

# Incremental Inference in Context

Helen Gaylard

Department of Computer Science, University of Exeter,  
Harrison Building, North Park Road, Exeter EX4 4QF, UK  
H.L.Gaylard@exeter.ac.uk

Allan Ramsay\*

Department of Computation, UMIST,  
PO Box 88, Manchester M60 1QD, UK  
Allan.Ramsay@umist.ac.uk

## Abstract

Constructing discourse models often involves inference from contextual and background knowledge. The inference engines used for this task are typically designed for other, quite different tasks. This makes them inefficient with the repeated applications which are required over even mini-discourses. The specific requirements of the task of building discourse models were analysed and used to propose two new inference algorithms. The first uses the presumption that there will normally be at least one model for an utterance as a basis for improving performance within the discourse update step. The second uses a forwards-chaining algorithm constrained by centering. This gives us incremental inference which generates only facts relevant to the current utterance.

**Keywords:** centering, context, discourse, inference, relevance

## 1 Introduction

Inference is a powerful tool in computational semantics. It allows us to take a static, context-independent representation of utterance meaning, and interpret it in context, at the same time updating this discourse context. Thus, inference takes us beyond the realm of computational semantics into

---

\*who is partially supported by EU grant IST-2000-29452 “Dynamic Universal Mobility for Adaptive Speech Interfaces”

that of computational pragmatics, where everything is – or, rather, *should be* – constrained by context. The problem addressed in this paper is that inference itself is overly powerful unless it too is constrained by the discourse context — as readily becomes apparent from the inefficiencies that arise when processing even mini-discourses.

There are many aspects of context which could be brought to bear in constraining inference and so keeping it relevant. However, the main focus of this paper will be on one particular problem, the need for *incremental inference* in discourse. What this means is simply the following: when we are processing the second utterance in a discourse all of our inferences should be relevant to that utterance. In particular, we should not be repeating ourselves by duplicating inferences already made in processing the first utterance. An obvious move is to use a forwards-chaining algorithm to ensure that all inferences are triggered by the current utterance, but this doesn't by itself lead to the desired gains in efficiency. One reason for this is that the current utterance will indirectly trigger inferences which have already been made; for instance, inferences involving temporal relations. Inference needs to be more tightly constrained if it is to be strictly incremental. One way to achieve this is to insist that all inferences involve *centred* entities; that is, the forward-looking centres of the current utterance. The advantage of this approach to representing contextual constraints is that it is easily implemented in any system of language understanding which already uses Centering Theory (Grosz and Sidner, 1986; Grosz et al., 1995). Below we describe how incremental inference has been implemented in the system of language understanding described in (Ramsay and Seville, 2000).

## 2 Background

### 2.1 Discourse Models

In (Ramsay and Seville, 2000), inference is used to build discourse models. These represent the *common ground* (Grice, 1975) of conversants, and contain the kinds of information needed to resolve presuppositions and other contextually-dependent aspects of interpretation. A discourse model is updated with a discourse state corresponding to an utterance as outlined below.

The input to the discourse update step is the logical form for the utterance. We obtained the logical form illustrated below by parsing the utterance *John slept*. This logical form can be paraphrased as follows. It says that there is some time,  $A$ , which is past with respect to the speech time of the current discourse state, that there is some sleeping event,  $D$ , which is

in a simple aspectual relationship to this time, and that John is the actor of this event. Referring expressions are represented in the logical form as terms of the kind  $ref(\lambda X(P.X))$ . The term  $ref(\lambda E(named(E, John)))$  can be read as a reference to the discourse entity which satisfies the *property*  $\lambda E(named(E, John))$ .

$$\begin{aligned} \exists A : \{ & A \text{ is interval} \\ & \&ends\_before(ref(\lambda B(speech\_time(B, ref(\lambda C(cdiscourse(C)))))), A) \} \\ \exists D : \{ & aspect(simple, A, D) \} \\ & \theta(D, actor, ref(\lambda E(named(E, John))) \\ & \&sleep(D) \\ & \&D \text{ is event} \end{aligned}$$

