

---

# Models and Discourse Models

Allan Ramsay and Helen Seville,  
*Centre for Computational Linguistics*  
*UMIST, PO Box 88, Manchester M60 1QD, UK*  
*Email: allan/heleng@ccl.umist.ac.uk*

*Keywords:* Model construction, update semantics, dynamic semantics, property theory

## Abstract

It is indeed true that ‘computational semantics has reached the stage where the exploration and development of inference is one of its most pressing tasks — and there’s a lot of interesting new work which takes inferential issues seriously’<sup>1</sup>. There are, however, a number of open issues about the exact role of inference within computational semantics. We will argue that using your inference engine for constructing models, rather than for drawing out consequences, may be a fruitful way forward.

## 1 What is inference for?

There has been a substantial move recently to take the role of inference seriously in computational semantics. For a long time there was very little difference between formal semantics and computational semantics: in both cases, the task was to construct some formal entity which captured some aspect of the meaning of a natural language utterance, and to make sure that the construction could be carried out on the basis of information that was explicitly present in the text (i.e. compositionally if possible, or using glue to sort out how the pieces went together if not).

This work is a necessary step on the way to developing a computational treatment of semantics, but it is not the whole story. What we really want when confronted with an utterance is a picture of how the speaker wanted us to change our world view. This notion is implicit in dynamic theories of semantics, notably in situation theory, where it is assumed that the ‘meaning’ of an utterance is something which can be used to change the context/common ground/discourse model/mutual knowledge/... A number of theories simplify this notion by assuming that the update step simply consists of anchoring any referring expressions and then adding the result to the current discourse state (e.g. [10, 13]), though [9] provides an early analysis which takes a rather more complex view of the update step. The development of effective theorem provers (stemming partly from improvements in theorem proving algorithms but also from the radical change in computing power over the last few years) makes it possible to revisit this issue, and to see whether we can make more interesting statements about the way the update step works.

We will argue for a particular view of the update step, namely that when you assimilate an utterance, your task is to construct a model that supports (i) the content

---

<sup>1</sup>From the call for papers for the *1st Workshop on Inference in Computational Semantics*.

of the utterance itself, (ii) your background encyclopædic knowledge of the world, and (iii) the content of the discourse so far. The key idea here is that you deal with utterances as they come in, updating your current view of the world by forcing it to include the current utterance, and that you then throw away the utterance itself. After you have heard an utterance and taken it on board, you no longer have access to the utterance itself: all you have left is the picture of the world that you created when assimilating it.

[25, 28] show how this approach makes it possible to obtain subtly different consequences of individual lexical and sub-lexical items by considering their interactions with other co-textual items. [29, 30] develop a theory of the relation between referential form and discourse structure using model construction as the basic engine for anchoring referring expressions. Section 2 provides an overview of the theorem prover, showing how it can be used for model construction and discussing some of the problems that arise when we extend it to deal with intensional logic. Section 3 provides examples of the results we obtain for a variety of input sentences.

## 2 Model construction

The inference engine we use for this work is an elaboration of [17]’s model generation theorem prover Satchmo. We will outline the basic operation of this theorem prover below: the crucial point for our purposes is that Satchmo can easily be seen as a tableau-based theorem prover. Tableau theorem provers are usually used for proving that some goal  $A$  follows from a set of assumptions  $\Delta$  by adding the negation  $\neg A$  of the goal to  $\Delta$  and showing that the resulting set has no models. They can, however, also be used for model construction, by adding  $A$  itself to  $\Delta$  and showing that the resulting tableau has an open branch. Such a branch constitutes a model, since it is an assignment of truth values to ground literals which supports the set  $\{A\} \cup \Delta$ .

There are, however, two problems with using Satchmo for our purposes:

1. The original presentation of Satchmo is extremely efficient when solving puzzles, i.e. problems where all the information that is present is required for the solution, and the task is to work out how to use it. It can, however, be made arbitrarily inefficient by including irrelevant information. [24] showed how to improve the basic algorithm’s performance when faced with irrelevant information, and [16] further elaborated these improvements.
2. We believe, along with [2] and [1], that a proper treatment of natural language semantics requires you to use a highly intensional logic. Such logics are notoriously intractable: property theory [31], which we use, is not recursively enumerable, and it seems unlikely that non-well-founded set theory, which underlies the [1] version of situation theory, is any better. [26] adapts Satchmo to work with property theory, but [3] show that it is impossible to provide a static normal form for property theory. As a consequence, many pruning strategies are no longer available.

Satchmo works by converting the problem to sequent form, and classifying sequents into Horn and disjunctive sequents, where a Horn sequent has a single literal as its consequent and a disjunctive one has a disjunction of literals as its consequent. Negations are treated as sequents with the absurd proposition  $\perp$  as their consequents.

