XML/XSL in the Dictionary: The Case of Discourse Markers

Daniela Berger and **David Reitter** and **Manfred Stede**
University of Potsdam
Dept. of Linguistics / Applied Computational Linguistics
P.O. Box 601553 / D-14415 Potsdam / Germany
{berger|reitter|stede}@ling.uni-potsdam.de

Abstract

We describe our ongoing work on an application of XML/XSL technology to a dictionary, from whose source representation various views for the human reader as well as for automatic text generation and understanding are derived. Our case study is a dictionary of discourse markers, the words (often, but not always, conjunctions) that signal the presence of a disocurse relation between adjacent spans of text.

1 Overview

Electronic dictionaries have made extensive use of SGML encoding in the past, but to our knowledge, the advantages of contemporary frameworks such as XML/XSL are only beginning to be explored. We are applying this framework to our work on a lexicon of 'discourse markers' and will outline the advantages of deriving a variety of views from the common underlying lexical resource: different views for different demands by the human eye, but also application-specific views that tailor the dictionary to either the parsing or the generation task (a third one would be machine translation but is not covered in this paper), and that respect the conventions of the specific underlying programming language. Using XSL style sheets for producing the views automatically is especially useful when the lexicon is still under development: the ramifications of particular modifications or extensions can be made visible easily by running the conversion and testing the applications in question.

Discourse markers are words (predominantly conjunctions) that signal the kind of semantic or rhetorical relationship between adjacent spans of text. In text generation, when given a representation of propositions and relations holding between them, the task is to select an appropriate discourse marker in the context. In text understanding, discourse markers are the most important clues for inferring the 'rhetorical structure' of the text, a task that has lately been called 'rhetorical parsing'. While these discourse markers are a somewhat idiosyncratic class of lexical items, we believe that our general approach to applying XML/XSL can be fruitful to other branches of the dictionary as well (in particular to the open-class "content words").

After reviewing some earlier work on XML-based dictionaries (Section 2) and discussing the notion of discourse markers (Section 3), we proceed to outline the particular requirements on a discourse marker lexicon from both the text generation and the text understanding perspective (Section 4). Then, Section 5 describes our XML/XSL encoding of the source lexicon and the views for the human eye, for automatic text generation, and for text understanding. Finally, Section 6 draws some conclusions.

2 Dictionaries and XML: Related work

Recent research in lexicology has been focused on two different goals: the mark-up process for existing print dictionaries, and the successful construction of machine-readable dictionaries from scratch.

The first approach has received more attention in the past. This is partly due to the fact that the transformation of existing print dictionaries into modules for NLP applications promises to be less time-consuming than the construction of a new machine-readable database. Lexicologists agree on the fact that a dictionary entry is inherently hierarchical, i.e., it consists out of atomic elements grouped together within non-atomic elements in a tree-like hierarchy. Many approaches place orthographical and phonological information together in one group, while grammatical information is put in a different group. This hierarchical approach also allows to denote scope by inserting information at different levels of the hierarchy. Again, information about orthography and phonology generally applies to every facet of the headword and are thus placed high in the hierarchy, while other information might only apply to single definitions and thus ranks lower hierarchically (Amsler/Tompa, 1988; Ide, Véronis, 1995; Ide et al., 2000).

A common problem of lexicologists working with print dictionaries is the fact that there is a certain variation between entries in any two given dictionaries or even within the same dictionary. This results in a neccessary trade-off between the descriptive power and the generality of an approach, i.e. to design a SGML application that is both descriptive enough to be of practical value and general enough to accomodate the variation.

There has been, on the other hand, only little research on machine-readable dictionaries that are not based on print dictionaries. To our knowledge, only (Ide et al., 1993) deals with this issue by reviewing several approaches towards encoding machine-readable dictionaries. One of these is the use of text models that apply a rather flat hierarchy to mark up dictionary entries. These text models might chiefly use typographical or grammatical information. Another approach is using relational databases, in which the information contained in a dictionary entry is distributed over several databases. A third approach is based on feature structures that impose a rich hierarchical structure on the data. The authors finally describe an example application that uses feature structures encoded in SGML to set up a machine-readable dictionary.

