A framework for utterance disambiguation in dialogue

Pont Lurcock, Peter Vlugter, Alistair Knott
Department of Computer Science
University of Otago
New Zealand
{pont,pvlugter,alik}@cs.otago.ac.nz

Abstract

We discuss the data sources available for utterance disambiguation in a bilingual dialogue system, distinguishing global, contextual, and user-specific domains, and syntactic and semantic levels. We propose a framework for combining the available information, and techniques for increasing a stochastic grammar's sensitivity to local context and a speaker's idiolect.

1 Introduction

Resolving the ambiguities present in an incoming utterance is a key task in natural language processing. Interpreting an utterance, whether semantically, syntactically or phonologically, is typically construed as a two-stage process: the first stage involves deriving a set of all possible analyses, using relatively well-defined principles, and the second stage involves selecting between these analyses, using principles that are harder to define and formalize.

This paper considers principles for disambiguating utterances in a human-machine dialogue system. Our main goal is to present a framework for integrating the different sources of information relevant to this task, and the available techniques for making use of this information. Working with a dialogue system highlights the need for such a framework; there are additional types of ambiguity that need to be considered (such as dialogue act ambiguity), and a particularly diverse set of informational resources to consult (many of which relate to the current dialogue context). At the same time, there are additional opportunities available—in particular, the ability to ask the user clarification questions if a given ambiguity cannot be otherwise resolved.

We begin in Section 2 with an analysis of the kinds of disambiguation needed in our dialogue system, and of the information available to the system to perform the task. In Section 3, we outline a general framework for performing disambiguation in a dialogue system. In Sections 4 and 5, we describe how our system implements this framework. We conclude in Section 6 with an account of our current work.

2 Sources of information available for utterance disambiguation

Utterance interpretation is typically modelled as a pipeline process, beginning with phonological interpretation, proceeding with syntactic analysis and semantic analysis, and concluding with discourse or dialogue attachment. We will illustrate by showing the pipeline used in our own bilingual English/Māori dialogue system, Te Kaitito (see e.g. Knott and Wright (2003); Knott et al. (2004)). Our pipeline is shown in Figure 1. Since our input is a written sentence

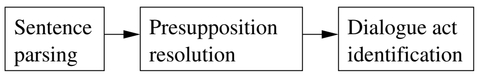

Figure 1: The utterance interpretation pipeline

we begin with sentence parsing, using the LKB system (Copestake and Flickinger, 2000). LKB works with HPSG-like grammars; the grammar we use is bilingual, simultaneously encoding a fragment of English and Māori (Knott et al., 2002). LKB's parser delivers 'flat' semantic representations in a format called Minimal Recursion Semantics (MRS) (Copestake et al., 1999), which are turned into Discourse Representation Structures (DRSs) distinguishing between an assertion and a set of presuppositions (Kamp et al., in preparation). These DRSs are then passed to a presupposition resolution module, which finds referents for anaphora, definite NPs, and other presuppositional constructions. The resolved DRSs are passed to a dialogue act identification module, which determines the dialogue act made by the utterance.

Ambiguities can arise at any point in this pipeline. There can be many parses of the sentence, many semantic interpretations of a parse tree, many ways to resolve the presuppositions in a semantic interpretation, and many ways to interpret the resulting structure as a dialogue act. By the end of the pipeline, there can be a large number of interpretations consistent with the original input sentence. How should one interpretation be chosen?

To answer this question, it is useful to survey the kinds of information available to a system for the resoluton of ambiguities. There are two ways of thinking about this information. Firstly, relevant information can appear at different **levels**, suitable for use at different points in the pipeline. Some information is **syntactic**: as is now well known[1] we can make use of statistics about the relative frequency of syntactic constructions in a corpus to decide between alternative analyses. Other information is **semantic**: for instance, we can specify a set of axioms about what are considered normal circumstances, and use a theorem-prover to determine which candidate interpretation is most consistent with these. Syntactic information can be used to resolve syntactic ambiguities, and semantic information can be used to resolve ambiguities in semantic interpretation and discourse attachment. Secondly, relevant information can come from different **domains**, of which we consider three prominent ones: the **world**, the **speaker model**, and the **dialogue context**. Domains and levels of information are roughly orthogonal; Table 1 summarizes their possible combinations.

		level	
		syntactic	semantic
domain	world	general corpus statistics	world knowledge axioms
	speaker model	language model of the user	axioms about user
	dialogue context	statistics about recent context	context-matching operations

Table 1: Domains and levels of information for use in utterance disambiguation

This division is remeniscent of Menzel and Schröder's (1999) concept of multi-level parsing, although they do not use the orthogonal domain and level classifications shown here.

