Using a Computational Situation Theoretic Language to investigate Contemporary Semantic Theories

Alan W Black
Centre for Cognitive Science, University of Edinburgh
2 Buccleuch Place, Edinburgh EH1 9NW
awb@ed.ac.uk

Abstract of talk given at
Dagstuhl Workshop on Semantic Formalisms

DRAFT – March 6,1993

Abstract

One of the problems with contemporary semantic theories of natural language, such as Situation Semantics, Discourse Representation Theory and Dynamic Semantics, is that although they address similar semantic phenomena their widely differing syntax and semantics make it difficult to compare their treatments of similar natural language semantic phenomena. It would be useful to describe these different semantic theories in a common framework so they could be more easily compared, and in the long run allow cross-pollination of their various treatments.

In this extended abstract I will give a brief overview of a language called ASTL. ASTL is based on fundamental aspects of situation theory. It is designed to be used as a meta-language for describing aspects of natural language semantic theories, offering a common framework for the description of semantic theories. Because ASTL has an implementation, descriptions in ASTL can be used directly to derive semantic translations of the theories they describe. Also because ASTL is given a situation theoretic semantics it offers a semantics for any description written in it, which allows closer comparison of treatments in a theory.

Apart from the advantages of close comparison of theories (e.g. DRT and Dynamic Semantics) that ASTL allows, we will also discuss what makes ASTL suitable as a framework and how it compares with other possible semantic meta-theories (e.g. Prolog, feature systems, etc.).

Acknowledgements: This work was funded by SERC and DYANA-2.

1 Motivation

Although, in the field of computational semantics, we have the same ultimate goal of providing a computational treatment of natural language semantics, many different theories have been proposed as possible solutions. These theories although often addressing the same semantic phenomena may have apparently quite different solutions. The difference in syntax and semantics of theories can be clearly illustrated just by looking at three semantic translations of the same simple sentence. Given the simple sentence *"A man walks"* three contemporary theories would give something like the following translations.

Situation Semantics	$S \models \ll man, X; 1 \gg$
	$S \models \ll walk, X; 1 \gg$
Dynamic Logic	$\exists x\ [man(x)] \wedge walk(x)$
DRT	$\begin{array}{\|l\|}\hline \texttt{X} \\ \hline \texttt{man(X)} \\ \texttt{walk(X)} \\ \hline \end{array}$

Note that only the dynamic logic representation has an explicit existential quantifier but even it differs from standard first order logic. The semantics of dynamic logic is such that the x in $walk(x)$ actually does fall within the scope of the preceding existential.

Given the above three apparently different treatments it would be useful if we could offer a framework in which they could be described such that a close comparison could more easily be made. This is exactly the work that is more fully presented in [Black 93].

There is an analogy here with what happened around ten years ago in the field of computational syntax. PATR-II ([Shieber 84]) offered a language in which contemporary syntactic theories could be specified. PATR-II descriptions could then be directly executed. PATR-II was not proposed as a syntactic theory itself but only as a mechanism in which theories could be described. Even though syntactic theories were never fully defined in PATR-II, descriptions of key aspects of theories lead to a greater understanding of them and later theories of natural language syntax have benefited from the close comparison that was possible in PATR-II.

Here we will give a brief overview of the computational situation theoretic language ASTL which is designed to allow executable specifications of aspects of natural language semantic theories such as Situation Semantics, Discourse Representation Theory and Dynamic Logic. The specification of a theory is sufficient to allow translations to be given to natural language utterances. Furthermore because such descriptions are in the same framework a much closer comparison of the semantic treatments is made possible.

2 ASTL

ASTL is designed as a computational language which will act in some way like a meta-language suitable for describing aspects of natural language semantic theories. ASTL is based on fundamental aspects of situation theory and is given a formal syntax and semantics within a situation theoretic model. ASTL offers a representation for: individuals, parameters, variables, relations, facts, situations, situation types and constraints. In addition to these static objects a set of inference rules (and interpreter) is provided in order that new propositions may be derived from a basic description of situations and constraints. The main purpose of the language is to allow a semantic theory to be described, such that given an utterance it is possible to derive its semantic translation.

Although language processing (particularly deriving semantic translations from utterances) is an important aspect of ASTL, we do not want to exclude the possibility of using the translations derived from the description of a theory within ASTL for more general inference. Although that has not been exploited in [Black 93] it does seem to be a reasonable possibility.

