Merging the two word streams naturally allows one speaker to continue or correct another. If a second speaker starts another utterance (such as a backchannel acknowledgment like *mm-hm*) the parser will be able to form phrase structure around this disruption using the *non-interference*

[2] For instance, a traditional way to accommodate editing terms might be via a metarule,
X -> Y Z ==> X -> Y editing-term Z, where X varies over categories and Y and Z vary over sequences of categories. However, this would produce phrases containing editing terms as constituents, whereas in our approach editing terms are separate utterances.

[3] The low speech recognition accuracy encourages users to produce short, carefully spoken utterances leading to few speech repairs. Moreover, the system does not speak until the user releases the speech input button, and once it responds will not stop talking even if the user interrupts the response. This virtually eliminates interruptions.

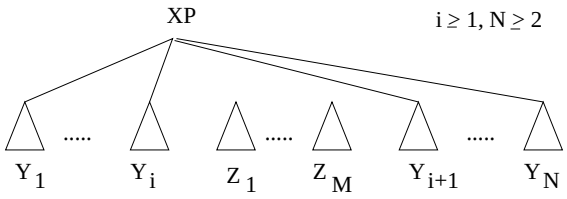


Figure 3.1: The Non-interference Metarule

metarule. Editing terms may occur by themselves in an abridged repair or as part of a repair with a reparandum. The *editing term metarule* will allow the parser to form phrase structure around editing terms in either case. The *repair metarule* will allow a corrected utterance to be formed around a reparandum.

3.1.1 Non-interference Metarule

Given that we have imposed a total ordering on words, changes in speaker are well-defined and occur at word boundaries. For each pair of consecutive changes of speaker, the non-interference metarule allows phrase hypotheses before the first change of speaker to be extended by grammatical structures following the second change of speaker.

Figure 3.1 defines the non-interference metarule graphically. Note that to simplify the diagram, XP is used to label the constituent constructed from Y_1 through Y_N. However, XP can be any constituent not just a phrasal one. XP can be an utterance, sentence, specifier, even an editing term (see section 3.1.2 for details on how editing terms are handled).

Assume a change of speaker follows Y_i and the next change of speaker follows Z_M. The phrase hypothesis for XP ending at Y_i is allowed to be completed with Y_{i+1} through Y_N via the non-interference metarule since Y_i and Y_{i+1} are only separated by the second-speaker interruption, Z. If two more changes of speaker appeared somewhere after Y_{i+1}, the metarule could be applied again allowing XP to be formed even though it had been interrupted more than once by a second speaker.

For two consecutive changes of speaker (call these points A and B), the actual implementation of the metarule in a chart parser makes copies of all arcs (incomplete phrase hypotheses) ending at A and extends them to end at B. This allows one speaker's phrases to skip over interruptions by a second speaker. Note that since the original, unextended arcs remain, it is still possible for one speaker to continue a phrase of the other. Extended arcs are marked with the feature value, **extend** +. All arcs are annotated with pointers to the constituents they contain, and all constituents are labeled with their positions in the input. Thus, exactly what an arc or constituent[4] marked **extend** + is skipping can be recovered.

Figure 3.2 illustrates how the non-interference metarule works. To simplify the example, the VP arcs are omitted and *mm-hm* is assumed to be an utterance. Note that the parser does not store completed constituents such as (`NP -> DET N .`) as arcs. It just adds the constituent to the chart. At the second change of speaker, copies are made of all arcs ending at the first change of speaker and their indices set to end at the current change of speaker. Observe that a complete analysis of a dialog segment is a set of utterances that jointly cover all the words uttered. Here, the analysis would be a parse tree for the UTT from 0 to 5 (with a hole from 2 to 3) and a parse tree for the UTT from 2 to 3.

[4] Arcs pass their feature values on to constituents that they form.

```
u: the train         leaves tomorrow
s:            mm-hm
  0   1     2     3        4         5

NP -> DET . N  from 0 to 1
NP             from 0 to 2
UTT -> NP . VP from 0 to 2
UTT            from 2 to 3
* second change of speaker seen *
UTT -> NP . VP from 0 to 3
VP             from 3 to 5
UTT            from 0 to 5
```

Figure 3.2: Non-interference Metarule in Action

When dealing with an example containing 3 or more changes of speaker, care must be taken to prevent arcs from being copied across all the changes of speaker. We will use the term contribution (contr) here to refer to an uninterrupted sequence of words by one speaker (the words between speaker changes). In the example below, consider change of speaker (cos) 2. Copies of all phrase hypotheses ending at change of speaker 1 are extended to end at change of speaker 2. In this way, speaker u can form a phrase from contr-1 and contr-3 skipping speaker s's interruption, or contr-1, contr-2, and contr-3 can all form one constituent. At change of speaker 3, all phrase hypotheses ending at change of speaker 2 are extended to end at change of speaker 3 except those hypotheses that were extended from the previous change of speaker. Thus, an utterance cannot be formed from only contr-1 and contr-4.

```
u: contr-1          contr-3
s:          contr-2          contr-4
cos        1        2        3
```

An alternative framework for handling a second speaker would be to parse the two streams of words separately and join them when necessary using some kind of continuation metarule. The corpus analysis of section 1.2 examined 3441 TRAINS-93 utterances containing 259 examples of second speaker interruptions. Only 40 of these 259 cases were continuations or repairs. Based on the frequency of true interruptions over continuations or repairs, separately parsing words of the two speakers seems the better alternative. However, the non-interference metarule is very straightforward to implement in a standard chart parser. Like the other two metarules, it merely specifies a section of input that can be bridged by grammatical structures. Parsing the two speakers separately requires a second chart and some mechanism for exploring phrase hypotheses that cross charts. The parsing algorithm would have to be changed to search both charts when trying to extend arcs. We chose to combine the two speakers' words into one stream to avoid changing the parser data structures or parsing algorithm. Given that second speaker interruptions only occur 259 times of out of 3441 utterances in the test corpus, the minor efficiency differences in the two methodologies should not matter in the course of processing an entire dialog.

3.1.2 Editing Term Metarule

[Heeman, 1997] lists the 36 editing terms that occur two or more times in the TRAINS-93 dialogs. Some of these editing terms are lexically ambiguous; for example, *okay* can be an editing

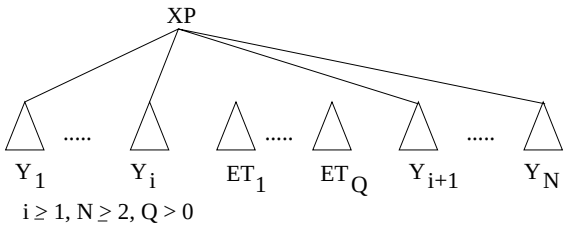


Figure 3.3: The Editing Term Metarule

term or an adjective. The editing term metarule gives the parser the option of forming phrase structure around a potential editing term. However, the parser is still free to include a potential editing term in the current utterance. Thus, we trigger the editing term metarule any time a potential editing term is seen.

The structure allowed by the editing term metarule is shown in figure 3.3. As in the non-interference metarule, XP stands for any constituent; thus XP could be another editing term. Usually XP will not be another editing term, but we did not want apriori to rule out examples such as *I um mean*. XP may be interrupted between two subconstituents by one or more editing terms, and a constituent can be interrupted in more than one location (not shown in figure). To implement the editing term metarule in a chart parser, copies of all phrase hypotheses ending before a potential editing term are extended to end after the editing term. Each of these extended arcs is annotated with a pointer to the editing term constituent that it skips. This annotation is passed on to the constituent formed from the arc.

3.1.3 Repair Metarule

Editing terms suggest the presence of a repair; other clues include ungrammatical input and word repetition. Once the presence of a repair has been hypothesized, possible starting points and end points of the reparandum (the corrected material) must be hypothesized as well. The repair metarule then allows constituents to skip over the hypothesized reparandum.

Figure 3.4 is a picture of the general structure allowed by the repair metarule. XP is a possibly broken constituent ending with Z_L, which may also be broken. It corresponds to the constituent restarted by Z'_1 through P_N. As in the other two metarules, XP can be any constituent, even an editing term. It may be impossible to disambiguate XP, especially if it is very short (e.g., *to*). In such cases, the parser will just have to chose the most likely syntactic analysis of XP. One or more editing terms may indicate the end of the reparandum; these will be skipped via the editing term metarule.[5] The corrected phrase, XP' includes the alteration (the direct correction of the reparandum) and possibly parts of the utterance before the reparandum and after the alteration. The words making up Y, Z, Z', P, and ET, can be spoken by either of the speakers. Usually one speaker will correct himself/herself but there are cases where another speaker helps in the correction. Figure 3.5 shows syntax trees for an example repair: *take E1 to the um E2 to Corning*.

This definition provides no details about the inner structure of XP and XP' but with high probability the reparandum and alteration consist of a series of phrases (Z and Z') that are immediate subconstituents of XP and XP'. In modification repairs, there is also a strong tendency

[5] Although not implemented as a preference in our parser, it should be noted that if an editing term is being repaired then it is unlikely that editing terms intervene between the reparandum and correction.

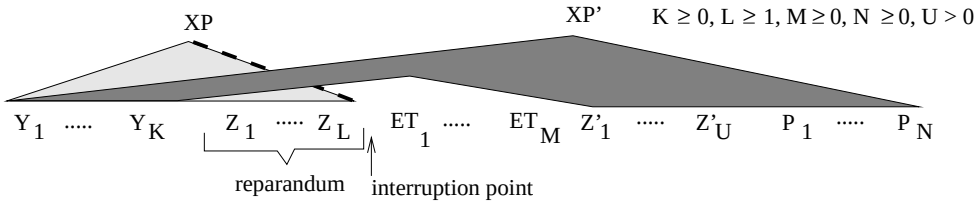


Figure 3.4: The Repair Metarule

for Z_i and Z_i' to be of the same type.[6] In all types of repairs, Levelt's rule [Levelt, 1983] tends to hold: if Z is suitably completed then *Z and Z'* should form an allowable coordinate structure.

The reason the definition of the repair metarule is somewhat vague is that there are exceptions to the above tendencies. Consider this example from TRAINS dialog d93-8.2: *how does the long does that take*. Here, XP is a wh-question where part of the initial wh-phrase is repaired along with the auxiliary verb, the subject, and the VP.

The chart parser implementation of the repair metarule is similar to the implementation of the other metarules. In this case, the reparandum is the disruption around which the parser needs to be able to form phrase structure. So, copies of arcs ending before the reparandum are extended to end after the reparandum, allowing phrase structure to bridge the disruption. Each of these arcs is labeled with the feature value **repair** + and this feature value is passed on to the constituent formed from the arc. Constituents are labeled with their subconstituents and starting and ending positions. Thus, material skipped due to repair can be recovered from the parser's output.

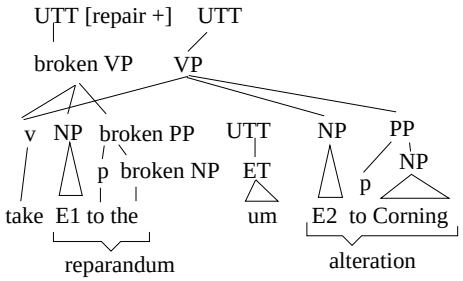


Figure 3.5: Sample Repair

3.2 Implementation

The previous section gave some details as to how the metarules are implemented in a chart parser. For the repair metarule, however, the problem of conjecturing probable reparanda in the input needs to be addressed, to avoid overgeneration of repair hypotheses. Chapter 5 discusses how to determine whether a word is likely to be the last word of a reparandum and if so, how to find the start of that reparandum. However, the process requires some lookahead. Section 3.2.1 discusses when to apply the repair metarule given that lookahead is needed. Section 3.2.2 describes how the three metarules interact.

[6] In an informal survey of the modification repairs in the corpus examined in section 3.4, only 4% of repairs did not follow these two tendencies.

3.2.1 When the Repair Metarule Should Be Applied

One of the most difficult parts of dialog parsing is detecting the starts and ends of reparanda. To determine if a word w ends a reparandum, lookahead is needed since editing terms may follow and increase the probability of w ending a reparandum. Furthermore, parallelism between material (words, parts of speech, syntactic structure) before and after w affects the probability that w ends a reparandum. In the example, *take the oranges no the bananas to Corning*, the repetition of *the*, the similarity between *oranges* and *bananas*, and the syntactic parallelism (both *the oranges* and *the bananas* are NPs) help indicate that *the oranges* is a reparandum.

Chapter 5 discusses two in-parser speech repair identifiers. Both were given utterance boundaries to constrain their search, and reparanda detection was done in a postprocessing phase once the utterance end was reached. The second speech repair identifier used information from the pre-parser speech repair identifier of [Heeman and Allen, 1997]. Heeman and Allen's identifier was run in batch mode on the test corpus producing an n-best list of repair hypotheses for each of the turns in the input. Theoretically, this process has the entire turn as context although the size of the search space dictates heavy pruning as the turn is processed. Based on the parsablity of the input with different repair hypotheses, the parser chooses from Heeman and Allen's n-best list. Heeman and Allen's module could be modified to work interactively and produce repair hypotheses as soon as an utterance ends or any point in the input. The first in-parser speech repair identifier could also be modified to work interactively.

Applying the repair metarule as soon as possible is beneficial to a best first parser. The parser's task as defined in section 3.3 is to produce a parse of the corrected utterance and a syntactic analysis of what the speaker started to say. Often, if there is a repair, a complete parse of the input will not be a possible before the repair metarule is applied. In such a case, if the repair metarule is not applied until after a best first parser is done processing the input, the parser will end up performing an exhaustive search of the possible parses of the input instead of finding best parses of the corrected utterance and what the speaker started to say. The repair metarule is the parser's mechanism of allowing for the possibility of a repair being present. If it is not applied until after parsing, that means the parser will not consider the possibility of a repair until after it exhausts all the possible fluent parses.

Implementing incremental speech repair detection is tricky because it is difficult to determine how much lookahead is necessary to avoid a large number of false alarms. A threshold on repair probabilities is one way to determine when to apply the repair metarule. If the probability of a reparandum end following a certain word is high enough then the repair metarule will be applied. Experiments need to be run to determine if such a simplistic threshold can provide incremental speech repair detection without causing a significant slow-down due to false alarms.

3.2.2 How the Metarules Interact

Even if one attempts to apply the repair metarule incrementally, it will always be applied after the two other metarules (with respect to excluding a word w from the main utterance) since they apply instantly once an editing term or two consecutive changes of speaker have been seen. Thus, the repair metarule can extend arcs created by the editing term and non-interference metarules. Care needs to be taken to allow the opposite case, allowing arcs created by the repair metarule to be extended by the editing term and non-interference metarules. The parser must keep track of where in the input the editing term and non-interference metarules have been applied to determine if they can possibly extend arcs later created by the repair metarule. The parser also needs to ensure that multiple applications of the repair metarule interact correctly. Problems can occur if the repair metarule is first applied to a reparandum going from point B to point C and then applied to a reparandum going from A to B (see example below). In such

a case, the parser must allow copies of arcs to be extended, not only from A to B but A to C as well.

```
u: pick up the boxcar of o- orange orange juice
                         A  B      C
```

A word w may be an editing term (e.g. *uh*) or complete an editing term (e.g., *mean*, in the term *I mean*). If processing w results in an editing term being added to the chart, the editing term metarule will extend copies of all arcs over the editing term. After processing w, the parser will consider whether a change of speaker follows w and if so, whether the non-interference metarule should be applied. If speech repair detection is being done incrementally, the parser will then look back in the input to see if the probability of a reparandum end is above threshold. If so the repair metarule will then be applied.

The parser should be sure to not accidently duplicate its efforts by applying two metarules to the same set of words. In the example below, a second speaker interrupts a first speaker by saying *mm-hm*. The editing term metarule extends copies of all arcs before *mm-hm* to end after it.

```
s: take the engine     to Corning
u:                mm-hm
```

At the change of speaker following *mm-hm*, the non-interference metarule should realize that arcs have already been extended over *mm-hm* and not duplicate the editing term metarule's efforts. In the previous sections, we remarked that the editing term metarule annotates these extended arcs with the editing term being skipped. The non-interference metarule marks extended arcs with the feature value **extend** +. In the case where both metarules apply, arcs should be annotated in both ways. Such an annotation will show that *mm-hm* is an editing term uttered by another speaker (an acknowledgment) and not a self-acknowledgment nor just a standard second speaker interruption.

In the case of a speech repair with editing terms, the speech repair identifier needs to be aware that editing terms should not be included as the end of a reparandum. Such an action would duplicate the efforts of the editing term metarule and complicate the process of analyzing what the speaker started to say before the repair.

The repair metarule and the non-interference metarule can apply to the same material if a second speaker tries to continue a first speaker's utterance but the first speaker rejects the continuation (such as in the example below). Note another analysis of this example is that u simply ignores s and there is no repair. Both analyses can be generated without duplication of effort. The non-interference metarule will allow u's utterance, *take the tanker to Corning*, to be parsed. The non-interference metarule extends arcs across *the boxcar* and marks these arcs with the feature value **extend** +. The repair metarule then does not have to extend arcs for the parser to consider *the tanker* as a correction to *the boxcar*. Instead arcs extended across the *the boxcar* by the non-interference metarule simply have to be also marked *repair* +.

```
u: take          the tanker to Corning
s:     the boxcar
```

3.3 What is a Dialog?

Treating repairs and other disruptions of phrase structure as extra-grammatical simplifies the definition of "utterance" since interruptions do not necessarily end an utterance. We will

use the term "utterance" to refer informally to a sentence, a single phrase (question answer), acknowledgment, editing term, or short conventionalized saying (*good bye*).

The input to the parser will not just be a series of sequential utterances. When a repair is made there will be a corrected utterance as well as a possibly broken-off utterance consisting of what the speaker started to say. These utterances may overlap and share words. Another speaker may start a second utterance while the first speaker is still finishing the original utterance. In some of these cases, the first speaker may not resume their original utterance; these broken-off utterances become part of the dialog analysis. Editing terms such as *um* will form embedded utterances inside the main utterance. This section specifies the output of the parser on such examples as well as specifying allowable input.

From a traditional parsing perspective, a text is a series of sentences to be analyzed. An analysis for a text would be a series of parse trees and logical forms, one for each sentence. An analogous view is often taken of dialog; dialog is a series of adjacent, disjoint "utterances" and a dialog analysis is a series of parse trees and logical forms, one for each successive utterance. Such a view either disallows editing terms, repairs, interjected acknowledgments and other disruptions, or else breaks semantically complete utterances into fragmentary ones. We analyze dialog in terms of a set of utterances covering all the words of the dialog. As is apparent from the metarules discussed earlier, and as explained further below, utterances can be formed by more than one speaker and the words of two utterances may be interleaved.

Speakers are allowed to interrupt their utterances with editing terms. These editing terms are regarded as separate utterances. At changes of speaker, the new speaker may: 1) add to what the first speaker has said, 2) start a new utterance, or 3) continue an utterance that was left hanging at the last change of speaker (e.g., because of an acknowledgment). Note that a speaker may try to interrupt another speaker and succeed in uttering a few words but then give up if the other speaker does not stop talking. These cases are classified as incomplete utterances and are included in the dialog analysis.

Except in utterances containing speech repairs, each word may belong to only one utterance. We form two syntactic analyses of an utterance with a speech repair. One analysis includes all the words up to the reparandum end but stops at that point; this is what the speaker started to say, and will likely be an incomplete utterance. The second analysis is the corrected utterance and skips the reparandum. In the example, *you should take the boxcar I mean the tanker to Corning* the reparandum is *the boxcar*. Based on our previous rules the editing term *I mean* is treated as a separate utterance. The presence of the speech repair causes two analyses to be formed: the utterance, *you should take the tanker to Corning*, and the utterance, *you should take the boxcar*.

For each utterance in the input, the parser needs to find a syntactic analysis that starts at the first word of the utterance and ends at the last word. When repairs are present, this syntactic analysis is based on one or more applications of the repair metarule, allowing the analysis to exclude one or more reparanda. For each reparandum skipped, the parser needs to find an analysis of what the user started to say. In some cases, what the user started to say is a complete constituent: *take the oranges I mean take the bananas*. Otherwise, the parser needs to look for an incomplete analysis ending at the reparandum end. Typically, there will be many such analyses; the parser searches for the longest analyses and then ranks them based on their category: UTT > S > VP > PP, and so on. The incomplete analysis chosen may not extend all the way to the start of the utterance, in which case the process of searching for incomplete analyses is repeated. Of course the search process is restricted by the first incomplete constituent. If, for example, an incomplete PP is found then an additional incomplete constituent would have to expect a PP.

Figure 3.6 shows an example of this process on utterance 62 from TRAINS dialog d92a-1.2. Assuming perfect speech repair identification, the repair metarule will be applied from position

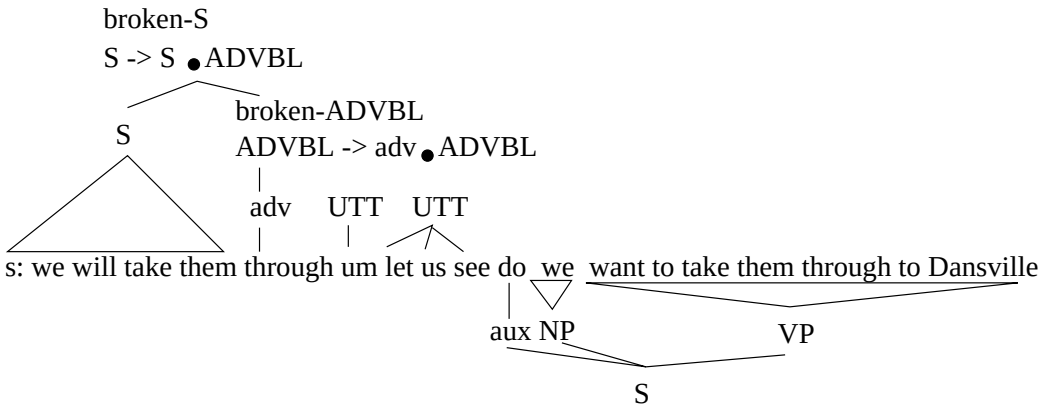


Figure 3.6: Utterance 62 of d92a-1.2

0 to position 5, meaning that the parser needs to construct a syntactic analysis starting at position 5 and ending at the last position in the input. This analysis (the corrected utterance) is shown under the words in figure 3.6. The parser then needs to construct an analysis of what the speaker started to say. There are no complete constituents ending at position 5. The parser instead finds the arc `ADVBL -> adv • ADVBL`.[7] This arc only covers the word *through* so another arc needs to be found. The arc `S -> S • ADVBL` expects an `ADVBL` and covers the rest of the input to the left, completing the analysis of what the user started to say (as shown on the top of figure 3.6). The editing terms are treated as separate utterances via the editing term metarule.