The papers mentioned above agree on using SGML for the mark-up. We found that their SGML code is, however, in general XML-compliant.

3 Discourse markers

Several contemporary discourse theories posit that important aspects of a text's coherence can be formally described (and represented) by means of *discourse relations* holding between adjacent spans of text (e.g. Asher, 1993; Mann, Thompson, 1988). We use the term *discourse marker* for those lexical items that (in addition to non-lexical means such as punctuation, aspectual and focus shifts, etc.) can signal the presence of such a relation at the linguistic surface. Typically, a discourse relation is associated with a wide range of such markers; consider, for instance, the following variety of CONCESSIONS, which all express the same underlying propositional content. The words that we treat as discourse markers are underlined.

We were in SoHo; {nevertheless | nonetheless | however | still | yet}, we found a cheap bar.

We were in SoHo, but we found a cheap bar anyway.

Despite the fact that we were in SoHo, we found a cheap bar.

Notwithstanding the fact that we were in SoHo, we found a cheap bar.

Although we were in SoHo, we found a cheap bar.

If one accepts these sentences as paraphrases, then the various discourse markers all need to be associated with the information that they signal a concessive relationship between the two propositions involved. Notice that the markers belong to different syntactic categories and thus impose quite different syntactic constraints on their environment in the sentence. Discourse markers do not form a homogeneous class from the syntactican's viewpoint, but from a *functional* perspective they should nonetheless be treated as alternatives in a paradigmatic choice.

A detailled characterization of discourse markers, together with a test procedure for identifying them in text, has been provided for English by (Knott, 1996). Recently, (Grote, to appear) adapted Knott's procedure for the German language. Very briefly, to identify a discourse marker (e.g., *because*) in a text, isolate the clause containing a candidate from the text, resolve any anaphors and make elided items explicit; if the resulting text is incomplete (e.g., *because the woman bought a Macintosh*), then the candidate is indeed a 'relational phrase', or for our purposes, a two-place discourse marker.

In addition to the syntactic features, the differences in meaning and style between similar markers need to be discerned; one such difference is the degree of specificity: for example, *but* can mark a general CONTRAST or a more specific CONCESSION. Another one is the notable difference in formality between, say *but ... anyway* and *notwithstanding*.

From the perspective of text generation, not all paraphrases listed above are equally felicitous in specific contexts. In order to choose the most appropriate variant, a generator needs knowledge about the fine-grained differences between similar markers for the same relation. Furthermore, it needs to account for the interactions between marker choice and other generation decisions and hence needs knowledge about the syntagmatic constraints associated with different markers. We will discuss this perspective in Section 4.1

From the perspective of text understanding, discourse markers can be used as one source of information for guessing the rhetorical structure of a text, or automatic rhetorical parsing. We will characterize this application in Section 4.2.

4 Requirements on a discourse marker lexicon

As the following two subsections will show, text generation and understanding have quite different preferences on the information coded in a discourse marker lexicon, or "DiMLex" for short. In addition, different systems employ different programming languages, and the format of the lexicon has to be adapted accordingly. Yet we want to avoid coding different lexicons manually and thus seek a common "core representation" for DiMLex from which the various application-specific instantiations can be derived. Before proposing such a representation, though, we have to examine in more detail the different requirements.

4.1 The text generation perspective

Present text generation systems are typically not very good at choosing discourse markers. Even though a few systems have incorporated some more sophisticated mappings for specific relations (e.g., in DRAFTER (Paris et al., 1995)), there is still a general tendency to treat discourse marker selection as a task to be performed as a "side effect" by the grammar, much like for other function words such as prepositions.

To improve this situation, we propose to view discourse marker selection as one subtask of the general lexical choice process, so that — to continue the example given above — one or another form of CONCESSION can be produced in the light of the specific utterance parameters and the context. Obviously, marker selection also includes the decision whether to use any marker at all or leave the relation implicit. When these decisions can be systematically controlled, the text can be tailored much better to the specific goals of the generation process.