There are two key observations:

1. If the antecedent of a sequent is true, then its consequent must be true. For disjunctive sequents, this means that at least one of the disjuncts must be true.
2. Horn sequents can be used in a Prolog-like backward chaining inference engine for proving their consequents. Indeed, they can be turned directly into Prolog clauses, with a sequent like  $P_1 \wedge P_2 \wedge \dots \wedge P_n \implies Q$  turning into the Prolog clause  $Q :- P_1, P_2, \dots, P_n$ . A sequent with  $\perp$  as its consequent, e.g.  $on(X, a) \implies \perp$  (which would arise from  $\neg \exists X on(X, a)$ ), will turn into **absurd**  $:- on(X, a)$ .

Satchmo exploits these two observations as follows. Suppose that we have a set of hypotheses  $\Delta$  and a target formula  $A$ . We convert  $\Delta \cup \{\neg A\}$  into a set of sequents (with the Horn ones turned directly into Prolog), and then proceed as follows:

1. Try to prove **absurd**, just using the Prolog clauses that arise from the Horn sequents. If you can, then clearly the set of sequents as a whole is inconsistent, since they include the Horn ones as a subset.
2. If not, try to find a disjunctive sequent whose antecedent is provable just from the Horn sequents/Prolog clauses but whose consequent is not currently provable. If you find such a disjunctive sequent, it is clear that at least one of its disjuncts must be true. So we add each disjunct in turn to our set of facts and Horn sequents and try to show that the result of this is inconsistent (either directly, by (1), or by repeated applications of (2)). If each disjunct leads to an inconsistent set, then we know that the original set was inconsistent, since at least one of them has to be true.

This algorithm solves a number of well-known puzzles extremely efficiently. Much of the efficiency arises from the way it switches between backward-chaining for Horn sequents and forward-chaining for disjunctive ones. For puzzles, where the disjunctive sequents are known to be relevant, this is a very effective strategy. For more general cases, however, it can be very inefficient. The problem is that each use of a disjunctive sequent with  $N$  disjuncts in the consequent spawns  $N$  new tasks. It may be that at least one of the disjuncts fails to help with either proving **absurd** or finding a disjunctive sequent. In this case, the extra work involved in exploring the  $N$  branches is entirely wasted.

[24, 16] deal with this by ensuring that disjunctive sequents only become available for splitting when either at least one disjunct [24] or every disjunct [16] has showed up as a goal in a failed backward chaining proof. It is easier to watch for single disjuncts becoming relevant, and more effective to watch for complete sets.

The move to property theory complicates the inference process considerably. What should we do with sequents which contain quantification over properties and propositions? Such items in the antecedent of a clause are easily dealt with: we decline to try to prove them until they are properly instantiated, at which point they can be subjected to perfectly standard operations. The situation is more awkward when they appear in the consequent. Consider, for instance, the rule that anything which is known is true:

$$\forall X \forall P (know(X, P) \rightarrow P)$$

This rule is often used in modal approaches to knowledge and belief, where it takes on the status of an axiom [11]. Such modal approaches run into problems associated with using possible worlds semantics, which mean that it is impossible to avoid assuming that everyone knows all the logical consequences of their basic belief sets (logical omniscience) and that people are unable to distinguish between logically equivalent propositions (logical blindness). By couching such rules as statements *within* the theory, rather than as axioms, it is possible to avoid some of these foundational problems. At the same time, they do make the operation of the inference engine rather harder.

Consider the given rule: it looks a bit like a Horn sequent – there is, after all, only one element on the right-hand side. This element is uninstantiated, of course, but does that matter?

Suppose we decide to treat it like a Horn sequent, and turn it into a bit of Prolog. It would look something like

```
P :- know(X, P).
```

This is illegal in Prolog. There are two reasons for this.

1. Prolog is supposedly based on first-order logic, where such rules are banned. They are banned from first-order logic largely because once you allow in quantification over propositions and properties, you are liable to run into the paradoxes of self-reference (such as the Liar Paradox). Most implementations of Prolog, however, provide plenty of meta-predicates which cannot be given coherent interpretations within first-order logic, so this cannot really be the reason for banning intentional operators.
2. Allowing in unrestricted intensional operators will cause the Prolog engine to go into infinite loops. Some such operators are, in fact, permitted: the use of variable subgoals is an example. However, if such a subgoal is not instantiated at the point when it is called the Prolog engine will throw a run-time error – not because trying to call an uninstantiated goal is meaningless or paradoxical, but because it will inevitably lead to an infinite loop. [26, 3] show how to avoid these problems. There is not space to recount this solution here: the critical point is that constructing a normal form is now a dynamic activity, which means that it is not possible to use optimisations that rely on static analysis of the problem.