[1]See e.g. Manning and Schütze (1999, Chapter 12).

To illustrate these categorizations, consider a well-known example of syntactic ambiguity:

(1) Fruit flies like a banana.

Under the most intuitive reading (call it Reading 1), the sentence is about what fruit flies like to eat. But it can also be interpreted (Reading 2) as being about the way fruit tends to fly through the air. We can use different sorts of knowledge to help decide between these alternatives.

Firstly, we could use world knowledge—for example, the fact that fruit doesn't typically fly, or that insects often like fruit. This information could take the form of axioms in some suitable logical language, directly encoding such propositions. But it could also take the form of statistics about common syntactic structures in wide-domain corpora, which indicate that the probability of the verb *fly* taking a subject headed by *fruit* is vanishingly low. These statistics can also be seen as a form of world knowledge, albeit one encoded in a far less explicit way than a set of logical propositions.

Knowledge about the speaker of the sentence is also of use in helping to disambiguate. For instance, if the speaker is known to be a geneticist, this would support Reading 1, while if she is an aerodynamicist, this might generate a weak preference for Reading 2. Again, knowledge of the speaker can either be semantic (e.g. taking the form of logical axioms) or syntactic (e.g. statistics derived from a set of her previous utterances).

Finally, an utterance's context provides strong constraints on disambiguation. Again, this information can be syntactic (a recent usage of *fly* as a verb or noun would lend support to the corresponding interpretation) or semantic, involving use of a logical representation of the dialogue context: an utterance is scored by the ease with which it can be incorporated into the context (Knott and Vlugter, 2003). For instance, if the utterance answers the question *How does fruit fly?*, we may assume Reading 2; if it answers the question *What do fruit flies like?*, we have very strong evidence for Reading 1.

In summary: we can distinguish six broad categories of information relevant to utterance disambiguation, classifying orthogonally by domain and level.

3 A framework for utterance disambiguation

The central question in this paper is: how can we incorporate disambiguation routines into our pipeline so as best to integrate these disparate categories of information? A simple approach would be to use a 'greedy' algorithm: choose the best syntactic analysis, using all the available syntactic data, then pick the best semantic interpretation that can be derived from this winning syntactic analysis, and so on. But this approach can throw out plausible interpretations: semantic information often *overrides* statistical information about frequency of syntactic constructions. Consider the following dialogue:

(2) System: I keep my pig in a pigpen.
User: Where is the pen?

If the system's grammar has two lexical entries for *pen*, one meaning 'writing pen' and one meaning 'pigpen', the user's utterance will be syntactically ambiguous. The intended interpretation is clearly 'pigpen', but corpus statistics are unlikely to support this; if anything, writing pens will be more common in a general corpus. (Even looking at syntactic constructions in the recent context will not help, since the word *pen* was not used in the first utterance.) However, at a higher level, presupposition resolution information can give us the right reading. The intended reading carries a presupposition that there is a pigpen, which can be successfully resolved, while the other reading presupposes a writing pen, which will not be found, and must be accommodated.

In summary, we need a way to *combine* information from different levels (syntactic and semantic) and domains (world, user and local). Two questions now arise. Firstly, when disambiguating an utterance at some point in the interpretation pipeline, how far should we **look ahead** along the pipeline? Secondly, how should we combine evaluations made at different points in the pipeline using different types of information?

3.1 Proposal for a disambiguation procedure

As to the question of how far to look ahead: we propose that we should always look to the very end of the pipeline. For instance, when performing syntactic disambiguation, we should take each possible reading, and perform semantic interpretation, presupposition resolution and dialogue attachment. Of course, each of these might themselves generate alternative possibilities. The resulting space of possibilities is rather like a conventional AI search graph, in which leaves are complete interpretations of the utterance. At each stage, the procedure which generates the ambiguities at that stage can assign each alternative a **local score**, using the information appropriate to that stage. When we have created the full set of complete interpretations, we can combine these local scores somehow (see Section 3.2) to create a **global score** for each complete interpretation.