### 2.1.1 Anchoring

Discourse update consists of two steps. In the first step, the logical form is *Skolemised* and all referring expressions are *anchored* to entities in the discourse model. To anchor a referring expression  $ref(\lambda X(P.X))$ , we demonstrate:

- **Existence:** by proving that  $P.Y$  holds of some entity  $Y$ <sup>1</sup>
- **Uniqueness** (relative to the discourse context): by failing to prove that there is an *equally salient* entity  $Z$  such that  $P.Z$  also holds

For instance, in anchoring  $ref(\lambda E(named(E, John)))$ , Discourse State 0, which represents everything which can be proved in the common ground prior to the start of the discourse, provides the discourse context. The background knowledge represented in Discourse State 0 includes the fact  $named(\#348, John)$ , and, as  $\#348$  has no competitors in Discourse State 0 also known to be named *John*, the referring expression's uniqueness presupposition succeeds. The anchored logical form we obtain is given below.

$$\begin{aligned} \#424 \text{ is interval} \\ \&ends\_before(\#288(1), \#424) \\ \&aspect(simple, \#424, \#425) \\ \&\theta(\#425, actor, \#348) \\ \&sleep(\#425) \\ \&\#425 \text{ is event} \end{aligned}$$


---

<sup>1</sup>This need not be an existing discourse entity. We may be able to prove that a referring expression, like *the first snowdrops of Spring*, denotes by virtue of its sense (Seville and Ramsay, 2000; Gaylard and Ramsay, 2002).

We record entities mentioned or introduced in this logical form as the list of forward-looking centres for the utterance.

### 2.1.2 Inference

Discourse State 1 will consist of the facts (or rules) explicitly represented in the logical form above together with those new facts which can be *inferred* from these and the existing discourse model. This discourse model contains, in addition to the proven facts evident in discourse states, further background knowledge in the form of rules, or *meaning postulates*; for instance:

$$\begin{aligned} \forall X \text{woman}(X) &\rightarrow \text{female}(X) \\ \forall X (\text{X is interval}) &\rightarrow \text{extended}(X) \vee \text{instantaneous}(X) \end{aligned}$$

The second stage of the discourse update step involves taking the existing discourse model, updating it with the information explicitly present in the anchored logical form, and then using an inference engine to update the model with further facts and rules which can be inferred. In the context provided by Discourse State 0, we obtain the following discourse state from the utterance *John slept*:

Discourse state 1

<hr/>	
<i>aspect</i> ( <i>simple</i> , #412, #412)	<i>sharedloc</i> (#412, <i>agent</i> )
<i>type</i> (#412, <i>interval</i> )	<i>static</i> (#412)
<i>type</i> (#412, <i>event</i> )	<i>sharedagentloc</i> (#412)
$\theta$ (#412, <i>actor</i> , #348)	<i>ends_before</i> (#288(1), #412)
<i>type</i> (#348, <i>animal</i> )	<i>type</i> (#289(#412), <i>instant</i> )
<i>atelic</i> (#412)	<i>start</i> (#289(#412), #412)
<i>sleep</i> (#412)	<i>completion</i> (#290(#412), #412)
	#289(#288(1)) > #290(#412)

## 2.2 The Inference Engine

The inference engine used to construct the above model has been described extensively in (Ramsay, 1995; Cryan and Ramsay, 1997; Ramsay, 2001). The basis for this inference engine is Manthey and Bry's SATCHMO (Manthey and Bry, 1988), which has been extended to deal with the kinds of intensional representations required for the task of natural language processing. SATCHMO's efficiency at solving a certain class of problems can be put down to its separation of the knowledge base into Horn and disjunctive sequents, with the former being dealt with very efficiently through

backwards-chaining and the latter being converted into Horn sequents by means of a kind of forwards chaining. In the extended version described in (Ramsay, 1995; Cryan and Ramsay, 1997; Ramsay, 2001), the knowledge base is divided into Horn and complex sequents, with the latter being used to represent equalities and  $\lambda$ -applications, in addition to disjuncts.

