



Modeling Task-Oriented Dialogue

MAITE TABOADA

Department of Linguistics, Simon Fraser University, Burnaby, B.C. V5A 1S6 Canada
E-mail: mtaboada@sfu.ca

Abstract. A common tool for improving the performance quality of natural language processing systems is the use of contextual information for disambiguation. Here I describe the use of a finite state machine (FSM) to disambiguate speech acts in a machine translation system. The FSM has two layers that model, respectively, the global and local structures found in naturally-occurring conversations. The FSM has been modeled on a corpus of task-oriented dialogues in a travel planning situation. In the dialogues, one of the interactants is a travel agent or hotel clerk, and the other a client requesting information or services. A discourse processor based on the FSM was implemented in order to process contextual information in a machine translation system. Evaluation results show that the discourse processor is able to disambiguate and improve the quality of the dialogue translation. Other applications include human-computer interaction and computer-assisted language learning.

Key words: discourse processing, finite state methods, machine translation, speech act assignment

1. Context and Ambiguity in Dialogue

For any given utterance out of what we can loosely call context, there is usually more than one possible interpretation. A speaker's utterance of an elliptical expression, such as the figure 'twelve fifteen', might have a different meaning depending on the discourse context, the way the conversation has evolved until that point, and the previous speaker's utterance. 'Twelve fifteen' could be the time 'a quarter after twelve', the price 'one thousand two hundred and fifteen', the room number 'one two one five', and so on. Although English can convey all those meanings with a single expression, the translation into other languages might require different expressions for the different meanings. Only with appropriate contextual information can we produce an accurate translation.

If this is a problem for human listeners, the problem grows considerably when a parser is doing the disambiguation. One of the difficulties in performing machine translation with the output of a speech recognition system is the presence of disfluencies and recognizer errors. A grammar designed to accept only perfectly formed sentences will fail on this type of input. The Phoenix parser (Ward, 1991, 1994) was designed to capture the content of spoken dialogue through *semantic grammars*. Semantic grammars assign different parts of an input string to a number of concepts, then combine those concepts to form a complete thought or sequence, and usually assign a speech act to the sequence. If we are able to mechanically

detect illocutionary force in the input language, we can produce the same illocutionary force in the output language, irrespective of the actual surface forms the two languages might use. In this sense, we depart from syntax-based, literal translation to provide a rough translation of content. However, the speech act, and the illocutionary force it carries, can only be fully interpreted when we make use of contextual information. In this paper I describe a discourse processor that uses contextual information to help a parser disambiguate among different possible parses for an input sentence. The final goal is to improve the translation in an end-to-end speech translation system.

2. Modeling Dialogue

Dialogue modeling for computational applications has been tackled both from descriptive and predictive points of view. Both linguists and computational linguists have converged towards the study of dialogue through what the former call *genres* and the latter call *domains*. Linguists have been interested in the study of texts “as staged goal-oriented social processes which integrate field [the social action, ‘what is actually taking place’], mode [the role structure, ‘who is taking part’] and tenor [the symbolic organization, ‘what role language is playing’] choices in predictable ways” (Halliday and Martin, 1993, p. 36). The use of the term genre was borrowed from the study of literary texts, and it was first applied to everyday texts by Bakhtin (1986). Bakhtin coined the term *speech genres* to refer to “relatively stable thematic, compositional and stylistic types of utterances” that correspond to the specific styles for certain spheres of human activity and communication (Bakhtin, 1986, p. 64). Linguistic applications of this idea of genre to the modeling of conversation include: flowcharts (Ventola, 1987), systemic flowcharts (Fawcett *et al.*, 1988) and dynamic networks (O’Donnell, 1990).

Computational linguists have been interested in the modeling of dialogue for obvious reasons: predictions about the structure of a dialogue simplify and improve the work of parsers and generators significantly. Within machine translation and in conversation specifically, the use of the contextual information gained from a formal representation of the structure of the conversation would improve the accuracy of the translation in many context-sensitive cases.

Let us just look at one example: the Spanish ‘*está bien*’ can mean ‘he/she is fine’, ‘are you okay?’, ‘is that alright?’, and ‘that is fine’, depending on the context. In examples taken from a Spanish corpus of scheduling dialogues (Ahlen, 1997) we find instances of some of those different meanings. Speaker A in Example (1) below proposes a date and asks for confirmation. Speaker B replies with an acceptance.¹ In both cases they use ‘*está bien*’. If the parser, as in the case in the JANUS system used here, cannot process intonation markers, then the only clue is in the context. Without context we cannot parse neither translate the different meanings.

- (1) A: El veintiséis no hay problema. Nos podemos reunir de once a una de la tarde. ¿Está bien?
B: Sí, está bien. Perfecto. Hasta luego.
(A: On the twenty-sixth there is no problem. We can meet from eleven to one in the afternoon. Is that alright?
B: Yes, that's fine. Perfect. See you later)

For comparison, I will discuss here two examples of computational attempts to model the structure of conversation: the GLR* discourse processor and the Verbmobil system.

The first GLR* discourse processor (Rosé *et al.*, 1995; Qu *et al.*, 1996a, b) was developed in order to reduce ambiguity and improve translation accuracy. GLR* is one of two parsers within the JANUS system (see Section 3). The discourse processor deals with task-oriented dialogues as well, this time within the scheduling domain. The discourse module is based on Lambert's tripartite model (Lambert and Carberry, 1992; Lambert, 1993). It disambiguates the speech act of each sentence and incorporates the sentence into a discourse plan tree; at the same time it updates a calendar to keep track of the dates being discussed by the speakers. The final disambiguation combines a series of scores from different sources: speech recognizer, parser, and discourse processor. The two main problems with the plan-based system are (1) its resource-intensive and time-consuming running, and (2) its sensitivity to cumulative error (produced when an incorrect hypothesis is chosen and incorporated in the model that serves as a basis for subsequent predictions).

Verbmobil is another recent example of discourse processing for a limited domain. In Verbmobil – which also deals with scheduling and travel planning dialogues – the system acts as a backup for a conversation carried in English by two non-native speakers of the language (Reithinger *et al.*, 1995). The system was in development for a number of years, and the first incarnation of the discourse processor used a plan processor with an embedded finite state machine (Reithinger and Maier, 1995). The most recent version also makes use of a fixed set of speech acts, in this case divided into dialogue and task acts (Alexandersson *et al.*, 2000a). The distinction is similar to Litman and Allen's (1990) between domain plans, used to model tasks, and discourse plans, which are domain-independent and relate to the mechanics of the dialogue.

The dialogue module in Verbmobil uses a dialogue memory, a plan processor and a dialogue processor (Kipp *et al.*, 2000). The dialogue memory keeps track of dates and decisions made during the dialogue, which are then used to produce a dialogue summary (Alexandersson *et al.*, 2000b). The plan processor also uses a layered approach, but instead of two layers, as described in this paper, four layers are used, based on Sinclair and Coulthard's (1975) classification: dialogue, phase, game and move. All four are built bottom-up from basic dialogue acts. The plan processor is used for the recognition of dialogue phases (similar to our subdialogues), and for the recognition of games and moves, useful in the generation of summaries. Finally, the dialogue processor combines dialogue acts and

content representation to produce structures that will be used in the generation of summaries.