We also need to be able to reason about equality. Much recent work on reasoning about equality relies on using rewrite rules to make sure that each constant is replaced by the canonical representative for the relevant equivalence class after each move that equates one item with another [8, 7]. Very roughly speaking, if you had a set of sequents like  $\{p(a) \implies q(a), \top \implies p(b)\}$  then if you discovered that  $a$  and  $b$  were equal you would rewrite your sequents to  $\{p(a) \implies q(a), \top \implies p(a)\}$ , from which you could infer  $q(a)$ .

There are two problems with this approach for the tasks that we want to address. The first is that the problems we are considering tend to have quite large rule sets, so that rewriting will tend to be a costly process. The second is that, just as with sequents with variable consequents, it is hard to know when it is a good idea to try to prove that two items are equal. Any time that you have a goal which has the same functor and arity as the head of some clause, but which fails to match it, the

possibility of proving that the items which failed to unify are in fact equal raises itself. We therefore choose to include sequents with equalities as their consequents with the set of forward rules. When such a rule is triggered, we use the result to update the current set of equivalence classes, and we use this dynamically when attempting to match subgoals with clause heads.

Allowing in sequents with uninstantiated literals makes it impossible to use any pruning strategies that depend on static analysis of the problem. [14]’s ‘pure literal deletion’, for instance, cannot be used in this context, since you might delete some sequent only to discover that one of the intensional rules has reinstated the relevant literal. Similarly, sequents that enable you to derive equalities can make it possible to unify two terms which at first sight did not match, so that again you might delete a sequent only to discover later that the relevant literal was not in fact pure. We therefore have to adapt such techniques so that their effects are only temporary: when an intensional operator introduces a literal into some sequent, we have to ‘impurify’ anything that we deleted using this rule, and likewise for the relevance checking operations discussed above.

With these adaptations we can use Satchmo for model construction. Each time Satchmo chooses some element of the consequent of a forward sequent, it is in essence extending a branch of a tableau. Proving that the current branch supports a proof of  $\perp$  is tantamount to showing that this branch is closed. In other words, if Satchmo gets to a point where there are no more forward sequents to explore, and yet no proof of  $\perp$  has emerged, then the choices it has made constitute a partial model. This can be fleshed out by looking for Horn sequents whose antecedents are provable and adding their consequents as well. The result is a minimal model, in the sense used for providing a semantics for circumscription [18]. We take it that the process of constructing such models can be illuminating when you are trying to interpret a natural language utterance.

### 3 Meanings and interpretations

What happens when you hear a sentence such as (1)?

(1) *I was reading a book.*

The words in this sentence are connected to various concepts and relations, and the way the words are organised tells you something about how those concepts and relations are to be linked up. It seems as though the major task you face when interpreting such a sentence is to construct some structure which captures the notions explicitly encoded in the text: to construct, in other words, a ‘meaning representation’.

$$\begin{aligned} \exists A :: \{ & \text{aspect}(\text{prog}, \text{ref}(\lambda B \text{centred}(B, 1) \wedge \text{now} > B), A) \} \\ \exists D :: \{ & \text{book}(D) \} \\ \forall E :: \{ & A.E \} \theta(E, \text{agent}, \text{ref}(\lambda F \text{centred}(F, 1) \wedge \text{speaker}(F))) \\ & \wedge \text{read}(E) \wedge \text{type}(E, \text{event}) \wedge \theta(E, \text{object}, D) \end{aligned}$$

FIG. 1. Logical form for (1) ‘*I was reading a book*’

Most work in computational semantics assumes that this is the key activity. The goal is to produce a formal paraphrase, in some appropriate logic, of the kind given

in Fig. 1. The paraphrase given here is obtained by fairly orthodox means, using an HPSG-like grammar to derive a structural analysis and then reading the interpretation off this structural analysis using a rather weak form of ‘glue’ [4] to combine the meanings of the constituents into a whole. Exactly how we do this is unimportant for the present paper. What matters now is, what should you do once you have such a paraphrase?

If you hear a sentence such as (1), you will form a mental picture of the scene. You will see me, sitting there with a book in my hand and my eyes focussed on its pages. The book will be open, and will be the right way up, and will have Roman characters on it. All sorts of things will spring into your mind. How does this happen?