The basic algorithm for dealing with complex sequents can be described as follows:

1. Attempt to prove **absurd** on the basis of the existing knowledge base of Horn sequents.
2. Find a complex clause whose antecedent is satisfied and assert its consequent. If the consequent is disjunctive, this means non-deterministically choosing and asserting one of the disjuncts. Repeat 1.

Once all complex sequents have been explored, assuming **absurd** has not been proved, it is a simple matter to flesh out the model by collecting those facts which can be proved through backwards-chaining. As Horn sequents can be translated directly into Prolog clauses, this final stage of model building is extremely efficient.

While SATCHMO was originally used to prove a goal  $G$  by asserting its negation  $\neg G$  and attempting to derive a contradiction, here it is used, not to prove any particular goal, but to attempt to construct a model, assuming one exists. The significance of this will be made apparent below.

### 3 Inference in Discourse

As has already been mentioned, the domain of natural language processing requires that we use an intensional logic. This inevitably makes inference less efficient as, for example, we need to deal with certain  $\lambda$ -applications through forwards chaining. This much is unavoidable.

Arguably the main *avoidable* source of inefficiency in building discourse models comes from the repeated application of the inference engine with each successive discourse state. SATCHMO was designed to be applied to a static knowledge base. In constructing a discourse model we are dealing with a very different situation. We need to re-apply the forwards-chaining mechanisms of the inference engine everytime the discourse is updated. This means exploring complex sequents whose antecedents are satisfied, only to find that their consequents have also already been satisfied in a previous discourse state. A further major cost comes from consistency-checking. We have

meaning postulates like the following in the system, representing general background knowledge:

$$\forall I :: \{I \text{ is instant}\} \neg I > I$$

This simply says that no instant can follow itself. Many other temporal constraints are needed to ensure consistency. For instance, another rule says that the end of an interval cannot follow its start. As the discourse model progresses, it amasses temporal information. Checking the consistency of this becomes more and more costly with each discourse state. Worse, as the discourse progresses, the consistency checks we are making which actually involve *new* facts become a shrinking proportion of the whole. Mostly, we are just repeating our past efforts.

The inefficiencies which occur with successive applications of the inference engine to successive discourse states are compounded by the inefficiencies resulting from the repeated application of consistency-checking within each discourse state. In the context of the kinds of puzzles SATCHMO is designed to solve, it makes sense to check for consistency as often as possible. The task of building the discourse model differs from such puzzles in at least two important respects:

1. We are looking for a model, not hoping to show there isn't one.
2. Not all of our complex sequents are disjunctive.

The relevance of the second point is as follows. It makes sense to check early on if a disjunct leads to a proof of **absurd**, as this allows you to backtrack to attempt an alternative disjunct. However, it is not clear that it is worth checking consistency after every complex sequent is explored, as there may be no possible remedy from backtracking if, for instance, the assertion of an equality results in inconsistency. The question which has to be asked is whether we want to spot inconsistencies as quickly as possible or arrive at models, where they exist, as quickly as possible. In the context of natural language processing, there is a strong presumption that there is *at least one* model of an utterance in the context in which it is spoken. This is point (1) above. If we are processing the utterance *John fired his secretary yesterday*, there will be an inconsistent interpretation involving the *firearm* sense of *fire* which we will want to reject, and a consistent one involving the *employment* sense of *fire* which we will want to accept. Arguably, then, consistency-checking should not be applied as often as possible but only in those situations where it will enable us to reject an inconsistent interpretation in favour of a consistent one.