Rather than present the formal aspects of the language, the following simple example will help illustrate the basic operation of the language.

```
Individuals   {h,t}
Relations     {happy/1, smiles/1}
Parameters    {S}
Variables     {*S, *T, *Y}
Situations
  ;; Basic situations -- i.e. who smiles where
  (SIT1 :: [S ! S != <<smiles,h,1>>
                S != <<smiles,t,1>>]
   SIT2 :: [S ! S != <<smiles,t,1>>] )
Constraints
  ;; if they smile they are happy
  *S : [S ! S != <<happy,*Y,1>>]
    <=
     *S : [S ! S != <<smiles,*Y,1>>].
```

Variables in ASTL are conventionally written as atoms prefixed with `*` (although the initial declarations actually define which atoms are variables, individuals, etc.). In the above ASTL description we have defined two basic situations, (`SIT1` and `SIT2`). In `SIT1` we know two facts, that `h` ("Hanako") smiles and that `t` ("Taro") smiles while in `SIT2` we only know that `t` ("Taro") smiles. In general constraints, which are between situations, state that in all ways that the right hand side of the constraint is true the left hand side is true also. So in this particular example we are stating that if something smiles in a situation then it is also happy, in that same situation. The variable `*S` at each side of the `<=` operator ensures that we are talking about the very same situation.

Given the above simple description we can ask simple queries about the system of situations and constraint. For example we can ask.

```
astl> query *T:[S ! S != <<happy,t,1>>].
```

This can be glossed as: is there a situation in which `t` ("Taro") is happy? The implementation rather than simply answering "yes" or "unknown" will display any situations for which the query proposition is true thus the answer will be given as

SIT1	SIT2
happy(h) smiles(h) happy(t) smiles(t)	happy(t) smiles(t)

The output in ASTL is typically shown as boxes based on the situation theoretic notations of EKN ([Barwise & Cooper 93]). This box notation which is different from ASTL's basic linear form gives a much better display of what a situation supports, especially when a situation supports facts that have situations as arguments.

A major aspect which distinguishes ASTL from a simple logic programming language is that ASTL treats situations as first class objects (as is the case in situation theory). Situations can be used as arguments to facts. Self-reference is not a problem. The use of situations as objects allows inference to be local to situations. It also allows situations not mentioned in the initial declaration to be derived thus offering a powerful descriptive language.

Using ideas from Situation Theoretic Grammar (STG) [Cooper 89] we can use the basic aspects of ASTL to offer a framework for natural language syntax and hence lead to a method for describing the construction of natural language semantic translations for utterances. If we take the general view that the utterance of a piece of natural language can be represented by a situation we can easily use ASTL to represent the syntax of utterances. More specifically when I utter the noun phrase *"Hanako"* we can talk about the situation in which I uttered it. It has a time and a place. We can also attribute various syntactic properties to that situation. For example

"*Hanako*" $\leadsto$

```
Sit123

cat(SIT123,ProperNoun)
use_of(SIT123,"Hanako")
```

Given this treatment of utterances if such a situation is "next to" one in which I say *"walks"* we can infer that there is also a sentence situation covering the two words. In ASTL terms we can capture this with a constraint of the form

```
*S:[S ! S != <<cat,S,Sentence,1>>
        S != <<start,S,*Start,1>>
        S != <<end,S,*End,1>>]
<=
    *NP:[NP ! NP != <<cat,NP,NounPhrase,1>>
              NP != <<start,NP,*Start,1>>
              NP != <<end,NP,*Mid,1>>],
    *VP:[VP ! VP != <<cat,VP,VerbPhrase,1>>
              VP != <<start,NP,*Mid,1>>
              VP != <<end,NP,*End,1>>].
```

Because one of ASTL's major uses is in language processing we have included a special form of constraint specifically to deal with grammar rules, and hence the start and end point need not be made explicit in a special grammar rule form of constraint. However this special form may be completely defined in terms of the basic ASTL constraints—this is analogous to the relationship between Prolog's Definite Clause Grammar (DCG) rules and basic Prolog rules.

3 Using ASTL for describing theories

As the ultimate goal of computational semantic theories is to provide a computational treatment of the semantics of natural language it seems useful if not necessary that theories are specified in enough detail so that they can actually be used. Descriptions in ASTL are intended to be executable specifications of theories. The idea being that a description contains a formalisation of a semantic theory that is specific enough to allow the construction of semantic translations of natural language utterances.

In [Black 93] three such descriptions are given: Situation Theoretic Grammar [Cooper 89], Discourse Representation Theory [Kamp 81], and a form of dynamic semantics (based on DPL [Groenendijk & Stokhof 91]). Each of these descriptions are given with respect to the same language fragment which is sufficient for simple sentences and discourses including inter-sentential anaphora and donkey anaphora.

It should be added that ASTL is not a general implementation language for large natural language processing systems. It may develop that way in the future but the current implementation is only really suitable for smaller investigations of contemporary semantic theories. The present work has concentrated on both the representation of the theories' semantic translation and the construction of that form. As yet little work has been done on drawing inferences from the generated semantic translation of utterances, although ASTL does seem to have the basic capabilities to act in that capacity.

One possible representation for DRSs in ASTL is as parametric situation types. That is a DRS is effectively the type of a situation where it is true—with respect to a set of anchors for each free parameter (i.e. discourse marker), and of course appropriate conditions for quantifiers. DRSs are threaded through the basic syntactic structure in a way similar to the threading in the DRT description in [Johnson & Klein 86]. Given a set of words that are to be uttered and the description of DRT in ASTL the implementation can derive the syntactic structure of the utterance as the DRSs at each stage in the discourse. The outgoing DRSs for *"a man walks"* is

```
  P1
P1

 man(X)
 walk(X)
```

Notably there is no explicit domain in the representation of the DRS. A useful consequence of the ASTL representation of DRSs is that it is a (parametric) situation type and has a natural interpretation within the model. This shows that key aspects of DRT can be encoded within a situation theoretic model and that the encoding is in no way contrived.