This prototype implementation does not check feature value restrictions before joining arcs. Such checking would result in better reconstructions of what the user started to say. However, in some cases, the parser will not find a chain of arcs (that agree on feature value restrictions) covering the input up to the reparandum end. Even if feature value restrictions are ignored, sometimes other types of robust parsing techniques will need to be employed to construct an analysis of what the user started to say. For example, discontinuous parts of the reparandum could be joined in order to form a syntactic analysis, or template filling could be used to try and match verbs to their arguments. In the worst case, a list of the verbs, NPs, and modifiers in the input could be returned to give some indication of what the user started to say.

When multiple repairs occur, the repair metarule allows the corrected utterance to skip multiple reparanda. For each repair the parser must construct a representation of what the speaker started to say. Currently the parser can only do this for the first repair. For repairs that follow, the parser needs to exclude previous reparanda to get an analysis of what the speaker conveyed before the current repair. Overlapping repairs (where a correction itself is corrected) can be handled in the same manner. Consider an example discussed in [Heeman, 1997], utterance 4 from d93-16.3: *what's the shortest route from engine from for engine two at Elmira*. The corrected utterance is *what's the shortest route for engine two at Elmira*. What the user started to say before the first repair is *what's the shortest route from engine* and what they started to say before the second repair is *what's the shortest route from*.

The parser must also be aware that the possibly broken utterance ending with the reparandum may be an embedded utterance and not start at the beginning of the input. For example, if an editing term is repaired *take them let let us see to corning* then the partial utterance before the repair is *let* not *take them let*.

The parser computes a logical form for each utterance in tandem with a syntactic analysis.

[7] Of course there are many arcs ending at position 5 including `PP -> p • NP`. The parser must disambiguate these possibilities and pick one arc to consider.

Functioning as part of the TRIPS dialog system [Ferguson and Allen, 1998], the parser sends these analyses to the system's dialog manager and clears its chart in anticipation of the next batch of input. An alternative architecture described in [Core and Schubert, 1997] is to treat the parser's chart as a resource shared with the dialog manager. Such a dialog chart would eliminate the need for the parser to send messages containing its entire output, and the dialog manager would not have to maintain a separate representation containing these analyses.

Many parsers (including the one presented here) use syntactic clues to help mark interrogative, imperative, and declarative utterances as possible questions, suggestions, requests, and informs. Working with a dialog chart, the parser could also mark inter-utterance relations such as grounding. Given that the positions of second speaker interruptions are represented in their parse trees, the parser can tell if an acknowledgment such as *mm-hm* appears in the middle of an utterance and thus does not ground all of the utterance's content.

The parser will make errors in structural disambiguation and utterance boundary detection. If the chart is a shared resource, a dialog manager could query it for alternative parses in cases where the dialog manager could not make sense of the parser output. One out of every 4 predictions made by the best statistical utterance boundary detector of chapter 4 is a false alarm. Thus, the chart should not be cleared after posited utterance boundaries because the parser may find that the current utterance continues past a posited boundary.

3.4 Verification of the Framework

To test this framework, data was examined from 31 TRAINS-93 dialogs[8] [Heeman and Allen, 1995], containing 3441 utterances,[9] 19,189 words, 259 examples of overlapping utterances, and 495 speech repairs. The first assumption to be checked was whether ordering the words of the two speakers based solely on word end points was reasonable. The second assumption is that the metarules handle all patterns of speech repairs, editing terms, and second speaker interruptions. For this corpus, these two assumptions held.

3.4.1 Overlapping Speech

The framework presented above covered all overlapping utterances and speech repairs with three exceptions. Ordering the words of two speakers strictly by word end points neglects the fact that speakers may be slow to interrupt or may anticipate the original speaker and interrupt early. The latter was a problem in utterances 80 and 81 of dialog d92a-1.2 as shown below. The numbers in the last row represent times of word end points; for example, *so* ends at 255.5 seconds into the dialog. Speaker *s* uttered the complement of *u*'s sentence before *u* had spoken the verb.

80 u:	so	the		total	is
81 s:			five		
	255.5	255.56	255.83	256	256.61

However, it is important to examine the context following:

[8] Specifically, the dialogs were d92-1 through d92a-5.2 and d93-10.1 through d93-14.1.

[9] This figure does not count editing term utterances nor utterances started in the middle of another speaker's utterance.

82 s: that is right
s: okay
83 u: five
84 s: so total is five

The overlapping speech was confusing enough to the speakers that they felt they needed to reiterate the content of the overlapping utterances. The same is true of the other two such examples in the corpus. It may be the case that a more sophisticated model of interruption will not be necessary if speakers themselves cannot follow completions that lag or precede the correct interruption area.

3.4.2 Parsing the Corpus

In addition to manually checking the adequacy of the framework on the cited TRAINS-93 data, we tested a parser implemented as discussed in section 3.1 on the same data. This parser is a modified version of the one in TRIPS system [Ferguson and Allen, 1998]. Although both TRIPS and TRAINS-93 deal with transportation, TRIPS users keep their utterances fairly simple (partly because of speech recognition problems) while humans talking to each other in the TRAINS-93 domain felt no need to observe such restrictions.

In a random sample of 100 utterances from the corpus (with all reparanda removed), the parser was able to produce a syntactic analysis of 65 of the utterances. However, 37 of these utterances were one word long (*okay*, *yeah*, etc.) and 5 utterances were question answers (*two hours*, *in Elmira*); thus on interesting utterances, likely to have repairs, grammar coverage is 39.7%. Parser accuracy is a stricter measure than grammar coverage and only records cases where the parser output the correct syntactic analysis for an utterance. Parser accuracy on this 100 utterance test set is 62% and 34.5% on the interesting utterances. Assuming perfect speech repair detection, the parser accuracy on the corrected utterances of the 495 speech repairs in the corpus is 25.3% (125 correct examples).[10]

Of the 259 overlapping utterances, 153 were simple backchannel utterances consisting only of editing terms (*okay*, *yeah*) spoken by a second speaker in the middle of a first speaker's utterance. If the parser's grammar handles the first speaker's utterance these can be parsed, as the second speaker's interruption can be skipped. The experiments focused on the 106 overlapping utterances that were more complicated. In only 24 of these cases did the parser's grammar cover both of the overlapping utterances. One of these examples, utterances utt39 and 40 from d92a-3.2 (see below), involves three independently formed utterances that overlap.

s: when it would get to bath
u: okay how about to dansville

We have omitted the beginning of *s*'s utterance, *so that would be five a.m.* for space reasons. Figure 3.7 shows the syntactic structure of *s*'s utterance (a relative clause) under the words of the utterance. *u*'s two utterances are shown above the words of figure 3.7. The purpose of this figure is to show how syntactic analyses can be formed around interruptions by another speaker and how these interruptions themselves are analyzed. The specific syntactic structure of the utterances is not shown. Typically, triangles are used to represent a parse tree without showing its internal structure. Here, polygonal structures must be used due to the interleaved nature of

[10] In 19 cases, the parser returned syntactic analyses but they were incorrect and not included in the above figure.

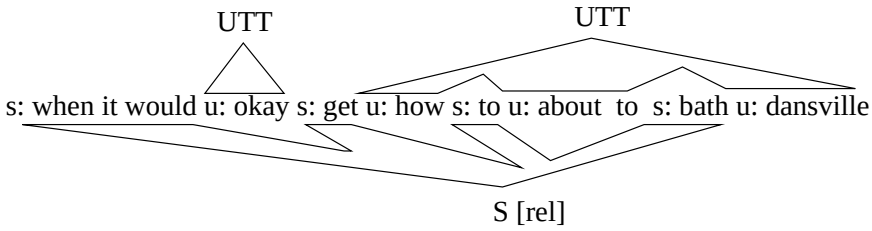


Figure 3.7: Utterances 39 and 40 of d92a-3.2

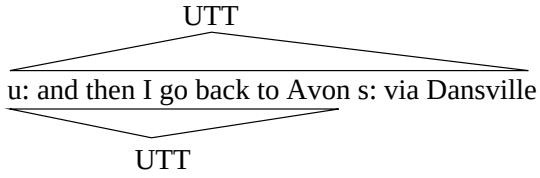


Figure 3.8: Utterances 132 and 133 from d92a-5.2

the utterances.

Figure 3.8 is an example of a collaboratively built utterance, utterances 132 and 133 from d92a-5.2, as shown below. *u*'s version of the utterance (shown below the words in figure 3.8) does not include *s*'s contribution because until utterance 134 (where *u* utters *right*) *u* has not accepted this continuation.

u: and then I go back to avon
s: via dansville

3.5 The Generative Power of the Metarules

Written descriptive text is composed of a series of strings (sentences), elements of the string set generated by the writer's grammar. Given a grammatical formalism which appears capable of describing the set of just those strings that could appear in such texts, it is then meaningful to talk about the generative power of that formalism - i.e., the classes of string sets it could be used in principle to describe (e.g., the context-free languages, the context sensitive languages, or some other class). It thus seems natural to ask about the generative power of the grammatical formalism employed here, namely context-free phrase structure rules together with parser metarules. However, for dialog it is not at all clear how to associate strings with the combined output of the two speakers. In contrast with the case of descriptive text, there is no obvious way to chop a dialog into consecutive, meaningful units. We could pick some arbitrary way of chopping – e.g., chop at utterance boundaries, but since utterances may be interleaved and embedded, this would break utterances into pieces. For example, a simple backchannel acknowledgment would break an example such as the one below into three pieces.

```
u: take     the engine
s:     mm-hm
```

It does not seem worth exploring whether pieces obtained in some such arbitrary way comprise strings of a context-free language or not.

To avoid breaking up utterances, we could restrict ourselves to splitting the dialog at utterance boundaries where there are no other utterances in progress. However, a string identified in this way may in principle include an arbitrarily large chunk of the dialog, possibly even all of it, in a chain such as the one below:

```
u: utterance-1a  utterance-1b  utterance-1c
s:      utterance-2a  utterance-2b
```

Such a string, based on the merged word stream, can consist of a rather arbitrary mix of words from syntactically dissimilar utterances. In view of this, it hardly seems of interest to analyze the formal status of sets of "strings" produced in this way, in terms of the Chomsky hierarchy.

The only thing that does seem to make sense is just to consider the strings corresponding to individual utterances (exclusive of intervening words – i.e., precisely the utterances in the sense in which the current dialog parser identifies them). But those strings are produced by a context-free grammar,[11] so from this perspective there is nothing further to be said about generative power obtained through metarules.

We can, however, look at monologs which contain editing terms and speech repairs but not second speaker interruptions. In particular, we could ask (a) whether the strings corresponding to utterances that are well-formed apart from containing repairs and editing terms, can be generated by a context-free grammar; and (b) whether context-free rules together with the editing term and repair metarules have greater than context-free generative power.

Concerning question (a), note first that a standard context free grammar can easily be modified to allow editing terms to intervene in the middle of utterances. Copies of lexical rules such as `det -> the` can be created to accommodate editing terms: `det -> the ET+` Before we consider how speech repairs could be represented in a context-free grammar, we must say what constitutes an allowable repair. Examples such as utterance 11 from TRAINS-93 dialog d93-8.2, *how does the long does that take*, show that we cannot require a repair's correction to be one constituent because *long does that take* does not form one constituent. Similarly, we cannot require that the reparandum be one constituent. We can construct examples such as *water meter cover screw um cover bolt*. If the structure of the words up to the end of the reparandum is `[[[water meter] cover] screw]` then the reparandum *cover screw* is not one constituent that is repaired by *cover bolt*.[12] What we can say about repairs is that speakers stop a grammatical utterance in the middle (the reparandum end) and restart at a previous position (the reparandum start). There must be a valid although possibly broken utterance ending at the reparandum end, and the words following the reparandum must form an utterance with the words before the reparandum.

To construct a context-free grammar allowing speech repairs as defined above seems to require rules of the form: `XP -> [XP [broken +|-]] XP` where the second XP can be missing words from its left side that appear on the left side of the first XP. For example, *how does the long does that take* would be handled by the rule `UTT -> [UTT [broken +|-]] UTT`. The second UTT would be missing the word *how* which is the leftmost word of the broken UTT *how does the*. First, we must specify how `[broken +]` XPs can be formed. Then we must allow the second XP in the above rule to be missing words on its left side that appear on the left side

[11] There is evidence that humans produce non-context-free utterances but we make the assumption in our parser that they do not.

[12] It may be the case that speakers do not make repairs in such left-branching structures. However, until corpus analysis can prove such a claim, we need to make allowances for such a possibility. The general evidence in the TRAINS-93 corpus is that speakers can "back up" to any point not too far back in the current utterance and resume from there, even though the majority of repairs restart a currently incomplete phrase.

of the first XP. This is tricky and we have only been able to specify context-free rules where the number of words shared between the XPs is bounded by a constant. We are not sure if a context-free grammar can be developed allowing an unlimited number of shared words, and are unsure of the answer to question (b), whether a context-grammar with metarules has greater than context-free generative power. Below we present a context-free grammar allowing speech repairs given a fixed bound on the number of shared words between a repaired constituent and its correction.

Broken constituent rules must allow multiple missing right hand subconstituents as well as allowing the rightmost subconstituent that is present to be also broken. In a phrase that normally has k right hand side (RHS) constituents with k≥1, if k=1, then the RHS can be a broken constituent. If k>1, then m rightmost constituents can be missing (1≤m<k), or the last constituent that is present can be broken, or both. For example, we would produce the following unbroken and broken constituent rules to replace an originally given context-free rule, `X -> U V W`:

```
(X (broken -)) -> (U (broken -)) (V (broken -)) (W (broken -))
(X (broken +)) -> (U (broken -)) (V (broken -)) (W (broken +))
(X (broken +)) -> (U (broken -)) (V (broken -))
(X (broken +)) -> (U (broken -)) (V (broken +))
(X (broken +)) -> (U (broken -))
(X (broken +)) -> (U (broken +))
```

2k rules result from each application of this transformation. The transformation results in a polynomial increase in the grammar (a linear increase, if there is a fixed bound on k, independent of grammar size).

Now that we can form broken constituents, we have completed the first step towards allowing rules of the form `XP -> [XP [broken +|-]] XP` to accommodate repairs. The next step is allowing the second XP to be missing words from its left side that appear on the left side of the first XP. Figure 3.9 shows the general structure of a repair; the repaired constituent, `[XP [broken +|-]]`, extends from w_i to w_k; the reparandum ends at w_k, but starts at w_{j+1}. If j+1 = i, then the reparandum and `[XP [broken +|-]]` are identical. The correction follows the reparandum (after any editing terms), and extends from w_{k+1} to w_m. If j+1>i, then the two XPs share words. We can use the grammar's feature system to require that the words missing from the front of the second XP are on the left of the first XP: `XP -> [XP [words ?w]] [XP [missing ?w]]`. This rule can work in two ways; **words** can be the words on the left of the first XP that are shared with the second XP, or **words** can be all the words of the first XP and **missing** will be words missing from the front of the second XP as well as words that are not shared but appear in the first XP. We take the first approach here although it will be clear that both result in the same complications. Furthermore, we take a bottom-up approach to feature propagation although an equivalent top-down view can be constructed. As we sketch the transformations to the grammar, we give an analysis of how much the grammar size increases. For example, introducing rules of the form `XP -> [XP [words ?w]] [XP [missing ?w]]` will in the worst case double the grammar size. However, given that ?w can be a sequence of words, we need to analyze how many context-free rules without features this rule results in. Take v as the vocabulary size and p as the maximum number of words that can be shared, the maximum length of **missing**. Adding these rules results in v^p new grammar rules, an increase polynomial in the vocabulary size if p is fixed.

As we sketch the remaining necessary grammar transformations, we will show how the new grammar can handle the repair: *meter cover screw # bolt* where # marks the reparandum end. Once the proper NBARs are built, the rule `NBAR -> [NBAR [words ?w]] [NBAR [missing ?w]]` will allow the repair.

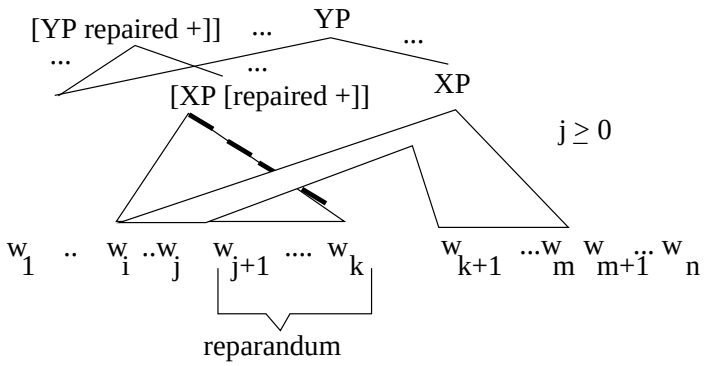


Figure 3.9: Detailed Structure of a Speech Repair

3.5.1 Building `missing`

To build **missing** we start at the lexical level by creating rules such as: `[N [missing meter]] -> ` ϵ, obtained from `N -> meter`. In passing along **missing** we have to be sure that **missing** corresponds to a sequence of words missing from the left side of the current constituent. No gaps in the sequence can be permitted. The **rhs** feature signals when a constituent has elements on its right hand side; i.e., no missing words should come after it. The lexical rules we just created would all have a [**rhs** -] feature value while standard lexical rules would have a [**rhs** +] feature value. To correctly propagate **missing**, a rule such as `X -> U V W` would have to be replaced by the following rules. Note, [**missing ?m1 ?m2**] is regarded as equivalent to [**missing ?m**] where ?m is the concatenation of the (nonempty) word sequences ?m1, ?m2. Similarly for [**missing ?m1 ?m2 ?m3**].

```
1. [X [missing -] [rhs +]] -> [U [missing -]] [V [missing -]] [W [missing -]]
2. [X [missing ?m] [rhs +]] -> [U [missing ?m]] [V [missing -]]
                                  [W [missing -]]
3. [X [missing ?m1 ?m2] [rhs +] -> [U [missing ?m1] [rhs -]] [V [missing ?m2]]
                                     [W [missing -]]
4. [X [missing ?m1 ?m2 ?m3] [rhs ?r]] -> [U [missing ?m1] [rhs -]]
                                           [V [missing ?m2] [rhs -]]
                                           [W [missing ?m3] [rhs ?r]]
```

To analyze the number of context-free rules without features produced by this transformation, take k as the maximum length of a rule's RHS. Each grammar rule will be expanded into k+1 rules with features such as in the example above. In the worst case, each RHS element contributes words to the left hand side (LHS) element's **missing** feature: `[missing ?m1..?mk]`. Each RHS element must contribute at least one word to **missing**, $|?mi| \geq 1$. The maximum number of shared words is p, $|?m1..?mk| \leq p$, and `?m1..?mk` can take on at most v^p different values. This string can be formed in $\binom{p-1}{k-1}$ different ways from words contributed by RHS elements: there are p-1 positions in the string at which k-1 divisions can be made splitting the string into k parts. If p and k are regarded as fixed, the increase in grammar size is thus polynomial in v: $O(v^p)$. The previous transformations discussed here (lexical rule transformations and repair rule introduction) resulted in grammar size increases of 2 and v^p; 2 is dominated by $O(v^p)$ and can be left out of the analysis.

For the *meter cover screw # bolt* example, `NBAR -> NBAR NBAR` would be replaced by the following rules:

```
5. [NBAR [missing ?b] [rhs +]] -> [NBAR [missing ?b]] [NBAR [missing -]]
6. [NBAR [missing ?b1 ?b2] [rhs ?r]] -> [NBAR [missing ?b1] [rhs -]]
                                        [NBAR [missing ?b2] [rhs ?r]]
```

`NBAR -> N` would be replaced by

```
7. [NBAR [missing ?b] [rhs ?r]] -> [N [missing ?b] [rhs ?r]]
```

Using these rules we give an analysis of the correction *bolt*. Given that the structure of the words before the correction is [[meter cover] screw], the derivation of the correction *bolt* is as follows. First we need to create the words missing in the second XP using epsilon rules:

`[N [missing meter] [rhs -]] -> ϵ from 3 to 4`

`[N [missing cover] [rhs -]] -> ϵ from 4 to 5`

To simplify the discussion, assume the input positions are labeled as 0 *meter* 1 *cover* 2 *screw* 3 ϵ 4 ϵ 5 *bolt* 6. Using rule 7, NBARs can be formed from the N constituents above. Then we combine [NBAR [missing meter] [rhs -]] and [NBAR [missing cover] [rhs -]] using rule 6 and form:

`[NBAR [missing "meter cover"] [rhs -]] from 3 to 5`

Assuming we have a standard lexical entry for *bolt*, [N [rhs +] [missing -]], we can form, [NBAR [rhs +] [missing -]] for *bolt* and combine this with the NBAR going from 3 to 5 using rule 5 and form:

`[NBAR [missing "meter cover"] [rhs +]] from 3 to 6`

3.5.2 Building `words`

The next step is to construct the **words** feature. In addition to standard lexical rules, we need rules that initialize **words**: `[N [words water] [wrhs -]] -> water`. When *water* is seen, two lexical constituents are created; one with the feature value [**words water**] and one without (due to the standard lexical rule). Note that the standard lexical rule for water would be the following `[N [words -] [wrhs +]] -> water`. The **wrhs** feature, like **rhs**, signals when the end of a word sequence occurs. Non-lexical rules such as `X -> U V W` would have to be replaced by rules such as:

```
8.  [X [words -] [wrhs +]] -> [U [words -]] [V [words -]] [W [words -]]
9.  [X [words ?m] [wrhs +]] -> [U [words ?m]] [V [words -]] [W [words -]]
10. [X [words ?m1 ?m2] [wrhs +] -> [U [words ?m1] [wrhs -]]
                                   [V [words ?m2]] [W [words -]]
11. [X [words ?m1 ?m2 ?m3] [wrhs ?r]] -> [U [words ?m1] [wrhs -]]
                                         [V [words ?m2] [wrhs -]]
                                         [W [words ?m3] [wrhs ?r]]
```

Building the **words** feature requires the same number of new rules as building the **missing** feature. Overall the number of lexical rules will be tripled. Non-lexical rules will expand into $O(v^p)$ rules without features. The increase is still polynomial.