The generation task imposes a particular view of the information coded in DiMLex: the entry point to the lexicon is the discourse relation to be realized, and the lookup yields the range of alternatives. But many markers have more semantic and pragmatic constraints associated with them, which have to be verified in the generator's input representation for the marker to be a candidate. Then, discourse markers place (predominantly syntactic) constraints on their immediate context, which affects the interactions between marker choice and other realization decisions. And finally, markers that are still equivalent after evaluating these constraints are subject to a choice process that can utilize preferential (e.g. stylistic or length-based) criteria. Therefore, under the generation view, the information in DiMLex is grouped into the following three classes:

— *Applicability conditions:* The necessary conditions for using a discourse marker, i.e., the features or structural configurations that need to be present in the input specification.

— *Syntagmatic constraints:* The constraints regarding the combination of a marker and the neighbouring constituents; most of them are syntactic and appear at the beginning of the list given above (part of speech, linear order, etc.).

— *Paradigmatic features:* Features that label the differences between similar markers sharing the same applicability conditions, such as stylistic features and degrees of emphasis.

Very briefly, we see discourse marker choice as one aspect of the *sentence planning* task (e.g. (Wanner, Hovy, 1996)). In order to account for the intricate interactions between marker choice and other generation decisions, the idea is to employ DiMLex as a declarative resource supporting the sentence planning process, which comprises determining sentence boundaries and sentence structure, linear ordering of constituents (e.g. thematizations), and lexical choice. All these decisions are heavily interdependent, and in order to produce truly adequate text, the various realization options need to be weighted against each other (in contrast to a simple, fixed sequence of making the types of decisions), which presupposes a flexible computational mechanism based on resources as declarative as possible. This generation approach is described in more detail in (Grote, Stede, 1998).

4.2 The text understanding perspective

In text understanding, discourse markers serve as cues for inferring the rhetorical or semantic structure of the text. In the approach proposed in (Marcu, 1997), for example, the presence of discourse markers is used to hypothesize individual textual units and relations holding between them. Then, the overall discourse structure tree is built using constraint satisfaction techniques. Our analysis method uses the lexicon for an initial identification and disambiguation of discourse markers. They serve as one of several other shallow features that determine through a statistical, learned language model the optimal rhetorical analysis.

In contrast to the use of markers in generation, the list of cues is significantly longer and includes phrasal items like *aus diesem Grund (for this reason)* or *genauer genommen (more precisely)*.

5 Our XML/XSL solution

In the following we show some sample representations and style sheets that have been abridged for presentation purposes.

5.1 Source representation

In our hierarchical XML structure, the `<dictionary>` root tag encloses the entire file, and every single entry rests in an `<entry>` tag, which unambigously identifies every entry with its `id` attribute. Within the `<entry>` tag there are four subordinate tags: `<form>`, `<syn>`, `<sem>`, and `<examples>`.

The `<form>` tag contains the orthographic form of the headword; at present this amounts to two slots for alternative orthographies. The `<syn>` area contains the syntactic information about the headword. In this shortened example, there is only the `<init_field>` tag present,

```
<?xml version="1.0" ?>
<?xml-stylesheet type="text/xsl"
  href="short_dictionary.xsl" ?>
<!DOCTYPE dictionary SYSTEM "DTD.dtd">
<dictionary>
   <entry id="05">
      <form>
         <orth>denn</orth>
         <alt_orth>none</alt_orth>
         <!-- . . . -->
      </form>
      <syn>
         <init_field>-</init_field>
         <!-- . . . -->
      </syn>
      <sem>
         <function>causal</function>
         <!-- . . . -->
      </sem>
      <examples>
         <example>Das Konzert muss ausfallen,
          *denn* die S&auml;ngerin ist erkrankt.
         </example>
         <example>Die Blumen auf dem Balkon sind
          erfroren, *denn* es hat heute nacht
          Frost gegeben.</example>
      </examples>
   </entry>
   <entry>
      <!-- more entries -->
   </entry>
</dictionary>
```

Figure 1: The XML structure

which shows whether the headword can be used in the initial field of a sentence. Correspondingly, `<sem>` contains semantic features such as the `<function>` tag, which contains the semantic/discourse relation expressed by the headword. Finally, `<examples>`, contains one or more `<example>` tags that may each give an example sentence.