Johnson-Laird [12] argues that when people perform reasoning tasks, the strategy that they employ involves constructing and inspecting ‘mental models’. The activity he describes is very reminiscent of the task performed by tableau theorem provers as they spawn and investigate new branches. Johnson-Laird suggests that people’s natural approach to problem solving involves exploring such models, which are constructed incrementally by adding new facts which are compatible with the problem statement. We suggest that this process underlies the ‘update step’ in discourse processing, and that it can be effectively mimicked by using a tableau-like theorem prover to construct a model that admits the formal paraphrase of the current utterance together with the hearer’s general world knowledge.

Consider the formal paraphrase in Fig. 1. This says that at some known time before now ( $ref(\lambda Bcentred(B, 1) \wedge now > B)$  – the time  $B$  which we have in mind and which precedes *now*) a set of events  $A$  each member of which was a reading event involving the speaker  $ref(\lambda Fcentred(F, 1) \wedge speaker(F))$  and some book was in progress. This analysis contains a number of debatable decisions:

- The referential terms correspond to pronouns and definite NPs – things which have to be anchored in the current situation in order for the utterance to make any sense.
- The fact that we treat the sentence as referring to a set of events rather than just a single event provides a handle on the kinds of problem for which [19] require ‘coercive’ operations.
- We assume that it makes sense to express the way that individuals participate in an event by drawing on a small number of ‘thematic roles’, rather than following [20, 21] and assigning each verb its own idiosyncratic set of roles. See [5] for a discussion of the merits and demerits of various approaches to thematic roles.

In order to decide on the merits and demerits of these choices, we need to do two things: (i) we need to spell out the consequences of each of them, and (ii) we need to encode these consequences in a way that is amenable to an inference engine. We spell out the consequences by writing ‘meaning postulates’ in some appropriate formal language. For reasons which we have elaborated elsewhere [25, 28], the formal language we choose is Turner’s property theory.

How should we make them amenable to an inference engine? If we wanted to use our inference engine for proving things, we would have to decide what we wanted to prove. For a given utterance, it is not reasonable to suppose that at the time when it is produced you are aware of all the questions that you might need to answer about it. So we can’t use the inference engine by anticipating all possible questions and

answering them (and if we did, we would waste a huge amount of effort, since many of them will eventually turn out to be irrelevant). We can't afford to carry the logical form along with us until the questions do get asked, since by that point the amount of information that we have will be overwhelming. We therefore prefer to use the inference engine *at the point when the utterance is assimilated*, but to acknowledge that, like a human listener, we may jump to the wrong conclusions from time to time<sup>2</sup>.

In order for the inference engine to do anything with our formal paraphrase, we have to have a body of background knowledge. Our background knowledge is encoded as a set of meaning postulates:

**MP: read(1)** Reading a specific book has a defined end point

$$\forall A :: \{read(A) \wedge \exists B \theta(A, object, B)\} telic(A)$$

**MP: read(2)** Reading can be seen as a mental activity. We employ the notion of a 'view' of an object to capture the fact that a single object can be seen in a number of different ways – that a book, for instance, can be seen both as a physical object made of paper and as a repository for ideas. This notion addresses many of the issues for which [23] introduces 'qualia': see [28] for a comparison between the two approaches.

$$\forall A :: \{read(A)\} \exists B view(A, B) \wedge B \text{ is mental}$$

**MP: read(3)** It can also be seen as a physical activity

$$\forall A :: \{read(A)\} \exists B view(A, B) \wedge B \text{ is concrete}$$

**MP: read(4)** The agent and object of a reading event are usually in the same place as the event itself

$$\forall A :: \{read(A)\} sharedagentloc(A) \wedge sharedobjectloc(A)$$

**MP: read(5)** Reading events take a perceptible amount of time

$$\forall A :: \{read(A)\} extended(A)$$

**MP: read(6)** If  $B$  is the agent of an event  $A$  whose agent should be in the same place as the event, and  $D$  and  $E$  are 'compatible ways of thinking about'  $A$  and  $B$ , then if  $F$  is the location of  $D$  then it is also the location of  $E$ .

$$\begin{aligned} \forall A \forall B :: & \{\theta(A, agent, B) \wedge sharedagentloc(A)\} \\ & \forall D :: \{view(A, D)\} \\ & \forall E :: \{view(B, E) \wedge D \ E\} \\ & \forall F :: \{location(D, F)\} location(E, F) \end{aligned}$$

**MP: prog(1)** If some set  $A$  of events is in the progressive aspect with respect to some time  $B$ , then  $A$  has at least one member  $C$  which started before  $B$  and at least one member  $F$  which ended after  $B$ , where  $C$  and  $F$  may be either the same or different.