To analyze the words before the reparandum end in *meter cover screw # bolt*, we start by forming the following lexical constituents:

```
[N [words meter] [wrhs -]] from 0 to 1
[N [words cover] [wrhs -]] from 1 to 2
[N [words -] [wrhs +]] from 2 to 3
```

The rule `NBAR -> NBAR NBAR` would be replaced by rules such as

```
12. [NBAR [words ?w] [wrhs +]] -> [NBAR [words ?w]] [NBAR [words -]]
13. [NBAR [words ?w1 ?w2] [wrhs ?r]] -> [NBAR [words ?w1] [wrhs -]]
                                        [NBAR [words ?w2] [wrhs ?r]]
```

The rule `NBAR -> N` would be replaced by
`[NBAR [words ?w] [wrhs ?r]] -> [N [words ?w] [wrhs ?r]]`
Using this rule we can produce NBARs for the N constituents above. Using rule 13, we can combine [NBAR [words meter] [wrhs -]] and [NBAR [words cover] [wrhs -]] to form the following constituent:
`[NBAR [words "meter cover"] [wrhs -]] from 0 to 2`

This constituent can be combined with [NBAR [words -] [wrhs +]] (going from 2 to 3) using rule 12:
`[NBAR [words "meter cover"] [wrhs +]] from 0 to 3`

Now that we have an analysis for the two parts of the repair, the rule
`NBAR -> [NBAR [words ?w]] [NBAR [missing ?w] [rhs +]]`
can combine them, allowing the repair.

We take allowable speech repairs to be of the form XP XP where the first XP may be broken and the second XP may be missing words from its left side that appear on the left of the first XP. The above context-free grammar allows such speech repairs when the number of words shared between the XPs is less than or equal to a fixed bound, p. The grammar is formed by transforming a standard grammar increasing its size by a factor of $O(v^p)$. Thus, the increase in grammar size, although very large, is polynomial, so that speech repairs can be accommodated by a context-free grammar when the maximum number of shared words has a fixed bound.

To see why assuming a fixed bound on the number of shared words is dubious, consider the example *meter cover screw # bolt*. Here the number of shared words is 2. However, this could be expanded to *water meter cover screw # bolt* where the number of shared words is 3. And the process could be repeated *ad infinitum*. To accommodate arbitrarily many shared words in a repair, the **words** feature would have to allow for arbitrarily many words. Such a feature could be used in a grammar for the language of repeated strings, a context sensitive language. Assume all words have a **words** feature corresponding to their lexical form, and construct the nonterminal XP from the following rules:

```
[Lexcat [words w]] -> w
[XP [words ?w]] -> [Lexcat [words ?w]]
[XP [words ?w1 ?w2]] -> [Lexcat [words ?w1]] [XP [words ?w2]]
```

Then only repeated strings are accepted by the following rule.

```
S -> [XP [words ?w]] [XP [words ?w]]
```

However, we are not allowing **words** to be used in such an arbitrary way. **words** and **missing** are constrained to be used only to answer the question: is the suffix of the string a valid continuation of an earlier part of the string? It is not clear whether this system with such a constraint is context sensitive.

3.6 The Complexity of Parsing with Metarules

The first problem is deciding how to divide the input into units of size n, whose parsing complexity we can then examine as a function of n. We could divide the input at utterance boundaries; however an utterance A may have an utterance B starting in the middle of it and continuing past the end of A. Theoretically, this chain could continue until the end of the dialog: utterance C could start in the middle of B and so on. So in general the only input segment for which we know *a priori* that no utterances are in progress at its end points is the dialog as a whole. The size of the input would thus be the number of words in the dialog - but if we take this as the basis of our analysis, we are apt to greatly overestimate the actual complexity of the parsing algorithm, since in practice these utterance chains do not extend very far. The alternative is to perform analysis on a per-utterance basis. However, some utterances may (1) be inside another utterance, (2) contain embedded utterances, (3) and contain utterances that continue past its end. The first point is the most problematic. In the context of chart parsing, if we are analyzing utterance B that is contained inside utterance A, utterance A will have an unknown number of arcs that can be continued by constituents starting at the beginning of B.

In light of this, it seems desirable to use a unit of analysis that can be assumed to have no ongoing utterances at its end points. Assume that such units can be detected through syntactic and prosodic information without the parser exploring phrase hypotheses that cross these boundaries. Instead of assuming the worst case, that n equals the dialog size, we can measure an average n on different corpora to give us an idea of the running time of the algorithm on these corpora.

The question then is "What is the running time of the dialog parser on an input of size of n, where the input starts at the beginning of an utterance where no other utterances are in progress, and ends when there are no more ongoing utterances?" In a regular parser there would be $O(n^2)$ possible constituents (given that there are $O(n^2)$ pairs of start and end points).[13] We can modify our parser to work like the parser in [Core and Schubert, 1996] in order to simplify the complexity analysis. However, we concede that the resulting analysis does not apply to our dialog parser but rather a parser whose output is identical but operates differently.

In the parser of [Core and Schubert, 1996], copies of words are made that extend across material to be skipped (editing terms, reparanda, second speaker interruptions). Consider additions to the chart for the example *the um engine*.

```
``the''            from 0 to 1
DET                from 0 to 1
NP -> DET . NBAR   from 0 to 1
``um''             from 1 to 2
``the''            from 0 to 2 (skipped 1 to 2)
DET                from 0 to 2 (skipped 1 to 2)
NP -> DET . NBAR   from 0 to 2 (skipped 1 to 2)
``engine''         from 2 to 3
NBAR               from 2 to 3
NP                 from 0 to 3 (skipped 1 to 2)
```

After *um* is seen, the previous word is extended over the interruption. This new "word" then forms a new DET constituent which starts an arc `NP -> DET . NBAR` that ends at 2.

The parser can introduce $O(n^2)$ new words that extend across interrupting material. We standardly form $O(n^2)$ constituents, so the total number of constituents will still be $O(n^2)$.

[13] Each pair of start and end points can be labeled with various categories. This number will be at most the size of the grammar which we take as constant.

From this point on, the complexity analysis is the standard one. The number of arcs in the chart will be $O(\alpha n^2)$, where α is the size of the grammar times the length of the longest right hand side of a grammar rule. This is $O(n^2)$ if we treat α as fixed. Each constituent and each arc is formed by rule instantiation or by extending arcs ending at no more than $O(n)$ different positions in the input. Thus, constructing the chart requires $O(n^3)$ work.

The next step is to determine if the chart contains a set of interleaved utterances (and in the case of repairs, possibly overlapping utterances) covering the input. Finding the top-level utterance constituents that make up such an analysis is a difficult problem. Each constituent on the chart can be formed from many different arcs, and some arcs may include lexical items that skip material. At the outset, we have no way of knowing whether overlapping utterance constituents constitute a legitimate parse. Ordinarily overlapping constituents do not compose valid syntactic analyses, but here repaired material and its correction may share words, and since constituents may skip words, constituents that appear to overlap may actually interleave. There does not seem to be an obvious non-exponential algorithm to test the $O(n)$ utterance constituents starting at the beginning of the input to see whether they cover the input or interleave with a chain of utterances that cover the input.

The exponential algorithm involves building a more detailed chart. From the first word, the algorithm generates all possible arcs and constituents (a constant amount, c) with pointers back to their subconstituent word. The second word will also generate c arcs and constituents (with suitable pointers) combining with the previous entries to produce c^2 chart entries. Total work for parsing in this way is $O(c^n)$.

At $O(n)$ positions in the input, an interruption will be formed (a reparandum, editing term, or second speaker interruption). These interruptions cause the formation of new lexical items which can be viewed as increasing the input size by a factor less than n meaning running time is still $O(c^n)$.[14] We mark these new lexical items with features indicating the position of material skipped. These features will be passed on to constituents formed from these lexical items.

At this point we have a chart with different entries corresponding to different derivations. For each utterance constituent starting at the beginning of the input, we know the possible intersection points with other utterances. For example, an utterance constituent starting at position 0, ending at position k, and with one gap from i to j can be interleaved with an utterance starting at position i or it can be adjacent to an utterance starting at position k. In the worst case of speaker interruption, speakers will utter two word utterances and always interrupt each other meaning an $O(c^n)$ search at $O(n)$ positions in the input to find each utterance constituent. Each utterance will then require $O(n)$ work for its tree to be built from the chart's pointers. Editing terms and repairs will occur mid utterance; lexical items will point to any editing term utterances that follow them, allowing editing term utterances to be easily retrieved when forming an analysis of the dialog.

Given a reparandum that interrupts an utterance, the $O(c^n)$ constituents starting at the input beginning and ending at the reparandum end need to be examined to see if there is a valid syntactic analysis of what the speaker started to say. We assume a grammar such as the one in section 3.5 that can form broken constituents. So even though what the speaker started to say is broken off, the grammar can form a broken utterance constituent assuming the reparandum fits the preceding phrase structure.[15] There will be at most $O(n)$ repairs in the input meaning $O(nc^n)$ time to process them which is dominated by the $O(n^2c^n)$ time to construct standard utterance analyses.

The TRIPS dialog system's parser [Ferguson and Allen, 1998] already included a chart built with pointers as described above, even before it was modified to perform dialog parsing. Such

[14] It is as if the parser received $O(2n)$ words to parse instead of n.

[15] In the actual parser, you would want to check this as soon as the speech repair identifier posited a repair.

repairs found	187
false alarms	223
recall	35.89%
precision	45.61%

Table 3.1: Accuracy of Speech Repair Identifier

a chart allows reasoning about ambiguity and easy extraction of parse trees. In practice, this parser's running times are quite reasonable as shown in section 3.7. This does not mean that we should abandon the search for an efficient algorithm for extracting dialog analyses from a chart. We only state that there is no obvious non-exponential implementation and leave the question open.

3.7 Time Comparison to a Standard Parser

Chapter 5 discusses different speech repair identifiers. Section 5.3 of that chapter shows that sacrificing precision allows gains in recall. Small decreases in precision are acceptable since the repair metarule merely gives the parser the option of skipping a posited reparandum. False alarms for the most part will simply force the parser to consider additional phrase hypotheses. The experiment described below judges the slow-down caused by the metarules and the speech repair identifier of section 5.1.

A test corpus of TRAINS-93 dialogs was used in this study; it contained 521 repairs, 19189 words, and 3660 utterances.[16] The recall and precision of the speech repair identifier on this corpus is shown in table 3.1. To run a standard parser (one with no metarules) on the TRAINS-93 dialogs, we either had to process the words of the two speakers separately or merge them into one stream (which allows one speaker to disrupt the syntax of another). We chose the former approach, producing 3802 utterances after the resegmentation. Note that the parser with metarules was run on a corpus where the two speaker streams were merged, but the per utterance results in table 3.2 are based on 3802 utterances for both parsers. Table 3.2 lists the running time of the parser without metarules, and with metarules and the speech repair identifier described in table 3.1.[17] On average, the increase due to dialog parsing is about half a second per utterance, a hardly noticeable difference. The average utterance is 5 words long so it not the case that utterances such as *okay* dominated the input.

Note that this experiment was performed early in the development of the parser, before the parser attempted to reconstruct partial utterances for reparanda as discussed in section 3.3. The reconstruction techniques are relatively straightforward: treat an incomplete arc in the chart as complete and reparse. Such a technique should not be prohibitively expensive but further experiments will be needed to test this assumption. Reconstruction could also be done on-demand when there is time to spare or the dialog manager is confused about the user state.

In this experiment no attempt was made to "clean up" the input for the standard parser by removing repairs and editing terms. It is not clear what the effect such a step would have on time, since the clean-up process might take more time than it would save.

[16] This is the same set of dialogs used previously, d92-1 through d92a-5.2 and d93-10.1 through d93-14.1. Collaboration with Peter Heeman resulted in re-annotation of speech repairs and changes in utterance segmentation.

[17] These results were obtained on a SUN UltraSparc 60 with two processors running at 360 Mhz and having 512 Megabytes of memory. The code does not take advantage of the extra processor but it was helpful in preventing interference from system programs.

no metarules	0.36 seconds per utterance
metarules	0.91 seconds per utterance
ratio	2.513

Table 3.2: Cost of Dialog Parsing

In accuracy tests, clearly the standard parser would need a pre-parser speech repair identifier to clean up editing terms and reparanda. If the same speech repair identifier were used to trigger the repair metarule of the dialog parser, then the ability of the dialog parser to skip false alarms could be measured.

The standard parser was tested on two separate streams of data for the two speakers to prevent backchannel acknowledgments and other interruptions from disrupting phrase structure. The corpus analysis of chapter 1 indicated that 40 out of 3441 TRAINS-93 utterances were cases where one speaker continued or repaired the utterance of another. The standard parser would miss all these cases. The dialog parser gets the correct parse in three of these cases.

Although this is a low number of successes, taken together with the 2.7% and 4.8% increases in speech repair recall due to parser intervention seen in chapter 5, these numbers support the case for including metarules in parsers of dialog systems. The strongest support though is the richer representation that metarules allow, a representation showing (1) what editing terms appear and where, (2) where second speaker speaker interruptions occur, and (3) what the user started to say during a repair. The experiment of this section shows that points (1) and (2) are achievable with reasonable overhead (even with mediocre speech repair identification) and future work will test whether point 3 is achievable in a real-time parser.

4 Finding Utterance Ends

In a grammar designed for written language, the top-level syntactic category is typically the sentence. Here we consider dialog as composed of utterances; utterances can be sentences but they can also be phrasal answers to questions as well as hesitations (*um*) and conventionalized expressions (*hello*, *let's see*). In addition, utterances are not always complete such as when one speaker interrupts another and the first speaker never resumes the interrupted utterance.

Often a speaker will produce several utterances before another speaker takes the turn (note that "turn" should simply be understood as uninterrupted words by one speaker). Speakers may also collaborate to form utterances, their utterances may overlap, or one speaker's utterance may be embedded inside the other's. Parsers must identify these embedded and overlapping utterances and identify standard utterance boundaries. Failure to find these boundaries will prevent the parser from correctly determining the structure of the input and can lead to misunderstanding. Many current dialog systems only allow users to speak one utterance per turn and do not allow mid-utterance interruptions. However, the next generation of dialog systems need to push beyond these limitations to truly engage users in dialog.

Statistical utterance boundary detection can help a parser sort out multi-utterance turns and utterances constructed by multiple speakers. To train such a detector, we must first devise a way to annotate utterances. We break dialogs into dialog units (DUs) where each DU has no utterances crossing its boundaries. Within each DU, words are labeled with identifiers indicating to which utterance they belong. The experiments of section 4.2 and 4.3 only consider DU boundary detection.[1] However in light of our success at DU boundary detection, statistical utterance boundary detection should be feasible and helpful for breaking turns into utterances and recognizing utterances that cross changes of speaker.

In this chapter, we first discuss previous work (section 4.1). In section 4.2, we discuss our basic model of DU boundary detection and its accuracy: 75.95% recall and 74.17% precision. In addition we present accuracy results after each piece of evidence in the model is removed in turn. This shows the predictive power of the pieces of evidence used. In section 4.3, we show that speech repair information, even if imperfect, can increase DU boundary detection precision. On the test corpus, our basic model with no speech repair information attains 78.29% recall and 77.11% precision and with imperfect speech repair detection (recall 50% / precision 56%) attains 78.21% recall and 78.38% precision. Of course utterance boundary information can be used to limit speech repair search. In section 5.3, the possibility of using the parser's knowledge of utterance boundaries to rescore speech repair hypotheses is discussed. Since the two processes are mutually beneficial they need to be performed simultaneously.

[1] In the corpus of section 4.2, only 7.5% of the utterances had second speaker interruptions. In the corpus of section 4.3, the percentage (4%) is even lower. Thus, in many cases DUs did not have internal utterance end points that could have confused our detection model.

4.1 Previous Work

[Stolcke and Shriberg, 1996a; Stolcke *et al.*, 1998] are the only other research projects investigating utterance boundary detection in a corpus with second speaker interruptions. Stolcke *et al.*'s goal is to aid speech recognition and provide linguistic boundaries for parsing and information extraction. Speech recognition can benefit from linguistic boundaries since the beginning of an utterance should be handled differently than the middle of the utterance [Stolcke and Shriberg, 1996b]. Stolcke and Shriberg give the example of fillers such as *um*. Fillers in the middle of utterances should be removed from the context of n-gram language models to improve their performance. At the beginning of an utterance, fillers provide better context to the language model than words of the previous utterance, and fillers at these positions should not be removed from the context.

To detect utterance boundaries in [Stolcke and Shriberg, 1996a], Stolcke and Shriberg use n-gram word models including tokens for utterance boundaries and turn markers. A sample of their training data is "A: and, uh, you know, colds and things like that <laughter> get -- B: Yeah <S> A: -- spread real easy and things <S>" (p. 1006). Note, <S> is the utterance boundary marker. Focusing only on utterance end points is somewhat simplistic since in the above example the embedded utterance end point is not marked differently from a standard utterance end point. In the training data, -- differentiates second speaker interruptions from new utterances by a second speaker. However, when testing on unmarked data, this information will not be available. This model achieves a recall of 76.9% and precision of 66.9%. Assuming perfect part-of-speech tagging, replacing words other than discourse markers, filled pauses, and conjunctions with their parts of speech in the n-gram model gives 79.6% recall and 73.5% precision.

[Stolcke *et al.*, 1998] continues this work extending its scope to include speech repair detection. Stolcke *et al.* train a CART-style decision tree [Breiman *et al.*, 1984] that uses prosodic features such as: duration (of pauses, final vowels, final rhymes), F0 (before and across a boundary, difference from baseline), and signal to noise ratio. Stolcke *et al.* also construct two 4-gram word models that include repair and boundary events. One "word model" has word tokens corresponding to speaker changes and long pauses. This enhanced word model performed the best at recognizing repair and boundary events, and combining it with the prosody-driven decision tree resulted in 93% accuracy on detecting repair and boundary events using transcribed words and 74.9% accuracy using recognizer output.[2] Stolcke *et al.* report separate utterance boundary detection results only for the enhanced word model without prosodic information: 76.31% recall and 79.89% precision.

There are two studies [Kompe *et al.*, 1997; Lavie *et al.*, 1996] that perform utterance boundary detection with the restriction that mid-utterance interruptions are not allowed. [Kompe *et al.*, 1997] uses statistical utterance boundary detection to reduce parse time by 92% and reduce the number of syntactic trees output by the parser by 96%. For the utterance boundary detection task itself, recall was 80.8% (not counting boundaries at change of speakers) and precision was 84.12%. Their utterance boundary detector has a trigram language model that includes boundary markers as well as words. A multi-layer perceptron was integrated with the language model and processed prosodic features such as syllable durations, F0 features (minimum, maximum, etc.), pause lengths, etc. (see [Batliner *et al.*, 1996] for more details). Kompe *et al.* do not consider utterance boundaries at change of speakers because in their corpus 100% of speaker changes mark utterance boundaries, rather than the 88.3% of speaker changes marking DU boundaries that we saw in our experiment with the best results (section 4.3). Our recall in this experiment on DU boundaries not occurring at speaker changes was about 60%. However,

[2] In these two cases, labeling everything as no-repair/no-boundary results in accuracies of 81.8% and 69.2% respectively.

the results are not totally comparable to Kompe *et al.* since we were performing DU boundary detection and DUs may be interrupted in the middle by another speaker. Clearly though, we need to include more prosodic information such as syllable durations, F0 features, energy features (minimum and maximum, etc.), and so on.

The JANUS machine translation system [Lavie *et al.*, 1996] starts utterance boundary detection by performing a coarse segmentation of speaker turns using prosodic cues (primarily silence) and probabilities of utterance boundaries given a four word context window. Within these coarse segments, the parser can make further divisions based on the syntactic structure of these segments. Translation accuracy is 54% when this coarse segmentation is performed and only 36% if it is not. If the parser is replaced by the Phoenix robust analysis system, the effect on accuracy is lessened; without coarse segmentation, translation accuracy is 49% and with coarse segmentation, translation accuracy is 52%. Despite the lesser effect in the robust parsing case, the results indicate that statistically based coarse segmentation should be a part of the solution to the utterance boundary detection problem.

4.2 DU Boundary Detection

Our basic model of DU boundary detection involves calculating for each word in the input the probability, using equation 4.2, that the word ends a DU, and calculating the probability that it does not end a DU, using a similar equation (replacing *end* with $\neg end$). Equation 4.2 is an approximation given that the evidence used is probably not conditionally independent given the presence (or absence) of a boundary. Thus, we are not guaranteed that the probability of *end* and $\neg end$ add up to one.

(4.2) $P(end|context) \approx P(end|SpkrChange)P(end|tone)P(end|sil)$
$P(end|CatEndingConstit)P(end|fragment)\Pi_{i=0}^{n}[P(end|dm_i)]$
$\Pi_{i=0}^{m}[P(end|e_i)]\Pi_{i=0}^{n+m+4}[1/P(end)]$

- SpkrChange = presence of a speaker change following this word.
- tone = presence of a boundary tone following this word.
- sil = presence of a certain duration of silence (rounded to closest tenth of a second).
- CatEndingConstit = highest constituent (UTT > S > VP > PP > NP) ending with this word and starting at the beginning of the DU.
- fragment = presence of a word fragment following this word.
- dm = set of discourse markers immediately following this word.
- e = set of editing terms immediately following this word.
- 1/P(end) = normalization factor, based on average DU length.

This model was tested on a set of TRAINS dialogs comprising 3441 DUs and 19,189 words.[3] Training was done using cross validation on a dialog basis[4] so the same data was never used simultaneously for training and testing.

The results are shown in table 4.1. The SpkrChange evidence of equation 1 can mislead the model because the change in speaker may be due to a backchannel acknowledgment rather than a second speaker starting a new DU. A modified experiment was run where SpkrChange was not counted for one word interruptions by the other speaker where the interruption was an editing term such as *okay* or *mm-hm*. This change slightly improved results: precision 74.17% and recall 75.95%.

[3] This corpus consisted of dialogs d92-1 through d92a-5.2 and d93-10.1 through d93-14.1. Actually only 3410 DU boundaries were ambiguous as the other 31 coincided with dialog end points.

[4] For example, when testing on dialog d92-1, dialogs d92a-1.1 through d93-14.1 would be used for training.

recall	75.13%
precision	72.91%
ends correctly guessed	2562
missed ends	848
false alarms	952

Table 4.1: Evaluation of DU Boundary Detector

evidence removed	accuracy	recall	precision
tone	88.26%	72.87%	68.08%
CatEndingConstit	88.74%	75.89%	68.78%
sil	89.34%	74.43%	71.53%
dm	89.66%	72.52%	73.67%
e	89.67%	74.69%	72.69%
SpkrChange	89.92%	74.90%	73.60%
fragment	90.17%	75.95%	74.15%
none	90.18%	75.95%	74.17%

Table 4.2: Effect of Evidence Terms on DU Boundary Prediction

To see the effect of each piece of evidence from equation 4.2, experiments were run where each evidence term was removed from the model in turn. Note, in these experiments, one word interruptions by another speaker consisting of an editing term were not counted as speaker changes. The results are shown in increasing order with respect to accuracy in table 4.2. Accuracy is defined as: CorrectAnswers/answers where CorrectAnswers include cases where the classifier found boundaries as well as cases where the classifier was correct in not positing a boundary. Precision and recall are given since accuracy can be misleading; never positing a boundary gives good accuracy but no recall or precision.

