

DiaP

A Dialogue Parser using Segmented Discourse Representation Theory

Nicholas Yates

Université Libre de Bruxelles & Ecole Royale Militaire

Brussels, Belgium

nyates@ulb.ac.be

1 Introduction

Segmented Discourse Representation Theory or SDRT (Asher & Lascarides, 2003) has gained much attention in recent years as one of the most formally precise and linguistically grounded accounts of discourse and dialogue interpretation. Using underspecified dynamic semantics along with a special logic for extracting information, SDRT builds a structured representation of a discourse, governed by Rhetorical Relations (or speech acts). The highly complex interaction of modular levels, using different logics, makes the thorough empirical testing of any theory quite difficult without the assistance of a computer. I present here an ongoing work on implementing a discourse and dialogue parser using SDRT. The program is written in Prolog and is eventually intended to become a discourse and dialogue theory development environment.

2 Overview of the system

One can think of SDRT as a discourse/dialogue grammar theory, having a set of lexical items (*i.e.* rhetorical relations) and a set of rules (*i.e.* glue axioms and rules for inferring rhetorical relations). It is in this spirit that DiaP has been conceived, like development tools for syntactic theories (*e.g.* LKB, ALE, ...)

On the one hand, there is a semantic ontology for the content-level information, an inventory of relations and a set of general axioms and rules for inferring these relations. These are generally specific to the domain under investigation. They form the ‘grammar’ *per se*.

On the other hand, there are a number of modules, using various kinds of formalisms, interacting to analyse a sentence, extract the discourse-level information (Glue Logic) from it, compute its attachment and relations and complete any possible semantic underspecification. These make up the core of the program or the general theory itself.

2.1 Defining a dialogue grammar

In this part I will briefly describe the source files that are provided by the user of DiaP.

The general ontology for the language in use is expressed in the Typed Feature Structure formalism (TFS), in an HPSG style. We need a rich formalism to define the semantic types of expressions, because we use information like indices, event structure and subcategorisation to resolve possible disambiguations, compatibility issues while processing anaphora resolution, implicit arguments, ... Eventually, we might want to use directly the semantic output of a syntactic parser (I plan on using LKB (Copestake, 1999) at some future point) but this is not necessary since DiaP has its own AVM engine and can assign an AVM representation to basic expressions from some Underspecified Semantics description like MRS, (Copestake et al., 1999). In this case, the user is required to input a full type description and vocabulary. This is done using the standard TDL (Type Description Language) syntax.

The SDRT-specific parts include a rhetorical relations inventory and a set of glue logic rules and axioms. The rhetorical relations are also given as TFSs in a separate file. They will inherit their spe-

cific properties (subordinating or coorodating relation, veridical,...) from their supertypes. These will constrain the kind of update that has to be operated on the discourse structure to accomodate for the new relations. Specific semantic updates can be expressed as features.

The Glue Logic axioms are defined in another file. They are inserted as rule schemas :

```
(be_introuble(_,X,Pi) & says(X,Pi) >
  request_assist(X,Pi)).
```

The variables get instanciated to specific terms in the discourse, then the predicate and all its arguments, save the label argument, become one predicate, taking the label as sole argument. These are quantifier free first order formulas that make up the logic for inferring rhetorical relations:

```
[be_introuble(e1,ag1)](p11)&
 [says(ag1)](p11) >
 [request_assist(ag1)](p11)
```

2.2 Modular approach

The task of the program will be to build an SDRS or Segmented Discourse Representation Structure for an input discourse or dialogue. Sentences are entered as underspecified logical forms (ULF). A full TFS description of the new ULF will be built by the program. Then the previous SDRS is checked for available attachment sites. For everyone of these, the Glue Module will transfer information from both ULFs into the shallower Glue logic. This information, together with the general Glue rules and axioms should allow the program to infer some rhetorical connection for the new constituent via the Commonsense Entailment Theorem Prover or CETP (Shlangen & Lascarides, 2002). We will typically have a list of possible updates with different attachment sites or anaphora resolution.

For each of these possible attachments, the Update module will update the previous SDRS according to the constraints implied by the inferred rhetorical relation and possible semantic disambiguations. The module for Maximal Discourse Coherence will then score each of these updates according to a number of criteria like rhetorical relation specificity, added semantic information, etc... The update getting the highest score will be selected and we can go on to the next utterance.

The representation used by the program for SDRS also allows for underspecification, making it possible to simply add an 'unknown' relation when none is inferrable. This underspecification can then be resolved later on when new information is added to the SDRS.

The conception of the program closely follows SDRT's modular approach and is intended to reflect the declarative way in which many sources of information combine. DiaP is a work in progress, but at this time, it can analyse small discourses and build coherent SDRSs for them using a wide array of SDRT techniques: lexically specified implicit arguments, event structure, anaphora and cataphora resolution, type inheritance,...

3 Using DiaP

One calls DiaP on an input file containing a discourse or a dialogue, *i.e.* a sequence of utterances entered as this :

```
(Speaker:)Sentence-ULF description
```

by invoking the following predicates in PROLOG:

```
?- dip(file).    to process a discourse.
```

```
?- diap(file).  to process a dialogue.
```

There is no GUI for DiaP at the moment, so most of the processing details are output to the terminal. There are some facilities though for automatically exporting the resulting SDRSs or AVMs in \LaTeX and viewing them.

Further work will include improving the user interface, linking the program to an HPSG parser like LKB and of course, testing the program on larger discourses with extensive sets of rules.

References

- Asher, N. and A. Lascarides. 2003. *Logics of Conversation*. Cambridge University Press.
- Copestake, A. 1999. *The (New) LKB System*. <http://www-csli.stanford.edu/aac/papers.html>
- Copestake, A., D. Flickinger, I. Sag and C. Pollard. 1999. *Minimal Recursion Semantics: An introduction*. Draft. <http://www-csli.stanford.edu/aac/papers.html>
- Shlangen, D. and A. Lascarides. 2002. *CETP: An automated theroem prover for a fragment of common sense entailment*. Edinburgh Informatics Report Series. EDI-INF-RR-0119.