Below we discuss in turn:

- those changes made to the forwards-chaining algorithm to avoid the unnecessary repetition of consistency-checking within discourse states
- the changes implemented to ensure that the inferences made with each successive discourse state are largely incremental rather than repetitive

## 4 Consistency-checking for Discourse Models

The changes made to the basic forwards-chaining algorithm depend on the observation that the only time consistency-checking can lead us to backtrack is when we are exploring a non-final disjunct of a disjunctive sequent. The following algorithm thus forces us to attempt non-disjunctive complex sequents before disjunctive ones. There is no point in consistency-checking after each of these, since we don't expect to find an inconsistency arising through non-disjunctive meaning postulates alone and even if we did find one no remedy would be available to us through backtracking. Consistency-checking is carried out after each non-final disjunct of a disjunctive sequent, with backtracking resulting if we are able to prove **absurd**<sup>2</sup>. The revised algorithm is as follows:

1. If possible, find a non-disjunctive complex clause whose antecedent is satisfied and assert its consequent. Repeat 1.
2. Find a disjunctive clause whose antecedents are satisfied. Assert one of the disjuncts from the consequent. If it is a non-final disjunct and you can prove **absurd**, then try a different disjunct; otherwise, repeat 1.
3. Check for consistency, i.e., attempt to prove **absurd**.

The changes made here are very worthwhile because they focus on the inefficient forwards-chaining stage of model generation while leaving the efficient backwards-chaining stage untouched. While this revised algorithm does not constitute incremental inference, the effects of making inference more efficient within the discourse state do become increasingly apparent as the discourse progresses. This is because, as the discourse model grows, each call of **absurd** becomes increasingly time-consuming.

---

<sup>2</sup>Of course, we don't *know* that the disjunct is responsible, but this is the presumption as any utterance in context is expected to have a model. Thus the decision has been made to attempt to build such models more quickly at the expense of taking longer to recognise those atypical cases where no model is available.

The limitation of this revised algorithm is that, since it relies on exploring non-disjunctive complex sequents before disjunctive ones, it can only be combined with approaches to incremental inference which also share this feature, as discussed below.

## 5 Incremental Inference in Context

The results of forwards-chaining in one discourse state are recorded and carried forwards to the next discourse state. However, this does not prevent us repeating much of the work we have already carried out. It is only once we have checked that the antecedents of a complex sequent are satisfied that we may recognise that its consequent is also satisfied and so no further exploration of it is needed. With each successive state we are performing a comprehensive search of our complex sequents, each of which may be satisfied in increasingly many ways as the discourse progresses.

What is needed is a way of limiting the exploration of complex sequents to those which are triggered by new facts. However, possible triggers include, not just those facts explicitly contained in the utterance which constitute an easily identifiable set, but also new facts which follow in turn from these. Identifying the latter means switching to a forwards-chaining algorithm even for those facts provable from Horn sequents. As such facts are proved extremely efficiently through backwards-chaining, this necessarily means a loss in efficiency which must be more than compensated for by performance gains in processing the complex sequents.

There are two further complications. With a forwards-chaining algorithm, we lose control over the order in which sequents are explored. This means that we potentially lose any efficiency gains which can be made from having only very carefully placed consistency checks, as described in the section above. A different approach to controlling consistency-checking is needed. In the context of a forwards-chaining algorithm, the obvious move is to attempt to prove **absurd** only when it itself is triggered by some new fact. However, this is potentially a very dangerous move. Some facts, such as equalities, only ever trigger other facts (including **absurd**) indirectly. This means that we have to check consistency *every time an equality is asserted*. This is a potentially a very inefficient move, a solution to which will be presented below.