The table shows that boundary tones have the largest impact, with their absence causing a 4.06% decrease in recall and a 8.21% decrease in precision. Word fragments have the least impact and their absence only causes a 0.03% decrease in precision and does not noticeably affect recall. Word fragments were added to the model based on the idea that they would not occur at DU boundaries. This assumption turned out to be wrong. Some DU are interrupted by interjections that are not repairs. These DUs may not be continued and may end in word fragments as in dialog d92a-3.1:

```
u: well for engine t-
s:                   which engine
```

Syntactically possible utterances may often be found in the middle of actual utterances in the corpus. For example the utterance, *take the train to Avon via Corning*, contains the potential utterances: *take the train* and *take the train to Avon*. Despite this problem, CatEndingConstit can be used to eliminate false alarms where utterance end points do not occur. Removing the CatEndingConstit evidence quadruples the number of false alarms. The coverage of the parser's grammar of sentences in the corpus was 65%. With a better grammar, CatEndingConstit would be even more effective at eliminating false alarms.

Discourse markers and boundary tones are good sources of evidence for the presence of DU boundaries and help increase recall. It is not surprising that boundary tones, the ends of

prosodic phrases, are good indications of DU boundaries. [Traum and Heeman, 1996] discusses clean transitions, changes of turn[5] where the second speaker does not overlap the previous speech. These will account for some of the utterance end points seen by a dialog system. Traum and Heeman state that more than 94% of the clean transitions in their corpus were marked by boundary tones.

It is interesting to note that the evidence tested here is partially redundant. The absence of boundary tones (the most predictive evidence) only decreases recall by 4.06%. Although reasonably accurate boundary tone detection is possible [Heeman, 1997], decent DU boundary predictions are possible without such information.

One obvious cause of inaccuracy in this model are DUs with more than one utterance. Utterances that start in the middle of a DU may be impossible to tell apart from those that start the DU without considering the full utterance boundary detection problem. Another problem is the assumption that pieces of evidence are independent given the presence or absence of a boundary. Each piece of evidence by itself can be misleading. For example, boundary tones can occur in the middle of an utterance and words that are possible discourse markers such as *so* can occur in the middle of an utterance. However, if a boundary tone and discourse marker are both seen then the likelihood of a DU boundary might be greater than $P(end|tone) * P(end|dm = SO)$.

Another cause of inaccuracy is that the parser's notion of utterance is not identical to that of speakers in the corpus. The utterance boundaries in the corpus were marked by the author in accordance with the parser's grammar. The grammar demands that input such as *take the engine to Avon and pick up the boxcars* be broken into two utterances. However, intonational cues (silence and boundary tones) may not mark this distinction as the speaker may think of the input as one piece.

A third source of errors is the susceptibility of syntactic information to misclassification errors. Thus, if one DU boundary is missed or a false alarm occurs, then it is unlikely that the parser will construct an utterance constituent ending at the next DU boundary.

Using prosodic information in addition to silence and boundary tones should also aid utterance boundary detection; [Kompe *et al.*, 1997; Koiso *et al.*, 1998; Wells and Macfarlane, 1998; Caspers, 1998] show that turn ends are marked by a rich variety of prosodic cues. Including more prosodic information into an utterance boundary detector will at least improve its performance at recognizing utterance ends occurring at turn ends.

Viewing the results in an "all or nothing" manner means about 3 out of 4 DU boundaries will be found and 1 out of 4 predictions is a false alarm. If we construct an actual utterance boundary detection model, the detector can pass along probabilities of utterance boundaries to the parser. The parser may overcome false alarms when all new hypothesized utterance analyses have lower probabilities than utterance analyses going past the posited boundary. Similarly, if an utterance constituent cannot be built, the parser can search for utterance boundaries that may have been missed. The next step in this work is to test such a model and evaluate the output of the parser to determine the effect of statistical utterance boundary detection on the correctness of the parser's output.

4.3 DU Boundaries and Speech Repair Detection

Since a word cannot both end a reparandum (the corrected material in a speech repair) and an utterance, perfect speech repair detection would aid an utterance boundary detector

[5] In Traum and Heeman's work, a distinction is made between change of speaker and change of turn. Backchannels are not considered changes of turn, only changes of speaker.

recall	78.29%
precision	77.11%
ends correctly guessed	2813
missed	780
false alarms	835

Table 4.3: Evaluation of Standard DU Boundary Detector

recall	78.15%
precision	79.39%
ends correctly guessed	2808
missed	785
false alarms	729

Table 4.4: Evaluation of a DU Boundary Detector with Perfect Repair Information

in its task. Although perfect automatic speech repair detection is not currently available, we can provide hand annotations to the parser to see the maximum extent to which speech repair information can help our DU detection model. We have performed a second experiment using automatic speech repair detection to see how much help current speech repair identifiers provide to DU boundary detection.

Slight modifications were made to the corpus for these experiments. Dialog d93-10.5 was removed from consideration since repair annotations were unavailable for it at the time of the experiment. A version of Heeman and Allen's speech repair identifier predating the version in [Heeman and Allen, 1997] was connected to the parser for the second experiment. The repair identifier does not allow multiple speakers to collaborate to form utterances. These cases had to be split into separate incomplete utterances. The resulting corpus had 3593 ambiguous DU boundaries and 19045 words.

Our DU boundary detector was rerun on this corpus to provide a baseline measure (see table 4.3 for the results). Table 4.4 shows performance of a model that has perfect speech repair information and never posits a DU boundary where a reparandum ends. The augmented model ends up missing 5 of the end points that the original model correctly predicted. Given that the augmented model makes different end point predictions, the syntactic evidence it has is different as well, and in this case leads it to miss 5 end points that it found before. Out of 3593 DU end points, 5 end points cannot be taken as evidence that speech repair information will always harm recall and more experiments need to be performed. The impact of the augmented model on precision is much stronger; it raises the precision by 2.96% (eliminates 106 false alarms).

It appears that speech repair information should be included in an utterance boundary detection model. But what if the speech repair information is not always correct? We ran a second experiment using Heeman and Allen's speech repair identifier as the source of speech repair information.[6] The performance of this speech repair identifier on the test corpus is shown in table 4.5 and the performance of DU boundary detection using this information is shown in table 4.6. The increase in precision is not as great as before but still amounts to eliminating 60 false alarms. Thus, even with currently available speech repair identification systems, speech repair information is beneficial to DU boundary identification.

[6] This speech repair identifier was never simultaneously trained on the same data that it was tested with.

recall	50.09%
precision	55.76%
repairs correctly guessed	271
missed	270
false alarms	215

Table 4.5: Heeman's Speech Repair Identification Results

recall	78.21%
precision	78.38%
ends correctly guessed	2810
missed	783
false alarms	775

Table 4.6: Evaluation of a DU Boundary Detector with Imperfect Repair Information

5 Finding Speech Repairs

Speech-based dialog systems often attempt to identify speech repairs in the speech recognition phase, so that speech repairs will not disrupt the speech recognizer's language model [Heeman and Allen, 1997], [Stolcke and Shriberg, 1996b], [Siu and Ostendorf, 1996]. In such a system, it is then tempting to remove conjectured reparanda (corrected material) and editing terms from the input prior to further processing. One reason not to do so is that the parser can help in speech repair identification, a hypothesis explored in the following chapter. One way to test whether the parser can help the speech repair identification process is to create an in-parser speech repair identifier and compare it to pre-parser speech repair identifiers. Sections 5.1 and 5.2 discuss a simple in-parser speech repair identifier and its performance. This simple identifier unfortunately was not able to outperform the state of the art in pre-parser speech repair identification. Success was instead gained from rescoring the output of a pre-parser speech repair identifier resulting in increases in recall varying from 2.7% to 4.8% as discussed in section 5.3. On a subset of the corpus where grammar coverage was 100%, a 8.89% increase in recall was seen.

Experiments of this chapter assume perfect utterance boundary detection. Utterance boundary detection is still under development; the statistical utterance boundary detector of chapter 4 only finds about 3 out of every 4 utterance boundaries. Relying on this detector would significantly lower the parsing accuracy. Since the grammar coverage (assuming correct utterance boundaries) is low (39.7%, on non-trivial utterances, as measured in section 3.4), perfect utterance boundary detection will be assumed.

5.1 In-parser Speech Repair Identification

For each word in the input, the parser must perform speech repair detection; it must decide whether the end of a reparandum follows the word. The detection process also involves determining whether a repair is a *fresh start* (corrects the entire utterance up to this point) or a *modification repair*. These terms are taken from [Heeman, 1997]. The reason this distinction is made during detection is that fresh starts and modification repairs appear in somewhat different contexts. As noted in [Heeman, 1997], editing terms such as *sorry* often occur with fresh starts while editing terms such as *uh* correlate more with modification repairs. In modification repairs, speakers repeat previous material signaling the location of the reparandum (e.g. *take the oranges with the box um with the tanker*). In a fresh start however, no previous material in the utterance is repeated. After a modification repair, the parser must perform repair correction; it must find the start of the repair.

5.1.1 Repair Detection

For each word, we calculate the probability that the word ends a fresh start reparandum, ends a modification repair reparandum, or does not end a reparandum. The equation below shows how the probability for a fresh start is estimated. Probabilities for modification repair reparandum ending and no reparandum ending are computed in the same manner and the most likely one chosen.[1]

(5.1.1) $P(freshstart|context) \approx P(freshstart|SpkrChange)P(freshstart|fragment)$
$P(freshstart|tone)P(freshstart|sil)P(freshstart|CatOfHighestConstit)$
$P(freshstart|DirectionOfExtension)P(freshstart|MaxMatchscore)$
$P(freshstart|MaxAtZero)\Pi_{i=0}^{n}[P(freshstart|e_i)]\Pi_{i=0}^{n+6}[1/P(freshstart)]$

- SpkrChange = presence of a speaker change following this word.
- fragment = presence of a word fragment following this word.
- tone = presence of a boundary tone following this word.
- sil = presence of a certain duration of silence (rounded to closest tenth of a second).
- e = set of editing terms immediately following this word.

The pieces of evidence, *CatOfHighestConstit* and *DirectionOfExtension*, are attempts to detect the syntactic disruption caused by a speech repair. The presence of a sentence parse is a good indication that no repairs are present as the last experiment of section 5.2.2 shows. The converse is true to a lesser degree; lack of an utterance parse gives evidence that repairs may be present (coverage of the parser's grammar on 39.7%, as measured in section 3.4). However, lack of an utterance parse does not tell us where in the input repair(s) may have occurred. To solve this problem, the parser computes a local measure of grammaticality.

For each word, the parser finds the largest constituent(s) containing that word according to the greedy/approximate algorithm below. The categories of these constituents are ordered (S > VP > PP > NP and so on). If more than one largest constituent is found then the constituent highest (leftmost) in this ordering is chosen. This is the CatOfHighestConstit evidence mentioned above. The DirectionOfExtension evidence is either left, right, both, or neither depending on whether the largest constituent covers words to the right of the current word, words to the left of the current word, both, or neither. The idea behind this evidence is that words in the middle of constituents such as VPs (with additional words to the left and right) are less likely to be involved in repairs. Words ending reparanda are less likely to be in constituents that extend to words on the right. Thus, CatOfHighestConstit and DirectionOfExtension can indicate cases where it is unlikely that a reparandum end follows the current word.

```
TO FIND LARGEST CONSTITUENT CONTAINING WORD W

assume for each position in input, we have the largest constituent(s)
 ending at this position;

i := maximum position;
repeat
  if largest constituent(s) ending at i contains w then return constituent(s);
  i := i - 1
```

[1] We assume that the context terms are independent given knowledge of what type of repair (if any) appears. If this assumption were always true, the probability of no reparandum could be calculated by $P(NoReparanum|context) = 1 - P(freshstart|context) - P(modification|context)$. However, when the assumption is wrong, the probability estimates for modification repair, fresh start, and no repair may not add to 1.

Parallelism scores, defined below, are used to find reparandum beginnings but can also be used to aid the process of repair detection. *MaxMatchScore* is the best parallelism score obtained by picking one of the previous words as a reparandum start and the current word as the reparandum end. A high MaxMatchScore is good evidence for the presence of a repair. The *MaxAtZero* evidence is simply whether this maximum occurs at the first input word. It is used to help identify fresh starts.

5.1.2 Repair Correction

Parallelism scores are computed from word matching and are used in both repair detection and repair correction. Parallelism is computed between a potential reparandum and its correction. The correction is defined as the material after the reparandum minus any editing terms immediately following the reparandum. When a word of the reparandum matches a word of the correction, 1 is added to the parallelism score whereas mismatches subtract 1 from the score. The matching algorithm is shown below.

```
score := 0;
x := word number of reparandum start;
y := word number of correction start;
while (x <= last_word_of_reparandum) and (y <= last_word_of_input)
    if word(x) = word(y) then
      score := score + 1;
      y := y + 1;
      x := x + 1;
    elseif word(x) can_be_found_after word(y)
      score := score - 1;
      y := y + 1;
    else
      score := score - 1;
      x := x + 1;

score := score - remainder_of_reparandum_if_any_remains
```

In later experiments, the word identity check in the algorithm was modified to allow for part-of-speech matching.

5.2 Speech Repair Identification Results

This section first discusses detection results including the predictive power of the various pieces of evidence in equation 5.1.1. In these results the distinction between fresh start and modification repairs is not made. Posited fresh starts and modification repairs are labeled correct if they coincide with any type of reparandum end point. It is in the repair correction results that this distinction is made: a repair that is labeled as a fresh start is a correct answer if the repair actually begins at the start of the utterance. A repair that is labeled as a modification repair is a correct answer if the repair correction process finds the start of the reparandum.

5.2.1 Detection

To test the repair detection model corresponding to equation 5.1.1, a corpus of TRAINS dialogs [Heeman and Allen, 1995] consisting of 3441 utterances, 19189 words, and 495 speech

recall	42.63%
precision	51.21%
repairs correctly guessed	211
false alarms	247
missed repairs	284

Table 5.1: Evaluation of In-parser Speech Repair Detector

evidence removed	recall	precision	repairs correct
MaxMatchscore	19.19%	37.11%	95
e	33.74%	59.86%	167
CatOfHighestConstit	35.35%	48.88%	175
sil	36.77%	55.15%	182
fragment	37.98%	51.23%	188
DirectionOfExtension	38.38%	51.08%	190
MaxAtZero	41.21%	56.20%	204
tone	42.42%	51.47%	210
SpkrChange	42.63%	50.97%	211
none	42.63%	51.21%	211

Table 5.2: Effect of Evidence Terms on Speech Repair Detection

repairs was used.[2] The results of applying the speech repair detection model to this corpus are shown in table 5.1. (Note that given the preceding numbers, approximately 1 out of 40 word boundaries is a reparandum end point. Always predicting no repair would give 82.1% accuracy, and on finding reparandum end points would lead to 0% recall and undefined precision.[3])

The purpose of the next experiments was to see the impact of removing each piece of evidence (from equation 5.1.1) in turn. False alarms slow down the system by creating additional phrase hypotheses but do not necessarily cause incorrect answers. Missing a repair eliminates the possibility of getting the correct parse. Thus, the results of these experiments (table 5.2) are ordered by recall. By this scale, MaxMatchscore is the most important piece of evidence since removing it results in the lowest recall and precision. Editing terms are the second most important piece of evidence with regard to recall. The removal of speaker change information had the least impact on recall. It was added to the model to help precision (repairs are less likely at changes of turn).

It is not surprising that match scores play an important role in speech repair detection since detecting repetitions also helps [Bear *et al.*, 1992] achieve good results. The syntactic evidence, CatOfHighestConstit and DirectionOfExtension, also ended up being fairly predictive. even though the parser grammar's only covered 65% of input utterances (39.7% of interesting utterances).

These experiments also involved using word matching to find the starts of repairs. Thus, the models can be ranked on their total performance. Table 5.3 shows the recall and precision at finding both the start and end of reparanda. It is not fair to judge the predictive powers of the models based on this data. It may be the case that the reparandum endings found by one

[2] The dialogs used were d92-1 through d92a-5.2 and d93-10.1 through d93-14.1.

[3] Precision is the percentage of "hits" divided by hits and false-alarms where hits in this case are correctly identified reparandum ends. Since hits and false-alarms are zero we cannot define precision here.

evidence removed	recall	precision	repairs correct
MaxMatchscore	13.33%	25.78%	66
e	27.88%	49.46%	138
CatOfHighestConstit	28.08%	38.83%	139
DirectionOfExtension	29.50%	39.25%	146
sil	29.90%	44.85%	148
fragment	30.10%	40.60%	149
MaxAtZero	31.52%	42.98%	156
tone	32.73%	39.71%	162
	33.33%	40.05%	165
SpkrChange	33.54%	40.10%	166

Table 5.3: Effect of Evidence Terms on Speech Repair Correction

repairs correctly detected	211
correct beginnings found	165
correction rate	78.20%
resulting overall recall	33.33%

Table 5.4: Correction with Only Word Information

model had easier to find reparandum starts. Notice that the model without SpkrChange gets one more correct repair than the standard model. These two give us the highest recall. One repair out of 495 does not necessarily mean that SpkrChange will always hurt total performance and more experiments are needed to see the true effect of the SpkrChange evidence.

5.2.2 Correction

In the previous section, repair correction results were included in order to describe the total performance achieved by various detection models. The repair correction information for the standard detection model is shown in table 5.4. If the correction rate could be improved, the overall repair recall rate would start to approach 42.63%, the recall rate of reparandum endings. The purpose of the next experiment was to see if part-of-speech (POS) information could improve repair correction. The matching algorithm from section 5.1.2 was modified to include POS information as shown in figure 5.1. Note, the function `POS` returns all the possible parts of speech of a word.

A slightly different corpus of TRAINS dialogs was used in this study; it had 521 repairs, 19189 words, and 3660 utterances.[4] Table 5.5 shows the results of running the standard detection and correction model on this corpus as well as the results including POS information.

The decrease in correction accuracy (4 misses out of 521 repairs) is not great enough to conclude that our word and POS matching algorithm necessarily decreases the correction rate. However, it definitely does not increase the correction rate. In [Heeman, 1997], POS information is used successfully and in general this should be the case. The problem with the system tested here is that the grammar's coverage is only 65% and even when parses can be found, the parser will not necessarily disambiguate among the competing alternatives. Thus, the system cannot

[4] This is the same set of dialogs used previously, d92-1 through d92a-5.2 and d93-10.1 through d93-14.1. Collaboration with Peter Heeman resulted in re-annotation of speech repairs and changes in utterance segmentation.

```
score := 0;
x := word number of reparandum start;
y := word number of correction start;
while (x <= last_word_of_reparandum) and (y <= last_word_of_input)
  if word(x) = word(y) then
      score := score + 1;
      y := y + 1;
      x := x + 1;
  elseif POS(x) intersects POS(y)
      score := score + 0.5;
      y := y + 1;
      x := x + 1;
  elseif word(x) can_be_found_after word(y)
      score := score - 1;
      y := y + 1;
  else
      score := score - 1;
      x := x + 1;
score := score - remainder_of_reparandum_if_any_remains;
```

Figure 5.1: Algorithm for Computing Parallelism Using Word and POS Information

repairs correctly detected	237
correct beginnings (word matching)	187
correct beginnings (word and POS matching)	183
correction rate (word matching)	78.90%
correction rate (word and POS matching)	77.22%
resulting overall recall (word matching)	35.89%
resulting overall recall (word and POS matching)	35.12%
resulting overall precision (word matching)	45.61%
resulting overall precision (word and POS matching)	44.63%

Table 5.5: Effect of Adding Part-of-Speech Information

assign a unique part of speech to a word but instead must assign a word all its possible parts of speech. Another problem is that the parser's lexicon uses 13 POS tags. The work of [Heeman, 1997] uses 55 POS tags and does part-of-speech tagging in order to assign unique parts of speech. To see the advantage of the 55 POS tag set, consider the example below. The 13 POS tag set considers *see* and *have* as identical whereas the 55 POS tag set differentiates them. Thus, the 13 POS tag set gives more credit to parallelism between and *us see* and *we have* and could lead to incorrectly identifying *us see* as a reparandum.

```
I have let us see we have two boxcars
```

For the in-parser repair identifier, word level parallelism is effective at finding repairs (increasing recall). To increase precision, one technique is to not posit repairs when a full parse of the input is available. If the repair detection experiment of section 5.2.1 is repeated with such a heuristic, the precision increases from 52.66% to 57.41% while the recall only decreases from 44.04% to 43.03%.[5] The recall of this model on repair correction is 32.93% and the precision is 43.94%. These results are still lower than the pre-parser speech repair identifier described in [Heeman, 1997] (recall 63.76% and precision 72.54% as measured on the entire TRAINS corpus). A more sophisticated in-parser model inside a parser with a broader coverage grammar and better disambiguation system should outperform a pre-parser model. However, the goal is not to replace pre-parser repair identifiers but rather prove that a parser can correct their output. The next experiments test this hypothesis directly by attempting to correct the output of the speech repair identifier of [Heeman and Allen, 1997].

5.3 Correcting a Pre-parser Speech Repair Identifier

One way for the parser to aid the speech repair identification process is to examine the n-best repair hypotheses output by a pre-parser speech repair identifier and pick the best one based on syntactic evidence. This technique was tested by connecting the pre-parser speech repair identifier of [Heeman and Allen, 1997] to our parser. This repair identifier produced up to 100 hypotheses about the speech repairs, boundary tones, and parts of speech associated with the words of each turn in the test corpus. Each hypothesis was given an estimated probability.

Both the parser and Heeman and Allen's speech repair identifier were developed and tested on the TRAINS corpus [Heeman and Allen, 1995]. However, Heeman and Allen's testing data was broken into two streams for the two speakers while the test data for the parser merged the two speakers' words into one data stream. The differences in segmentation resulted in different speech repair annotations. The parser's test data was used in experiment one while Heeman and Allen's testing data was used for experiments two and three.

In all experiments, the repair hypothesis allowing the most words to parse was chosen with repair probabilities used only to break ties. Experiment three used only the subset of the corpus that the parser's grammar covered. This experiment showed what factors other than grammar coverage caused the parser to make mistakes in selecting repair hypotheses.