The disambiguation is based on both rules and weighted rules (Koch *et al.*, 2000). The rules take into account prosodic, syntactic and semantic features of the utterance, plus the previous dialogue act. The rules take the form of weighted defaults (Schmitz and Quantz, 1995), where a particular piece of information (syntactic, presence of keywords, structural information) establishes a preference of weight for the utterance to be assigned a dialogue act. Dialogue acts can be reinterpreted at any point in the processing, even by doing backtracking and reinterpreting all previous dialogue act assignments. The reinterpretation is not feasible in the system described here, since the final goal is to produce real-time translations, where there is no possibility of revising previous speech act assignments.

The main drawback of the discourse processors in both GLR* and Verbmobil was their lack in accuracy and the need for an extra component that would keep track of the global structure of the discourse. The drawback, in general, of systems that rely on plans is their complexity, which makes them slow and resource-intensive. The system described in the next section differs from other systems in that it merges both global and local structure in a single component which is simple, robust and fast. Speed is a concern in the JANUS system, whose task is to translate on-line. The following section explains the system in which the discourse processor is integrated, and Section 4 looks at the major components and the algorithm for the discourse processor in more detail.

3. System Background

JANUS is a multi-lingual speech-to-speech translation system that translates spontaneous dialogue between two speakers in a limited domain (Waibel, 1996; Lavie *et al.*, 1996b). It is designed to deal with the kind of problems that naturally occur in spontaneous speech – such as mispronunciations, restarts, noises and the lack of clear sentence boundaries – as well as additional errors introduced by the speech recognizer. The machine translation component of JANUS handles these problems using two different approaches: GLR* and Phoenix. The GLR* parser (Lavie and Tomita, 1993; Lavie, 1995) is more accurate, whereas the Phoenix parser (Ward, 1991, 1994) is more robust. Both are language-independent and follow an interlingua-based approach. The system translates spontaneous dialogues in two domains: scheduling domain (two speakers trying to set up an appointment) and travel planning (a client making travel arrangements with a travel agent or hotel clerk). English, Spanish and German are both source and target languages (Lavie *et al.*, 1997). Figure 1 shows an outline of all the system components.

The input string in the source language is first analyzed separately by the parsers, to produce a language-independent content representation. From that representation the generation component in each of the modules generates the output string in the target language. Additionally, the GLR* module contains a

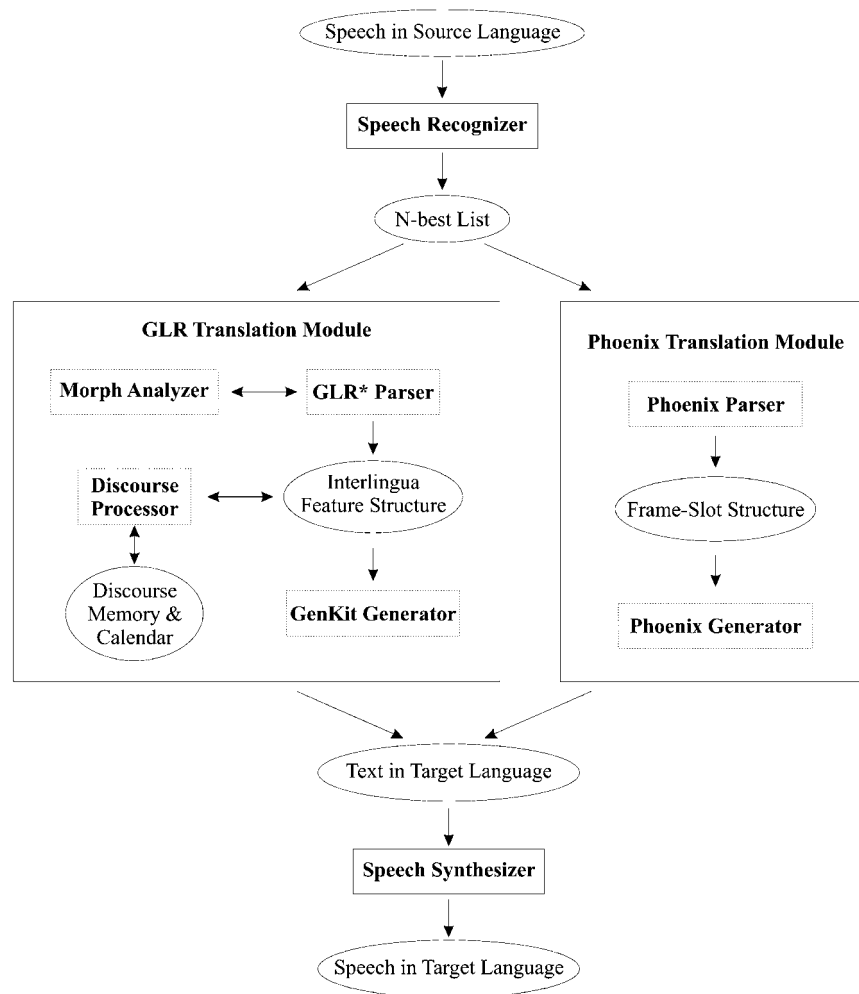


Figure 1. The JANUS system.

discourse processor that disambiguates the speech act of each sentence, normalizes temporal expressions and incorporates the sentence into a discourse plan tree.

This project focuses on the Phoenix module of the machine translation component. The JANUS Phoenix translation module (Mayfield *et al.*, 1995) was designed for semantic grammars. The parsing grammar specifies patterns in order to introduce grammatical constraints at the phrase level rather than at the sentence level. This method captures the semantic content of a complete input string, regardless of the ungrammaticalities often occurring between phrases. The patterns in the grammar help the parser extract a structure of concepts by means of tokens. Top-level tokens represent speech acts, whereas the intermediate and lower-level tokens capture the more specific parts of the utterance – such as days of the week or times in the scheduling domain. The inputs to the parser are fragments of a turn, each

one of them a complete thought, although not necessarily a complete sentence. The segmentation of input is described in more detail in Lavie *et al.* (1996a). Before the introduction of the discourse processor described here, each one of those segments was parsed and generated devoid of contextual information.

4. The Discourse Module

The approach used for incorporating contextual information consists of combining discourse information with the output of the Phoenix parser, which is a set of possible parses for an input string. The new discourse module interacts with the parser, selecting one of these possibilities. The decision is based on the information provided by the previous discourse context together with pragmatic considerations, such as the structure of adjacency pairs (Schegloff and Sacks, 1973; Sacks *et al.*, 1974), and the responses to speech functions (Halliday, 1994; Martin, 1992). The context module keeps a history of the conversation in order to estimate, for instance, the likelihood of a greeting once the opening phase of the conversation is over. A more local history determines the expected second part in any adjacency pair, such as a question-answer sequence. The parser performs *late disambiguation*, collecting as much information as possible before proceeding on to disambiguation, rather than restricting the parser's search early on. The two-layered approach allows us to take into account both global and local context in the same module. This provides robust and efficient processing, avoiding multiple components for dialogue processing, which is the approach in the Verbmobil system.