We now need to use these scores to decide on the best interpretation. Local and global scores provide only a heuristic measure of which interpretation is best, so we can only use them as a rough indicator of which is the best reading. If one interpretation has a global score far exceeding those of all other interpretations, we can safely choose it. But if there are several alternatives with roughly the same global scores, we need to ask a clarification question, so that the user can disambiguate overtly. The selection of an appropriate clarification question is problematic in its own right, since we may be trying to distinguish between several interpretations whose high global scores originate in different stages of the pipeline. Section 3.3 discusses this topic.

3.2 Combining information types

How do we combine local scores due to statistical parsing, presupposition resolution and dialogue attachment? To date, we have considered two approaches. We first considered a **weighting formula**. In this method, all local scores are numerical, and the global score is a weighted sum of the local scores, with weights chosen so as to reflect the importance of different sources of information. However, this fails to capture potential interactions between different information sources. We now use a **conditional formula**—basically a simple procedural algorithm. In this scheme, we can specify, for instance, that a certain local score is only used if there is a tie as regards some other local score. Our general observation is that higher levels of information tend to trump lower levels; for instance, if there are two alternative interpretations of an utterance at the dialogue act level, and only one of these is consistent with a coherent dialogue, then it shouldn't matter if the other interpretation scores more highly in the syntactic or presupposition-resolution domains. Section 4 discusses this algorithm further.

3.3 How to generate clarification questions

A key component in a dialogue-based disambiguation system should be the ability to ask the user for clarification. There are well-known dialogue strategies for doing this; the concept of a **clarification subdialogue** is well-established as a dialogue structure—see, for example, Schmitz (1997).

It seems that different clarification questions target ambiguities at different points in the interpretation pipeline. At the syntactic level, clarification questions tend to have a **multiple-choice** structure, in which the alternative syntactic possibilities are disambiguated by rephrasing. For instance:

(3) User: Fruit flies like a banana.
System: Do you mean (1) 'A banana is liked by fruit flies,' or (2) 'Fruit flies just as a banana does'?

At the presupposition resolution level, clarification questions tend to take the form of *wh*-questions. For instance:

(4) User: The dog barked.
System: *Which* dog barked?

These questions are sometimes termed **echo questions** (c.f. Ginzburg (1996)). They are syntactically different from ordinary questions: for instance, the *wh* element receives a certain kind of stress, and interestingly, they can be embedded inside questions:

(5) User: Who did the dog chase?
System: Who did *which* dog chase?

Questions about dialogue act assignment are much less common. Questions of this sort would probably include **meta-level questions** such as the following:

(6) System: Are you talking to me?

(7) System: Are you asking me, or telling me?

Recall from Section 3.1 that a clarification question will be asked if the highest-ranked interpretation is close enough in score to one or more lower-ranked interpretations. What sort of clarification question should we then generate?

Our key suggestion here is that we need to nail ambiguities *in the order they arise in the interpretation pipeline.* We do not want to ask a question about an ambiguity at one stage in the pipeline if there are still ambiguities remaining at an earlier stage. We therefore propose traversing the interpretation space a second time for the interpretations to be clarified; as soon as an ambiguity is generated, we should ask a question to resolve it. If ambiguities still remain after this question has been answered, we continue to traverse the search space for the remaining interpretations, and ask further clarification questions about points further on in the pipeline. For example, we might have three potential interpretations A, B and C. If B and C have the same syntactic analysis, but A's analysis differs, we ask a multiple-choice question about these two syntactic possibilities. If the user answers that the syntax is that of B/C, we need to look further in the pipeline. If we find that B and C have a presupposition resolved in different ways, we then ask an echo question to nail this remaining ambiguity.

The ambiguities that we find on this second pass can be thought of as those that we find to be ambiguous *in hindsight*, in the light of processing further on in the pipeline. Even though we are asking a question about syntactic ambiguity, we have actually worked out one or more full interpretations for each possibility. To be helpful to the user, the system could perhaps be configured to include information with a 'parenthetical' flavour in a clarification question, indicating what subsequent interpretation decisions follow as a corollary to the alternatives she is currently being asked about. For example:

(8) User: The fruit flies like a banana.
System: Do you mean (1) 'The (fresh) fruit (in the bowl) flies like a banana,' or (2) 'The (mutant) fruit flies (in the genetics lab) like a banana'?

As described in section 5.1, we cannot be sure that the system's grammar will always be sufficient to form a syntactically unambiguous rephrasing such as *The drosophila are fond of a banana.* In these cases, parenthetical annotations of semantic information could be very useful.

