Manfred Stede

Local coherence analysis in a multi-level approach to automatic text analysis

We characterize a text-technological approach to text analysis as combination of a multi-level representation framework and XML-based document processing techniques. The main advantages of such an approach are the chance to flexibly combine modules for constructing different applications, and the overall robustness resulting from the operational principle of higher-level modules combining the — possibly partial — results of lower-level ones. We illustrate the approach with the specific task of local coherence analysis, i.e. the computation of coherence relations between text spans.

1 Introduction

What does it mean to analyze or — more ambitiously — to understand a text? Over the years, Artificial Intelligence and Computational Linguistics have responded in quite different ways to this question. The present paper argues in favour of a text-technologically-inspired multi-level approach to automatic text analysis: *Text technology* emphasizes the utility of XML-based document processing techniques (see Lobin (2000)), while the *multi-level* conception views text analysis as the systematic composition of distinct levels of information that can be produced by independent analysis tools. By bringing these two realms together, we aim at designing robust systems that can be easily configured for dealing with different kinds of text and perform different tasks (which usually share a number of generic subtasks such as tagging, noun phrase chunking, etc.). When levels are created independently of one another, the analysis tools might very well produce better or worse results for different portions of the text. Looking then across all levels of analysis, we can end up with more or less information for different segments — depending on how difficult the text is, and on how good the tools are. The result is what had been envisaged by Hirst and Ryan (1992) as a "mixed-depth representation" of text content. Utilizing contemporary text-technological approaches, this can be achieved by a highly distributed analysis approach much better than with "holistic" analysis schemes that had been *en vogue* when Hirst and Ryan had put forward the idea.

After outlining this general approach (Sections 2 to 4), the second half of the paper (Section 5) will illustrate the idea by focusing on one particular subtask of text understanding: *local coherence analysis*, i.e., the inferring of semantic or pragmatic relations holding between text segments. As long as "deep semantics" and knowledge processing are absent, the most useful information for such an analysis module comes from the *connectives*: closed-class lexical items that express, more or less specifically, the semantic or pragmatic relationship between text segments (or, more precisely, between their interpretations).

After illustrating the task of local coherence analysis with a simple example, we will enumerate the problems that connectives can create for text analysis, and then sketch an approach to automatic local coherence analysis that is embedded in the multi-level framework.

2 Looking back (1): Text Understanding in Artificial Intelligence

In the 1970s, automatic text understanding was one of the central goals of the flourishing discipline of Artificial Intelligence, which at the time aimed at reproducing human cognitive behaviour with machines employing symbolic representations and inference mechanisms. For a cognitive agent, understanding a text was largely conceived as aligning the text with the prior knowledge of that agent – a mechanism that was quite explicitly formulated in the *script* representations and alignment procedures in the tradition of Schank and Riesbeck (1981). As an extension of the popular paradigm of encoding static, factual knowledge with *semantic networks*, scripts were meant to represent an agent's procedural knowledge about stereotypical events. The best-known example is the *restaurant* script that encodes the steps of entering a restaurant; choosing and ordering food; eating; paying the bill, and leaving. One of the early programs, SAM, was able to match simple English stories against this generic script and thereby to "understand" a particular story about somebody eating something in some restaurant. It is important to notice that SAM did not perform anything like a syntactic analysis but directly matched surface patterns of English words against a meaning representation in the framework of *conceptual dependency* theory, an approach that aimed to represent word meaning (and in particular that of verbs) by decomposing it into semantic nets consisting of a set of *primitives* and relations between them. Consequently, programs like SAM always operated on carefully hand-crafted sample texts to which the (equally carefully handcrafted) conceptual dependency patterns would fit.

An important step forward from such toy settings was the FRUMP program by DeJong (1982). FRUMP in fact took news messages as input, which were — in contrast to SAM — not required to belong to a single small domain. Instead, FRUMP first inspected the text for possible topics and then actively selected the script which it surmised to be most suitable. For doing that, FRUMP drew on a set of 40 precoded scripts, which represented typical flows of reports on certain types of news. This difference in coverage made FRUMP much more impressive than SAM, but at the same time, the work in retrospect made it clear that the overall approach of pre-coding "story scripts" was a dead-end: In order to extend the program to further coverage, one would have to write many more scripts; moreover, the approach of "semantics-only" pattern matching was not very tolerant to mild deviations in the story and/or its linguistic formulation. Clearly, some general notion of paraphrasing was needed in order to detect that a great many linguistic variants could in effect report on the exact same event.