The discourse module interacts with other modules within the overall system, as diagrammed in Figure 2. The module is able to operate both on output from the speech recognizer and on transcribed data. To develop the module, transcriptions of the dialogues were used, rather than the less reliable output of the speech recognizer. The dialogue domain – a speech genre (Bakhtin, 1986) – chosen for this project was the *travel planning domain*. This domain consists of dialogues where a customer makes travel arrangements with a travel agent or a hotel clerk, in order to book hotel rooms, flights or other forms of transportation. They are *task-oriented dialogues*, because the speakers have specific goals of carrying out a task that involves the exchange of both information and services. Since the travel planning domain is a new domain for the JANUS system, I also needed to write the appropriate grammars and decide on the set of speech acts for this domain. The task is thus divided in four different areas: (1) selection of appropriate speech acts, (2) parsing and generation grammars, (3) coding of the discourse module, and (4) use of a training corpus to obtain probabilities for disambiguation.

4.1. SPEECH ACT TAXONOMY

The selection of speech acts is a very important part of the project, since we are relying on their specificity or generality to produce the appropriate predictions.

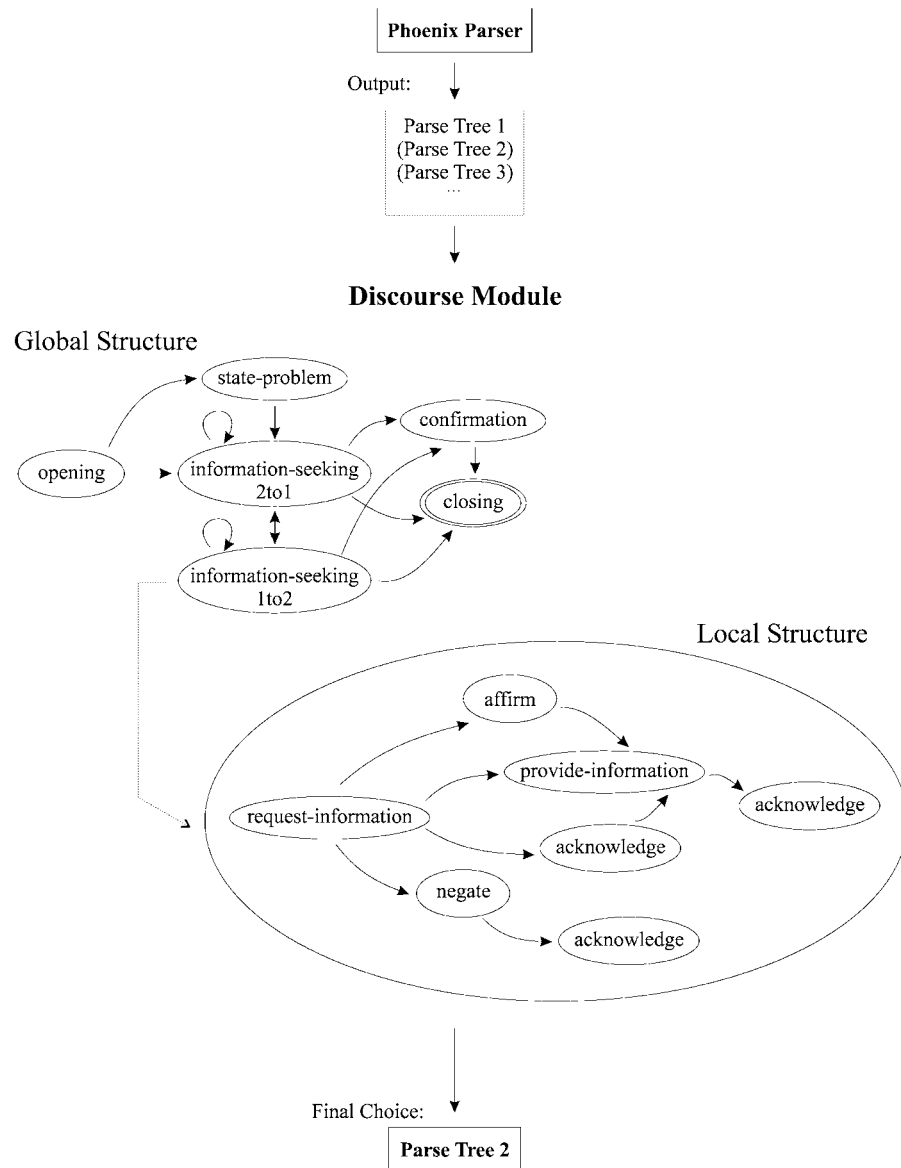


Figure 2. The discourse processor.

For the selection of speech acts I took several taxonomies into account. Searle’s classification of speech acts (Searle, 1979) was slightly adapted to deal with negotiation dialogues, as in Halliday (1994) and Martin (1992). In the computational arena, both the Enthusiast (Rosé and Qu, 1996) and the Verbmobil classifications (Jekat *et al.*, 1995) provide a very extensive set of speech acts already tested and evaluated as regards to their efficiency in machine translation. The Clarity project has also devised a set of speech acts for Spanish dialogue annotation (Levin *et*

Table I. Subdialogue and speech act taxonomy

Subdialogue	Speech act
Opening	greeting offer-help identify-self affirm acknowledge
State-problem	provide-info request-info acknowledge affirm
Information-seeking-2to1	request-info provide-info acknowledge affirm accept
Information-seeking-1to2	request-info provide-info negate acknowledge affirm
Confirmation	request-confirmation affirm
Closing	farewell thank promise acknowledge

al., 1999). Some effort was made to use the classification derived from a joint effort by the Discourse Resource Initiative (Allen and Core, 1997; Core *et al.*, 1999), which attempts to develop a standard classification scheme for discourse processing. Unfortunately, because the annotation scheme is still under revision, I was not able to make the taxonomy completely compatible with the Discourse Resource Initiative scheme.

Since the finite state machine is divided in two layers, the speech acts are grouped according to the subdialogue in which they can occur. A set of 6 subdialogues and 14 speech acts seemed the most appropriate for this domain, as listed in Table I.

Opening and closing subdialogues are self-explanatory. They include speech acts that have to do with the social purpose of those subdialogues (introduce each other, ask about each other's well-being, etc.), but also speech acts that serve a purpose within the task, as are the farewells and thanks within the closing subdialogue. Those serve to provide closure to the conversation, indicating that no further discussion is needed.

Example (2) is a typical opening, in which the first speaker (the travel agent or hotel clerk) identifies the company and provides his or her name. The first turn constitutes the opening subdialogue of this conversation (the boundary is indicated by two vertical bars). In the next turn, the caller starts the state-problem subdialogue, by providing information about the reason for the call.

- (2) S1: [identify-self] Holiday Inn Pittsburgh. [identify-self] This is Mary Ellen.
 [offer-help] How may I help you? ||
 S2: [provide-info] I'm calling to reserve a hotel room.

In (3) we can see a closing subdialogue. The first three speech acts are the end of an information-seeking subdialogue. After the acknowledgment on that information ("okay"), the caller thanks the travel agent, and at that point the closing subdialogue begins. The subdialogue is completed with another round of thanks, and a final farewell by the caller, which provides closure to the conversation.

- (3) S2: [request-info] And what time does it leave the hotel?
 S1: [provide-info] Leaves the hotel at four thirty.
 S2: [acknowledge] Okay. || [thank] Thanks very much.
 S1: [acknowledge] Alright. [thank] Thanks for calling Pittsburgh Travel.
 S2: [farewell] Goodbye.

In the state-problem subdialogue, the client explains what he or she wants to have accomplished during the conversation, as illustrated in Example (4).