Parse	Probability	Attachment	Saliency	Presuppositions	Accommodations	Dialogue act
P1		P1.A1				
		P1.A2				
P2		P2.A1				
P3		P3.A1				
		P3.A2				
		P3.A3				
⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 2: Structure for aggregation of utterance disambiguation data in Te Kaitito

4 Disambiguation in Te Kaitito

Te Kaitito provides a variety of sources for disambiguation data: a DRS representation of the current discourse context, a saliency list of referents ranked by how recently they have been mentioned, and records of recent utterances and preferred parses. We also have a corpus of sentences hand-annotated with their correct parses.

As an utterance travels along Te Kaitito's processing pipeline, a **disambiguation table** (See Table 2) is progressively filled with all the information necessary for a disambiguation decision. When the utterance reaches the end of the pipeline, a disambiguation module uses this information to make a decision—either selecting an interpretation outright, or initiating generation of clarification questions if there is insufficient information for a clear decision.

The structure of the disambiguator is, broadly speaking, an iterative pruning process traversing the table from right to left. Each column is consulted in turn and sufficiently implausible parse/attachment combinations discarded. If at any stage only a single interpretation remains, it is chosen as the correct one; otherwise the next column to the left is used to prune the remaining parses. If multiple interpretations remain after the final stage (the stochastic grammar), clarification questions are generated (see Section 5).

This model fits the observation, made in Section 3.2, of prioritising information from higher semantic levels: the lower levels are only consulted as tie-breakers for the higher levels.

The exact mechanism for combining disambiguation is subject to some experimentation: in particular, it is not clear that presuppositional weight is always a more reliable indicator than saliency. However, the self-contained nature of the disambiguator lets us modify it without affecting the rest of the system.

4.1 Syntactic disambiguation using statistics

Probabilistic rule annotation using statistical data is an established technique for resolving syntactic ambiguity (Manning and Schütze, 1999, Chapters 11 and 12). Augmenting a context-free grammar with rule probabilities is relatively straightforward; the application of similar techniques to a Head-Driven Phrase Structure Grammar such as that used in Te Kaitito is a more complex issue (Brew, 1995).

We use the techniques discussed by Toutanova et al. (2002) for stochastic HPSG parsing: augmentation of the derivation trees with probabilities in the manner of a probabilistic CFG, and an expectation-maximization technique on selected features of the derivation tree.

To construct a stochastic grammar, we require a source of statistical data. In our case this takes the form of an annotated treebank. Using the LKB parser and the [incr tsdb()] package[2], we can parse a corpus of test sentences and manually select preferred parses. [incr tsdb()] stores the human annotator preferences as first-class data, making it relatively immune both to changes in the nature of the statistics extracted and in the underlying grammar.

4.1.1 Contextually augmented probabilities

Commonly, probabilities in stochastic grammars are static: they are inferred off-line from a corpus and remain fixed thereafter. It would clearly be desirable to adapt the probabilities to the current dialogue context. Consider ex-

[2][incr tsdb()] provides an integrated grammar development environment with a range of facilities for diagnostics, evaluation and benchmarking (see Oepen (1999) for details). Here we are mainly concerned with its facility for maintaining a database of test items, parses, and human correctness annotations.

ample 1: If *fruit flies like bananas* follows hard upon *meat flies rather inelegantly* and *vegetables fly like a dream*, then we would like to assign greater weight to the *fly-as-verb* reading. This can be done by treating the dialogue as an additional corpus, albeit with different learning parameters.

We propose augmenting the usual probability of a rule or feature with another value representing its 'weight' in the immediately preceding context. This value is then combined with the corpus-derived probability to give the overall probability used in disambiguation.

The contextual 'weight' of a feature is simply a counter which is incremented whenever the feature appears in an utterance by the user or the system. Heavy damping is applied at each dialogue turn so that features which stop appearing in the context soon regain their base probabilities. Careful tuning of the damping factor will be necessary.

It is not at present clear how best to combine the contextual weight of a feature with its base probability. Our current proposal is simple addition after scaling by an empirically tuned factor, but further evaluation will be needed.

4.1.2 The user model

The distribution of grammatical features varies with the speaker as well as with the context, both in vocabulary and higher-level constructs. Our dialogue system functions in a language learning environment, where users are learners of Māori. In this environment, there is even greater variation between users' idiolects, as learners often have widely differing skill levels.