---

<sup>2</sup>The fact that the inference engine is not (cannot be) a decision procedure means that we sometimes have to cut attempted proofs short. Consistency checking is an undecidable problem even for first-order logic: the extra computational complexity of property theory does not make things any better. We will therefore sometimes conclude that there is no proof of  $\perp$ , when in fact all that has happened is that we have not found such a proof. Under those circumstances, we will conclude that the current open branch of the tableau constitutes a model when in fact this is not the case, just as people will sometimes fail to spot that their beliefs are inconsistent.

$$\begin{aligned}
& \forall A \forall B \text{Aspect}(\text{prog}, B, A) \\
& \quad \rightarrow \exists C :: \{A.C\} \\
& \quad \quad \forall D :: \{\text{view}(C, D) \wedge D \text{ is interval}\} \\
& \quad \quad \quad \forall E :: \{\text{start}(E, D) \wedge E \text{ is instant}\} B > E \\
& \quad \quad \wedge \exists F :: \{A.F\} \\
& \quad \quad \quad \forall G :: \{\text{view}(F, G) \wedge G \text{ is interval}\} \\
& \quad \quad \quad \quad \forall H :: \{\text{end}(H, G) \wedge H \text{ is instant}\} B < H \\
& \quad \quad \quad \wedge C = F \vee \text{different}(C, F) \\
& \dots
\end{aligned}$$

If you don't have rules like these for all the terms that appear in your formal paraphrases, there will be nothing for your inference engine to work on. If you do have them, you can use them to explore the space of possible models. From the meaning postulates given above, and a number of others like them, we construct the model in Fig 2 for (1).

|                      |                               |                      |                          |
|----------------------|-------------------------------|----------------------|--------------------------|
| #338 < #358          | read(#354)                    | view(#354, #360)     | type(#358, instant)      |
| #352.#354            | sharedobjectloc(#354)         | view(#354, #361)     | end(#358, #356)          |
| type(#353, mental)   | sharedagentloc(#354)          | type(#355, mental)   | type(#359, solid)        |
| book(#353)           | telic(#354)                   | type(#356, interval) | type(#360, mental)       |
| envelope(#353, #355) | $\theta$ (#354, object, #353) | extended(#356)       | type(#361, concrete)     |
| view(#353, #359)     | $\theta$ (#354, agent, #340)  | type(#357, instant)  | 1 > #338                 |
| type(#354, event)    | view(#354, #356)              | start(#357, #356)    | aspect(prog, #338, #352) |

FIG. 2. Model for (1) ‘I was reading a book’

The only things in this model that are contained explicitly in the formal paraphrase are the past time #338, the speaker #340, the book #353 and the event set #352. Everything else emerged when the model was constructed: the meaning postulate describing the progressive aspect introduced two members of the event set, and then coalesced them into a single event #354 on the grounds that is indeed possible for a single extended event to have started before #338 and finished after it. Then all the different views of #354 were introduced, and the various temporal properties of this event were recorded (note that it is the *temporal* view of the event which is taken to be extended).

We will end this section with a second example:

(2) *An alleged murderer was caught in the park.*

$$\begin{aligned}
& \exists A :: \{\text{aspect}(\text{simple}, \text{ref}(\lambda B \text{centred}(B, 2) \wedge \text{now} > B), A)\} \\
& \quad \exists D :: \{\text{alleged}(D, \lambda E \text{murderer}(E))\} \\
& \quad \quad \forall F :: \{A.F\} \\
& \quad \quad \quad \theta(F, \text{object}, D) \\
& \quad \quad \quad \wedge \text{catch}(F) \wedge \text{type}(F, \text{event}) \wedge \text{in}(F, \text{ref}(\lambda G \text{park}(G)))
\end{aligned}$$

FIG. 3. Logical form for (2) ‘An alleged murderer was caught in the park’

We can do nothing with this until we have some appropriate meaning postulates. The most interesting issue in (2) concerns the notion of an ‘alleged murderer’. An alleged murderer is *not* a murderer who has been alleged. There is no reason, in fact, to

suppose that an alleged murderer is a murderer at all. All you can really conclude from the fact that X is an alleged murderer is that someone else has said that X is a murderer, which we capture with

$$\forall A \forall B :: \{alleged(A, B)\} \\ \exists C \exists D event(D) \wedge say(D) \wedge \theta(D, agent, C) \wedge \theta(D, object, B.A)$$

This meaning postulate draws upon the expressive power of property theory, which allows us to quantify over properties which might be ascribed to individuals, and upon the fact that we have chosen to treat adjectives as two-place relations between individuals and descriptions. Without this we would be unable to provide a satisfactory analysis of non-factive adjectives such as ‘*alleged*’ or ‘*fake*’, which comment on the truth or falsity of the description encoded by the modified  $\bar{N}$ . There are numerous other examples of places in natural language where some part of an utterance says something about the veracity of some other part, and in all such cases you need an intensional logic if you want to provide a coherent account of what is going on.