We have shortened this presentation considerably; the full lexicon contains more fine-grained features for all three areas: within `<form>`, information on pronounciation, syllable structure, and hyphenation; within `<syn>`, information about syntactic subcategorization and possible positions in the clause; within `<sem>`, for example the feature whether the information subordinated by the marker is *presupposed* or not.

5.2 HTML views

The listing in Figure 4 shows a style sheet that provides an HTML by listing the XML data in a format that roughly resembles a print dictionary. Figure 2 shows the output that results from applying this XSL file to the XML source in figure 1.

05: denn
Occurrences: middle field / Nullstelle
Semantics: kausal
Related markers: weil da
Examples: Das Konzert muss ausfallen, *denn* die Sängerin ist erkrankt.
Die Blumen auf dem Balkon sind erfroren, *denn* es hat heute nacht Frost gegeben.

Figure 2: One HTML view of the data

We assume that the general structure of the formatting part of XSL is familiar to the reader. We would like to highlight some details.

XLINK is used to ensure that the entry contains an HTML-anchor named after the headword (ll. 14-20). This way it is possible to link to a certain entry from the `<rel>` tag of a different entry (39-45).

We also employ the XSL equivalent to an if/then/else construct (24-31). The `<xsl:choose>` tag encloses the choices to be made. The `<xsl:when>` tag contains the condition `match=".[alt_orth='none']"` that does nothing if the `<alt_orth>` tag contains the data `none`. Every other case is covered by the `<xsl:otherwise>` tag that prints out the `<alt_orth>` information if it is not `no entry`.

Entry	`alt_orth`	`init_field`	`mid_field`	...
denn	none	-	+	...
da	none	+	+	...
zumal	none	-	-	...
weil	none	-	-	...
als	none	-	-	...

Figure 3: Another possible HTML view of the data

Figure 3 shows another possible view for the data. In this case the data is presented in table form, ordered by the value of the `mid_field` tag. It would be easy to show that it is possible to use a `<xsl:choose>` construct as shown in the example before to print out only those entries that satisfy a certain condition.

5.3 The text generation view

For the lexicon to be applied in our text generation system 'Polibox' (Stede, 2002), we need a Lisp-based version of DiMLex. Using the `(defstruct <name> <slot1>`

```
    <?xml version="1.0"?>
    <xsl:stylesheet
     xmlns:xsl=
       "http://www.w3.org/1999/XSL/Transform">
5     <xsl:template match="/">
        <FONT SIZE="-2">
          <xsl:apply-templates/>
        </FONT>
      </xsl:template>
10    <xsl:template match="dictionary">
        <xsl:apply-templates/>
      </xsl:template>
      <xsl:template match="entry">
        <P><font size="2"><B><A>
15            <xsl:attribute name="NAME">
                <xsl:value-of select="form/orth"/>
              </xsl:attribute>
              <xsl:value-of select="./@id"/>:
              <xsl:value-of select="form/orth"/>
20        </A></b></font>
          <xsl:apply-templates/></P>
      </xsl:template>
      <xsl:template match="form">
        <xsl:choose>
25        <xsl:when match=".[alt_orth='none']">
          </xsl:when>
          <xsl:otherwise>
            <BR/><B>Alternative orthography:</B>
              <xsl:value-of select="alt_orth"/>
30        </xsl:otherwise>
        </xsl:choose>
      </xsl:template>
      <xsl:template match="sem">
        <BR/><B>Semantics:</B>
35      <xsl:value-of select="ko_sub"/>
        / <xsl:value-of select="function"/>
        <br/><b>Related markers:</b>
          <xsl:for-each select="rel">
            <A><xsl:attribute name="HREF">
40               #<xsl:value-of select="."/>
              </xsl:attribute>
              <xsl:value-of select="."/></A>
          </xsl:for-each>
      </xsl:template>
45    <xsl:template match="syn">
        <BR/><B>Occurrences:</B>
        <xsl:choose>
          <xsl:when match=".[init\_field='-']">
          </xsl:when>
50        <xsl:otherwise>
            initial field /
          </xsl:otherwise>
        </xsl:choose>
      </xsl:template>
55    <xsl:template match="examples">
        <BR/><B>Examples:</B>
        <xsl:for-each select="example">
          <xsl:value-of select="."/><BR/>
        </xsl:for-each>
60    </xsl:template>
    </xsl:stylesheet>
```