- (4) S1: [greeting] Hi. [offer-help] Can I help you?
 S2: [affirm] Yeah. || [provide-info] I'd like to go out to dinner this evening.
 [provide-info] And I'm looking for some local restaurants. [request-info] Do you know of any around here?
 S1: [acknowledge] Sure.

Information-seeking-2to1 includes mainly requests for information on the part of the client ('2to1' refers to the order of initiation: the client is usually the second speaker in the conversation). In the following example, the caller asks for flight times, which are then promptly provided by the travel agent.

- (5) S2: [request-info] Could . . . Can I have some flight times that would leave some time around June 6th?
 S1: [provide-info] There are several flights leaving D.C. [provide-info] There'd be one at . . . (*conversation continued*)

Information-seeking-1to2 typically involves requests for information on the part of the travel agent, such as personal information or a credit card number. In Example (6) below, the caller is requesting information on the reservations. Before that task can be accomplished, the travel agent needs the number of travelers, and

thus she asks, in the second turn, how many people there are in the party. The beginning of the Information-seeking-1to2 subdialogue is marked with two vertical bars.

- (6) S2: [request-info] Can you make reservations for me leaving on Saturday the 8th on the 10 o'clock flight and coming back on, let's see, Sunday the 16th on the, let's make it the 5:55?
 S1: [affirm] Sure. || [request-info] How many people do you have traveling with you?
 S2: [provide-info] Well, it will be myself, my husband and my mother and my twelve-year old son. [provide-info] We'll all be coming. [provide-info] So that will be four.

Finally, in the confirmation subdialogue, both speakers ground their knowledge and make sure that they agree on what has been accomplished during the conversation. In the following example, the travel agent repeats previously established information, and starts the confirmation subdialogue towards the end of the turn, by requesting a confirmation.

- (7) S1: [acknowledge] Okay. [provide-info] I have one king-size room for \$84 a night, reserved for . . . for the middle . . . for the end of April, starting the 20th at a total of \$672. [provide-info] Your MasterCard number is 7193 5523 0186 2197, and the expiration date is 5 98. || [request-confirmation] Is this correct?
 S2: [affirm] That is correct.

The speech acts are described in more detail in Taboada (1997), which also provides further examples. The speech acts were created and compiled by only one annotator, the author, and thus no reliability measures in the annotation could be calculated (for instance, the *kappa* statistic, described in Carletta, 1996, and applied in Stolke *et al.*, 2000). Although a set of speech acts tested for reliability would be desirable, the taxonomy is based on others that have been created and tested by a number of annotators, such as the DRI taxonomy.

4.2. PARSING AND GENERATION GRAMMARS

The selected speech acts are encoded in the Grammar – in the case of Phoenix a semantic grammar, the tokens of which are concepts that the segment in question represents. Any utterance is divided in SDUs – Semantic Dialogue Units – which are fed to the parser one at a time. SDUs represent a full concept, expression or thought, but not necessarily a complete grammatical sentence. Let us take an example input and a possible parse for it:

- (8) Could you tell me the prices at the Holiday Inn?
 ([request] (COULD YOU
 ([request-info] (TELL ME
 ([price-info] (THE PRICES
 ([establishment] (AT THE
 ([establishment-name] (HOLIDAY INN))))))))))

The top-level concepts of the grammar are speech acts themselves, the ones immediately after are usually further refinements of the speech act and the lower-level concepts capture the specifics of the utterance, such as the name of the hotel in the above example.

Both parsing and generation grammars were coded for this domain. In the configuration described in this paper, the parsing grammar is English, with Spanish generation. The grammars were ported from another domain, the scheduling domain. The steps involving in adapting and modifying the entire domain are described in Lavie *et al.* (1997).

The grammar parses most of the information contained in the dialogues, although it still fails in some cases. For instance, in Example (9), the parser could not process the first part of the sentence, and it only understood the location “from Pittsburgh to D.C.”. It also assigned the wrong speech act, request-confirmation, to the entire sequence.

- (9) US Air has the most frequent flights to and from Pittsburgh to D.C.
 [request-confirmation]
 ([temp-loc] ([local] ([from-to] (FROM [place] (PITTSBURGH)TO [place] (D C)))))

4.3. DISCOURSE MODULE

The discourse module proper processes the global and local structure of the dialogue in two different layers. The first one is a general organization of the dialogue’s subparts; the layer under that processes the possible sequence of speech acts in a subpart. My assumption is that negotiation dialogues develop in a fixed way with three clear phases: *opening*, *negotiation* and *closing*. The same assumption was made for scheduling dialogues in the Verbmobil project (Maier, 1996), and it has been observed in many different types of task-oriented conversation, and in telephone conversations – also usually oriented towards a goal (Schegloff and Sacks, 1973).

The middle phase in these dialogues is referred to as the *task-performance phase*, since it is not always a negotiation *per se*. Within the task performance phase different subdialogues can take place – information-seeking, decision-making, payment, clarification, etc. The order of these subdialogues is less predictable than the *opening – task performance – closing* sequence.

Discourse processing has frequently made use of sequences of speech acts as they occur in the dialogue, through bigram probabilities of occurrences, or through modeling in a finite state machine. However, if we only take into account the speech act for the previous segment in order to pick a correct parse for the current one, we might have insufficient information to decide – as is the case in some elliptical utterances which do not follow a strict adjacency pair sequence:

- (10) (talking about flight times)

S1: I can give you the arrival time. Do you have that information already?

S2: No, I don't.

S1: It's twelve fifteen.

If we are parsing the segment 'it's twelve Fifteen', and our only source of information is the previous segment, 'no, I don't', we cannot possibly find the referent for 'twelve fifteen', unless we know we are in a subdialogue discussing flight times, and arrival times have been previously mentioned. This concept of subdialogue is equivalent to Grosz and Sidner's (1986) discourse focus, or Chafe's (1994) focus of consciousness: it refers to the events and entities present in the speakers' focus of attention.

My approach aims at obtaining information both from the subdialogue structure and the speech act sequence by modeling the global structure of the dialogue with a finite state machine, with Opening and Closing as initial and final states and other possible subdialogues in the intervening states. Each one of those states contains a finite state machine itself, which determines the allowed speech acts in a given subdialogue and their sequence. The gains from such an approach are: (1) the constraints imposed on the possible speech acts in a subdialogue – disallowing a greeting interpretation if we are not in the greeting phase of the dialogue, for instance; (2) the information obtained to process ambiguous parses – deciding that the figure 'twelve fifteen' is a flight time if we are in a subdialogue where there is a negotiation for an appropriate flight time.

The discourse component takes as its input the output of the parser. It does not provide any information for the parser in its search for an appropriate parse for a given segment, but works with whatever output it receives from the parser. The three options at that point are:

1. The discourse processor will choose one of the parses if there are multiple ones, or else the only one available, and incorporate it into its history;
2. It might find itself in a situation where it is unable to make a choice among the parses returned;
3. It might find its predictions very different from the available choices.

The last two situations may come about because there is a previous error in the information the discourse module contains – cumulative error (Qu *et al.*, 1996a) – or because the input from the dialogue could not possibly be expected in a standard conversation within this domain or genre. In those situations, the two possible solutions are either to let the parser heuristics decide or have a statistic-based probability for the input, extracted from a corpus. The following sections describe the workings of the finite state machine in more detail.