4 Properties of ASTL

ASTL has sufficient expressive power for describing at least the core aspects of the three semantic theories mentioned above. It is interesting to step back and try to identify which properties of ASTL make it suitable as a meta-language for the formal specification of executable specifications of semantic theories. We can identify three main aspects:

The ability to represent complex structured objects and allow general relations between them.

The ability to have constraints between general objects and draw inferences from them.

A mechanism for reasoning about "variables" in the object language (as distinct from variables in the meta-language) and be able to describe binding mechanisms for these object language variables.

There is perhaps another property which is discussed as a possible extension to ASTL which is not currently included. That property is some form of abstraction, application and reduction which seems to be a basic property of some natural language semantic theories. However simple aspects of this can be modelled in the current version of ASTL.

With respect to situation theory the above three properties are core aspects of the theory. We can identify each of the properties with well accepted key parts of situation theory:

- situation, abstractions and types
- constraints
- parameters and anchoring

However we do not say that situation theory is therefore the only possible base for a general semantic meta-language. The above aspects are also found in other frameworks. It seems quite possible to define an attribute logic (of the general form as described in, for example [Johnson 88]) which has exactly those properties. Feature logics are very expressive and a wide selection of constructs are available which can be included in a logic. But arguably we could say that even if such a logic were defined it seems reasonable to claim that any implementation of that logic would equally be an implementation of ASTL as it would have exactly those properties which are fundamental to ASTL. We could also build such properties within Prolog. Although Prolog has only one "situation" it is easy to encode structured objects like situations in Prolog in fact some other logic programming languages already do this. Another area where we might also find such properties is the framework of object-oriented programming currently popular in computer science. The conclusion we wish to draw from this is that the essential properties of ASTL as stated above are not uncommon in other frameworks but as they are fundamental properties of situation theory it seems not unreasonable to present them within a situation theoretic framework. Also because we are working in the area of computational semantics it seems good that we can find the abstract properties we require in an already existing semantic theory.

5 Conclusion

ASTL offers both a method for formalisation of key aspects of contemporary semantic theories and because ASTL is a computational systems the formalisations in ASTL are directly executable. Once theories have been described in ASTL much closer comparisons can be made. The descriptions of DRT and dynamic semantics in ASTL given in [Black 93] seems to show that these theories differ only at some low level information content of expressions rather than any predictive or functional level. Also once descriptions are given in the same framework we can start to re-use treatments in one theory within another (e.g. adding a DRT like treatment of pronouns to a simple situation semantic treatment of language). This second aspect depends on there being at least some basic commonality between the theories but as we are interested in semantic theories that are describing similar phenomena this will often be the case.

In conclusion ASTL is a simple tool which does seem to help solve our original problem—how can we compare contemporary theories when they are presented with such different syntax and semantics. Although there is lots more to be done, ASTL seems to be a good start.

References

[Barwise & Cooper 93] J. Barwise and R. Cooper. Extended Kamp Notation: a graphical notation for situation theory. In *Situation Theory and its Applications, III*, CSLI Lecture Notes. Chicago University Press, forthcoming 1993.

[Black 93] A.W. Black. *A Situation Theoretic approach to computational semantics.* Unpublished PhD thesis, University of Edinburgh, Edinburgh, UK., 1993.

[Cooper 89] R. Cooper. Information and grammar. Technical Report RP No. 438, Dept of Artificial Intelligence, University of Edinburgh, 1989. Also to appear in J. Wedekind, ed. *Proceedings of the Titisee Conference on Unification and Grammar.*

[Groenendijk & Stokhof 91] J. Groenendijk and M. Stokhof. Dynamic Predicate Logic. *Linguistics and Philosophy*, 14:39–100, 1991.

[Johnson & Klein 86] M. Johnson and E. Klein. Discourse, anaphora and parsing. In *Proceedings of the 11th International Conference on Computational Linguistics*, pages 669–675, Bonn, West Germany, 1986.

[Johnson 88] M. Johnson. *Attribute-value logic and the theory of grammar.* CSLI Lecture Notes Number 16. Chicago University Press, 1988.

[Kamp 81] H. Kamp. A theory of truth and semantic representation. In J. Groenendijk, T. Janssen, and M. Stokhof, editors, *Formal Methods in the Study of Language.* Mathematical Center, Amsterdam, 1981.

[Shieber 84] S. Shieber. The design of a computer language for linguistic information. In *Proceedings of the 10th*

International Conference on Computational Linguistics/22nd Annual Conference of the Association for Computational Linguistics, pages 362–366, Stanford University, California, 1984.