Figure 4: The XSL file for the HTML view shown in Figure 2

`.. <slotn>)` construct, we define a class of objects for discourse markers, where the features needed for generation are stored in the slots. Again, we abbreviate slightly:

```
(defstruct DiscMarker
   Relation N-Complexity S-Complexity
   Ortho POS ... Style)
```

Now, a Lisp-object for each individual discourse marker entry is created with the function `make-Discmarker`, which provides the values for the slots. Figure 5 shows the shape of the entry for *denn*, whose XML-source was given in figure 1.

Again, we aim at deriving these entries automatically via an XSL sheet (which we do not show here). Notice that the mapping task is now somewhat different from the HTML cases, since the transformation part of XSL (XSLT) comes into play here. Instead of merely displaying the data in a web browser as in the examples before, an XSLT processor may transform data for use in some XML based client application.

As explained in Section 4.1, in the generation scenario we are given a tree fragment consisting of a discourse relation node and two daughters representing the related material, the nucleus and the satellite of the relation. In order to decide whether a particular marker can be used, one important constraint is the "size" of the daughter material, which can be a single proposition or an entire sub-tree. The generator needs to estimate whether it will fit into a single phrase, clause, sentence, or into a sequence of sentences; a subordinating conjunction, for instance, can only be used if the material can be expressed within a clause. Thus, the Lisp-entry contains slots `N-Complexity` and `S-Complexity`, which are highly application-specific and thus do not have a simple corresponding feature in the XML source representation of the dictionary. The XSL sheet thus inspects certain combinations of daughter attributes of `<syn>` and maps them to new names for the fillers of the two `Complexity` slots in the Lisp structure. (Similar mappings occur in other places as well, which we do not show here.)

```
(make-Discmarker
   :Relation cause
   :N-Complexity sent
   :S-Complexity sent
   :Ortho denn
   :POS coordconj
   :Style unmarked)
```

Figure 5: Lisp-version of generation-oriented dictionary entry for *denn* (abridged)

5.4 The text understanding view

Our analysis method recasts rhetorical parsing as a set of classification decisions, where a parsing framework builds a tree structured analysis. Each of the decisions is based on a set of features. Feature types range from syntactical configuration to the presence of a certain discourse marker. The mapping from a pattern of observed features to a rhetorical relation may be learned automatically by a classification learning algorithm.

Learning and analysis applications use a parsing framework that gives us a set of text span pairs. Every two text spans are subject to a classification learning algorithm (during training) or the actual classifier. So, a rhetorical relation is assigned to these two spans of text along with a score so that the parsing framework may decide which of several competing classifications to accept.

Learning and actual rhetorical analysis are accomplished by a set of distinct tools that add specific annotations to a given input text, before resulting relations are learned or guessed. These tools include a data mining component, a part-of-speech tagger and a segmenter. They all access data organized in an XML syntax. The central learning and parsing application makes use of a Document Object Model (DOM) representation of the corpus. This data structure is effectively used for information interchange between several components, because it allows us to easily visualize and modify the current data at each processing step during development.

With the present corpus data, the learning algorithm is theoretically able to identify rhetorical markers automatically and could thus compile a marker lexicon. However, markers are highly ambiguous. Even though many of them can be tagged as adverbials or conjunctions, markers often lack distinctive syntactic and/or positional properties; some of them are phrasal, some are discontinuous. To identify significant cue - relation correlations, a lot of annotated data is necessary: more than is usually available. In a sparse data situation, we want to easen the learning task for the rhetorical language model: It makes sense to use a discourse marker lexicon.

On the other hand, we do not expect a hand-crafted lexicon to contain all contextual constraints that would enable us to assign a single rhetorical relation. These constraints can be very subtle; some of them should be represented as probabilistic scalar information.

Thus, DiMLex contributes to initial discourse marker disambiguation. From each entry, we interpret syntactic positioning information, morphosyntactic contextual information and a scope class (sentential, phrasal, discourse-level) as a conjunction of constraints. The presence of a certain discourse marker in a specified configuration is one of the features to be observed in the text.