4.3.1. *Inputs and Outputs of the Discourse Processor*

The discourse processor takes as inputs the parses produced by the Phoenix parser. Those are sequences of concepts to which the input tokens are matched, as in Example (8) above.

The input string might be ambiguous with respect to the speech act it represents. Thus, in the following (invented) examples, 'okay' represents three different

speech acts, namely a prompt for an answer (11), an acceptance of a previous offer (12), or a backchanneling element (Yngve, 1970), that is, an acknowledgment that the previous speaker's utterance has been understood (13).

- (11) S1 So we'll switch you to a double room, okay?
- (12) S1 So we'll switch you to a double room. S2 Okay.
- (13) S1 The double room is \$90 a night.
S2 Okay, and how much is a single room?

In such cases, the parser will return different speech acts for that same input, as in the examples below.

- (14) [prompt] (OKAY)
- (15) [accept] (OKAY)
- (16) [acknowledge] (OKAY)

Once these three possibilities have been fed into the discourse processor, the context will determine which one to choose, and that will be the output. Presently, the discourse processor does not admit the whole parse tree as input, only the top-level speech act. Below there are two typical input-output sequences, one where the speech act is ambiguous (17), and one where there is no ambiguity (18).

- (17) Input string:
OKAY
Parser's output:
[prompt] (OKAY)
[accept] (OKAY)
[acknowledge] (OKAY)
Input to the discourse processor:
[prompt],[accept],[acknowledge]
Output of the discourse processor (dependent on context):
[accept]
Output sent to the generator:
[accept] (OKAY)
- (18) Input string:
CAN YOU GIVE ME YOUR CREDIT CARD NUMBER
Parser's output:
[request] ([request-info] ([cc-number] (CAN YOU GIVE ME YOUR CREDIT CARD NUMBER)))
Input to the discourse processor:
[request]
Output of the discourse processor:
[request]
Output sent to the generator:
[request] ([request-info] ([cc-number] (CAN YOU GIVE ME YOUR CREDIT CARD NUMBER)))

4.3.2. *The Finite State Machine Grammar*

The finite state machine grammar is a representation of the states and transitions in the finite state machine, as in Figure 2 above. It defines the sequence of possible subdialogues and speech acts, together with their possible adjacencies. Thus, a state in the finite state machine is defined through the subdialogue where it appears plus the speech act it represents. In (19) we can see the speech acts contained in the opening subdialogue.²

- (19) opening – zero
 opening – identify-self
 opening – offer-help
 opening – greeting
 opening – affirm
 opening – acknowledge

For each one of these states there is a possible sequence of follow-up states, again represented by a subdialogue-speech act combination, plus information on whether the follow-up happens when the speaker is the same or when the speaker changes. The opening – identify-self state can be followed by either an opening – acknowledge, if the speaker changes, represented here by a 1, or it can be followed by an opening – offer-help, when the speaker is the same, represented by a 0.

- (20) opening – identify-self
 opening – acknowledge
 1
 opening – identify-self
 opening – offer-help
 0

The grammar was hand-coded, based on 29 dialogues from the English corpus of travel-planning dialogues, and also informed by experience with the scheduling corpus. It represents the most typical flow of the conversation, although it does not account for every single conversation, given that there are conversations that deviate from this general characterization.

4.3.3. *The Algorithm*

The discourse module processes one conversation at a time, represented in the top-level speech acts of the parses the Phoenix parser produces. For each speech act received by the discourse module there are two possible situations: that the speech act is unambiguous or that it is ambiguous, that is, there is more than one possibility to choose from. In turn, an ambiguous speech act produces three new possibilities, as represented in Figure 3. In the figure, “FSM” stands for “finite state machine.”

A. Unambiguous Speech Act

In the situation where the parser returns just one speech act for the input string, the discourse processor might find that the parse matches one of the possibilities for

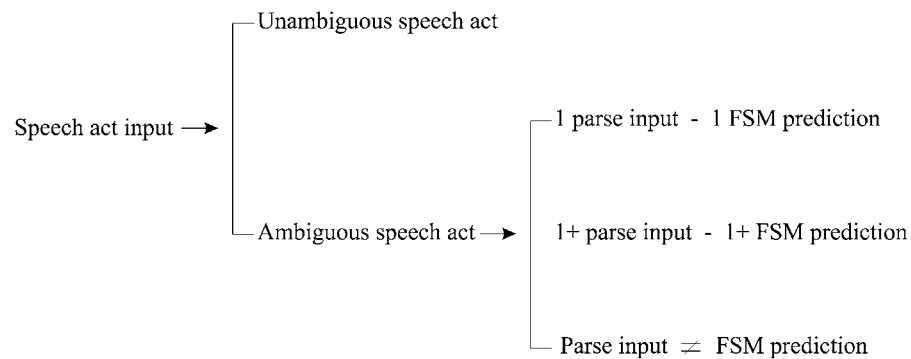


Figure 3. Discourse processor options.

next speech act given the current state or that there is no match between finite state grammar prediction and parser output, as seen in examples (21) and (22).

(21) Parser's output:

[provide-info]

Discourse Processor state:

Current state: request-info

Possible next states: provide-info, acknowledge

Match found!

(22) Parser's output:

[provide-info]

Discourse Processor state:

Current state: greeting

Possible next states: greeting

No match found, jump

In the first example, the speech act provide-info is one of two the finite state grammar predicts. Since there is a match, we add provide-info to the conversation history and proceed. In (15), there is no match of the finite state grammar predictions and the parser's output. In that case, there is a *jump* from the current state of the finite state machine to another, unpredicted state. A jump entails the resetting of the context. The systems attempts to jump first to another state within the same dialogue. If the current subdialogue does not contain the selected speech act, then the system jumps to a speech act within the next subdialogue. If that fails again, then the finite state machine is traversed from the beginning, until a matching speech act is found in any of the global subdialogues.

B. Ambiguous Speech Act

An ambiguous speech act is encountered when the parser returns more than one possible top-level speech act for the same input string. At this point there are three possibilities:

B.1 One parse matches only one possibility.

B.2 More than one parse matches more than one possibility.

B.3 There is no match between possibilities and parses.

B.1 One Parse Matches only One Possibility

This is the simplest case. As illustrated in example (23) below, the discourse processor adds the matching parse to the conversation history.

(23) Parser's output:

[provide-info], [request-info]

Discourse Processor state:

Current state: request-info

Possible next states: provide-info, acknowledge

Match found! Speech act provide-info added to conversation history

B.2 More than One Parse Matches more than One Possibility

In this case the finite state grammar itself does not solve the ambiguity. We resort to probabilities of speech acts and choose accordingly. The use of probabilities is explained in more detail in Section 4.4 below.

(24) Parser's output:

[provide-info], [request-info]

Discourse Processor state:

Current state: request-info

Possible next states: provide-info, request-info, acknowledge

Both parses were possible. Looking up probabilities . . .

Speech act provide-info added to history

B.3 There Is No Match between Possibilities and Parses

As in the previous case, we need to consult the probabilities to decide among the possible speech acts, in this case because none of them were predicted in the finite state grammar. Once we have chosen a speech act according to probabilities, we have also performed a jump, since we did not follow the sequence predicted by the finite state machine.

(25) Parser's output:

[provide-info], [request-info]

Discourse Processor state:

Current state: request-confirmation

Possible next states: provide-confirmation, affirm, negate

No match found. Looking up probabilities . . .