An abridged copy of the model that results from combining the MP for ‘*alleged*’ with the logical form for (2) is shown in Fig. 4.

|   |                                       |   |
|---|---------------------------------------|---|
| <i>alleged</i> (#512, $\lambda A$ murderer( <i>A</i> )) | <i>type</i> (#515, <i>event</i> )     | <i>type</i> (#527, <i>solid</i> )         |
| <i>event</i> (#513)                                     | <i>catch</i> (#515)                   | <i>park</i> (#527)                        |
| <i>say</i> (#513)                                       | <i>in</i> (#515, #527)                | <i>accommodated</i> ( <i>park</i> (#527)) |
| $\theta$ (#513, <i>agent</i> , #514)                    | $\theta$ (#515, <i>object</i> , #512) |   |
| $\theta$ (#513, <i>object</i> , murderer(#512))         |                                       |   |

FIG. 4. Model for (2) ‘*An alleged murderer was caught in the park*’

(4) contains a catching event #515 whose object is #512. #512 is an alleged murderer, and as a consequence the model contains a saying event #513 whose object is the proposition *murderer*(#512). It is notable that the meaning postulate for ‘*alleged*’ required us to quantify over the property *B* and then apply this to the individual *A*, and that the result of this is to leave a proposition as the argument of a relation – moves that are not available unless you use some intensional logic such as property theory. It is also worth noting that the model contains an individual, #527, which is classified as a park, but that we record the fact that this individual is present only because we *accommodated* the referring expression ‘*the park*’.

## 4 Models, ‘mental models’, and discourse

The model constructed above for (1) contains a considerable amount of information. We believe that this is both correct and inevitable, and that the major flaw in what we have done so far is that the background knowledge that we draw upon for such models is only a tiny part of what is needed. The details of each specific rule are debatable: is our treatment of tense and aspect reasonable, do we have a good set of thematic roles, are we right to classify sleeping as an extended homegeneous state and hiccuping as an instantaneous atelic action, should a non-factive adjective be treated as a relation between an individual and a property, and so on? But there is no doubt that particular configurations of aspect markers and aktionsarts do evoke

interpretations which include multiple events, and there is no doubt that there is something odd about adjectives like ‘fake’ and ‘alleged’. When people assimilate language, they can bring a huge amount of world knowledge to bear — the dozen or so facts that we have recorded about reading events are just scratching at the surface for this one word, and any serious NLP system will have to cope with tens of thousands of words. We currently have just over 200 meaning postulates. Some of these are very general principles, such as the general rules for partial orders:

**MP: partial order** Partial orders are transitive<sup>3</sup> and irreflexive

$$\begin{aligned} &\forall A\forall B\forall C(A < B \wedge B < C \rightarrow A < C) \\ &\forall A\neg(A < A) \end{aligned}$$

Others are rather more specific – there is not room to present any more here, but our current rule set can be viewed at <http://ubatuba.ccl.umist.ac.uk>. Clearly we need a large number of rules. The 200 we have are quite a good test for the theorem prover, and serve to illustrate the kind of thing that can be done taking this approach. We would like to have 100000 rules, and we cannot promise that the performance of the inference engine will remain acceptable if and when we get such a rule set. Nonetheless, we believe that even these initial attempts at fleshing out what the various lexical and sub-lexical items mean are very illuminating.

The crucial point is that by constructing a model at the point when the utterance is assimilated we can get a detailed picture of the world without asking ourselves a large number of specific questions. Consider (2). What questions would you ask in order to get at the information embodied in Fig. 4? Just looking at the paraphrase in Fig. 3, you would be unlikely to ask whether someone had said that the person who had been caught was a murderer. This is implicit in what ‘alleged’ means, but unless you went and looked at the meaning postulates you could hardly guess it. What questions would you ask about the size of the event set in each of (1) and (2)?

By constructing models, we simply let these facts emerge. The model construction process strongly resembles the way Johnson-Laird believes that people make up mental models when asked to perform abstract reasoning tasks: we try to find a set of assignments of N-tuples of individuals to properties and relations which will make our set of beliefs true. We are not making any strong claims about the ‘psychological reality’ of the algorithm we use. It does, however, seem to be a sensible way to integrate the content of an utterance into the context in which it is uttered.

One advantage of proceeding this way is that it provides a simple mechanism for dealing with referring expressions. If an utterance contains a referring expression, such as ‘the park’ in (2), then what we have to do is to show that there is exactly one item for which the current situation supports the claim that this item satisfies the specified property<sup>4</sup>. A number of awkward points, however, remain.