The other complication is that forwards-chaining, while it starts with new facts, can trigger old and/or irrelevant facts as well as new and/or relevant facts. For instance, consider updating Discourse State 1, as given

in Section 2, with the utterances *He died. Mary cried. She loved him.* Discourse State 4 will contain, amongst others, the fact that the time when Mary loved John preceded the speech time of utterance 4:

*ends\_before*(#288(4), #418)

This triggers the new fact that the speech time of Discourse State 3 precedes that of Discourse State 4:

*ends\_before*(#288(4), #288(3))

which in turn triggers a series of old and uninteresting facts, such as the following (which says that the speech time of Discourse State 2 in turn precedes that of Discourse State 3):

*ends\_before*(#288(3), #288(2))

Some new facts triggered also appear relatively uninteresting, including:

*ends\_before*(#288(4), #412)

This one says that the time associated with John's sleeping (the event described by utterance 1) preceded the speech time of utterance 4. It is important that we are able to prove this fact if we need it, and indeed we are able to do so through backwards-chaining whenever required. However, even though this is a new fact, its *relevance* to Discourse State 4 is not obvious. Indeed, if we included it in our models it would be arbitrarily placed in the new discourse state, 4, although if anything it seems more relevant to an old discourse state, 1. Therefore, arguably forwards-chaining should involve some cut-off point beyond which inferences for inclusion in the new discourse state are not triggered.

It is relatively straightforward to cut off forwards-chaining at an appropriate point. What we need is some criterion for determining whether a triggered fact is *relevant* to the current discourse state<sup>3</sup>. We already use Centering Theory for such purposes as reference resolution. This means that we have a list of forward-looking centres associated with each discourse state, and thus a readymade indicator of its focus of attention. When we are carrying out inference, the second stage of the discourse update step, we already have available to us the forward-looking centres of the *current* utterance. By checking whether a triggered fact incorporates any centred entities, we can determine its relevance as a fact and as a trigger. A fact like *ends\_before*(#288(3), #288(2)) which contains no entities mentioned or introduced by the current utterance will be considered irrelevant. By ignoring such facts we will also discount as irrelevant those facts which can only be

---

<sup>3</sup>This is not exactly the same as *new*. As has been shown, some new facts are not relevant. It is also the case that some relevant facts may be triggered by new facts although they are not themselves new.

triggered via irrelevant facts, including *ends\_before*(#288(4), #412).

## 5.1 Triggered Forwards Chaining

The revised algorithm which takes into account the above considerations uses the existing representation of Horn and complex sequents. This is important as we should be able to perform backwards-chaining proofs using Horn sequents, represented in Prolog, whenever required. What is also required to drive forwards chaining is a record of triggers. Thus whenever a fact is asserted, any corresponding triggers are also recorded<sup>4</sup>:

```
female(LABEL, X):-
    woman(LABEL, X).

trigger(woman(LABEL, X), female(LABEL, X)).
```

The heads of rules for **absurd** used to contain a single argument, the label. Now they are asserted with an extra argument which is a list consisting of all the variables in the rule's antecedents:

```
absurd(LABEL, [X, Y]):-
    >(LABEL, X, Y),
    >(LABEL, Y, X).
```

This extra argument enables the goal **absurd** to be instantiated by the fact which triggers it, a critical consideration from the viewpoint of efficiency.

The forwards-chaining algorithm operates by collecting a set of relevant facts for the discourse state and using these to trigger further relevant facts. The initial list of facts is derived directly from the logical form for the utterance. The following algorithm is applied to each fact in the list in turn:

1. Check for consistency by calling any **absurd** triggered by the fact.
2. Add the fact to the current discourse state.
3. Call any other rules triggered by the fact. Collect any facts generated which contain centred entities. Assert a Horn sequent corresponding to any such fact derived from a complex sequent. Test any non-final disjunct for consistency and, if necessary, retract and replace it with an alternative disjunct.

---

<sup>4</sup>The extra argument in the Prolog rules is a label used in labelled deduction (Gabbay, 1996). Space precludes a detailed description of the use of labels here.

4. Add the newly triggered facts to the list and repeat.

The results for this algorithm are rather mixed. It was compared with the modified version of the original algorithm described in the section above on the following examples:

1. *John is hiccuping.*
2. The mini-discourse *John slept. He died. Mary cried. She loved him.*

Example (1) was chosen as a processing-expensive utterance, as it involves deriving and resolving inconsistencies. Example (2) was chosen to evaluate the effect of incremental inference on processing times as the discourse progresses.