For this reason it makes sense to augment the probabilistic model with per-user information. Again, the dialogue itself is used as an ancillary corpus, but in this case statistics are only extracted from the user's utterances, not those of the system. The results of this on-line learning can then be combined with the corpus-derived probabilities in the same way as the contextual counts. Again, some damping is desirable. Personal usage patterns don't change as fast as conversational topics, but they are subject to gradual variation, particularly when the speaker is learning a new language.

There are additional benefits to keeping track of a learner's usage patterns: they can be compared with a corpus of sentences at the learner's intended level to find gaps in their knowledge of a language. It may be useful to bias the system's generation system in favour of features that a learner's speech *lacks*, in order to give them more exposure to constructs that they find more difficult.

4.1.3 Question-answering and priming

In human conversation, even the most unlikely parse can become plausible when primed by a question with the appropriate syntactic structure. For example, a hugely improbable interpretation of the sentence *Matt cooks lunch* can be primed by prepending the question *What do matt cooks do when they get hungry in the middle of the day?*. We would like to be able to prime our system similarly.

This kind of priming can be incorporated in similar fashion to the context and user models, using a very short-term skewing of feature probabilities. When a question is asked, the probabilities of its features are substantially increased for the next dialogue turn only. In this case, *matt-as-adjective* and *cook-as-noun* would receive greater than normal weight.

This problem can also be considered at other levels of processing: at a semantic level, Te Kaitito will prefer the *cook-as-noun* interpretation if there are cooks in the current discourse context. At a dialogue act level, the uncommon interpretation can be preferred because it is the only one which answers the question. One advantage of incorporating this kind of priming at the syntactic level is efficiency: with a large grammar it might be necessary to prune less likely parses before the semantic stage of processing, even though this runs counter to our disambiguation technique. In this case it's vital to incorporate priming at the syntactic level, or the correct parse may be removed before the semantic and dialogue-act layers can perform disambiguation.

4.2 Semantic disambiguation using presupposition resolution

We have already described how our system uses information about the presupposition resolution process to generate preferences between interpretations: see Knott and Vlugter (2003). Here is a quick summary.

There are three possibilities to consider when resolving a presupposition: we may find no antecedents in the discourse context that match the presupposition, we may find exactly one antecedent that matches, or we may find more than one antecedent. If we cannot find any discourse entities to bind a presupposition to we can be generous and resolve this presupposition by accommodating the information provided. If there is exactly one possible binding the presupposition is simply resolved. If there is more than one possible binding then we add further ambiguity to the intended meaning of an utterance as each possible binding can be considered a separate interpretation. After presupposition resolution each interpretation has its presuppositions resolved either through binding to some entity in the discourse context, or by accommodating the content of the presupposition, and the number of possible interpretations may have increased.

In disambiguation through presupposition resolution we keep to three principles. Firstly, we prefer interpretations that resolve through binding over those that resolve through accommodation. Secondly, when presuppositions are resolved through binding we prefer those with greater presuppositional content. Thirdly, we prefer interpretations where presuppositions are resolved to more salient entities in the discourse.

4.3 Dialogue act disambiguation

Dialogue act disambiguation is currently done procedurally. As noted in Section 3.2, it seems unlikely that a dispreferred dialogue act interpretation could ever be redeemed by high local scores on syntactic or presupposition-resolution grounds. In the context of a question, an assertion is checked to see if it can be interpreted as an answer to the question. If not, it is considered to be a new assertion (with the question being ignored). Note that it could be an answer that the system cannot interpret as such, because of some misunderstanding or the limitations of matching questions and answers. In the context of an ungrounded assertion, a question is first considered as a possible clarification question and then, if this fails, as a new query (with implicit grounding of the previous assertion). In dialogue act disambiguation we prefer interpretations that answer a question over those that ignore the question and assert new information, and so on.

5 Clarification questions in Te Kaitito

Te Kaitito currently generates two kinds of clarification question. These will be described in turn.

5.1 Multiple-choice questions

We generate all possible sentences realizing each remaining candidate interpretation, and then search for sentences that unambiguously present one interpretation. Here is an example:

(9)	User:	The dog chased the sheep.
	System:	Do you mean (1) 'The sheep were chased by the dog,' or (2) 'The sheep was chased by the dog'?