5.3.1 Experiment One

The first experiment used the parser's speech repair annotations. The pre-parser speech repair identifier that was used is an older version of Heeman and Allen's identifier described in [Heeman and Allen, 1997]. Correspondingly, the recall and precision of this identifier are lower

[5] These numbers are slightly different than the previous experiment due to resegmentation of the corpus.

Repairs correctly guessed	271
False alarms	215
Missed	270
Recall	50.09%
Precision	55.76%

Table 5.6: Heeman and Allen's Speech Repair Results from Exp 1

Repairs correctly guessed	284
False alarms	371
Missed	257
Recall	52.50%
Precision	43.36%

Table 5.7: Augmented Speech Repair Results from Exp 1

than current versions. The recall and precision of the identifier on the test corpus is shown in table 5.6. The test corpus consisted of 541 repairs, 3797 utterances, and 20,069 words.[6] To correct Heeman and Allen's output, the parser starts by trying their module's first choice. If this results in a parse covering the input, that choice is selected as the correct answer. Otherwise the process is repeated with the module's next choice. If all the choices are exhausted and no parses are found, then the first choice is selected as correct. This approach is similar to an experiment in [Bear *et al.*, 1992] except that Bear et al. were more interested in reducing false alarms. Thus, if a sentence parsed without a posited repair then the repair was ruled a false alarm. Here the goal is to increase recall by trying lower probability alternatives when no parse can be found.

The results of such an approach on the test corpus are listed in table 5.7. Recall increased by 4.8% (13 cases out of 541 repairs), showing promise in the technique of rescoring the output of a pre-parser speech repair identifier.[7] One factor relating directly to the effectiveness of the parser at speech repair identification is the percent of fluent or corrected utterances that the parser's grammar covers. The grammar's coverage of non-trivial utterances (utterances that are not phrasal question answers or acknowledgments) in this corpus, as measured in section 3.4 is 39.7%. When a fluent or corrected utterance cannot be parsed, the parser may pick a low scoring repair hypothesis that eliminates the unparsable material (this may be most of the utterance). This situation results in a false alarm and actual repairs in the input may be missed.

5.3.2 Experiment Two

Experiment two confirms the results of experiment one using the latest version of Heeman and Allen's speech repair identifier [Heeman and Allen, 1997], Heeman and Allen's speech repair annotations, and a better baseline measurement. The baseline measure used in experiment one neglected the role that utterance boundaries play in repair detection. The parser is given utterance boundaries, but Heeman and Allen's identifier did not have access to this information. Since speech repairs are intra-utterance corrections, utterance boundaries help limit the search space that must be considered. The baseline measure of this experiment was adjusted to control for this advantage.

[6] Specifically the dialogs used were d92-1 through d92a-5.2; d93-10.1 through d93-10.4; and d93-11.1 through d93-14.2. The speech repair identifier was never simultaneously trained and tested on the same data.

[7] In this thesis, increases and decreases are reported in percentages rather than differences. 4.8% is obtained by dividing the difference in recall, 2.41, by the original amount 50.09.

Repairs correctly guessed	445
False alarms	125
Missed	250
Recall	64.03%
Precision	78.07%

Table 5.8: Heeman and Allen's Speech Repair Results from Exp 2

Heeman and Allen's segmentation broke the input into two parts, one for each speaker, and further divided those into turns. The author broke turns into utterances as defined by the parser's grammar. Heeman and Allen's scoring module worked on a per-turn basis, meaning that if a turn contained several utterances the parser was not allowed to pick one repair hypothesis for the first utterance and a different one for the second. The parser scored the different hypotheses based on the number of words that parsed for each hypothesis. So if one hypothesis allowed the construction of two corrected utterances, one containing 5 words and another containing 7 words, its score would be 12. The hypothesis with the highest score was picked. In the case of ties, the hypothesis with the higher probability (as assigned by Heeman and Allen) was chosen.

To construct a baseline measurement taking into account the effect of utterance boundaries, hypotheses output by Heeman and Allen's module that crossed utterance boundaries were eliminated. The top scoring hypothesis out of those remaining was selected for each turn. The resulting recall and precision are shown in table 5.8. The maximum attainable recall and precision are not 100% as the hypotheses output by Heeman and Allen's module do not always contain the correct answer. The maximum attainable recall and precision are 89.78% and 88.26%.

The test corpus for this experiment includes one additional dialog (d93-10.5) giving it a total of 20,213 words. The additional dialog and different segmentation and repair annotations result in a corpus of 2295 turns, 3953 utterances and 695 speech repairs. Involving the parser as described above produces the results shown in table 5.9.[8] Recall increases by 2.7% (12 repairs out of 695). Actually, there are 33 cases where the parser corrected the output of Heeman and Allen's module, but there are also 21 cases where the parser incorrectly rejected Heeman and Allen's first choice creating a false alarm and causing a repair to be missed. These instances occurred when the parser's grammar did not recognize the corrected utterance.

Because three aspects of the experiment were changed between experiments one and two, it is difficult to say whether 2.7% is a more valid measure of recall increase than the 4.8% measured in experiment one. One aspect that changed was the segmentation and, as a result, the grammar coverage changed, since segment boundaries sometimes broke up utterances. We measured the parsability of 60 turns randomly drawn from this corpus and containing 100 utterances. 65% of the turns parsed but if we do not consider turns consisting only of one-word utterances and phrasal question answers then grammar coverage is 36.4%. Since experiment one was utterance-based and had a grammar coverage of 39.7% on non-trivial utterances, the change in segmentation could have affected the recall rate. Clearly more experiments need to be run to get the correct figure.

[8] Note that these results are slightly better than those reported in [Core and Schubert, 1999a]. Both the parser and Heeman and Allen's repair identifier attempt to handle editing terms. In three cases, the parser picked a repair hypothesis that excluded an editing term. However, the editing term metarule allowed this editing term to be identified during parsing, correcting the initial error. In [Core and Schubert, 1999a], these corrections were not identified and these cases were counted as wrong.

Repairs correctly guessed	457
False alarms	746
Missed	238
Recall	65.76%
Precision	37.99%

Table 5.9: Augmented Speech Repair Results from Exp 2

5.3.3 Experiment Three

In experiment two, low grammar coverage caused the parser to select incorrect repair hypotheses that removed unparsable but fluent material from the input. Rerunning the experiment on a small corpus of parsable turns[9] would indicate how close the parser could come toward achieving the maximum repair recall allowed by the pre-parser repair identifier's n-best lists. From the corpus of experiment two, we selected a set of 103 parsable turns where the correct repair hypothesis was among those output by Heeman and Allen's module. Thus, 100% recall should be possible. This mini-corpus contained 179 utterances, 1229 words, and 127 repairs.

Table 5.10 lists the results from simply choosing the highest ranked of Heeman and Allen's repair hypotheses. Like experiment two, repair hypotheses that cross utterance boundaries are not considered. Table 5.11 lists the results from using the parser to chose from Heeman and Allen's repair hypotheses based on which hypothesis allows the most words to parse.

The increase in recall from using the parser is 8.89%, more than experiment two but not as much as if the recall had approached 100%. Given that all the turns parsed, mistakes in this experiment occurred when the parser formed a parse that included at least some of the reparandum. The parser picks the hypothesis allowing the most words to appear in the corrected utterance. Thus, if part of the reparandum can be accommodated, this hypothesis is incorrectly favored. 19 mistakes in this experiment were due to the parser including part or all of a reparandum in a corrected utterance.[10] In four of these cases, the parser used a discourse marker from a reparandum in the corrected utterance as in utterance 48 from d93-12.3:

```
so we need to oh we need boxcars
```

Here, the reparandum is *so we need to* but the parser identifies *so we need boxcars* as the corrected utterance.

In some cases the grammar does not impose enough constraints on possible parses and implausible parses are constructed as in utt 210 from d92a-5.2:

```
engine one is going from Dansville from Avon
```

Here the grammar allows a series of path describing PPs to combine with *go* to form a VP without disallowing multiple *from* PPs.

The grammar performs some rudimentary semantic checks; however, in one example (utterance 23 from d93-10.3), a semantically unlikely parse containing the reparandum is formed:

```
and then go to hm go to Corning
```

[9] Parsable assuming reparanda are correctly found.

[10] The other mistakes in the experiment can likely be easily fixed and involve the parser not understanding contractions and problems in the scoring mechanism involving editing terms.

Repairs correctly guessed	90
False alarms	13
Missed	37
Recall	70.87%
Precision	87.38%

Table 5.10: Heeman and Allen's Speech Repair Results from Exp 3

Repairs correctly guessed	98
False alarms	23
Missed	29
Recall	77.17%
Precision	80.99%

Table 5.11: Augmented Speech Repair Results from Exp 3

One can construct similiar cases *go to hm drive to Corning* and *and then go to hm take the engine to Corning* where a repair does not occur. Thus, such a semantic construction cannot simply be disallowed. However, in utterance 23 the parallelism between the two instances of *go to* should make the probability of a parse without the first *go to* more likely than the parse including it.

In certain situations, other examples of repairs are perfectly plausible as fluent utterances or utterances with shorter reparanda. Consider utt13 from d93-10.3 containing the reparandum *how does u-*. A plausible parse of this word stream assuming a shorter reparandum would be *or how do tankers move by themselves.* If another speaker had been moving tankers without engines, it would be reasonable for the first speaker to ask how tankers move by themselves.

```
or how does u- do tankers move by themselves
```

It may be the case that statistical methods can help with such examples. A statistical parser may notice a difference between (1) *how do tankers move by themselves* and (2) *do tankers move by themselves* based on its training data. Repair probabilities output by Heeman and Allen favor choice 2. In terms of total words in the corrected utterance, 1 is favored by one word; however proper combination of all these sources of evidence will allow the parser to choose alternative 2 despite the shorter reparandum in choice 1.

5.3.4 Discussion

Experiment two highlighted the sensitivity of parser repair identification to grammar coverage. The parser sometimes incorrectly rejected the most highly ranked choice output by Heeman and Allen's repair identifier, selecting instead another hypothesis that removed unparsable but fluent material from the input. Despite this limitation, parser rescoring did increase repair recall by 2.7%.

This repair recall increase along with the 4.8% increase seen in experiment one should be considered in light of the fact that the parser can perform syntactic analyses on the corrected uttterances of these 2.7% and 4.8% additional correct identifications. An experiment described in [Core and Schubert, 1999b] measured grammar coverage on a corpus of 495 repairs from the TRAINS dialogs. Even though the parser was given perfect speech repair information (100% recall), the parser's grammar only covered the corrected utterance in 144 of the 495 repairs (29% recall on correctly parsing the corrected utterance of repairs).

Experiment three showed that grammar coverage is not the only limiting factor in increasing repair recall. Parses that include parts of reparanda may be constructed and lead to choosing incorrect repair hypotheses. It remains to be seen if this problem can be overcome by considering probabilities of repair hypotheses and resulting parses as well as the number of words in the corrected utterance.

To gain full advantage of the parser's knowledge of grammar and the syntactic structure of the input, it may be necessary to involve the parser more directly in the repair identification process. Repairs can be incorrectly identified as part of a coordinate structure as in utterance 24 from d93-13.2:

```
um to pick up two boxes of oranges or two boxcars of oranges
```

The parser will need to reason about the parallelism between *two boxes of oranges* and *two boxcars of oranges* to determine that *two boxes of oranges* is a reparandum. An informal search of the test corpus revealed that 11% of modification repairs (7% of all repairs) were corrections of complete phrases or clauses. The parser could also search for parallelism between incomplete phrases and complete phrases (their possible corrections) in order to rescore the probabilities output by a pre-parser speech repair identifier.

The results even without the use of phrase-level parallelism are promising: a 8.89% increase in recall with perfect grammar coverage and increases in recall from 2.7% to 4.8% in standard tests. Consideration of repair probabilities along with parse probabilities, percentage of words in the corrected utterance, and phrase-level parallelism should allow even better performance.

6 Speech Act Annotation

There are two classes of applications that require the automatic analysis of dialogs: a computer system may act as a participant in a dialog with users, or it may act as an observer attempting to interpret human-human dialogs. In both cases, the system must keep track of how each utterance changes the commonly agreed upon knowledge (common ground, [Clark and Schaefer, 1989]) including the conversational agents' obligations and plans. Dialog text annotated with the communicative actions of each utterance would aid in training and testing such systems. In addition, linguists studying dialog would greatly benefit from annotated corpora that could be used to reveal the underlying structures of dialogs.

Speech act theory [Searle, 1975b] was one of the first attempts at developing a set of communicative actions. Searle's action classification included Representatives, that introduce information into the common ground; Directives, that attempt to create an obligation on the listener; and Commissives, that involve speakers attempting to introduce an obligation on themselves. Over the years, many researchers [Allwood, 1995; Cohen and Levesque, 1990; Hancher, 1979] have noticed that a major problem with speech act theory is that it attempts to capture an utterance's purpose(s) with one label.

A second problem is that research groups have extended Searle's speech acts in different ways to better handle their domains making it difficult to share speech act annotated corpora. Two examples are the VERBMOBIL research group [Reithinger and Maier, 1995] and the TRIPS group [Ferguson and Allen, 1998]. It is not clear how VERBMOBIL's DIGRESS speech act relates to any of the TRIPS speech acts and TRIPS's ACKNOWLEDGE APOLOGY speech act is similarly confusing for the VERBMOBIL group. This confusion prevents sharing of speech act labeled corpora between these two research groups. Giving researchers access to more speech act labeled data would aid the analysis of speech act patterns in dialog.

In order to address these problems, we worked with the Multiparty Discourse Group in the development of the Backward- and Forward-Looking annotation scheme (henceforth the BF scheme) [Allen and Core, 1997], a set of primitive communicative actions that can be used to analyze dialogs. Note, in previous work the BF scheme was called DAMSL (Dialog Act Markup in Several Layers); the Multiparty Discourse Group adopted the new name at their last meeting. The Multiparty Discourse Group is composed of representatives from various research groups studying dialog. The group has convened three times at meetings of the Discourse Resource Initiative (DRI).[1] For the purposes of this chapter, we will define communicative actions as referring to explicit manipulations of the common ground, and not include more subtle phenomena such as listeners forming opinions about speakers based on the tone and style of their speech. The BF scheme allows multiple communicative actions to be assigned to each

[1] See the DRI home page for more details: http://www.georgetown.edu/luperfoy/Discourse-Treebank/dri-home.html

utterance addressing the first problem discussed above. The BF scheme was also designed to be a top-level speech act hierarchy into which other speech act taxonomies could be easily mapped. For example, VERBMOBIL's DIGRESS speech act and TRIPS's TELL speech act might both map to the BF scheme's ASSERT action. Two corpora from different research groups could be combined into one BF annotated corpus that could be analyzed for speech act patterns to answer questions noted in [Core *et al.*, 1998] such as:

- What are the prosodic, lexical, and syntactic patterns associated with different speech acts?
- How do speech acts differ across languages?
- What types of speech act patterns do you see as speakers try to come to agreement?

Section 6.1 introduces the current version of the BF scheme's communicative actions. We labeled a collection of dialogs from the TRAINS-93 corpus with BF tags. In this initial annotation effort, the BF labels were sufficient to cover the speech acts of interest and we did not define a finer-grained domain-specific set of speech acts. Section 6.2 describes the inter-annotator reliability measured on these annotations to see if the multidimensional BF labeling was done reliably. If the BF scheme is supposed to act as a high-level speech act hierarchy to which research groups should provide mappings, then its categories should correspond to phenomena that can be consistently identified. Reliability is measured across BF dimensions, sets of tags that force annotators to make mutually exclusive choices. Some dimensions were reliable while others indicate that revisions in tag definitions are necessary. Section 6.3 discusses a simple analysis of speech act usage in the TRAINS-93 data showing the type of information that can be learned from a speech act annotated corpus. Section 6.4 describes the multi-functionality of utterances as shown by their BF tags, and section 6.5 discusses what future versions of the BF scheme may be like.

6.1 The BF Scheme

The BF scheme has three main layers: Forward Communicative Functions, Backward Communicative Functions, and Information Level. The Forward Communicative Functions consist of a taxonomy in a similar style as the actions of traditional speech act theory. The Backward Communicative Functions indicate how the current utterance relates to the previous dialog, such as accepting a proposal, confirming understanding, or answering a question. Information Level encodes whether the utterance deals with the dialog task, the communication process, or meta-level discussion about the task.

6.1.1 Forward Communicative Function

The main Forward Communicative Functions are shown below. Each function is labeled independently of the others, so, for example, an utterance can be a statement as well as a question. STATEMENTs are defined as claims about the world, and are further subdivided based on whether the speaker is trying to affect the beliefs of the hearer, or is repeating information for emphasis or acknowledgment. Directives fit under the more general category, INFLUENCE-ON-LISTENER, which includes all utterances that discuss potential actions of the addressee. Directives are subdivided into two categories: INFO-REQUEST, which consists of questions and requests such as *tell me the time*, and ACTION-DIRECTIVE, which covers requests for action such as *please take out the trash* and *close the door*. INFLUENCE-ON-LISTENER also includes

OPEN-OPTION where a speaker gives a potential course of action but does not show preference toward it, *how about going to Joey's Pizza.* Commissives are given the more descriptive name, INFLUENCE-ON-SPEAKER, and are subdivided into OFFERS and COMMIT(ments). The OTHER-FORWARD-FUNCTION category is a default choice for communicative actions that influence the future of the dialog in a way not captured by the other categories. Sentence initial words such as *okay* are often separated into separate utterances and marked as OTHER-FORWARD-FUNCTION. These words may have Forward Communicative Functions such as signaling a repair or change in topic or holding the turn (while the person is thinking) as well as Backward Communicative Functions such as accepting and acknowledging. Future work in this annotation effort will include developing classes of OTHER-FORWARD-FUNCTIONS.

- Statement
  - Assert
  - Reassert
  - Other-Statement
- Influence-on-listener
  - Open-option
  - Directive
    - Info-Request
    - Action-Directive
- Influence-on-speaker
  - Offer
  - Commit
- Other-Forward-Function

6.1.2 Backward Communicative Function

The Backward Communicative Functions of the BF scheme are shown in figure 6.1. The classes AGREEMENT, UNDERSTANDING, and ANSWER are independent so an utterance may simultaneously accept information and acknowledge that the information was understood as well as answer a question.

AGREEMENT has several subclasses; ACCEPT and REJECT refer to fully accepting or rejecting an utterance or set of utterances. ACCEPT-PART and REJECT-PART refer to partially accepting or rejecting a proposal. HOLD refers to utterances such as clarification questions that delay the listener's reaction to a proposal or question. MAYBE refers to cases where the listener refuses to make a judgment at this point. The examples in figure 6.2 illustrate each type of agreement in response to the offer *Would you like the book and its review?*.

The UNDERSTANDING dimension concerns whether the listener understood the speaker. The listener may signal understanding or non-understanding or attempt to correct the speaker (showing that they either did not understand or that they did understand but that the speaker misspoke). Non-understanding can be indicated by utterances such as *huh?*, clarification questions (*To Dansville?*) and by explicit questions about what the speaker said or meant. Understanding can be indicated by acknowledgments such as *right* or *okay*, by repeating some of the speaker's utterance, or by continuing or completing the speaker's sentence.

- Agreement
  - Accept
  - Accept-Part
  - Maybe
  - Reject-Part
  - Reject
  - Hold
- Understanding
  - Signal-Non-Understanding
  - Signal-Understanding
    - Acknowledge
    - Repeat-Rephrase
    - Completion
  - Correct-Misspeaking
- Answer

Figure 6.1: Backward Communicative Function

```
Context:      A: Would you like the book and
                  its review?
Accept        B: Yes please.
Accept-Part   B: I'd like the book.
Maybe         B: I'll have to think about it
                  (intended literally)
Reject-Part   B: I don't want the review.
Reject        B: No thank you.
Hold          B: Do I have to pay for them?
```

Figure 6.2: Example Annotations Using the Agreement Label

The ANSWER dimension indicates that an utterance is supplying information explicitly requested by a previous INFO-REQUEST act. This is a highly specific function that you might expect could be generalized into some other form of response, but we have not as yet been able to identify what the generalization would be.

6.1.3 Information Level

The third part of the BF scheme, the Information Level annotation, is shown below and encodes whether the utterance deals with the dialog task, the communication process, or meta-level discussion about the task. This dimension eliminates the need to have tags such as COMMUNICATION-INFO-REQUEST, for utterances such as *What did you say?*, and TASK-INFO-REQUEST for utterances such as *What times are available?*. With this information, we can identify three independent subdialogs within a single dialog. The topic motivating the dialog is developed and discussed in the TASK part of the dialog. The TASK-MANAGEMENT part of a dialog involves explicit planning and monitoring of how well the task is being accomplished. The physical requirements of the dialog (such as being able to hear one another) are maintained in the COMMUNICATION-MANAGEMENT part of the dialog. Note that in some sense all utterances have a COMMUNICATION-MANAGEMENT component. It is only marked, however, when the utterance has no TASK or TASK-MANAGEMENT component.

- Information Level

  Task

  Task Management

  Communication Management

  Other

6.2 Inter-Annotator Reliability

This section presents reliability results from labeling a subset of TRAINS-93 dialogs with BF tags. A more detailed discussion of this work is presented in [Core and Allen, 1997]. Unfortunately, this subset of TRAINS dialogs and that used in chapters 4 and 5 was not the same. In the experiments of chapters 4 and 5, utterance segmentations were hand-annotated in accordance with the parser's grammar. Since speech act detection takes place during parsing, it would have been helpful to work with the same utterance units. However, in this project, a separate utterance segmentation was performed. Given that the segmentation was intended only for speech act labeling, utterance segment boundaries were made only where annotators thought the communicative actions of the speaker changed. In some cases, a series of main clauses each performed different actions and the utterance segmentation was mostly syntactic:

```
utt1 u: we'll get that couch
utt2    how about that end table?
```

Other times, a series of main clauses performed the same communicative actions and were grouped together:

```
utt1 u: we'll take the train to Corning
|       then we'll pick up boxcars in Avon
|       and go on to Dansville to pick up oranges
```

Our speech act annotation tool does not support discontinuous utterances. Instead, where a second speaker interrupts a first speaker, segmentation works as follows:

```
utt1 u: let's take the boxcars + to + Dansville
utt2 s: + mm-hm +
utt3    that will be four hours
```

Plus signs mark where utterance 2 interrupted utterance 1.

Three undergraduates and a graduate student were given informal training consisting of annotating some dialogs and having their results compared against canonical annotations as well as comparing their results against one another. A GUI-based annotation tool, DAT[2] was developed to test the BF scheme. This tool displays the dialogs and can play audio for individual utterances so annotators can listen to the actual dialogs as well as studying the transcripts. DAT also gives warnings to users when a suspicious pattern of inputs is entered and allows them to correct the annotation if desired. Here is a list of what the tool defined as suspicious.