In the 1980s, The German LILOG project (Herzog and Rollinger, 1991) set out to avoid such fallacies by adopting a highly modular approach that clearly distinguished between knowledge sources such as syntax, lexical semantics, sentence meaning, and pre-

coded background knowledge. It performed a thorough syntactic and semantic analysis and linked the resulting meaning representations of sentences with a knowledge base encoding domain and world knowledge, in order to account for "text meaning" beyond the sentence. All modules were developed with intensively-researched formalisms, and much care was taken in devising the scheme of interaction between the modules. In this way, LILOG produced many important results on linguistic representation and processing, and it also lead to a working implementation — but this was able to analyze hardly more than one sample text that had been chosen beforehand as illustration of the phenomena that had to be tackled. In essence, the problem was that each module relied on the completeness and correctness on the output of those modules that preceded it in the processing pipeline, and since all modules obviously had their individual weaknesses and gaps of coverage, the *overall* coverage of the system was low. Thus, while adopting a much more principled and linguistically-minded approach than the aforementioned works in the conceptual dependency tradition, LILOG also ran into the problem that breadth of coverage was extremely difficult to achieve: the problem that has become well-known as that of *robustness*.

3 Looking back (2): Statistical Methods for Text-oriented Applications

What is today called the "statistical turn" of Computational Linguistics in the early 1990s was quite probably a consequence of two distinct developments: on the one hand the growing frustration with AI-style systems that sometimes produced interesting toy solutions but invariably failed to "scale up"; on the other hand the impressive results of acoustic speech recognition, which surfaced in the late 1980s and which were entirely based on statistical methods, with no linguistic representations involved.

Now, emphasis shifted to research that clearly aimed at practical applications, that was able to process realistic data, and that followed strict methodologies of evaluating one's work in quantitative manners. Accordingly, Computational Linguistics grew more into an engineering-like discipline, and towards the end of the decade, the term *Language Technology* became widely accepted as a label for efforts to bring language-processing applications into the "real world".

For text-oriented research, a significant milestone was produced with the *message understanding conference* (MUC) competitions sponsored by DARPA in the 1990s (Grishman and Sundheim, 1996). The goal was to identify specific pieces of information from authentic texts (news messages) that belonged to a particular genre, such as terrorist attacks. In this case, the software should be able to extract, e.g., the type of attack, its target, the purported agent(s), date and location of the attack. MUC was organized as an open competition for interested research teams; performance was evaluated by clear quantitative criteria, and the conferences generated both much attention and considerable progress. Quite soon, the evaluations were run not only with respect to overall performance but also for various subtasks involved, such as reference resolution. Hence, a team could also demonstrate its strength by focusing their attention on specific NLP tasks, which in turn lead to more sophisticated methods for handling such tasks. Subsequent events that

were run in a similar manner were the *Document Understanding Conference* (DUC) and the *Text Retrieval Conference* (TREC). Here, the original task of information extraction was extended to challenges on machine translation, cross-linguistic information retrieval, text summarization, or question answering.

For all these purposes, many approaches were developed, the vast majority of which was based almost exclusively on statistical methods, i.e., by training automatic classifiers on labelled data. From the applications-oriented perspective, this definitely lead to success: Open-domain question-answering, for instance, nowadays can be done to an extent that just a few years ago few would have thought to be possible. On the other hand, since the statistical models are largely intransparent "black boxes", and furthermore, the MUC/DUC/TREC modules have usually been extensively tailored to the particular domain and genre in question, not too many generalizable insights into the principles of document structure and text analysis have been gathered in this way. In short, while task performance increased significantly, the interest in (text-)linguistic insights has simultaneously shrunk.[1]

4 A Text-Technological Approach to Text Analysis

On the basis of the (necessarily subjective) recap of the history of automatic text analysis given above, we will now sketch the idea of a *text-technological* perspective, which tries to avoid the pitfalls of AI-inspired approaches (Section 1) and at the same time tries not to be as narrow-minded as many purely-statistical approaches (Section 2). Instead, we aim at viewing the task of text analysis as a matter of XML-based document processing steps that can work together in a manner as modular and flexible as possible.

4.1 Analysis on Multiple Levels

As is well-known, most text analysis applications nowadays share a certain base set of processing steps, such as identifying the logical document structure, performing part-of-speech tagging and possibly some sort of phrase chunking; on top of these results, more abstract and possibly more application-specific modules can be run. The idea of *multi-level analysis* is to implement this approach in a highly systematic way, i.e., by extending the text document step by step with linguistic and other information, which in turn can be used by further analysis steps to add even more information.