Performance on (1) was marginally degraded with the new algorithm - 5.28 seconds, as compared with 5.18 seconds. This is not surprising since, in processing the *first* utterance in a discourse, the forwards-chaining part of the original algorithm is fairly efficient, insofar as it is not wasting any time exploring disjuncts which have already been resolved. This means we are not gaining anything by stipulating that complex sequents must be triggered by the utterance. Added to this, the backwards-chaining part of the original algorithm is extremely efficient, and we are losing this efficiency<sup>5</sup>.

The mini-discourse in (2) is an example of where incremental inference is required, and performance in this case is indeed significantly improved. Processing times for the four utterances with the original algorithm (as optimised in the section above) are  $1.746 + 2.325 + 5.404 + 14.932$ . With the new algorithm, the times are reduced to  $1.401 + 1.761 + 2.089 + 5.878$ <sup>6</sup>. This improvement indicates that any loss in efficiency through moving to a purely forwards-chaining algorithm is more than compensated for by the exploration of complex sequents only when they are triggered by the current utterance. It should be noted that the use of centering to restrict forwards

---

<sup>5</sup>The reason the degradation is not more significant is that the original algorithm involves *two* passes through the backwards-chaining part of the algorithm, one to collect all the facts available before forwards chaining and one to collect all those provable after, with the contents of the discourse state being derived by comparing the two sets to isolate new facts. It is possible to significantly reduce the time to 3.491 seconds by using a single pass, with the constraints of centering being used to weed out irrelevant facts. However, this does result in a rather overinformative discourse state, repeating *every* fact known about *every* centred entity.

<sup>6</sup>Even if, as in the footnote above, the two passes through backwards chaining are replaced by a single pass using centering constraints, the processing times for the original algorithm are still significantly worse at:  $1.047 + 1.896 + 4.754 + 13.844$ .

chaining to facts concerning centred entities is absolutely critical to this improvement. Without the constraints of centering in place the new algorithm generates a lot of old and/or irrelevant facts and is significantly less efficient than the original algorithm.

Performance is still a lot worse for utterance 4 than for utterance 1. One reason for this is that, because of the way equalities are represented, it is not possible *safely* to ensure that every goal triggered is instantiated, and uninstantiated goals become more costly as the discourse progresses. This remains an obvious, but non-trivial, area for future improvement.

## 6 Conclusion

State-of-the-art inference engines are freely available to computational semanticists (Blackburn et al., 1999), so why should we concern ourselves with developing our own? If to you take a fairly efficient inference engine like SATCHMO, adapt it for intensional logic, and use it together with a set of meaning postulates to draw inferences from a single utterance, it will do so fairly efficiently. However, if you apply it to even a mini-discourse, i.e., several related utterances in succession, its performance will rapidly become degraded. This paper analysed the reasons for this by examining the differences between the task of constructing discourse models and the kinds of puzzles for which inference engines are typically designed and optimised. The obvious difference is that discourse models are built incrementally, with each discourse update step requiring a further application of the inference engine. However, there are also more subtle differences which shouldn't be neglected. There is a strong presumption that a model of an utterance will be found, and this means that it makes sense to use consistency-checking (which becomes increasingly costly with each discourse state) sparingly, only at those decision points where an inconsistent disjunct might be replaced with an alternative.

Two revisions to a SATCHMO-based inference engine (Ramsay, 2001) were implemented on the basis of the above considerations. The first was a simple change to the way consistency-checking was applied. This improved the efficiency of the forwards-chaining part of inference without affecting the efficient backwards-chaining part. However, it did nothing to alter the fact that each application of the inference engine remained wholesale rather than incremental. Implementing incremental inference required a more global change to the inference algorithm, although SATCHMO's use of Horn sequents wherever possible, translated directly into Prolog rules, was retained.