- Question and answer have different Information Levels
- An acceptance that is not an acknowledgment
- An acknowledgment (but not acceptance) that is not at the COMMUNICATION-MANAGEMENT Information Level
- Answers that are not ASSERTs.
- A check question[3] whose answer does not have an AGREEMENT label.
- A response to an ACTION-DIRECTIVE or OPEN-OPTION that does not have an an AGREEMENT label.
- A response to a question that is not an answer.

After training, the students independently annotated a series of dialogs as shown in table 6.1[4]:

6.2.1 Results

The statistics used to measure interannotator reliability are percent pairwise agreement (PA), expected pairwise agreement (PE), and kappa (PA adjusted by PE): $K = \frac{PA - PE}{1 - PE}$. These are defined formally in [Siegel and Castellan, 1988]. For PA, statistics were collected for each tag dimension over each dialog. Then an average PA was computed as follows: average $= \sum(PA_i * TAPT_i) / \sum TAPT_i$ where TAPT is total annotations per tag and PA_i is the PA for a tag over dialog i.

[2] Available at http://www.cs.rochester.edu/research/trains/annotation/

[3] A check question is defined in the annotation manual as a statement about the world made for the purposes of confirmation, as in *We're using the blue sofa, right?* or *We're using the blue sofa* uttered with a rising intonation. Check questions are labeled as both STATEMENTs and INFO-REQUESTs and their answers are both ASSERTs and ACCEPTs (or possibly REJECTs).

[4] d1-d8 correspond to TRAINS dialogs d92a-2.1, d92a-2.2, d92a-3.1, d92a-4.1, d92a-4.3, d93-13.2, d93-13.3, and d93-16.1.

Dialog	Utts	Annotators	Total Annotations/Tag
d1	133	2 UG	266
d2	72	2 UG	144
d3	40	2 UG 1 GR	120
d4	41	1 UG 1 GR	82
d5	19	1 UG 1 GR	38
d6	88	1 UG 1 GR	176
d7	159	1 UG 1 GR	318
d8	52	1 UG 1 GR	104
total	604		1248

UG = undergraduate
GR = graduate student

Table 6.1: Experimental Setup

To calculate PE all 8 dialogs were concatenated, and PE was measured for each tag dimension. To see why PE must be calculated differently, consider how the PE for the statement dimension is calculated:

$$PE = prob(NoStatement)^2 + prob(Assert)^2 + prob(Reassert)^2 + prob(OtherStatement)^2$$

Since the probability estimates are squared, it makes more sense to calculate PE once for all the data rather than eight times (with poorer probability estimates for each of these calculations).

Siegel and Castellan in [Siegel and Castellan, 1988] show how to test the significance of kappa scores, to see whether a kappa score was a result of chance or reflects the agreement among the annotators. The equations given by Siegel and Castellan are shown below. It is assumed that kappas are normally distributed. Siegel and Castellan give a table of one tailed significance levels indexed by z values. The least significant level given is 0.1 so dashes in the results tables represent cases where the level does not reach 0.1. The best level given is 0.000005 so entries marked 0.000005 mean 0.000005 or better.

$$var(K) \approx \frac{2}{Nk(k-1)} \frac{PE - (2k-3)PE^2 + 2(k-2)\sum p_j^3}{(1-PE)^2}$$

$$z = \frac{K}{\sqrt{var(K)}}$$

Tables 6.2 and 6.4 show the PA, PE, and kappas for the BF tag dimensions. IOL is INFLUENCE-ON-LISTENER, and the Resp-to abbreviation refers to Response-to, an annotation of which utterances a response responds to.

Tables 6.3 and 6.5 show the significance levels of kappa for individual dialogs. Because we included 3 annotators in dialog 3, it is unclear how to calculate global kappa variance since one of the parameters to variance is number of annotators. The significance levels in tables 6.2 and 6.4 are based on two annotators which is close to correct since only 40 tags were contributed by the third annotator on dialog 3.

The global kappa for INFLUENCE-ON-SPEAKER turned out to be non-significant and is not included in the results. INFLUENCE-ON-SPEAKER (offers and commits) are not rare tags as

Measure	Statement	IOL	Other For Funct	Info level
PA	0.82	0.88	0.92	0.82
PE	0.47	0.60	0.85	0.56
Kappa	0.67	0.71	0.48	0.59
Signif	0.000005	0.000005	0.000005	0.000005

Table 6.2: Reliability for Main Forward Function Labels

Dialog	Tags	Statement	IOL	Other For Funct	Info level
d1	266	.000005	.000005	-	.00005
d2	144	.000005	.0005	-	.005
d3	120	.000005	.000005	-	.000005
d4	82	.000005	.00005	-	.0005
d5	38	.1	-	-	.025
d6	176	.000005	.000005	.005	.000005
d7	318	.000005	.000005	.001	.000005
d8	104	.000005	.0005	-	.001

Table 6.3: Significance Levels for Main Forward Function Labels

commitments occur when speakers agree to an action. It is, instead, annotator disagreements that make the kappa estimate non-significant.

According to [Carletta, 1996] even for tentative conclusions to be drawn, kappas must be above 0.67 with above 0.8 being considered reliable. The results suggest that with revisions to the annotation manual, annotators should be able to produce labelings of at least usable quality (between 0.67 and 0.8).

The AGREEMENT dimension has one of the lowest kappas and the kappa for INFLUENCE-ON-SPEAKER was not significant due to annotation variance. The major reason for disagreements in these dimensions is that annotators have a hard time deciding whether a response is an acceptance (labeled under the AGREEMENT dimension) or just an acknowledgment. In the example below, it is unclear whether *u* thinks going through Corning is a good idea or is waiting to hear more before making a judgment.

```
s: so we'll take the train through Corning
u: okay
s: and on to Elmira.
```

Hearing the audio sometimes helps, but there are many cases where the annotator would have to be able to read the speaker's mind in order to make the distinction. To make matters

Measure	Understand	Agree	Ans	Resp-to
PA	0.83	0.78	0.95	0.83
PE	0.59	0.61	0.72	0.28
Kappa	0.58	0.43	0.81	0.77
Signif	0.000005	0.000005	0.000005	0.000005

Table 6.4: Reliability for Backward Function Labels

Dialog	Tags	Understand	Agree	Ans	Resp-to
d1	266	.000005	.00005	.00005	.000005
d2	144	.000005	.0005	.1	.000005
d3	120	.0005	-	.000005	.000005
d4	82	.005	.025	.0005	.000005
d5	38	.025	-	-	.01
d6	176	.0005	.005	.000005	.000005
d7	318	.000005	.000005	.000005	.000005
d8	104	-	.005	.0005	.000005

Table 6.5: Significance Levels for Backward Function Labels

Dimension	Interp 1	Interp 2
Understanding	ACK	ACK
Agreement	N/A	ACCEPT
IOS	N/A	COMMIT
Info Level	COMM-MANAGE	TASK

Table 6.6: Two Interpretations of an Utterance such as "okay".

worse, this one decision also affects two other dimensions: the INFLUENCE-ON-SPEAKER dimension because acceptances many times mean commitment but acknowledgments do not, and the Information Level dimension since acknowledgments are at the COMMUNICATION-MANAGEMENT level while agreements are at the TASK level. Thus, we have differences in at least three dimensions based on a single subtle distinction that often cannot be made. The two interpretations are summarized in table 6.6.

This problem, where a slight change in interpretation causes major changes in the annotation, clearly indicates a need for revision. One possibility would be to introduce some labels that capture the ambiguity, but this would have to be done in each dimension and might serve to aggravate the problem by introducing additional choices. The other possibility is to force an agreement reading based on how the proposal/request is eventually treated in the dialog. Thus in the example above, unless the speaker goes on to reject or question the proposal, the response would count as an implicit accept, Interpretation 1 would not be allowed, and the response would have to be labeled with some AGREEMENT tag. Following this rule could be encouraged by having DAT give the user a warning every time an utterance is tagged an acknowledgment but no AGREEMENT tag is specified.

The OTHER-FORWARD-FUNCTION category also has a low kappa score; this is partially due to the fact that the expected agreement for it is high since its value is usually NOT-PRESENT. This category applies most often to words such as *okay* that are very ambiguous in their meaning even when heard in context. It will be interesting to develop subcategories of OTHER-FORWARD-FUNCTION such as "turn holding" and "signaling a repair" to give us a better idea of what phenomena annotators are having trouble labeling.

Most of the other labels have kappas around 0.6 meaning the annotations are fairly reliable but that some problems still remain. One problem that affects several labels involves check questions. Check questions are statements about the world made for the purposes of confirmation, as in *We're using the blue sofa, right?* or *We're using the blue sofa* uttered with a rising intonation. Check questions are labeled as both STATEMENTs and INFO-REQUESTs and their answers are both ASSERTs and ACCEPTs (or possibly REJECTs). However, it is difficult for an-

notators to consistently recognize a check question, leading to disagreements in the STATEMENT and INFLUENCE-ON-LISTENER dimensions (is it a statement, is it a question?), and disagreements about whether the next utterance is an ANSWER and ASSERT or is only labeled ACCEPT (or REJECT).

Another problem arises with indirect speech acts such as requests made by statements such as *it would be nice to have some light.* There is a continuum of interpretations for such an utterance, ranging from a pure ASSERT act through to a pure ACTION-DIRECTIVE act depending on the annotator's view of what the speaker intended and how the utterance was taken in the dialog. The BF scheme alleviates this problem somewhat by not forcing an annotator to choose between the two options. They can mark an utterance as both acts. In practice, however, we still see a fair amount of inconsistency and some more specific guidance appears to be needed. This may have to be done on a domain-by-domain basis, however. For instance, in the TRAINS-93 domain, the users often state their goals, as in *I have to get trains there by noon.* We have been taking these utterances simply as ASSERTs, but this is somewhat arbitrary as there is a sense in which such utterances influence the hearer's future action as with ACTION-DIRECTIVEs.

Another difficult example in TRAINS occurs when the speaker summarizes a plan that has already been developed, as in:

```
utt1: s: we'll go through Corning
utt2: u: mm-hm
utt3: s: pick up the oranges, and
          unload at Dansville
```

If utt1 and utt3 are really just descriptions of what has been agreed upon, they would be REASSERTs, but annotators often want to add an ACTION-DIRECTIVE interpretation as well because of their surface form. Such cases may be resolved with domain-specific instruction, but it is unclear whether unambiguous generic instructions can be found.

Another problem with the STATEMENT dimension is the label, REASSERT. When information is asserted that has been discussed previously, the annotators have to decide whether the speaker thinks the information was forgotten by the hearer (and thus constitutes an ASSERT) or whether the speaker is trying to reintroduce the information to make a point (and hence it would be a REASSERT). A similar confusion occurs with the REPEAT-REPHRASE tag of the UNDERSTANDING level where annotators have to decide how far back a REPEAT-REPHRASE utterance can refer and how close the paraphrase must be. Annotators also get confused if a speaker simultaneously makes a repetition and goes on to make a correction or completion. Some work needs to be done to clarify the definitions of these labels.

Another label that confuses annotators is the TASK-MANAGEMENT label of the Information Level dimension. In TRAINS-93, the domain is planning so an utterance such as *we can't do that because there is a train already on that track* is TASK level but something like *we could do that another way. do you want to change the plan?* would be considered TASK-MANAGEMENT since it explicitly discusses the course of the dialog while the first only implicitly signals a possible change in the course of the dialog. The difference is very subtle and hard to annotate.

6.3 Corpus Analysis

In this section, we describe a simple statistical analysis of speech act patterns in BF labeled TRAINS-93 data. A more detailed discussion is presented in [Core, 1997]. Our focus in this study is how speakers respond to one another's speach acts and which speech acts co-occur on

Tag Set	Most Freq. Value	Prob
Other-ForF	No Other-ForF	0.926
Info-level	Task	0.772
Statement	Not Statement	0.541
IOL	No Influence	0.726
IOS	No Commitment	0.762
Agreement	No Agreement	0.636
Understanding	No Understanding	0.684
Answer	Not an Answer	0.853

Table 6.7: Highest Frequency Tag Values and their Probabilities

utterances. We look at speech act bigrams and the conditional probabilities of BF tags given the presence of another BF tag on the same utterance. The unigram probabilities of BF tags in the corpus are also presented. Future work will include looking at speech acts patterns beyond just utterances and their responses as well as looking at how speech acts are marked by prosody and lexical choice.

The eight dialogs discussed in section 6.2 were annotated by two or more labelers. In seven of those dialogs, the labelers met and produced an annotation that they agreed on.[5] Annotation continued, producing 11 more annotated dialogs resulting in a total of 1524 utterances.[6] The most frequent values for each type of tag and their estimated probabilities[7] are listed in table 6.7

Using the estimated probabilities of all the tag values for the 8 major BF tag sets, cross entropy and perplexity can be computed. In this case, we are using cross entropy to measure how random the distributions of a tag's values are (given no context). The formula used here for cross entropy is:

$$H = -(1/K)\sum_{i=1}^{N} c_i * log_2(p(v_i)) \tag{6.1}$$

where v_i indicates a tag value, K is the size of the corpus (1524), c_i is how often v_i appears in the corpus, and $p(v_i) = c_i/K$. Perplexity is simply 2^H; an intuitive view of perplexity is that it gives the average number of choices faced by a tag prediction model. In the case of a two valued tag such as OTHER-FORWARD-FUNCTION, the worst perplexity is two; the values are completely random and each is equally likely. The best perplexity will always be one; only one value is possible.

Table 6.8 shows the cross entropy and perplexity of the 8 major BF tag sets. The results show that the high frequency tag values keep the perplexity of tag prediction low.

The unigram probabilities of table 6.7 do not tell us much about our corpus of dialogs. The fact that 54.1% of utterances are not statements is not surprising since we would expect a mixture of commands and questions as well as statements.[8] To investigate more interesting

[5] The eighth dialog was left out because one of the labelers left the project before a new labeling could be worked out.

[6] These included d92a-1.3, d92a-1.4, d92a-3.2, d92a-4.4, d92a-5.1, d93-10.1 d93-12.1 d93-12.2, d93-12.4, d93-16.2, and d93-16.3.

[7] frequency divided by the total number of utterances

[8] Commands and questions can have statement aspects to them but typical examples such as *take the Corning path* or *how far is that?* do not.

Tag Set	Cross Entropy	Perplexity	# Tags
Other-ForFunct	0.381	1.30	2
Info-level	0.957	1.94	5
Statement	1.4	2.64	4
IOL	1.2	2.29	4
IOS	0.97	1.96	3
Agreement	1.25	2.37	7
Understand	1.19	2.29	6
Answer	0.6	1.52	2

Table 6.8: Entropy and Perplexity of Current Annotations

Conditional Probability	Value	Unigram
$P(Assert_i$|Task-management$_i)$	0.586	0.373
$P(Assert_i\|Answer_i)$	0.955	0.373
$P(Assert_i$|Other-Statement$_{i-1})$	0.841	"
P(Info-request$_i$|Other-statement$_i$)	0.889	0.152
$P(Assert_i$|Info-request$_{i-1})$	0.791	0.373
$P(Assert_i$|Signal-non-understand$_{i-1})$	0.647	"
P(Info-request$_i$|Signal-non-understand$_i$)	0.882	0.152
$P(Hold_i$|Signal-non-understand$_i)$	0.706	0.033
$P(Assert_i\|Hold_{i-1})$	0.740	0.373
$P(Assert_i$|Open-option$_i)$	0.806	0.373
$P(Offer_i$|Open-option$_i)$	1	0.052
$P(Assert_i$|Action-directive$_i)$	0.558	0.373
$P(Assert_i\|Reject_i)$	0.971	"
$P(Assert_i\|Competition_i)$	0.909	"
$P(Accept_i$|Action-directive$_{i-1})$	0.699	0.306
$P(Accept_i$|Other-statement$_{i-1})$	0.683	"
$P(Accept_i\|Repeat_i)$	0.641	"
$P(Accept_i\|Acknowledge_i)$	0.797	"

Table 6.9: Useful Conditional Probabilities

probabilities, we measured conditional probabilities that indicate cases where a tag is more probable than the most likely tag (given no context). These probabilities are shown in table 6.9 with the third column showing the relevant unigram tag probability. The subscripts refer to utterance numbers; some of the conditional probabilities involve tags of the same utterance while others refer back to a previous utterance.

The first set of probabilities in the table shows that many ASSERTs are also labeled TASK-MANAGEMENT. TASK-MANAGEMENT utterances in the BF scheme are defined as meta-level discussion about planning and experimental procedure: *let's work on the orange juice problem first*, *Are we done?*. 34 of the 58 task management utterances in the corpus were ASSERTs. Many of these utterances state that the task is completed: *we are done*. The rest are various assertions about the problem solving process: *we need to do this part first*, *I added wrong*, *either solution is fine*. Since in general ASSERTs only occur 37.3% of the time, this information is valuable and gives us insight into meta-level planning discussion.

The second set of probabilities in the table involves questions and answers. OTHER-STATEMENT is highly co-related to INFO-REQUESTs as the combination serves to mark check questions. ASSERTs are highly co-related to answers. Thus, the probability of an ASSERT is higher after an OTHER-STATEMENT. Interestingly, the conditional probability of an ASSERT immediately following an INFO-REQUEST is lower. Sometimes, the responder will start off with an acknowledgment or an editing term *um*, *let's see*. Other times, the question will be answered with a question or old material (REASSERT).

Signaling non-understanding is commonly done through questions; answers are likely to follow so that the next utterance is likely an ASSERT. When a speaker makes a request or assertion, the listener is obligated to at least implicitly accept or reject the utterance. Sometimes the listener will ask for clarification or otherwise delay their judgment. These utterances are marked HOLDs. Since clarification is needed if a listener does not understand the previous utterance, HOLDs and SIGNAL-NON-UNDERSTANDINGs co-occur. Answers that are ASSERTs often follow HOLDs.

The third set of probabilities in the table concerns OPEN-OPTIONs and ACTION-DIRECTIVEs. OPEN-OPTIONs are weak suggestions by the speaker. These are likely to be ASSERTs (*there are oranges at the Corning warehouse you could use*). The annotation manual states OPEN-OPTIONs are usually OFFERs explaining why annotators have so far marked every OPEN-OPTION as an OFFER. ACTION-DIRECTIVEs are requests by the speaker that the listener perform an action. In the TRAINS domain, 55.8% of the ACTION-DIRECTIVEs are also ASSERTs. Since the TRAINS domain is about planning, these are utterances such as *the train will go through Corning* that serve as statements but are also requests that the listener accept the action as part of a plan. A common way to make a rejection is through an ASSERT, and completions of another speaker's utterance are often marked ASSERT as well.

The last set of probabilities in the table involves the frequency of acceptances. In TRAINS, the two speakers have roughly the same expertise although one speaker has travel time information and the other does not. 70% of the responses to ACTION-DIRECTIVEs were ACCEPTs because the listener is not likely to immediately to see a problem in a speaker's request since they both have similar levels of expertise and knowledge. The same is true of check questions indicated by their OTHER-STATEMENT tag. Given that an utterance is an acknowledgment (*okay*, *right*) or an acknowledgment made through repetition, it is likely that the utterance is also an acceptance.

Using a sophisticated statistical tool such as a decision tree should bring out additional results based on looking at more complex patterns of BF tags. The addition of cue words and sentence forms to the context should bring new developments as well. Another area for exploration is the use of data mining techniques to extract interesting patterns from the corpus. [Dehaspe and Raedt, 1997] tries such an approach on part-of-speech tags in a corpus of Wall Street Journal text.

6.4 The Multi-Functionality of Utterances

Section 6.3 estimated the conditional probabilities of various BF tags being present on an utterance given that it had a BF tag from another dimension: $P(A_i|B_i)$. If this probability is greater than the apriori probably of the most likely tag in A's dimension then it was reported in table 6.9. Some of the co-occurrences can be explained by the fact that some BF tag definitions make reference to syntax. Thus, the co-occurrence between INFO-REQUEST and SIGNAL-NON-UNDERSTANDING simply means that questions are often used to signal non-understanding. Co-occurrences with deeper meaning are signaled by $P(Hold_i|\text{Signal-non-understand}_i)$ and $P(Offer_i|\text{Open-option}_i)$. To understand these co-occurrences and to search for other such patterns, we extracted all patterns of two or more BF tags appearing on the same utterance 6 or more times in the corpus of section 6.3 (see tables 6.10 - 6.18 for the results).[9] These patterns correspond to full annotations of utterances (the corpus has 1524 utterances). Thus, the patterns ACCEPT,ACKNOWLEDGE and ACCEPT,ACKNOWLEDGE,COMMIT are disjoint since the first pattern is really ACCEPT,ACKNOWLEDGE,NO-COMMITMENT.

In the current version of the BF scheme, not all of the dimensions are independent. For example, if an utterance has a function at the AGREEMENT level then it also should have a function at the UNDERSTANDING level as shown in table 6.10. ACCEPT occurs often with the UNDERSTANDING tags ACKNOWLEDGE and REPEAT-REPHRASE. The probability $P(Hold_i|\text{Signal-non-understand}_i) = 0.71$ (from table 6.9) indicates that non-understanding is linked to the ACCEPTANCE dimension through the HOLD tag (which indicates that a speaker is delaying their agreement judgment). Commits often also appear with acceptances and acknowledgments since acceptance can mean commitment as shown in table 6.11.

Although Information Level is usually tied to the content of the utterance, with utterances such as *okay* it is heavily tied to the AGREEMENT, OTHER-FORWARD-FUNCTION, and UNDERSTANDING dimensions. The annotation manual states that an acceptance has the same Information Level as what it accepts. Annotators followed these instructions as shown in table 6.12. These are patterns belonging to acceptances of COMMUNICATION-MANAGEMENT and TASK-MANAGEMENT utterances.

The annotation manual also states that utterances tagged OTHER-FORWARD-FUNCTION, or tagged ACKNOWLEDGE or REPEAT-REPHRASE (but not ACCEPT) will be marked as COMMUNICATION-MANAGEMENT. Table 6.13 shows that the annotations reflected these instructions.

Some BF tags have ties to an utterance's surface form as shown in table 6.14. The ASSERT tag is serving to identify directives, rejections, and answers in the form of declarative sentences or short phrasal answers. The INFO-REQUEST tag indicates that offers and holds are sometimes stated in question form.

Table 6.15 shows how joint action requests and suggests have both commissive and directive functions. Requesting that someone helps in a joint action means the speaker is committing to performing their part as long as the listener agrees to help. Suggestions (OPEN-OPTIONS) of joint actions take a similar form: OPEN-OPTION,OFFER. Since the utterance is a suggestion rather than a request, the speaker is merely offering to perform the action not requesting and committing to it.