1. As we proceed through a discourse, the model we grow has to incorporate each utterance that we encounter. This can lead to considerable redundancy, since we will record (i) facts that are explicitly mentioned in the utterance, (ii) facts

---

<sup>3</sup>The rule for this clearly runs the risk of circularity. We include a certain amount of loop-checking in the theorem prover, and although we cannot guarantee to catch all potential loops, ones like this do not in fact cause any problems.

<sup>4</sup>See [27] for a detailed discussion of what to do with ‘donkey sentences’ and other cases where referring expressions occur inside hypothetical sentences.

that emerged as we explored the current open branch, and (iii) *facts which are implicit in the Horn sequents that arise from our meaning postulates*. If we keep these implicit facts in our model, subsequent inference steps can involve multiple, redundant, proofs of the same result. It is therefore important to have the implicit facts available for inspection, but not to use them for subsequent model building.

2. If, as a consequence of the above remarks, we decide that the model we obtain should not include facts that can be derived on the basis of Horn sequents, we face a problem with ‘bridging references’. It would be perfectly reasonable to follow (2) with *‘His accuser had told the police to look there for him’*. If we constructed the full model, with all the implicit facts, then we would indeed have a referent for the NP *‘his accuser’* (in the example, it would be #514, the agent of the saying event). But if we only keep the facts obtained from the utterance itself and the disjunctive sequents, this individual will not be available. When we want to anchor a definite NP such as *‘the park’*, we try to show that the current discourse state supports a proof that there is a park. When we get a proof, we may be able to extract a specific individual from it: for bridging references, this individual will be marked as being dependent on some entity which is explicitly present in the discourse, but the individual itself need not be visible in the current discourse state.
3. It frequently happens that a given goal has a number of alternative proofs. This can arise when we discover that two items, say X and Y, are equal, since then anything we know about either X or Y becomes available. It can also arise simply because we have alternative descriptions of the same item – if we have someone referred to once as a father and another time as a husband, then any attempt to prove that they are male can use either the rule that all husbands are male or the rule that all fathers are. The problem of alternative proofs of the same fact is endemic in the use of inference engines when faced with extended discourses. We deal with it by checking whether our current goal is ground. If it is, we assume that if we find a proof of it, there is no need to look for alternative solutions. Thus a rule like  $\forall X(\text{bird}(X) \rightarrow \text{fly}(X))$  becomes <sup>5</sup>

```
fly(A, B) :-
    testground([B], C),
    bird(A, B),
    (C==ground -> !; true).
```

This says that you can prove that B can fly if you can prove that it is a bird; and that if you know what B is then there is no point in trying to find any other solutions.

The approach described in this paper has been implemented, and may be tested at <http://ubatuba.ccl.umist.ac.uk>. It’s not guaranteed, and there are considerable overheads in running it over the web, so it may be slower than it should be. The critical question is: how much slower will it get as we scale it up to cope with thousands of meaning postulates rather than hundreds? To answer this, we have to do two things: (i) we have to acquire thousands of meaning postulates, and (ii) we have to

---

<sup>5</sup>The first argument in the Prolog version of the rule is a ‘label’ [6], i.e. somewhere to keep non-logical information about the proof. We use it for a number of purposes, notably for keeping a picture of the calling stack in order to perform loop-checking.

keep working on the inference engine. Acquiring thousands of meaning postulates is an extremely time-consuming and labour intensive activity, as the CYC project [15] has shown. We are trying to develop a substantial set of meaning postulates in the domain of economics, as part of a computer-aided language learning projects where we will use the model construction process to compare what the learner thinks something means with what it really means [22]. In order to maintain the performance of the inference engine as our knowledge base gets bigger, we need to reinstate the improvements to Satchmo introduced by [16, 24]. The problem is that these, like almost any pruning strategies, rely on a static analysis of the problem. Such a static analysis is inappropriate for intensional logics like property theory, since literals that are absent when you perform the analysis may be introduced by the intensional operators. We therefore need to develop dynamic versions of these optimisations.

