# Rhetorical Parsing with Underspecification and Forests

**Thomas Hanneforth**
Dept. of Linguistics
University of Potsdam
P.O. Box 601553
14415 Potsdam, Germany
tom@ling.uni-potsdam.de

**Silvan Heintze**
Dept. of Linguistics
University of Potsdam
P.O. Box 601553
14415 Potsdam, Germany
heintze@ling.uni-potsdam.de

**Manfred Stede**
Dept. of Linguistics
University of Potsdam
P.O. Box 601553
14415 Potsdam, Germany
stede@ling.uni-potsdam.de

## Abstract

We combine a surface based approach to discourse parsing with an explicit rhetorical grammar in order to efficiently construct an underspecified representation of possible discourse structures.

## 1 Introduction

The task of rhetorical parsing, i.e., automatically determining discourse structure, has been shown to be relevant, inter alia, for automatic summarization (e.g., Marcu, 2000). Not surprisingly, though, the task is very difficult. Previous approaches have thus emphasized the need for heuristic or probabilistic information in the process of finding the best or most likely rhetorical tree.

As an alternative, we explore the idea of strictly separating "high-confidence" information from hypothetical reasoning and of working with underspecified trees as much as possible. We create a parse forest on the basis of surface cues found in the text. This forest can then be subject to further processing. Depending on the application, such further steps can either calculate the "best" tree out of the forest or continue working with a set of structured hypotheses.

Section 2 briefly summarizes our proposal on underspecified rhetorical trees; section 3 introduces our grammar approach to text structure; section 4 compares this strategy to earlier work.

## 2 Parse forests and underspecification

We will illustrate the underspecification of ambiguities with the following example:
"(1) Yesterday the delegates elected their new representative by a narrow margin. Even though (2) Smith got only 234 votes, (3) he accepted the position. But (4) his predecessor was rather irritated by the results."

We take it that *even though* unambiguously marks a CONCESSION between the embedded clause (2, satellite) and the matrix clause (3, nucleus). For the purpose of illustration, we also assume that "but" can only signal a bi-nuclear CONTRAST relation with the second nucleus (4); the span of the first nucleus is in this case ambiguous (1-3 or 2-3). For linking (1) to the remaining material, we suppose that either ELABORATION (with nucleus (1)) or SEQUENCE holds. Further relations are possible, which will add to the possibilities, but our points can be made with the situation as just described.

Instead of enumerating all possible rhetorical trees for our example text, we use a *parse forest representation* which compactly encodes the different analysises. A parse forest is basically an attributed And-Or-graph with the properties of *subtree sharing* and *containment of ambiguities*. The first property means that a subtree, which plays different roles in some bigger structure, is represented only once. The second property ensures that two subtrees which have in common the same category and the same terminal yield, but which differ in the first step of a leftmost derivation are unified together.

Fig. 1 shows a simplified parse forest for the example text.

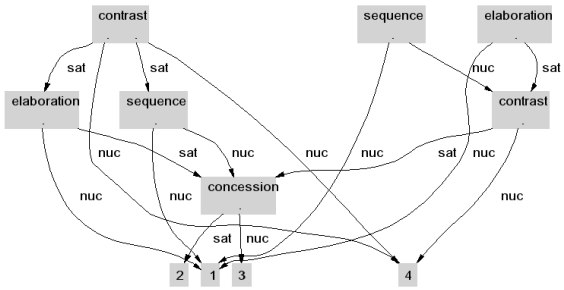


*Fig.1: Parse forest for the input text*

Subtree sharing is indicated by nodes (e.g. "1") which have several incoming edges. Containment of ambiguities is exemplified in fig. 1 by the upper left contrast node which represents a disjunctive hypothesis concerning the span of the relation.

Reitter and Stede (to appear) developed an XML-DTD scheme to represent such parse forests in XML notation.