These additional modules may compute "classical" linguistic information such as syntactic structures, be it in the form of dependency trees or constituent structures. One task that needs to be performed in addition to standard parsing is *named-entity recognition*; while these entities usually correspond to linguistic constituents (noun phrases), they

[1] Granted, we are simplifying here: There has also been work on the listed applications with more linguistics or knowledge processing involved; see, for example, Hovy et al. (2002) for knowledge-based question answering. Nonetheless, it is certainly fair to say that in Language Technology, purely statistical work has a clear majority. One side-effect, demonstrated with a quantitative analysis by Reiter (2007), is the rapidly shrinking amount of research that methodologically is situated on the border to Psychology or Cognitive Science.

can be quite complex and are, by definition, not covered by the standard lexicon used by a parser. Then, however, there are several tasks in text analysis that do *not* correspond to levels of standard linguistic analysis. On the contrary, some of the issues are even responsible for the robustness problems of early syntactic parsers. For example, a module for identifying time and date expressions performs a very useful job for any application that needs to track the temporal unfolding of events; and precisely this class of time and date expressions was one prominent cause of disturbance for syntactic parsing based on symbolic grammars in the early 1980s (see also Stede (1992)). A well-known approach to annotating temporal expressions in English is the TimeML framework[2], which also extends to capturing the linguistic marking of event structure. For German, an inventory of temporal expressions has for instance been compiled as part of the *Verbmobil* project by Endriss et al. (1998). Building on this theoretical proposal, our implementation of a time/date analyzer (Luft, 2007) operates immediately on the tokenized text and contributes a new level of information independent of syntactic analysis. In subsequent analysis steps, the temporal expressions can be interpreted (i.e., projected onto the calendar) and then support any application that can profit from this information.

There are quite a few other examples of "non-standard" levels for representing information found in texts. For the applications of opinion mining or question-answering, for instance, it is important to gather from the text whose viewpoint is expressed in a certain passage: A particular piece of information might not be a statement of the author but in a more or less complicated fashion be attributed to someone else. The most obvious step here is to identify quoted speech and the speaker whom it is attributed to. With indirect speech and the many ways of expressing it in text, this becomes more complicated. The "ultimate" solution would go as far as tracking the point-of-view as it develops in the text, demonstrated with a rather complex algorithm for the narrative text type by Wiebe (1994). Our current implementation of an 'attibution recognizer' is much more modest and merely tries to recognize quoted as well as indirect speech. It identifies the former on the level of plain text tokens and the latter on the output of a dependency parser, using search patterns formed with communication verbs. Again, it contributes to the pool of analyses a new layer that marks attributed content and links it to the reported source of that information. This level can in turn be used by a generic rhetorical parser — see for instance Marcu (2000), who had pointed out the problems that attributed content poses for rhetorical parsing.

To give a third and final example, for many applications it is important to know whether a certain passage in a text is presented by the author as "objective" information or marked as a subjective statement, either on behalf of the author herself or on behalf of somebody mentioned in the text — in which case this task links up to that of viewpoint identification mentioned above. To make this more concrete, consider again the application of question-answering and suppose the user asked "When was Barack Obama born?" In order to answer this reliably, a system should not just match the keywords but be able to distinguish the follwing text passages:

[2] http://www.timeml.org

1. *Barack Hussein Obama, Jr. ([...]; born August 4, 1961) is the junior United States Senator from Illinois.* (en.wikipedia.org, March 3rd, 2008)

2. *Some Republicans claimed that Obama was born in 1965.* (fictitous)

3. *Obama was probably born on the Fourth of July in 1961.* (fictitous)

While (1) purports to "state the facts", (2) explicitly attributes the information on the year of birth to a third party (pardon the pun); with (3), finally, the author indicates that he is not quite sure whether the information he provides is actually correct. We use the term "subjectivity identification" for the overall task of noticing expressions of epistemic stance (as in (3)) and of discovering opinion, i.e., finding out whether the author merely presents a fact or indicates that she likes or dislikes a particular state of affairs.

There are several other useful examples of such intermediate processing tasks that are relevant not to all applications of text understanding but to many of them. Accordingly, tasks like these should not be implemented from scratch with any new application or research project but treated as independent modules that can contribute their share to the bigger task of making sense of a text. Thus, in a multi-level framework, "making sense" ist not an "all-or-nothing" endeavour; instead, the idea is to employ a set of specialized modules, some of which will run independently while some will build on the results of others, so that in the end all information gathered about a text can be read off a stack of separate analyses — and combining information from different stacks can in turn yield more information. Robustness arises when the modules do not break in case some information on "lower" levels is absent but produce either some underspecified representation, or gracefully move on to the next sentence, leaving a gap in the that particular analysis. Obviously, the resulting mixed-depth representations (cf. Section 1) are useful only to the extent that the higher-level modules can actually make use of them, by evaluating underspecifications or by ignoring gaps.