Speech act provide-info added to history. Jump

4.4. CODING OF THE TRAINING CORPUS FOR PROBABILITIES

As explained in the previous section, there are situations where the path followed in the two layers of the structure does not match the parse possibility we are trying to accept or reject. In those cases, the transition is determined by unigram probabilities of the occurrence of the speech act we are trying to disambiguate. To obtain those probabilities, a corpus of 29 dialogues, totalling 1,344 utterances and over 2,500 speech acts, was coded with speech acts. The probabilities were smoothed using a standard absolute discounting algorithm (e.g., Ney *et al.*, 1994).

When the discourse processor needs to look up the probabilities it just opens a file that contains the name of the speech act plus its corresponding probability. An example is shown in (26).

(26) provide-info 0.917
request-info 0.268

The system, as described in this paper, uses only unigram probabilities, that is, it chooses the most frequent speech act in the overall corpus (29 dialogues). Using bigram probabilities, which take into account the previous speech act, would be more appropriate and would also reflect what we try to achieve with the FSM, namely, the use of contextual information. Possible extensions of this work include the use of bigram, even trigram probabilities (likelihood of the current speech act given the previous, or the two previous speech acts), and the use of probabilities of traversal of the FSM, which would be equivalent to using a hidden Markov model, as in Stolke *et al.* (2000).

5. Evaluation

The evaluation of the discourse module was performed on unseen data, consisting of 5 dialogues that the system had not used before for training or testing purposes. There were a total of 228 utterances and 398 speech acts, translated from English into Spanish. Two different evaluations were performed, which followed two related criteria. A first evaluation was based on the overall improvement in translation, i.e., an end-to-end evaluation. The second type of evaluation considers the number of speech acts that were disambiguated, whether they contributed to an improvement in translation or not.

In the end-to-end evaluation, since the module can be either incorporated into the system, or turned off, the evaluation shows the system's performance with and without the discourse module. For such purposes, two tests are performed.

A first test determines the translation quality based on a comparison of the input and the output, with the assignment of a grade to the translation: "perfect" for translations where the content and the illocutionary force were conveyed in a perfect way in the target language, Spanish in our case; "okay" for translations that conveyed the content, but which were awkward or not completely perfect; and "bad" for SDUs that were not translated at all or for translations that would not be understood. As with all evaluations of the machine translation components in

Table II. Translation evaluation

	Without discourse processor	With discourse processor
Perfect	20.72%	24.09%
OK	30.31%	26.94%
Bad	48.96%	48.96%

JANUS, the grading was done by independent graders, who had native fluency in the input language and native or near-native fluency in the output language. In this test the discourse processor was not used. The disambiguation process follows simple heuristics incorporated into the parser. If there is more than one possibility, the parser picks the one that skipped less words of the input and has a shallower parse tree.

The second test includes the discourse module, and uses that module to pick among the possibilities that the parser returns. The idea behind this evaluation is that since the choice of parse tree will be more appropriate for the situation, the translation will also be more contextually accurate. The results are shown in Table II.

As the results show, there is no improvement in translation quality for translations that were “bad” in the first place. That comes as no surprise, since the discourse module could not possibly improve faulty parses returned by the parser. However, when we look at the “perfect” and “okay” grades, we can see an increase in perfect translations when the discourse processor is introduced. The increase is produced by a shift of about four percentage points from the “okay” to the “perfect” category. This means that the translation accuracy was, in fact, improved by choosing the right speech act in the right context.

The number of “bad” translations was, however, very high. This is due to the poor performance of the Phoenix grammar. The grammar had too few development dialogues to train on, and thus the parses returned by the parser were not informative enough in the first place. In Example (27), the word “separate” was not predicted in the frame “separate room”, as a type of room. As a consequence, the parse could only get a bit of information on this particular sentence, extracting “want to” and “room” as dialogue acts provide-info, and “a” as an indication of number that could not be interpreted as any dialogue act. The translation only picked up these few words, without even being able to join “a” to “room”, and providing the appropriate gender (it should be “una habitación”), but instead translating it as if it were the number “one”.

(27) Do you want to get a separate room for like your mother or your son or something?

Parse:

– I’ve found SEPARATE as a new word

Table III. Ambiguity evaluation

Total # of speech acts	Ambiguous cases	Successfully disambiguated
398	108 (27.1%)	73 (67.6%)

[provide-info] want to get
 [number] a
 [provide-info] room
 Translation:
 quiero ... uno ... habitación ...

In some cases, there were problems with the segmentation of the input. For instance, in Example (28), the input was segmented before the relative clause “that starts on June the 7th”. The parser then interpreted that as a single utterance, and assigned it the dialogue act request-confirmation. The dialogue acts for the different segments are also provided.

(28) Hello. I need to get to Pittsburgh. I am in Washington D.C. and I hear there’s an arts festival that starts on June the 7th.

[provide-info] I need to get to Pittsburgh
 [provide-info] I am in Washington DC
 [provide-info] I hear there’s an arts festival
 [request-confirmation] that starts on June the 7th

Since I felt that the first evaluation did not provide enough information with respect to the performance of the discourse processor, a second type of evaluation was performed on the parsing side, looking at the ambiguous cases returned by the parser, and at how many of those were resolved by the discourse processor. The idea behind this evaluation is to abstract away from the performance of the grammar and the parser, to focus on the performance of the discourse processor in isolation.

The procedure was to manually annotate each SDU with the possible speech acts it could fill, and to let the discourse processor choose among them. Then, the choice was checked against the input, to decide whether the discourse processor had made an appropriate choice. The same 5 dialogues were used in this evaluation. Table III contains the number of ambiguous cases, and how many of them were successfully disambiguated.

For example, (29) provides the process for one turn in the dialogue. After the travel agent offers help, the caller produces three different speech acts in one turn. The third one, “can you tell me about that”, is sometimes erroneously parsed as offer-help. For the first speech act, the processor stays in the opening subdialogue, and then moves onto state-problem subdialogue when “I want to find out about hotels in Pittsburgh” is processed. Within that subdialogue, the speech act request-

info is more likely than offer-help, therefore “Can you tell me about that?” is disambiguated to request-info.

(29) S1: . . . Can I help you?

S2: Yes. I want to find out about hotels in Pittsburgh. Can you tell me about that?

1. [affirm] Yes
2. [provide-info] I want to find out about hotels in Pittsburgh.
3. [request-info],[offer-help] Can you tell me about that?
 1. Opening subdialogue
Speech act: affirm
 2. Opening subdialogue
Speech act: provide-info
We moved to the state-problem subdialogue
 3. State-problem subdialogue
Parsed speech act: request-info
Parsed speech act number 2: offer-help
Current speech act: provide-info
Possibility: request-info
Match found for the first parse
Possibility: offer-help
No match found for this possibility
Possibility request-info added to history; proceed

The results here show that the discourse processor can resolve 67.6% of the ambiguous cases it encountered. With a better performance by the parser, the discourse processor would yield a more accurate translation in most of those cases.

There is, obviously, room for improvement in the translation. There are a number of avenues that could be taken to that end. The first one is independent of the discourse processor, and has to do with improving the parsing and generation grammars. Secondly, both the speech act inventory and the FSM grammar can be revisited and modified to accommodate different conversation structures. Finally, the use of more refined probabilities (bigram or n-gram probabilities) could also improve the performance.