## 3 Discourse structure parsing

In our approach, we combine a standard chunk parser which identifies the relevant units for discourse processing with a feature-based grammar which builds larger rhetorical trees out of these chunks. The categories and features we use are summarized in table 1.

| Cat. | Feat. | Values | Comment |
|---|---|---|---|
| **rst** | | | RST-tree |
| | cat | macro_seg, s, ip, pp, … | The category of the RST-tree: macro segments, phrases sentences etc. |
| | type | ns, nn, term | Type of RST-tree: nuc-sat, multi-nuclear or terminal |
| | role | nuc, sat | Nucleus or satellite |
| | relation | elaboration, contrast, cause, … | The relation which combines the daughters of the RST-tree. |
| | dp | no_dp, but, although, … | The discourse particle triggering the relation, or no_dp, if absent. |
| **dp** | See above | | Discourse particle |
| **chunk** | | | Phrase or sentence |
| **punct** | | | Punctuation |

*Table 1: Grammar categories and features*

There are three groups of grammar rules:

1. Rules combining chunks to terminal RST-trees
2. Rules combining discourse particles and sentence fragments to non-primitives RST-trees
3. Rules combining sentences or groups of sentences (so called macro segments) to non-primitive RST-trees.

An example for a rule in group 1 is the one which builds a terminal RST-tree of category mc (main clause) out of a discourse particle, and sentence fragment and a full stop (all examples are given in Prolog-style notation, with curly brackets indicating feature structures):

(1)

```
rst({cat:mc, dp:DP,  type:term}) --->
      dp({cat:pav, dp:DP}),
      chunk({cat:ip}),
      punct({cat:fullstop}).
```

Rules like this one are used to build terminal RST-trees for sentences like (4) in our example text.

The second group of rules is exemplified by a rule which combines two terminal RST-trees - a subordinate clause containing a conjunction like *even though* and another clause - to a hypotactic RST-tree:

(2)

```
rst({cat:mc, rel:concession, dp:no_dp, type:ns}) --->
      rst({cat:sc, dp:even_though, role:sat}),
      rst({cat:mc, dp:no_dp, role:nuc}).
```

The macro segment building rules of the third group can be divided into two subclasses. The first class is constituted by rules which construct RST-trees on the basis of a relation that is triggered by a discourse particle. An example of this type is the possible contrast-relation between segments 4 and 2-3 in (1), which is triggered by the discourse particle *but*.

(3)

```
rst({cat:macro_seg, rel:contrast,
                        dp:no_dp, type:ns}) --->
      rst({cat:macro_seg, role:sat}),
      rst({cat:macro_seg, role:nuc, dp:but}).
```

The other subclass contains rules which freely construct branching RST-trees without the overt evidence of discourse particles. The relations which are typically involved here are SEQUENCE and ELABORATION. Relations which have in common the same type of nucleus-satellite-configuration are unified into a single rule using the list-valued form of the relation-feature:

(4)

```
rst({cat:macro_seg, rel:[sequence,elaboration],
                        dp:no_dp, type:nn}) --->
      rst({cat:macro_seg, role:nuc, dp:no_dp}),
      rst({cat:macro_seg, role:nuc, dp:no_dp}).
```

Fig. 2 shows a parse tree which reflects one analysis of our example text. Note that the segments into which the input is broken usually smaller than sentences.

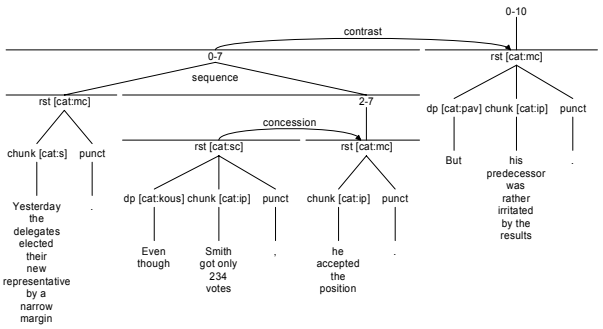


*Fig.2: Sample parse tree for the input text*

Rules like (4) ensure the robustness of the grammar as they can be used to combine partial structures without any structure triggering discourse particles.