4.2 Processing Architectures

When it comes to actually implementing a multi-level approach, the text-technological perspective adds to the conceptual framework the technical notion of XML-based document processing, which plays a central role in enabling the interoperability of the various analysis modules. The principal challenge is to channel the same source document – possibly enriched with some annotations already – through various black-box analysis modules and to align the output of those modules so that all annotations are in fact correctly assigned to the intended spans of text. An early and successful implementation of the idea was the GATE system (Cunningham et al., 2002), which comes with a number of pre-installed components for analyzing English documents, but turns out somewhat cumbersome to use when "own" modules are to be added to the analysis pipeline. The more recent UIMA architecture (Götz and Suhre, 2004) by IBM follows similar goals as GATE but is more ambitious, targeting large-volume data processing of not only textual but also speech and video data. UIMA defines the XML interface that components must

adhere to, and then manages these components and the data flow between them, which essentially amounts to a pipeline architecture. Components need to be written in Java or C++, or wrappers need to be provided.

Prior to the release of UIMA, several smaller-scale approaches to XML-based document processing frameworks have been developed. The system developed in the 'Sekimo' project (Goecke et al., 2003) maps the information from a set of annotation layers to a Prolog fact base, upon which further computation can be done. For example, Lüngen et al. (2006) use this approach to build a chart parser that analyzes coherence relations between text spans (see next section). In the architecture developed at Potsdam University, a generic standoff-XML representation format called PAULA (Potsdamer AUstauschformat für Linguistische Annotation; Dipper (2005)) has been defined, along with conversion scripts that map the output of various annotation tools and analysis modules to PAULA (Chiarcos et al., 2008). In this approach, the aim is to use the same architecture both for the scenario of manual annotation and for automatic processing. Regarding the first goal, the linguistic database ANNIS serves to integrate multiple annotations of the same text (e.g., syntax, coreference, focus/background structure) and enables querying across levels, as well as some statistical analyses. ANNIS serves as the central repository for data collected within the *Sonderforschungsbereich* 632 'Information Structure' at Potsdam University and Humboldt-University Berlin, but is also available to external researchers.[3] The first version of ANNIS relied on a representation of the information as Java objects in main memory, which are designed to specifically support effective visualization and querying. At present, version 2 is under development, which adds an interface to a relational database management system (PostgreSQL) so that larger amounts of data can be handled.

Besides the scenario of manual annotation of linguistic data on multiple levels, the PAULA framework is also used in applications of automatic text processing. Our implementation of a 'Modular Text analysis System' (MOTS) currently integrates about a dozen different modules (both freely available tools and modules developed by ourselves) equipped with wrappers that ensure their compatibility with PAULA. One important aspect of wrapping is to ensure consistent tokenization: All analyses in the various layers (often transitively) refer to a unique 'token' layer, which in turn refers to the source document. Hence, for analysis modules that come with their own built-in tokenizer, a workaround must be defined as part of the wrapping. Our pilot application was a text summarizer built in the SUMMaR project (Stede et al., 2006), which combines the results of various statistical and symbolic analysis modules to compute an informative (extractive) summary of the input text. The MOTS workbench relies on two mechanisms: (1) a generic merging script that converts the PAULA standoff data to a standard inline XML representation, used for effective visualization of the various analysis results; (2) a Java API that allows uniform access to the PAULA data and permits construction of additional layers of analysis. One such layer we are constructing on the basis of "lower-level" input layers is that of local coherence analysis.

[3] `http://www.sfb632.uni-potsdam.de/ANNIS`

5 Local Coherence Analysis

Having described our general approach to automatic text analysis, we now focus our attention on one particular subtask: hypothesizing coherence relations between adjacent spans of text. This is a central aspect of computing text meaning with "deep" approaches, but it is also relevant for robust applications based on "surfacy" methods. For example, Marcu (2000) demonstrated that an algorithm using patterns operating on the surface string can to a certain extent identify the "rhetorical structure" of the text and use this information for the purpose of automatic summarization.