6. Conclusion

In this paper I have presented a model of dialogue structure in two layers, which accounts for the sequence of subdialogues and speech acts in a type of task-oriented dialogue. The goal of the discourse processor derived from this model is to select the most appropriate parse among the possibilities returned by a parser.

The model structures dialogue in two levels of finite state machines. The discourse processor handles global and local structures within a single component, a finite state machine, that can easily be modified and extended to cover different types of dialogues. The only modification needed would be in the finite state

machine grammar. Of course, the system is able to handle with elegance only those dialogues that show the stereotypical structure modeled in the finite state machine grammar. Dialogues that incorporate elements from other genres – storytelling, gossip, casual conversation – would not benefit as much from the disambiguation of the discourse processor.

The compactness of the processor in a single component avoids the cost of exchanging information between two modules to deal with global and local structure. The system is robust with respect to unpredicted input, since it can jump to a different place in the finite state machine, creating a new context for the unexpected input. Evaluation results show that the use of the discourse processor improves the quality of translations (English to Spanish) in a speech translation system. In a different type of evaluation, it was also found that the discourse processor helps assign the most appropriate speech act in context.

Applications of modeling dialogue are not limited to machine translation, or even computational linguistics. In education, for example, there is an increasing trend to use sophisticated agents to guide students in their interaction with tutoring systems, or educational games. Socially-intelligent agents model students' cognitive and emotional states, and generate tailored interventions (Conati and Klawe, 2002). They also need to model students' expectations with respect to how a dialogue proceeds in normal settings.

Similarly, in Computer Assisted Language Learning (CALL), most of the interactions between a CALL system and a student would benefit from a model of where the conversation can go, when a clarification or a correction is necessary, or when the student has moved to a different subdialogue in the conversation. This is what Baker (2000) calls a model of the teaching or learning process, which enables the system to adapt its interventions to the learner.

In Human-Computer Interaction (HCI), context plays a key role. A recent special issue of the journal *Human Computer Interaction* is devoted to “context-aware computing” (Moran and Dourish, 2001). Ziegler (2002) proposes models for cooperative work processes (writing a paper together, designing and developing a new product), in which the high-level process is decomposed in smaller components. He presents a hierarchical framework, which can integrate different tasks. An FSM model of each of those tasks could be part of the framework, modeling the conversational structure of each task. The structure could also contain non-verbal acts, such as sending a message, or producing an outline. Another example of a possible application is the design of spoken prompts by a system (Hansen *et al.*, 1996).

Conversation structure is also of concern in the area of Software Agents. These are autonomous or semi-autonomous systems which perform certain tasks, in isolation (e.g., looking up a database), or in coordination (ordering goods from another agent). Agents need to have a common language, in order to perform tasks together. The proposals for such a language (agent communication language) are based on speech act theory, and on the notion of the structure of a conversation, in this area

known as a conversation protocol or conversation policy. These protocols capture the types of acts that can be performed in a given task (Elio *et al.*, 2000). The specifics of the language itself and of the protocols vary (Labrou *et al.*, 1999; FIPA, 2001; Labrou and Finnin, 1997), but they are all based on the notion that there is a usual path of traversal through a series of acts that are part of a task.

Acknowledgements

This project was completed during my stay at Carnegie Mellon University, which was made possible by a grant by “la Caixa” Fellowship Program. This work was part of the JANUS Project, funded by different agencies in the United States, Germany, and Japan. I am indebted to the members of the JANUS project, and the faculty and students in the Computational Linguistics program at CMU, as well as to the anonymous reviewers, for providing helpful comments and suggestions. This work was also supported in part by the Ministry of Science and Technology of Spain, under project MCYT-FEDER BFF2002-02441 (Ministerio de Ciencia y Tecnología/Fondo Europeo de Desarrollo Regional).

Notes

- ¹ The transcript has been slightly modified, and presented with standard orthographic conventions.
- ² The first state, opening – zero, is the initialization sequence for the finite state machine; it does not have any linguistic relevance.

References

- Ahlen S. (1997) *Enthusiast Data Collection*. Language Technologies Institute Technical Report, Carnegie Mellon University and The University of Pittsburgh.
- Alexandersson J., Engel R., Kipp M., Koch S., Küssner U., Reithinger N., Stede M. (2000a) Modeling Negotiation Dialogs. In Wahlster W. (ed.), *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer, Berlin, pp. 441–451.
- Alexandersson J., Poller P., Kipp M. (2000b) Generating Multilingual Dialog Summaries and Minutes. In Wahlster W. (ed.), *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer, Berlin, pp. 507–518.
- Allen J., Core M. (1997) *Draft of DAMSL: Dialog Act Markup in Several Layers*. Draft produced by the Multiparty Discourse Group at the Discourse Research Initiative (DRI) meetings at the University of Pennsylvania and at Schloss Dagstuhl. (<http://www.georgetown.edu/luperfoy/Discourse-Treebank/dri-home.html>).
- Baker M. (2000) The Roles of Models in Artificial Intelligence and Education Research: A Prospective View. *International Journal of Artificial Intelligence in Education*, 11, pp. 122–143.
- Bakhtin M. (1986) *Speech genres and Other Late Essays*. University of Texas Press, Austin.
- Carletta J. (1996) Assessing Agreement on Classification Tasks: The Kappa statistic. *Computational Linguistics*, 22/2, pp. 249–254.
- Chafe W. (1994) *Discourse, Consciousness and Time: The Flow and Displacement of Conscious Experience in Speaking and Writing*. University of Chicago Press, Chicago.