Depending on the depth of the syntactic and semantic analysis carried out by the text understanding system, different features provided by DiMLex can be taken into account. Certain structural configurations can be tested without any deep understanding; for instance, the German marker *während* is generally ambiguous between a CONTRAST and a TEMPORAL-COOCCURRENCE reading, but when followed by a noun phrase, only the latter reading is available (*während* corresponds not only to the English *while* but also to *during*).

In the parsing client application, DiMLex serves as resource for the identification of cue phrases in specific structural configurations. Rhetorical information from the DiMLex entries may serve as one of several cues for the classification engine. The final linking from cue patterns to rhetorical relations is learned from a corpus annotated with rhetorical structures.

6 Summary

We have presented our ongoing work on constructing an XML-based dictionary of discourse markers, from which a variety of views are derived by XSL sheets: For the dictionary designer or application developer, we present the dictionary in tabular form or in a form resembling print dictionaries, but with hyperlinks included for easy cross-referencing. Similarly, text generation and understanding systems are on the one hand written in different programming languages and thus expect different dictionary formats; on the other hand, the information needed for generation and parsing is also not identical, which is accounted for by the XSL sheets. Evaluation of the approach will depend on the client applications. Their implementation will determine the final shape of DiMLex.

References

R.A. Amsler and F.W. Tompa, "An SGML-Based Standard for English Monolingual Dictionaries." In: *Information in Text: Fourth Annual Conference of the UW Center for the New Oxford English Dictionary*, University of Waterloo Center for the New Oxford English Dictionary, Waterloo, Ontario, 1988, pp. 61-79.

N. Asher. *Reference to Abstract Objects in Discourse.* Dordrecht: Kluwer, 1993.

B. Grote, M. Stede. "Discourse marker choice in sentence planning." In: *Proceedings of the Ninth International Workshop on Natural Language Generation,* Niagara-on-the-Lake, Canada, 1998.

B. Grote. "Signalling coherence relations: temporal markers and their role in text generation." Doctoral dissertation, Universität Bremen, forthcoming.

N. M. Ide, J. Le Maitre, and J. Véronis, "Outline of a Model for Lexical Databases." *Information Processing and Management*, 29, 2 (1993), 159-186.

N. M. Ide and J. Véronis. Encoding Dictionaries.*Computers and the Humanities*, 29:167-180, 1995

N. M. Ide, Kilgarriff, A., and Romary, L.A Formal Model of Dictionary Structure and Content. In *Proceedings of EURALEX 2000*, pp. 113-126, Stuttgart.

A. Knott. "A data-driven methodology for motivating a set of coherence relations." Doctoral dissertation, University of Edinburgh, 1996.

W. Mann, S. Thompson. "Rhetorical structure theory: Towards a functional theory of text organization." In: *T*EXT, 8:243-281, 1988.

D. Marcu. "The Rhetorical Parsing of Unrestricted Natural Language Texts." *P*roceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics, Somerset, New Jersey, 1997.

C. Paris, K. Van der Linden, M. Fischer, A. Hartley, L. Pemberton, R. Power and D.R. Scott, "Drafter: A Drafting Tool for Producing Multilingual Instructions", *Proceedings of the 5th European Workshop on Natural Language Generation*, pp. 239-242, Leiden, the Netherlands, 1995.

M. Stede. "Polibox: Generating, descriptions, comparisons, and recommendations from a database." In Proceedings of COLING-2002.

L. Wanner, E. Hovy. "The HealthDoc Sentence Planner." Proceedings of the 8th International Workshop on Natural Language Generation, Hearstmonceux Castle, 1996.

Web References

Domain Object Model
W3C Recommendation, 13 November 2000
`http://www.w3.org/TR/DOM-Level-2-Core`

Extensible Stylesheet Language (XSL) 1.0
W3C Recommendation, 15 October 2001
`http://www.w3.org/TR/xsl`

XML Base
W3C Recommendation 27 June 2001
`http://www.w3.org/TR/xmlbase`

XSL Transformations (XSLT) 1.0
W3C Recommendation, 16 November 1999
`http://www.w3.org/TR/xslt`