# Bibliography

- [1] J Barwise and J Perry. *Situations and Attitudes*. Bradford Books, Cambridge, MA, 1983.
- [2] G Chierchia and R Turner. Semantics and property theory. *Linguistics and Philosophy*, 11(3), 1987.
- [3] M Cryan and A M Ramsay. 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, 1997. Springer-Verlag.
- [4] M Dalrymple, J Lamping, F C N Pereira, and V Saraswat. A deductive account of quantification in LFG. In M Kanazawa, C Piñón, and H de Swart, editors, *Quantifiers, deduction and context*, pages 33–58, 1996.
- [5] D R Dowty. On the semantic content of the notion of ‘thematic role’. In G Chierchia, B H Partee, and R Turner, editors, *Properties, Types and Meaning II: Semantic Issues*, pages 69–130, Dordrecht, 1989. Kluwer Academic Press.
- [6] D M Gabbay. Labelled deductive systems. Technical report, Dept. of Computing, Imperial College, 1989.
- [7] J Gallier, P Narendran, D Plaisted, S Raatz, and W Snyder. An algorithm for finding canonical sets of ground rewrite rules in polynomial time. *Journal of the ACM*, 40(1):1–16, January 1993.
- [8] J Gallier, P Narendran, S Raatz, and W Snyder. Theorem proving using equational matings and rigid  $E$ -unification. *Journal of the ACM*, 39(2):377–429, April 1992.
- [9] G Gazdar. *Pragmatics: Implicature, Presupposition and Logical Form*. Academic Press, New York, 1979.
- [10] J Groenendijk and M Stokhof. Dynamic predicate logic. *Linguistics and Philosophy*, 14:39–100, 1991.
- [11] J Hintikka. *Knowledge and Belief: an Introduction to the Two Notions*. Cornell University Press, New York, 1962.
- [12] P N Johnson-Laird. *Mental Models: towards a cognitive science of language, inference and consciousness*. Harvard University Press, Cambridge, Mass., 1983.
- [13] H Kamp. A theory of truth and semantic representation. In J A G Groenendijk, T M V Janssen, and M B J Stokhof, editors, *Formal Methods in the Study of Language*, pages 277–322, Dordrecht, 1984. Foris Publications.
- [14] R Kowalski. A proof procedure using connection graphs. *JACM*, 22(4):572–595, 1975.
- [15] D B Lenat and R V Guha. *Building large scale knowledge based systems*. Addison Wesley, Reading, Mass., 1990.
- [16] D W Loveland. Near-horn Prolog and beyond. *Journal of Automated Reasoning*, 7:1–26, 1991.
- [17] R Manthey and F Bry. Satchmo: a theorem prover in Prolog. In R Lusk and R Overbeek, editors, *Proceedings of the 9th International Conference on Automated Deduction (CADE-9)*, volume 310 of *Lecture Notes in Artificial Intelligence*, pages 415–434, Berlin, 1988. Springer-Verlag.
- [18] J McCarthy. Circumscription: a form of non-monotonic reasoning. *Artificial Intelligence*, 13:27–39, 1980.
- [19] M Moëns and M Steedman. Temporal ontology and temporal reference. *Computational Linguistics*, 14(2):15–28, 1988.
- [20] C J Pollard and I A Sag. *An Information Based Approach to Syntax and Semantics: Vol 1 Fundamentals*. CSLI lecture notes 13, Chicago University Press, Chicago, 1988.
- [21] C J Pollard and I A Sag. *Head-driven Phrase Structure Grammar*. Chicago University Press, Chicago, 1994.

- [22] The LARFLAST project. LARFLAST: LeARning Foreign LAnguage Scientific Terminology. Technical report, <http://www-t.fmi.uni-sofia.bg/larflast>, 1999.
- [23] J Pustejovsky. The generative lexicon. *Computational Linguistics*, 17(4):409–441, 1991.
- [24] A M Ramsay. Generating relevant models. *Journal of Automated Reasoning*, 7:359–368, 1991.
- [25] A M Ramsay. The co-operative lexicon. In H C Bunt, editor, *1st International Workshop on Computational Semantics I*, pages 171–180, University of Tilburg, 1994.
- [26] A M Ramsay. A theorem prover for an intensional logic. *Journal of Automated Reasoning*, 14:237–255, 1995.
- [27] A M Ramsay. Does it make any sense? updating = consistency checking. In K Turner, editor, *The Semantics-Pragmatics Interface from Different Points of View*, London and Amsterdam, 1999. Elsevier.
- [28] A M Ramsay. Weak lexical semantics and multiple views. In H C Bunt and E G C Thijsse, editors, *3rd International Workshop on Computational Semantics*, pages 205–218, University of Tilburg, 1999.
- [29] H L Seville. Experiments with discourse structure. In H C Bunt and E G C Thijsse, editors, *3rd International Workshop on Computational Semantics*, pages 233–247, University of Tilburg, 1999.
- [30] H L Seville and A M Ramsay. Reference-based discourse structure for reference resolution. In *ACL Workshop on Discourse/Dialogue Structure and Reference*, pages 90–99, University of Maryland, June 1999.
- [31] R Turner. A theory of properties. *Journal of Symbolic Logic*, 52(2):455–472, 1987.