Incremental inference was implemented using a forwards-chaining algorithm whereby every fact generated in the discourse state was either directly or indirectly triggered by the explicit contents of the utterance. A further constraint was needed to ensure that only facts relevant to the current utterance were triggered, and this was implemented by rejecting any fact which did not mention an entity in the forward-looking centres of the current utterance. This is a controversial move as, while the centering constraint only directly weeds out old facts, it indirectly weeds out facts which may be new but are nevertheless considered irrelevant. This raises the following question: is the purpose of inference to give us as *complete* as possible a model of the consequences of an utterance in discourse, or as *relevant* as possible a model? Although this takes us out of the realm of inference as we normally understand it and into the realm of Mental Models (Johnson-Laird, 1983) and Relevance Theory (Sperber and Wilson, 1986), we would argue for the latter. Recent similarly motivated work on controlling inference in discourse (Kohlhase and Koller, 2000; Kohlhase and Koller, 2002) appears to reflect the same choice. This in turn raises a further question: is a relevant model to be preferred at the risk of inconsistency? As any possible inconsistency could only be discovered through exploring new facts which are irrelevant to the current discourse state, we would argue that *yes* may be the answer to this question as well.

## References

- Blackburn, P., Bos, J., Kohlhase, M., and de Nivelle, H. (1999). Inference and computational semantics. In Bunt, H. C. and Thijsse, E. G. C., editors, *3rd International Workshop on Computational Semantics*, pages 5–19, University of Tilburg.
- Cryan, M. and Ramsay, A. M. (1997). A normal form for Property Theory. In *Proceedings of the 14th International Conference on Automated Deduction (CADE-14)*, volume 1249 of *Lecture Notes in Artificial Intelligence*, pages 237–251, Berlin. Springer-Verlag.
- Gabbay, D. M. (1996). *Labelled Deductive Systems*. Oxford University Press, Oxford.
- Gaylard, H. and Ramsay, A. (2002). A unified theory of reference resolution. In *Proceedings of the 4th Discourse Anaphora and Anaphor Resolution Colloquium*, University of Lisbon. Edicoes Colibri.

- Grice, H. P. (1975). Logic and conversation. In Cole, P. and Morgan, J. L., editors, *Speech Acts*, pages 41–58. Academic Press, London.
- Grosz, B. J., Joshi, A. K., and Weinstein, S. (1995). Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.
- Grosz, B. J. and Sidner, C. L. (1986). Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204.
- Johnson-Laird, P. N. (1983). *Mental Models: towards a cognitive science of language, inference and consciousness*. Harvard University Press, Cambridge, Mass.
- Kohlhase, M. and Koller, A. (2000). Towards a tableaux machine for language understanding. In *Proceedings of ICOS-2*, Dagstuhl.
- Kohlhase, M. and Koller, A. (2002). Resource-adaptive model generation as a performance model. *Journal of Language and Computation*. To appear.
- Manthey, R. and Bry, F. (1988). Satchmo: a theorem prover in Prolog. In *Proceedings of the 9th International Conference on Automated Deduction (CADE-9)*, volume 310 of *Lecture Notes in Artificial Intelligence*, pages 415–434, Berlin. Springer-Verlag.
- Ramsay, A. M. (1995). A theorem prover for an intensional logic. *Journal of Automated Reasoning*, 14:237–255.
- Ramsay, A. M. (2001). Theorem proving for untyped constructive  $\lambda$ -calculus: implementation and application. *Logic Journal of the Interest Group in Pure and Applied Logics*, 9(1):89–106.
- Ramsay, A. M. and Seville, H. (2000). Models and discourse models. *Journal of Language and Computation*, 1(2):159–174.
- Seville, H. and Ramsay, A. (2000). Making sense of reference to the unfamiliar. In Kay, M., editor, *Proceedings of the 18th International Conference on Computational Linguistics (COLING-2000)*, pages 775–781, Universität des Saarlandes.
- Sperber, D. and Wilson, D. (1986). *Relevance*. Blackwell, Oxford.