There's no guarantee that we will find such a sentence for each interpretation. However, we can take certain measures to improve the chances of doing so. Firstly, we can adapt our grammar to include devices for resolving common ambiguities. For instance, we can include a lexical item *you (by yourself)*, to unambiguously signal a singular second-person pronoun. Naturally we do not want to generate this term *except* when we are nailing ambiguities. But if we use the probabilistic grammar to rank alternative sentences when generating, and we ensure that special constructions are rare in the test suite, we should ensure that they are only used in clarification questions. Secondly, our grammar is bilingual, so we can generate sentences that realize an interpretation in another language. For example:

(10)	User:	Kia ora, e hoa mā.
	System:	Do you mean (1) 'Hello friends,' or (2) 'Hello, O white friend'?

This also increases the likelihood of paraphrases that successfully nail syntactic ambiguities.

5.2 Echo questions

We ask echo questions to resolve referential ambiguities generated during presupposition resolution. Example 4 demonstrates this. To be a bit more useful, the system can also generate multiple-choice alternatives:

(11)	User:	The dog barked.
	System:	*Which* dog barked? The black dog? Or the white dog?

Finally, if ambiguities are encountered during the processing of questions, rather than produce a nested echo-question, the system simply provides an answer to all of the questions the user can be understood as asking.

(12)	User:	Which dog chased the cat?
	System:	The black dog chased the white cat. The white dog chased the black cat.

6 Summary and further work

We have created a general framework within which to combine dialogue information from different domains and semantic levels, for the purpose of disambiguating user utterances in a dialogue system. We propose that disambiguation information be roughly prioritised according to its position in the processing pipeline: high-level semantic information carries more weight than low-level syntactic information. Clarification questions, if required, are generated in increasing order of semantic level. We also discuss a technique for augmenting a stochastic grammar with statistics drawn from the current dialogue context and from a particular user's dialogue history, giving it a better chance of selecting the correct parse in a given context.

The framework we describe is already in place, but many of the variables in the process (for example, the weighting and damping factors used to augment the stochastic grammar) still require some careful tuning. Further testing may also expose unforeseen interactions between levels, which may complicate the current straightforward iterative pruning algorithm; however, no changes to the framework itself should be necessary.

References

Chris Brew. 1995. Stochastic HPSG. In *Proceedings of EACL-95.*

A Copestake and D Flickinger. 2000. An open-source grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings of LREC 2000*, Athens, Greece.

A Copestake, D Flickinger, I Sag, and C Pollard. 1999. Minimal Recursion Semantics: An introduction. Manuscript, CSLI, Stanford University.

J Ginzburg. 1996. Interrogatives: Questions, facts and dialogue. In S Lappin, editor, *The handbook of contemporary semantic theory*, pages 385–422. Blackwell, Oxford.

H Kamp, J van Genabith, and U Reyle. in preparation. Discourse representation theory. In *Handbook of Philosophical Logic*. Springer-Verlag.

A Knott and P Vlugter. 2003. Syntactic disambiguation using presupposition resolution. In *Proceedings of the 4th Australasian Language Technology Workshop (ALTW2003)*, Melbourne.

A Knott and N Wright. 2003. A dialogue-based knowledge authoring system for text generation. In *AAAI Spring Symposium on Natural Language Generation in Spoken and Written Dialogue*, Stanford, CA.

A Knott, I Bayard, S de Jager, L Smith, J Moorfield, and R O'Keefe. 2002. Syntax and semantics for sentence processing in English and Māori. In *Proceedings of the 2nd Australasian Natural Language Processing Workshop*, pages 33–40, Canberra, Australia.

A Knott, I Bayard, and P Vlugter. 2004. Multi-agent human-machine dialogue: issues in dialogue management and referring expression semantics. In *Proceedings of the 8th Pacific Rim Conference on Artificial Intelligence (PRICAI 2004)*, Auckland.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing.* The MIT Press, Cambridge, Massachusetts.

Wolfgang Menzel and Ingo Schröder. 1999. Error diagnosis for language learning systems. *ReCALL.*

Stephan Oepen. 1999. [incr tsdb()] competence and performance laboratory. user and reference manual. http://citeseer.ist.psu.edu/457852.html.

Birte Schmitz. 1997. Collaboration in automatic dialogue interpreting. In *Proceedings of the Workshop "Collaboration, Cooperation and Conflict in Dialogue Systems", IJCAI-97*, pages 79–88.

Kristina Toutanova, Christopher D. Manning, Stuart M. Shieber, Dan Flickinger, and Stephan Oepen. 2002. Parse disambiguation for a rich HPSG grammar. In *First Workshop on Treebanks and Linguistic Theories (TLT2002)*, pages 253–263.