Check questions are a special syntactic construction where a declarative clause functions as a question such as *there are oranges in Corning right?*. Not all check questions have a word such as *right* following them and may only be marked by intonation. The annotation manual states that check questions should be labeled as OTHER-STATEMENT or REASSERT depending on whether the question repeats material from previous utterances. This explains the co-occurrence

[9] The tag, TASK was not included because of its high frequency.

accept acknowledge	173
accept acknowledge communication-management	13
accept repeat-rephrase	12

Table 6.10: Agreement Means Understanding

commit accept acknowledge	102
commit accept	7
commit accept acknowledge communication-management	6

Table 6.11: Acceptance Can Mean Commitment

of these tags with the INFO-REQUEST tag as shown in table 6.16. In some cases, the annotators did not follow directions and labeled check questions as ASSERTs. The syntactic construction of the check question may be used to make an offer or a clarification (HOLD).

Many times, questions deal with actions: *so we'll take the Corning engine right?*. In such cases, the responses are labeled as answers as well as rejects and accepts (see table 6.17). Table 6.18 shows how *can I help you* was annotated in 17 dialogs.

6.5 Future Work

Searle's different categories of illocutionary force are useful ways to think about the actions performed by an utterance. These categories are not mutually exclusive; an utterance can make a commitment as well as a statement about the world. These categories as translated into dimensions of the BF scheme are also not independent as shown in the previous sections. This makes annotation difficult and hinders translation from other speech act taxonomies to the BF scheme.

[Core *et al.*, 1998] starts to specify how to make the dimensions of the BF scheme less interdependent. A separate check question tag will prevent the need to label these as both questions and statements. The ANSWER and AGREEMENT dimensions will likely be merged so that a response is either negative, positive, or the answer to a wh question.[10]

The bad inter-annotator reliability seen in section 6.2 and the analysis of the speech act labeled TRAINS-93 data in sections 6.3 and 6.4 suggest that a low dimensional BF scheme might give better reliability without loss of information. In many cases, utterances received more than one BF label in order to capture syntactic as well as speech act information. Since most syntactic information can be captured by a shallow parser it makes sense to only require annotators to label speech acts not directly connected to syntactic form. The analysis of sections 6.3 and 6.4 should be repeated with other BF annotated corpora to see if the TRAINS-93 data

[10] In keeping with the previous version of the BF scheme, partial-negative-response and partial-positive-response may be included although these are low frequency phenomena.

task-management accept acknowledge	11
communication-management assert commit accept answer	8

Table 6.12: Info-level Same for Proposal and Acceptance

communication-management other-forward-function	93
communication-management acknowledge	63
communication-management repeat-rephrase	13
communication-management other-forward-function acknowledge	9

Table 6.13: Communication Management by Definition

assert answer	85
assert action-directive commit	63
assert task-management	24
info-request offer	17
assert completion	14
assert reject	12
assert action-directive	9
info-request hold	6
info-request task-management	6

Table 6.14: Encoding Surface Form

action-directive commit assert	63
action-directive commit	28
open-option offer assert	17
action-directive commit reassert	8
open-option offer	6

Table 6.15: Encoding Joint Actions

other-statement info-request	34
assert info-request	15
reassert info-request	10
other-statement info-request offer	6
other-statement info-request hold	6

Table 6.16: Check Questions

assert accept answer	52
assert accept answer commit	19
assert reject answer	16

Table 6.17: Answers to Requests or Open-options

communication-management info-request offer	17

Table 6.18: "Can I help you"

simply did not illicit certain combinations of tags. Below, we propose a low-dimensional BF scheme based on our experience with the TRAINS-93 data. This proposal may need to be modified as other corpora are analyzed or as trouble is encountered annotating with the low-dimensional BF scheme. Currently this direction seems the most promising.

The BF tags REASSERT and REPEAT-REPHRASE are attempts to identify information repeated in the dialog. Originally Old/New Information was a separate dimension with bad inter-annotator reliability. However, it is not clear that including this distinction under STATEMENT and UNDERSTANDING is helping. If REASSERT and OTHER-STATEMENT were removed then STATEMENT could be easily merged with INFLUENCE-ON-LISTENER since its only remaining tag would be ASSERT.[11] Only general claims that were not made to suggest or request would be labeled ASSERT.

As a first step at eliminating the separate commissives dimension, ACTION-DIRECTIVE and OPEN-OPTION need to have joint action counter-parts JOINT-ACTION-DIRECTIVE and JOINT-OPEN-OPTION. COMMIT and OFFER would still have to exist but merged with the tags in INFLUENCE-ON-LISTENER and thus mutually exclusive with requests and suggests. In such a scheme, COMMIT would only refer to pure commitments (that are not also requests or suggests), *I will do the dishes tonight*, and likewise for OFFERs, *I am willing to cook dinner*.

OTHER-FORWARD-FUNCTION, INFLUENCE-ON-LISTENER, and INFLUENCE-ON-SPEAKER will then form a single FORWARD-COMMUNICATIVE-FUNCTION dimension. Currently OTHER-FORWARD-FUNCTION labels turn-taking actions such as holding a turn by saying *well* or *okay* and should be mutually exclusive with the rest of the forward function tags.

The backward communicative functions could also be merged into a single dimension given the interdependency between the UNDERSTANDING and AGREEMENT dimensions. If REPEAT-REPHRASE is eliminated, the resulting BACKWARD-COMMUNICATIVE-FUNCTION dimension would have the following tags: WH-ANSWER, NEGATIVE-RESPONSE, POSITIVE-RESPONSE, MAYBE, HOLD, SIGNAL-NON-UNDERSTANDING, ACKNOWLEDGE, COMPLETION, AND CORRECT-MISSPEAKING. SIGNAL-NON-UNDERSTANDING should be a subtype of HOLD since a speaker cannot judge their agreement with an utterance they did not understand. ACKNOWLEDGE would now be mutually exclusive with ACCEPT. The problem of differentiating between them would still exist but at least it would be confined to one dimension. COMPLETION and CORRECT-MISSPEAKING would also be mutually exclusive with ACCEPT. Intuitively this seems right as speakers can complete and correct without acceptance:

```
u: I want to borrow
s:                  the lawn mower
```

Here, *s* may not be happy about loaning out the lawn mower and may reject the request: *yeah, well, I was planning on using it today.*

However, this will have to be worked out through corpus analysis. The relation between forward and backward communicative functions will also need to be specified. Speakers can answer a question with another question. This second question clearly has both a forward and backward function. However, labeling standard question answers as ASSERTs seems redundant with automatic syntactic annotation. Also we still have the problem of deciding whether ACCEPTs mean commitment.

It remains to be seen whether a low dimensional BF scheme will easily map to existing speech act taxonomies. However, a low dimensional BF scheme should allow annotators to achieve good inter-coder reliability while ensuring that all the main communicative actions of utterances are encoded.

[11] The primary purpose of OTHER-STATEMENT has been for check questions.

7 Conclusion and Future Work

Sections 7.1, 7.2, 7.3, and 7.4 present future work for chapters 3, 4, 5, and 6 respectively. Section 7.5 presents a general conclusion.

7.1 Future Work in Dialog Parsing

Chapter 3 introduced our parsing framework and brought up several issues. Although the framework has been tested on 31 TRAINS-93 dialogs (19,189 words), it needs to be tested on more data especially with regard to overlapping words. We linearize the words of the two speakers using word end points. This works for the 31 TRAINS-93 dialogs but it is unclear whether a more sophisticated technique will be needed in general.

The question explored in section 3.5 was "what is the generative power of a grammar accepting speech repairs?".[1] Let p equal the number of words shared between the repair's correction and the smallest XP including the reparandum. We showed that if we take p as having a fixed bound, we can construct context-free grammars accepting speech repairs. However, such a restriction seems arbitrary. If p is unbounded then unlimited length strings must passed as feature values in the grammar. In general this allows context-sensitive languages to be recognized. However, the features were used in a limited fashion in our analysis to implement the rule `XP -> XP XP` where the second XP may be missing words from its left side that appear on the first XP's left side. The question is still open as to what is the generative power of the grammar formed by adding this rule to a context-free grammar.

Another question we left open is whether the worst-case running time of the parser is necessarily exponential in the input size as discussed in section 3.6. Here the input size, n, refers to the number of words in a discourse unit (DU); a DU is defined as having no utterances crossing its boundaries. The algorithm we present to construct the chart and extract a syntactic analysis is exponential; however, we have not proven that parsing with metarules is necessarily exponential and a more sophisicated algorithm may be able to perform better.

An additional question about the framework is "how could it be extended to multi-party (more than two person) dialogs?". Much of the framework is independent of the number of speakers in the dialog. However, future work will include specifying details such as: "Can speakers collaborate to form a repair of another speaker's utterance?", "If speaker B interrupts speaker A, can speaker C then continue A's utterance?".

The dialog parser needs further testing and development of its ability to construct a representation of what the user started to say before a repair. Our prototype implementation

[1] We have to examine this question in the context of monologs since switching between speakers is not something a context-free grammar can accommodate without metarules or a similar mechanism.

currently does not check feature value restrictions as it builds a broken constituent representation and uses heuristics to select among the many possible continuations of a broken constituent. Corpus analysis is needed to determine exactly how useful reparanda are to a dialog system, and answer questions such as: "can word fragments provide evidence to a speech recognizer or dialog manager about what a speaker does not mean (priming what they might say next)?", "can reparanda provide evidence about a speaker's intentions?", and "can reparanda provide specific evidence about speaker confusions?". Since our parser is a modified version of the one in the TRIPS dialog system [Ferguson and Allen, 1998], after suitable changes are made to the TRIPS dialog manager, we can test our parser with the whole system. These tests will measure performance decreases from speech recognition errors and performance increases from reasoning about editing terms and reparanda and allowing users to make continuations and mid-utterance corrections.

7.2 Future Work in Finding Utterance Endings

The only utterance boundary detection work other than our own that considers mid-utterance interruptions by other speakers is that of [Stolcke and Shriberg, 1996a] as discussed in detail in section 4.1. To detect utterance boundaries, Stolcke and Shriberg use n-gram word models including tokens for utterance boundaries and turn markers. Stolcke and Shriberg's utterance boundary detector performs comparably to our utterance boundary detector (as presented in chapter 4). We ran our detector on two slightly different corpora achieving 75.95% recall and 74.17% precision on one corpus and 78.29% recall and 77.11% precision on the other. Stolcke and Shriberg achieve 76.9% recall and 66.9% precision. Making the assumption of perfect part-of-speech tagging they achieve 79.6% recall and 73.5% precision. Although we used a parser, we did not assume perfect part-of-speech tagging or perfect parsing.

Second speaker interruptions are bracketed in Stolcke and Shriberg's data and utterance endings are marked by the tag <S>. A sample of their data is "A: and, uh, you know, colds and things like that <laughter> get -- B: Yeah <S> A: -- spread real easy and things <S>" (p. 1006). The -- delimit the backchannel, *Yeah*. Such an annotation scheme is somewhat simplistic. For expository purposes assume A is female and B male. Stolcke and Shriberg's data does not indicate whether B starts a new utterance or continues an utterance he had already started. B can be marked as continuing A's utterance (by removing the interruption annotations). However, if A ignores B's continuation and continues her utterance after B's attempted continuation, this cannot be differentiated from A and B collaborating to build an utterance.

Our approach was also inadequate. As explained in section 2.1, we break dialogs into dialog units (DUs) ending where there are no more ongoing utterances. The utterance boundary detection in chapter 4 was really DU boundary detection. Usually DUs correspond to single utterances but sometimes consist of overlapping utterances. Also DUs may have embedded backchannel utterances. We need a model that considers the following possibilities:

- there is no change of speaker and,
  - (1) the current utterance is continued
  - (2) a new utterance is begun
- there is a change of speaker and the new speaker:
  - (3) starts a new utterance
  - (4) continues the previous utterance

(5) continues their last utterance

After each word, the event variable BT (boundary type) equals one of these values (1-5) depending on how the input continues. If we used the same types of evidence as the detector in chapter 4 to estimate BT we would get the following expression:
$P(BT_i|SpkrChange_i, tone_i, sil_i, CatEndingConstit_{0,i}, fragment_i)$.

- $SpkrChange_i$ = presence of a speaker change following word i.[2]
- $tone_i$ = presence of a boundary tone following word i.
- sil_i = length of silence following following word i.
- $fragment_i$ = presence of a word fragment following word i.

The evidence $CatEndingConstit_{0,i}$ assumes that previous BT variables have been assigned values. Thus for each word in the input, we know the starting point of the utterance to which it belongs. $CatEndingConstit_{0,i}$ is defined as the "best" (UTT > S > VP > PP > NP) syntactic categories of constituents ending at word 0 through i and starting at the utterance beginnings associated with these words. This evidence helps determine if a previous utterance was incomplete and thus likely to be continued.

In deciding whether a new utterance begins after a speaker change, additional prosodic information may be important. The energy and intonational contour of a continuation should be different than that of a new utterance. If the continuation refers back to a previous utterance (5), its energy and prosodic contour may be different than that of a direct continuation (4). Word overlap information can indicate when speakers are ignoring each other (not continuing each other's utterance). The distance between an utterance and its potential continuation in case (5) is also an important factor to consider.

Determining boundary type must be done in an integrated fashion with speech repair identification as suggested by [Heeman and Allen, 1997].[3] Section 4.3 shows that speech repair information can reduce false alarms in DU boundary detection since reparanda by definition cannot extend past the utterance in which they start. Similarly knowing utterance start and end points constrains the speech repair search.

In the current DU boundary detection experiments, lookahead is minimal and confined to whether a series of editing terms or discourse markers follow the current word. In future boundary type identification experiments, we need to see the effect of passing along the probabilities of various BT values to the parser. The parser may be able to overcome misclassifications such as an incorrectly posited utterance start or missing utterance start. When all new utterance analyses have lower probabilities than utterance analyses going past a posited new utterance start, the parser can rule the utterance start a false alarm. Similarly, if utterance analyses cannot be found, the parser can search BT values for possible utterance beginnings that were not chosen.

7.3 Future Work in Finding Speech Repairs

The state of the art in speech repair identification needs to be improved to reach human levels of performance. Although parser intervention was able to increase the recall of a state-of-the-art speech repair identifier by 2.7%, the resulting recall, 65.76% is not sufficient to achieve

[2] We know $P(BT = 1 \text{ or } 2|SpkrChange = T) = 0$ and $P(BT = 3 \text{ or } 4|SpkrChange = F) = 0$.

[3] They integrate boundary tone and speech repair identification given that boundary tones often correspond to utterance boundaries.

acceptable performance. In addition, precision decreased from 78.07% to 37.99% due to parser intervention. One reason for these performance problems was the low grammar coverage, 39.7% (on non-trivial utterances as measured in section 3.4). Many times the parser selected repair hypotheses that removed fluent but unparsable words from the input. However, even in an experiment with perfect grammar coverage (presented in section 5.3.3) recall only increased by 8.89%. However this time precision only dropped from 87.38% to 80.99%. The problem there was that the parser favors repair hypotheses allowing the most words to parse. So if part (or all) of a reparandum can be included in the main utterance, the parser favors such an alternative.

First the grammar coverage problem must be addressed to eliminate the large number of false alarms and increase repair recall. In addition to adding new rules to the grammar, a robust parser could be used to help the problem. The robust parser could try to construct a syntactic analysis for the input with no repairs and with repairs posited. The confidence scores of the two analyses could be compared to choose between them. By confidence score, we mean probability of the syntactic analysis adjusted by penalties for any robust parsing techniques used (skipping words, relaxing constraints, etc.).

The speech repair identifier used here [Heeman and Allen, 1997] assigns probabilities to repair hypotheses. Considering confidence scores and speech repair probabilities, we may be able to prevent material from the reparandum from being incorporated in the main utterance. One example of a repair missed by the dialog parser is utt23 from TRAINS-93 dialog d93-10.3, *and then go to hm go to Corning*. Since the whole input (excluding *hm*[4]) can be parsed as one utterance, the parser rejects the hypothesis that the first *go to* is a reparandum. However, the parallelism between the two instances of *go to* should lead to *go to* having a high probability of being a reparandum. Depending on how confidence scores are calculated, including the reparandum in the main utterance may result in a low score since the complement of *go* is not often (if ever) an infinitive phrase headed by *go*.

The speech repair identifier [Heeman and Allen, 1997] used in these experiments employs word and part-of-speech parallelism. It may be the case that phrase-level parallelism is also needed to achieve high recall and precision in repair identification. In addition to searching for complete phrases of the same type, the parser could search for an incomplete phrase followed by a complete phrase of the same type. Heeman and Allen's pre-parser speech repair identifier [Heeman and Allen, 1997] uses silence, boundary tones, and word fragments as evidence. Automatic word fragment recognition needs to be developed to avoid dependence on human annotations. The effect of adding more prosodic information such as that used in [Nakatani and Hirschberg, 1994] also needs to be explored.

In some cases, parser intervention will increase precision since the repair metarule does not require the parser to skip posited reparanda in analyzing the main utterance. Thus, the parser will always explore the possibility of including reparanda in the main parse. If a posited reparandum is included in a high probability main parse, then the reparandum can be ruled a false alarm. Currently we apply the editing term metarule to all potential editing terms, allowing the parser to skip these words in analyzing the main utterance or to include them in the main utterance. Sometimes both alternatives lead to a syntactic analysis. We need to perform disambiguation in these cases and evaluate our results.

7.4 Future Work in Speech Act Annotation

Most of the future work regarding the BF scheme is detailed in section 6.5. The initial 8 dimensional BF scheme was designed so that any possible combinations of communicative

[4] The editing term metarule allows the parser to separately construct an utterance from *um*.

functions could be encoded. Results on labeling TRAINS-93 data suggest that many of the combinations allowed do not occur. Furthermore inter-annotator reliability scores for some dimensions are lower than suggested cutoffs [Carletta, 1996]. Future work involves using a reduced dimension BF scheme and seeing whether this improves inter-annotator reliability and whether it properly captures utterance multi-functionality. One problem for both the original and new BF scheme is lack of testing on multi-party dialogs. Modifications may be needed such as allowing requests and commitments to be directed at particular speakers.

A separate issue is sharing of speech-act annotated data. A speech act taxonomy designed to fully capture an utterance's multi-functionality may not perform well as an intermediary representation for translating between current speech act taxonomies. A separate representation may be needed, or perhaps translation will not be necessary if research groups can establish a standard speech act taxonomy.

7.5 Conclusion

Section 1.6 discusses the contributions of this thesis in detail. To summarize, we have made contributions in the four major dialog-specific challenges in natural language processing: (1) determining an utterance's speech act, (2) allowing for the possibility that speakers may continue each other's utterances and interrupt each other, (3) handling speech repairs and editing terms, and (4) finding utterance boundaries

To address challenge one, we developed the BF scheme with the Multiparty Discourse Group. The BF scheme can assign multiple speech acts to each utterance to capture utterance multi-functionality. With such a speech act taxonomy, we can label corpora to see how people mark their speech acts through lexical choice, prosody, and syntax. We have made some preliminary discoveries about how people communicate in the TRAINS-93 dialogs as discussed in sections 6.3 and 6.4. In labeling TRAINS-93 dialogs, we discovered that labeling utterance multi-functionality is critical for the following reasons. Simply coding the syntactic form of an utterance is important because in addition to their primary speech act all declaratives seem to have an ASSERT function, all imperatives seem to have an ACTION-DIRECTIVE function, and all interrogatives seem to have an INFO-REQUEST function. It is also important to encode whether any actions discussed are joint actions since requests and suggestions of joint actions involve commitment on the part of the speaker in addition to introducing obligations on the listener.

We have developed a parsing framework that allows for the possibility that speakers may continue each other's utterances (challenge 2), and accommodates speech repairs and editing terms (challenge 3) assuming they can be correctly identified. Our parser is the first to allow second speaker interruptions and continuations without having interruptions necessarily disrupt ongoing utterances by the first speaker. Our approach is more sophisticated than simply treating the two speakers separately, and represents exactly where interruptions were made and how many interruptions occurred.

[Cori *et al.*, 1997] and [McKelvie, 1998] discuss the idea of representing speech repairs and editing terms through phrase structure but make simplifying assumptions that are not always correct about the structure of speech repairs. Furthermore, they do not test their formalisms through application to a corpus separately annotated with speech repairs. Our formalism (described in chapter 3) is designed to allow any constituent (including a possibly broken constituent) to be restarted at any point by a following correction. Although many times the reparandum forms a constituent, there are cases where it does not, suggesting that our formalism is correct in not making any stricter claims. In a corpus of 31 TRAINS-93 dialogs[5], comprising

[5] Specifically the dialogs were d92-1 through d92a-5.2 and d93-10.1 through d93-14.1.

3441 utterances, 19,189 words, and 495 speech repairs, our formalism accommodated all speech repairs. The formalism allows our parser to construct a representation containing what the user started to say before a repair, types and positions of editing terms, repair frequency, and repair positions. This is important information; previous psycholinguistic studies as reviewed in section 1.3 show that editing terms convey meaning and that reparanda can be used as a source of information by listeners.

Although [McKelvie, 1998] makes simplifying assumptions that are not always correct about the structure of speech repairs, this approach may seem attractive because it only requires the addition of grammar rules and does not require changes to the parser. However, other work, [Lavie, 1995] has shown that even fluent dialog requires a mechanism for forming phrase structure from non-adjacent constituents (when the parser's grammar does not cover the input). Allowing a parser to skip words and parse them separately is a minor change especially considering that McKelvie more than doubles the number of grammar rules having more than one element on their right hand side. Another advantage to changing the parser is that the output of such an approach is a more accurate representation of the input. A correction and reparandum are never included in the same syntactic tree and editing terms are separate from their surrounding utterances. And given that McKelvie cannot accommodate speech repairs where the correction is more than one constituent, it seems clear that parsers must allow non-adjacent constituents to combine. Such an approach allows us to handle examples such as utterance 11 from TRAINS-93 dialog d93-8.2: *how does the long does that take* and utterance 81 from TRAINS-93 dialog d93-10.4: *the two boxcars of orange juice should er of oranges should be made into orange juice* (these examples are taken from [Heeman, 1997]).

Regarding finding speech repairs, our work is the first to demonstrate that a parser's grammatical knowledge can improve the recall of a state-of-the-art speech repair identifier [Heeman and Allen, 1997] as shown in chapter 5. The grammar's coverage of the input (39.7%, as measured on non-trivial utterances in section 3.4) turns out to be crucial. Increases in speech repair recall due to parser intervention vary from 2.7% to 4.8% on different corpora. On a subset of the data where the grammar has 100% coverage, speech repair recall increases to 8.89% due to parser intervention.