Furthermore, rules of the kind shown in (4) are on the one hand necessary to produce all possible branching structure over a given sequence of terminal elements. On the other hand they introduce massive ambiguities into the grammar which causes the number of analyses to grow according to the Catalan numbers (cf. Aho and Ullman, 1972, p. 165).

It is therefore crucial that during parsing the construction of parse trees is strictly avoided because that would turn an otherwise polynomial parsing algorithm like chart parsing into an exponential one. Instead we incrementally build the parse forest mentioned in section 2. This is done by assigning a unique id to each edge introduced into the chart and by storing the ids of the immediate daughters within the edge. After parsing the parse forest is constructed by partitioning the set of edges into equivalence classes. Two chart edges E1 and E2 are in the same equivalence class if they a) have identical start and end positions and b) the categories of E1 and E2 subsume each other. For the subsumption test it is necessary to ignore the role-feature, because this feature is an attribute of the parse forest edges and not of the parse forest nodes.

Besides keeping the parsing algorithm polynomial it is of equal importance to keep the grammar constant low. For example, rule (4) which establishes a SEQUENCE/ELABORATION relation between two macro segments also connects two simple clauses (of category mc), a macro segment and a simple clause, or a simple clause and a macro segment. The standard move to avoid this kind of rule multiplication is to introduce an unary chain rule of the form

rst({cat:macro_seg}) ---> rst({cat:mc})

which ensures the desired level shifting.

Because of the inherent relational nature of RST trees this solution is blocked. Instead we use an inheritance hierarchy like that in fig. 3 and replace rule (4) with the following one, which is underspecified w.r.t to the category feature.

(5)

```
rst({cat:macro_seg, rel:[sequence,elaboration],
                          dp:no_dp, type:nn}) --->
     rst({cat:rst_tree, role:nuc, dp:no_dp}),
     rst({cat:rst_tree, role:nuc, dp:no_dp}).
```

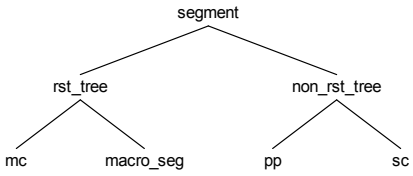


*Fig 3: Simplified inheritance hierarchy for cat*

## 4 Related work

Similar to Marcu (2000) we assume discourse markers as indicators for rhetorical relations.

But contrary to Marcu (1999) and also to Schilder (2002) we use a full-fledged discourse grammar and a standard parsing algorithm, which makes it, in our opinion, unnecessary to propose special rhetorical tree building operations, as suggested e.g. by Marcu (1999).

By using the chart parsing algorithm combined with the construction of an underspecified parse forest, it can easily be shown that our method is of cubic complexity. This is a crucial property, because it is commonly assumed that the number of distinct structures that can be constructed over a sequence of *n* discourse units is exponential in *n*, (as it is for example implicit in the DCG based algorithm proposed by Schilder, 2002).

Our system is robust in the same way as the one in Schilder (2002) because the grammar admits underspecified rhetorical trees in the absence of overt discourse markers.

## 5 Conclusion

We have shown that a grammar based approach to rhetorical parsing is suitable for efficient and robust construction of underspecified rhetorical structures.

## References


Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translating and Compiling*. Volume 1. Prentice-Hall, Englewood Cliffs, NJ.

Daniel Marcu. 1999. A decision-based approach to rhetorical parsing. *The 37th Annual Meeting of the Association for Computational Linguistics* (ACL'99), pages 365-372, Maryland, June 1999.

Daniel Marcu. 2000. The Rhetorical Parsing of Unrestricted Texts: A Surface-Based Approach. *Computational Linguistics*, 26 (3), pages 395-448.

David Reitter and Manfred Stede. to appear. Step by Step: Underspecified Markup in Incremental Rhetorical Analysis. To appear in: *Proc. Of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03)*. Budapest.

Frank Schilder. 2002. Robust Discourse Parsing via Discourse Markers, Topicality and Position. *Natural Language Engineering* 8 (2/3).