- Conati C., Klawe M. (2002) Socially Intelligent Agents in Educational Games. In Dautenhahn K., Bond A., Cañamero D. and Edmonds B. (eds.), *Socially Intelligent Agents: Creating Relationships with Computers and Robots*. Kluwer Academic Publishers, Dordrecht, pp. 213–220.
- Core M., Ishizaki M., Moore J., Nakatani C., Reithinger N., Traum D., Tutiya S. (1999) *Report of The Third Workshop of the Discourse Resource Initiative*. Chiba Corpus Project Technical Report No. 3 (CC-TR-99-1), Department of Cognitive and Information Sciences, Chiba University, Japan.
- Elio R., Haddadi A., Singh A. (2000) Task Models for Agent Conversation Policies. *Proceedings of Autonomous Agents-2000*, pp. 229–230.
- Fawcett R., van der Mije A., van Wissen C. (1988) Towards a Systemic Flowchart Model for Local Discourse Structure. In Fawcett R. and Young D. (eds.), *New Developments in Systemic Linguistics*, Vol. 2, Frances Pinter, London, pp. 116–143.
- FIPA (2001) Foundation for Intelligent Physical Agents ACL Message Structure Specification. Technical Report XC00061E. <http://www.fipa.org>.
- Grosz B., Sidner C. (1986) Attentions, Intentions, and the Structure of Discourse. *Computational Linguistics*, 12/3, pp. 175–204.
- Halliday M.A.K. (1994) *An Introduction to Functional Grammar* (2nd edition). Edward Arnold, London.
- Halliday M.A.K., Martin J. (1993) *Writing Science: Literacy and Discursive Power*. The Falmer Press, London.
- Hansen B., Novick D., Sutton S. (1996) Systematic Design of Spoken-Dialogue Interfaces. *Proceedings, Conference on Human Factors in Computing Systems (CHI'96)*, pp. 157–164.
- Jekat S., Klein A., Maier E., Maleck I., Mast M., Quantz J.J. (1995) *Dialogue Acts in Verbmobil*. Verbmobil Technical Report 65.
- Kipp M.J., Alexandersson R., Engel, Reithinger N. (2000) Dialog Processing. In Wahlster W. (ed.), *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer, Berlin, pp. 452–465.
- Koch S., Küssner U., Stede M. (2000) Contextual Disambiguation. In Wahlster W. (ed.), *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer, Berlin, pp. 466–477.
- Labrou Y., Finnin T., Peng Y. (1999) Agent Communication Languages: The Current Landscape. *IEEE Intelligent Systems*, 14/2, pp. 45–52.
- Labrou Y., Finnin T. (1997) *A Proposal for a New KQML Specification*. Technical Report CS-97-03, Computer Science and Electrical Engineering Department, University of Maryland Baltimore County.
- Lambert L. (1993) *Recognizing Complex Discourse Acts: A Tripartite Plan-Based Model of Dialogue*. PhD Thesis, University of Delaware.
- Lambert L., Carberry S. (1992) Modeling Negotiation Subdialogues. In *Proceedings of 32nd Annual Meeting of the ACL*.
- Lavie A. (1995) *A Grammar Based Robust Parser for Spontaneous Speech*. PhD Thesis, Carnegie Mellon University, Pittsburgh, PA.
- Lavie A., Tomita M. (1993) GLR*: An Efficient Noise Skipping Parsing Algorithm for Context Free Grammars. *Proceedings of the Third International Workshop on Parsing Technologies, IWPT 93*, Tilburg, The Netherlands.
- Lavie A., Gates D., Coccaro N., Levin L. (1996a) Input Segmentation of Spontaneous Speech in JANUS: A Speech-to-Speech Translation System. *Proceedings of ECAI 96*, Budapest, Hungary.
- Lavie A., Gates D., Gavaldà M., Mayfield L., Waibel A., Levin L. (1996b) Multi-lingual Translation of Spontaneously Spoken Language in a Limited Domain. In *Proceedings of COLING 96*, Copenhagen.
- Lavie A., Levin L., Zhan P., Taboada M., Gates D., Lapata M., Clark C., Broadhead M., Waibel A. (1997) Expanding the Domain of a Multi-lingual Speech-to-Speech Translation System. *Proceedings of the Spoken Language Translation Workshop*, 35th Annual Meeting of the Association for Computational Linguistics, ACL/EACL '97, Madrid, Spain, pp. 67–72.

- Levin L., Ries K., Thymé-Gobbel A., Lavie A. (1999) Tagging of Speech Acts and Dialogue Games in Spanish Call Home. *Proceedings, ACL '99 Workshop on Discourse Tagging*.
- Litman D., Allen J. (1990) Discourse Processing and Commonsense Plans. In Cohen P.R., Morgan J. and Pollack M.E. (eds.), *Intentions in Communication*. MIT Press, Cambridge, MA, pp. 365–388.
- Maier E. (1996) Context Construction as Subtask of Dialogue Processing: The Verbmobil Case. *Proceedings of the Eleventh Twente Workshop on Language Technology, TWLT 11*.
- Martin J. (1992) *English Text: System and Structure*. John Benjamins, Philadelphia/Amsterdam.
- Mayfield L., Gavaldà M., Seo Y-H., Suhm B., Ward W., Waibel A. (1995) Parsing Real Input in JANUS: A Concept-Based Approach. *Proceedings of TMI 95*.
- Moran T., Dourish P. (2001) Introduction, Special Issue on Context-Aware Computing. *Human Computer Interaction*, 16(2–4), pp. 87–96.
- Ney H., Essen U., Kneser R. (1994) On Structuring Probabilistic Dependencies in Stochastic Language Modelling. *Computer Speech and Language*, 8, pp. 1–38.
- O'Donnell M. (1990) A Dynamic Model of Exchange. *Word*, 41/3, pp. 293–327.
- Qu Y., Di Eugenio B., Lavie A., Levin L., Rosé C.P. (1996a) Minimizing Cumulative Error in Discourse Context. *Proceedings of ECAI 96*, Budapest, Hungary.
- Qu Y., Rosé C.P., Di Eugenio B. (1996b) Using Discourse Predictions for Ambiguity Resolution. *Proceedings of COLING 96*, Copenhagen.
- Reithinger N., Maier E. (1995) Utilizing Statistical Dialogue Act Processing in Verbmobil. *Proceedings of ACL*.
- Reithinger N., Maier E., Alexandersson J. (1995) Treatment of Incomplete Dialogues in a Speech-to-Speech Translation System. *Proceedings of the ESCA Workshop on Spoken Dialogue Systems*, Denmark.
- Rosé C.P., Qu Y. (1996) Discourse Information for Disambiguation. Manuscript, Carnegie Mellon University, Pittsburgh, PA.
- Rosé C.P., Di Eugenio B., Levin L., Van Ess-Dykema C. (1995) Discourse Processing of Dialogues with Multiple Threads. *Proceedings of ACL*, Boston, MA.
- Sacks H., Schegloff E., Jefferson G. (1974) A Simplest Systematics for the Organization of Turntaking for Conversation. *Language*, 50, pp. 696–735.
- Schegloff E., Sacks H. (1973) Opening up Closings. *Semiotica*, 7, pp. 289–327.
- Schmitz B., Quantz J.J. (1995) Dialogue Acts in Automatic Dialogue Interpreting. *Proceedings, 6th Conference on Theoretical and Methodological Issues in Machine Translation*, pp. 33–47.
- Searle J. (1996 [1979]) A Taxonomy of Illocutionary Acts. Reprinted in Martinich A. (ed.), *The Philosophy of Language* (3rd edition). Oxford University Press, New York.
- Sinclair J., Coulthard M. (1975) *Towards an Analysis of Discourse: The English Used by Teachers and Pupils*. OUP, Oxford.
- Stolke A., Ries K., Coccaro N., Shriberg E., Bates R., Jurafsky D., Taylor P., Martin R., Van Ess-Dykema C., Meteer M. (2000) Dialogue Act Modeling for Automatic Tagging and Recognition of Conversational Speech. *Computational Linguistics*, 26/3, pp. 339–373.
- Taboada M. (1997) *Discourse Information for Disambiguation: The Phoenix Approach in Janus*. M.Sc. Thesis, Carnegie Mellon University, Pittsburgh, PA.
- Ventola E. (1987) *The Structure of Social Interaction: A Systemic Approach to the Semiotics of Service Encounters*. Pinter Publishers, London.
- Waibel A. (1996) Interactive Translation of Conversational Speech. *IEEE Computer Society*, 29/7.
- Ward W. (1991) Understanding Spontaneous Speech: the Phoenix System. *Proceedings of ICASSP*.
- Ward W. (1994) Extracting Information in Spontaneous Speech. *Proceedings of ICSLP*.
- Yngve V. (1970) On Getting a Word in Edgewise. *Papers from the Sixth Regional Meeting of the Chicago Linguistics Society*. Chicago Linguistics Society, Chicago.
- Ziegler J. (2002) Modeling Cooperative Work Processes: A Multiple Perspectives Framework. *International Journal of Human-Computer Interaction*, 14/2, pp. 139–157.