We chose to concentrate on increasing recall because the metarules give the parser the option of ignoring a posited repair and including its reparandum in the main utterance. However, our current efforts to increase recall have led to drops in precision due to poor grammar coverage and the parser including all or part of a reparandum in the main utterance. Future work will explore whether a parser can increase both speech repair identification recall and precision. If the parser can only increase one or the other, performance results need to be examined to determine the impact on dialog system performance attributable to increased recall versus increased precision.

Both our project (see chapter 4 for details) and [Stolcke and Shriberg, 1996a; Stolcke *et al.*, 1998] address the problem of utterance boundary detection (challenge 4) when second speaker interruptions and continuations are possible. Although both projects ignore embedded utterance end points, our results suggest that detection of utterance end points (including embedded ones) is feasible. We took the additional step of comparing the predictive power of the difference pieces of evidence in our model. The evidence tested (syntactic information, boundary tones, silence, discourse markers, editing terms, speaker changes, and word fragments) turned out to be partially redundant. Boundary tones are the most predictive evidence but even without them recall only decreased by 4.06%. Discourse markers are the second most predictive evidence term. Syntactic evidence (the "best" constituent ending at the current word) cuts the false alarm rate to a quarter of its original value. We also ran experiments performing speech repair detection before utterance boundary detection. The speech repair information reduced the utterance boundary detector's false alarm rate. Of course, utterance boundaries constrain where speech repairs (intra-utterance corrections) can occur. The two processes are intertwined and should

be performed together.

Thus, we have made progress in all four dialog-specific areas of natural language processing. We have built an utterance boundary detector that works on data with second speaker interruptions and continuations. We have developed a parsing framework that accommodates these second speaker interruptions and continuations as well as speech repairs and editing terms. Unlike previous approaches [Cori *et al.*, 1997; McKelvie, 1998] we do not make simplifying assumptions about the form of speech repairs. We have also been able to increase speech repair identification recall by including the parser in the process. In the speech act recognition area, we have worked with the Multiparty Discourse Group to develop a speech act taxonomy that captures utterance multi-functionality, and we have started to label data to determine how speech acts are marked by lexical choice, prosody, and syntax.

Bibliography

[Allen and Core, 1997] J. Allen and M. Core, "Draft of DAMSL: Dialog Annotation Markup in Several Layers". available at ftp://ftp.cs.rochester.edu/pub/packages/dialog-annotation/manual.ps.gz, March 1997.

[Allwood, 1995] J. Allwood, "An Activity Based Approach to Pragmatics," Gothenburg Papers in Linguistics 75, Dept of Linguistics, Univ. of Goteborg, 1995.

[Batliner *et al.*, 1996] A. Batliner, R. Kompe, A. Kiessling, H. Neimann, and E. Noeth, "Syntactic-Prosodic Labeling of Large Spontaneous Speech Data-bases," in *Proc. of the Fourth International Conference on Spoken Language Processing (ICSLP-96)*, 1996.

[Bear *et al.*, 1992] J. Bear, J. Dowding, and E. Shriberg, "Integrating Multiple Knowledge Sources for Detection and Correction of Repairs in Human-Computer Dialog," in *Proc. of the 30th Annual Meeting of the Association for Computational Linguistics (ACL-92)*, pages 56–63, 1992.

[Bortfeld *et al.*, 1999] H. Bortfeld, S. D. Leon, J. E. Bloom, M. F. Schober, and S. E. Brennan, "Which Speakers Are Most Disfluent in Conversation, and When?," in *Notes of the ICPhS Satellite Meeting on Disfluency in Spontaneous Speech*, pages 7–10, July 1999.

[Branigan *et al.*, 1999] H. Branigan, R. Lickley, and D. McKelvie, "Non-Linguistic Influences on Rates of Disfluency in Spontaneous Speech," in *Proc. of the 14th International Congress of Phonetic Sciences (ICPhS-99)*, San Francisco, August 1999.

[Breiman *et al.*, 1984] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth and Brooks/Cole, Monterey, California, 1984.

[Brennan and Schober, 1999] S. Brennan and M. Schober, "Speech Disfluencies in Spoken Language Systems: A Dialog-Centered Approach," in *Proc. of the NSF Human Computer Interaction Grantees' Workshop (HCIGW-99)*, Orlando, FL, 1999.

[Brennan and Kipp, 1996] S. E. Brennan and E. G. Kipp, "An Addressee's Knowledge Affects a Speaker's Use of Fillers in Question-Answering," in *Abstracts of the Psychonomic Society, 37th Annual Meeting*, page 24, Chicago, IL, 1996.

[Brennan and Williams, 1995] S. E. Brennan and M. Williams, "The Feeling of Another's Knowing: Prosody and Filled Pauses as Cues to Listeners about the Metacognitive States of Speakers," *Journal of Memory and Language*, 34:383–398, 1995.

[Carletta, 1996] J. Carletta, "Assessing Agreement on Classification Tasks: the Kappa Statistic," *Computational Linguistics*, 22(2), 1996.

[Caspers, 1998] J. Caspers, "Who's Next? The Melodic Marking of Question vs. Continuation in Dutch," *Language and Speech*, 41(3-4):375–398, 1998.

[Chu-Carroll, 1997] J. Chu-Carroll, "A Statistical Model for Discourse Act Recognition in Dialogue Interactions," Technical Report SS-98-01, American Association for Artificial Intelligence, 445 Burgess Drive, Menlo Park CA 94025, 1997. Papers from Symposium on Applying Machine Learning to Discourse Processing.

[Clark, 1996] H. H. Clark, *Using Language*, Cambridge University Press, New York, 1996.

[Clark and Schaefer, 1989] H. H. Clark and E. F. Schaefer, "Contributing to Discourse," *Cognitive Science*, 13:259–294, 1989.

[Cohen and Levesque, 1990] Philip R. Cohen and Hector J. Levesque, "Rational Interaction as the Basis for Communication," in Philip R. Cohen, Jerry Morgan, and Martha E. Pollack, editors, *Intentions in Communication*, SDF Benchmark Series, pages 221–255. MIT Press, 1990.

[Core *et al.*, 1998] M. Core, M. Ishizaki, J. Moore, C. Nakatani, N. Reithinger, D. Traum, and S. Tutiya, "The Report of the Third Workshop of the Discourse Resource Initiative," Chiba Corpus Project Technical Report 3 (CC-TR-99-1), Chiba University and Kazusa Academic Hall, May 1998. contact: corpus-project@CogSci.L.chiba-u.ac.jp.

[Core and Schubert, 1999a] M. Core and L. Schubert, "Speech Repairs: A Parsing Perspective," in *Notes of the ICPhS Satellite Meeting on Disfluency in Spontaneous Speech*, pages 47–50, 1999.

[Core and Schubert, 1999b] M. Core and L. Schubert, "A Syntactic Framework for Speech Repairs and Other Disruptions," in *Proc. of the 37th Annual Meeting of the Association for Computational Linguistics (ACL-99)*, College Park, MD, June 1999.

[Core, 1997] M. G. Core, "Analyzing and Predicting Patterns of DAMSL Utterance Tags," Technical Report SS-98-01, American Association for Artificial Intelligence, 445 Burgess Drive, Menlo Park CA 94025, 1997. Papers from Symposium on Applying Machine Learning to Discourse Processing.

[Core and Allen, 1997] M. G. Core and J. F. Allen, "Annotating Dialogs with the DAMSL Annotation Scheme". Presented at the AAAI Fall Symposium on Communicative Action in Humans and Machines. Available at http://www.cs.rochester.edu/u/mcore, 1997.

[Core and Schubert, 1996] M. G. Core and L. K. Schubert, "Dialog Parsing in the TRAINS System," Technical Report 612, Department of Computer Science, University of Rochester, Rochester, NY 14627-0226, 1996.

[Core and Schubert, 1997] M. G. Core and L. K. Schubert, "Handling Speech Repairs and Other Disruptions Through Parser Metarules," Technical Report SS-97-04, American Association for Artificial Intelligence, 445 Burgess Drive, Menlo Park CA 94025, 1997. Papers from Symposium on Computational Models for Mixed Initiative Interactions.

[Cori *et al.*, 1997] M. Cori, M. de Fornel, and J.-M. Marandin, "Parsing Repairs," in R. Mitkov and N. Nicolov, editors, *Recent Advances in Natural Language Processing*, volume 136 of *Current Issues in Linguistic Theory*. John Benjamins Publishing Company, Philadelphia, PA, 1997.

[Dehaspe and Raedt, 1997] L. Dehaspe and L. D. Raedt, "Mining a Natural Language Corpus for Multi-Relational Association Rules," in W. Daelemans, A. van den Bosch, and A. Weijters, editors, *Workshop Notes of the ECML/MLnet Workshop on Empirical Learning of Natural Language Processing Tasks*, pages 35–48. Prague, Czech Republic, 1997.

[Dowding *et al.*, 1993] J. Dowding, J. M. Gawron, D. Appelt, J. Bear, L. Cherny, R. Moore, and D. Moran, "Gemini: A Natural Language System for Spoken-Language Understanding," in *Proc. of the 31st Annual Meeting of the Association for Computational Linguistics (ACL-93)*, pages 54–61, Columbus, Ohio, 1993.

[Ferguson and Allen, 1998] G. Ferguson and J. F. Allen, "TRIPS: An Intelligent Integrated Problem-Solving Assistant," in *Proc. of the National Conference on Artificial Intelligence (AAAI-98)*, pages 26–30, Madison, WI, July 1998.

[Finke *et al.*, 1997] M. Finke, M. Lapata, A. Lavie, L. Levin, L. Mayfield-Tomokiyo, T. Polzin, K. Ries, A. Waibel, and K. Zechner, "Clarity: Inferring Discourse Structure from Speech," Technical Report SS-98-01, American Association for Artificial Intelligence, 445 Burgess Drive, Menlo Park CA 94025, 1997. Papers from Symposium on Applying Machine Learning to Discourse Processing.

[Fox Tree, 1995] J. E. Fox Tree, "The Effects of False Starts and Repetitions on the Processing of Subsequent Words in Spontaneous Speech," *Journal of Memory and Language*, 34:709–738, 1995.

[Fox Tree, 1999] J. E. Fox Tree, "Between-Turn Pauses and Ums," in *Notes of the ICPhS Satellite Meeting on Disfluency in Spontaneous Speech*, pages 15–17, July 1999.

[Frazier and Fodor, 1978] L. Frazier and J. D. Fodor, "The Sausage Machine: A New Two-Stage Parsing Model," *Cognition*, 6:291–325, 1978.

[Gibbon and Tseng, 1999] D. Gibbon and S.-C. Tseng, "Toward a Formal Characterisation of Disfluency Processing," in *Notes of the ICPhS Satellite Meeting on Disfluency in Spontaneous Speech*, pages 35–38, July 1999.

[Goodwin, 1991] C. Goodwin, *Conversational Organization: Interaction between Speakers and Hearers*, Academic Press, New York, 1991.

[Görz *et al.*, 1996] G. Görz, M. Kesseler, J. Spilker, and H. Weber, "Research on Architectures for Integrated Speech/Language Systems in Verbmobil," in *Proceedings of the 17th International Conference on Computational Linguistics (COLING-96)*, 1996.

[Hancher, 1979] M. Hancher, "The Classification of Cooperative Illocutionary Acts," *Language in Society*, 8(1):1–14, 1979.

[Heeman and Allen, 1995] P. Heeman and J. Allen, "The TRAINS 93 Dialogues," TRAINS Technical Note 94-2, Department of Computer Science, University of Rochester, Rochester, NY 14627-0226, 1995.

[Heeman, 1997] P. A. Heeman, *Speech Repairs, Intonational Boundaries and Discourse Markers: Modeling Speakers' Utterances in Spoken Dialog*, PhD thesis, Department of Computer Science, University of Rochester, 1997.

[Heeman and Allen, 1997] Peter A. Heeman and James F. Allen, "Intonational Boundaries, Speech Repairs, and Discourse Markers: Modeling Spoken Dialog," in *Proc. of the 35th Annual Meeting of the Association for Computational Linguistics (ACL-97)*, pages 254–261, Madrid, July 1997.

[Hindle, 1983] D. Hindle, "Deterministic Parsing of Syntactic Non-Fluencies," in *Proc. of the 21st annual meeting of the Association for Computational Linguistics (ACL-83)*, pages 123–128, 1983.

[Jurafsky *et al.*, 1997] D. Jurafsky, R. Bates, N. Coccaro, R. Martin, M. Meteer, K. Ries, E. Shriberg, A. Stolcke, P. Taylor, and C. Van Ess-Dykema, "Automatic Detection of Discourse Structure for Speech Recognition and Understanding," in *Proc. of the 1997 IEEE Workshop on Speech Recognition and Understanding*, Santa Barbara, 1997. Available at http://www.Colorado.EDU/linguistics/jurafsky/.

[Kelley and Pohl, 1990] A. Kelley and I. Pohl, *A Book on C, Programming in C*, Benjamin/Cummings, New York, second edition, 1990.

[Kikui and Morimoto, 1994] Gen-ichiro Kikui and Tsuyoshi Morimoto, "Similarity-Based Identification of Repairs in Japanese Spoken Language," in *Proceedings of the 3rd International Conference on Spoken Language Processing (ICSLP-94)*, pages 915–918, 1994.

[Koiso *et al.*, 1998] H. Koiso, Y. Horiuchi, S. Tutiya, A. Ichikawa, and Y. Den, "An Analysis of Turn-Taking and Backchannels Based on Prosodic and Syntactic Features in Japanese Map Task Dialogs," *Language and Speech*, 41(3-4):295–321, 1998.

[Kompe *et al.*, 1997] R. Kompe, A. Kiessling, H. Niemann, E. Noeth, A. Batliner, S. Schachtl, T. Ruland, and H. U. Block, "Improving Parsing of Spontaneous Speech with the Help of Prosodic Boundaries," Verbmobil Report 210, German Research Center for Artificial Intelligence (DFKI) GmbH, Erwin-Schrdinger-Strae, D-67608 Kaiserslautern, 1997.

[Lau *et al.*, 1997] R. Lau, G. Flammia, C. Pao, and V. Zue, "Webgalaxy - Integrating Spoken Language and Hypertext Navigation," in *Proceedings of the 5th European Conference on Speech Communication and Technology (Eurospeech-97)*, pages 883–886, Rhodes, Greece, September 1997.

[Lavie, 1995] A. Lavie, *GLR*: A Robust Grammar Focused Parser for Spontaneously Spoken Language*, PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1995.

[Lavie *et al.*, 1996] A. Lavie, D. Gates, N. Coccaro, and L. Levin, "Input Segmentation of Spontaneous Speech in JANUS: a Speech-to-Speech Translation System," in *Proc. of ECAI Workshop on Dialog Processing in Spoken Language Systems*, Budapest, August 1996.

[Levelt, 1983] W. J. M. Levelt, "Monitoring and Self-Repair in Speech," *Cognitive Science*, 14:41–104, 1983.

[Lickley and Bard, 1996] R. J. Lickley and E. G. Bard, "On Not Recognizing Disfluencies in Dialogue," in *Proceedings of the 4rd International Conference on Spoken Language Processing (ICSLP-96)*, 1996.

[Lickley and Bard, 1998] R. J. Lickley and E. G. Bard, "When Can Listeners Detect Disfluency in Spontaneous Speech?," *Language and Speech*, 41(2):203–226, 1998.

[Lickley *et al.*, 1991] R. J. Lickley, R. C. Shillcock, and E.G. Bard, "Processing Disfluent Speech: How and When Are Disfluencies Found?," in *Proceedings of the 2nd European Conference on Speech Communication and Technology (Eurospeech-91)*, 1991.

[Mast *et al.*, 1996] M. Mast, H. Niemann, E. Nöth, and E. G. Schukat-Talamazzini, "Automatic Classification of Dialog Acts with Semantic Classification Trees and Polygrams," in S. Wermter, E. Riloff, and G. Scheler, editors, *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*, Lecture Notes in Artificial Intelligence, pages 217–229. Springer, New York, 1996.

[McKelvie, 1998] D. McKelvie, "The Syntax of Disfluency in Spontaneous Spoken Languages," Technical Report HCRC/RP-95, Human Communication Research Centre, University of Edinburgh, 2 Buccleuch Place, Edinburgh EH8 9LW, UK, 1998.

[Nagata and Morimoto, 1994] M. Nagata and T. Morimoto, "First Steps Towards Statistical Modeling of Dialogue to Predict the Speech Act Type of the Next Utterance," *Speech Communication*, 15:193–203, 1994.

[Nakatani and Hirschberg, 1993] C. Nakatani and J. Hirschberg, "A Speech-First Model for Repair Detection and Correction," in *Proc. of the 31st Annual Meeting of the Association for Computational Linguistics (ACL-93)*, pages 46–53, 1993.

[Nakatani and Hirschberg, 1994] C. H. Nakatani and J. Hirschberg, "A Corpus-Based Study of Repair Cues in Spontaneous Speech," *Journal of the Acoustical Society of America*, 95(3):1603–1616, March 1994.

[O'Shaughnessy, 1994] D. O'Shaughnessy, "Analysis and Automatic Recognition of False Starts in Spontaneous Speech," in *Proceedings of the International Conference on Audio, Speech and Signal Processing (ICASSP-94)*, 1994.

[Oviatt, 1994] S. Oviatt, "Predicting and Managing Spoken Disfluencies during Human-Computer Interaction," in *Proceedings of the ARPA Human Language Technology Workshop*, pages 222–227, 1994.

[Reithinger and Maier, 1995] N. Reithinger and E. Maier, "Utilizing Statistical Dialogue Act Processing in Verbmobil," in *Proc. of the 33rd annual meeting of the Association for Computational Linguistics (ACL-95)*, 1995.

[Rosè, 1997] C. P. Rosè, *Robust Interactive Dialogue Interpretation*, PhD thesis, School of Computer Science, Carnegie Mellon University, 1997.

[Ruland *et al.*, 1998] T. Ruland, C. J. Rupp, J. Spilker, H. Weber, and K. L. Worm, "Making the Most of Multiplicity: A Multi-Parser Multi-Strategy Architecture for the Robust Processing of Spoken Language," Verbmobil Report 230, German Research Center for Artificial Intelligence (DFKI) GmbH, Erwin-Schrdinger-Strae, D-67608 Kaiserslautern, 1998.

[Samuel *et al.*, 1997] K. Samuel, S. Carberry, and K. Vijay-Shanker, "Computing Dialogue Acts from Features with Transformation-Based Learning," Technical Report SS-98-01, American Association for Artificial Intelligence, 445 Burgess Drive, Menlo Park CA 94025, 1997. Papers from Symposium on Applying Machine Learning to Discourse Processing.

[Schubert, 1986] L. K. Schubert, "Are There Preference Trade-offs in Attachment Decisions," in *Proc. of the National Conference on Artificial Intelligence (AAAI-86)*, pages 601–605, Philadelphia, PA, 1986.

[Searle, 1975a] J. R. Searle, "Indirect Speech Acts," in P. Cole and J. L. Morgan, editors, *Syntax and Semantics*, volume 1, pages 59–82. Academic Press, New York, 1975.

[Searle, 1975b] J. R. Searle, "A Taxonomy of Illocutionary Acts," in K. Gunderson, editor, *Language, Mind, and Knowledge. Minnesota Studies in the Philosophy of Science*, pages 344–369. University of Minnesota Press, Minneapolis, Minnesota, 1975.

[Seneff, 1992] S. Seneff, "TINA: A Natural Language System for Spoken Language Applications," *Computational Linguistics*, 18(1):61–86, 1992.

[Shriberg, 1996] E. Shriberg, "Disfluencies in Switchboard," in *Proceedings of the 4rd International Conference on Spoken Language Processing (ICSLP-96)*, Philadelphia, PA, October 1996.

[Shriberg *et al.*, 1997] E. Shriberg, R. Bates, and A. Stolcke, "A Prosody-Only Decision-Tree Model for Disfluency Detection," in G. Kokkinakis, N. Fakotakis, and E. Dermatas, editors, *Proceedings of the 5^{th} European Conference on Speech Communication and Technology (Eurospeech-97)*, volume 5, pages 2383–2386, Rhodes, Greece, 1997.

[Siegel and Castellan, 1988] S. Siegel and N. J. Castellan, *Nonparametric Statistics for the Behavioral Sciences*, McGraw-Hill, second edition, 1988.

[Siu and Ostendorf, 1996] M.-h. Siu and M. Ostendorf, "Modeling Disfluencies in Conversational Speech," in *Proceedings of the 4rd International Conference on Spoken Language Processing (ICSLP-96)*, pages 386–389, 1996.

[Smith and Clark, 1993] V. L. Smith and H. H. Clark, "On the Course of Answering Questions," *Journal of Memory and Language*, 32:25–38, 1993.

[Stent *et al.*, 1999] A. Stent, J. Dowding, J. M. Gawron, E. O. Bratt, and R. Moore, "The CommandTalk Spoken Dialogue System," in *Proc. of the 37^{th} Annual Meeting of the Association for Computational Linguistics (ACL-99)*, pages 183–190, College Park, MD, June 1999.

[Stolcke and Shriberg, 1996a] A. Stolcke and E. Shriberg, "Automatic Linguistic Segmentation of Conversational Speech," in *Proceedings of the 4rd International Conference on Spoken Language Processing (ICSLP-96)*, volume 2, pages 1005–1008, Philadelphia, 1996.

[Stolcke and Shriberg, 1996b] A. Stolcke and E. Shriberg, "Statistical Language Modeling for Speech Disfluencies," in *Proceedings of the International Conference on Audio, Speech and Signal Processing (ICASSP-96)*, May 1996.

[Stolcke *et al.*, 1998] A. Stolcke, E. Shriberg, R. Bates, M. Ostendorf, D. Hakkani, M. Plauche, G. Tür, and Y. Lu, "Automatic Detection of Sentence Boundaries and Disfluencies Based on Recognized Words," in *Proceedings of the 6th International Conference on Spoken Language Processing (ICSLP-98)*, Sydney, Australia, 1998.

[Taylor *et al.*, 1998] P. Taylor, S. King, S. Isard, and H. Wright, "Intonation and Dialog Context as Constraints for Speech Recognition," *Language and Speech*, 41(3-4):493–512, 1998.

[Traum and Heeman, 1996] D. Traum and P. Heeman, "Utterance Units in Spoken Dialogue," in *ECAI Workshop on Dialogue Processing in Spoken Language Systems*, pages 84–91, Budapest, August 1996.

[Wells and Macfarlane, 1998] B. Wells and S. Macfarlane, "Prosody as an Interactional Resource: Turn-projection and Overlap," *Language and Speech*, 41(3-4):265–294, 1998.