Within the step of coherence analysis, the key role is played by *connectives*: lexical units with a relational meaning that contribute to cohesion and coherence by indicating a connection between adjacent text segments. In the following, Subsection 5.1 first briefly introduces the task of local coherence analysis, and then 5.2 looks in more detail into the treatment of connectives, using a dedicated lexical resource holding information about them for a variety of purposes and applications (5.3). Finally, subsection 5.4 will discuss the embedding of coherence analysis in a multi-level analysis framework.

5.1 "Rhetorical Parsing": The Idea

Among discourse researchers, it is generally taken for granted that *coherence relations*, semantic or pragmatic relationships between (mostly adjacent) text segments, are (besides coreference) a central aspect of text coherence. Similarly, there is agreement that connectives are the primary linguistic means for signalling such relations at the linguistic surface. Views differ, however, on the number and definitions of relations, and also on the formal properties of the structures resulting from assigning coherence relations first to "minimal units" of text and then recursively to larger segments. For different views, see Polanyi (1988), Mann and Thompson (1988), Asher and Lascarides (2003), Wolf and Gibson (2005). For our purposes here, we focus on the role of coherence analysis within the larger enterprise of document processing. Some researchers take the position that coherence relations can be computed all the way from minimal units of analysis up to the document level, i.e., that the relations also hold between paragraphs, sections, and so forth. For certain types of text genre, this is certainly a feasible assumption, and Lüngen et al. (2006) have shown that this approach can be implemented to account for the structure of certain kinds of scientific papers. In general, however, it seems useful to distinguish between phenomena of *local* coherence (viz. semantic or pragmatic relationships between adjacent spans of text) and the *global* structure of a document. The latter is largely determined "top-down" by genre-dependent conventions and schema-like structuring principles; the former arises "bottom-up" when understanding the connections between clauses and sentences. We thus use the term 'local coherence analysis' for the task of identifying such connections, and regard this task as usually being applicable within paragraphs. Occasionally it may very well happen that some coherence relation applies to join neighbouring paragraphs — but in general, we surmise that different types of

relationships hold between larger units of text on the one hand, and between clauses and sentences on the other.

Turning then to the local level, our primary source of information are connectives. These are lexical items belonging to different morphosyntactic classes (conjunctions, adverbials, prepositions) that can either explicitly signal a semantic relationship between text segments (e.g., *nonetheless* quite clearly signals a concessive relationship) or merely invite the reader to construct the type of relationship (e.g., *but* signals some sort of adversarial relation such as contrast, concession, substitution, or correction; *and* is much more general in meaning). Linguists have undertaken many detailled investigations of individual connectives or groups thereof; for German, an invaluable resource to be mentioned here is (Pasch et al., 2003), which defines connectives by means of five features: they are not inflectable, do not assign case to their syntactic environment, denote two-place relations, take as arguments states of affairs *(Sachverhalte)*, which must be expressable as sentences. When, as several authors do, prepositions are added to the class of connectives, the second criterion needs to be weakened.

In automatic text analysis, connectives have been employed, *inter alia*, by Sumita et al. (1992), Corston-Oliver (1998), and Marcu (2000). Marcu's system, as mentioned above, operated on the text surface: He had rules for disambiguating punctuation symbols in order to segment the text into minimal units (sentences or clauses), and then associated connectives with coherence relations to obtain structures according to Rhetorical Structure Theory Mann and Thompson (1988). We illustrate the approach with this short text:[4].

> Because well-formed XML does not permit raw less-than signs and ampersands, if you use a character reference such as `&#60;` or the entity reference `<` to insert the < character, the formatter will output `<` or perhaps `&#60;`.

Supposing that we are able to identify the connectives and punctuation symbols correctly (here in particular: note that *to* is not a spatial preposition; distinguish between commas in enumerations and those finishing clauses), we can identify the "scaffold" of this short text as the following:

```
Because A, if B or C to D, E or F
```

with A to F representing the minimal units of analysis. Next, fairly simple rules will be sufficient to guess the most likely overall bracketing of this string:

```
(Because A, (if ((B or C) to D)), (E or F))
```

And finally, it happens that the connectives *because, if, to* and *or* are quite reliable signals of the coherence relations *Reason, Condition, Purpose* and *Disjunction*, respectively. Combining this information with the bracketing, we obtain the tree structure (in spirit of RST) shown in Figure 1.

Obviously, few texts behave as nicely as the one we just investigated. For one thing, it is known that most coherence relations are *not* explicitly signalled at the text surface. And furthermore, even if a connective is present, we have to reckon with several problems, to which we will attend in the next subsection.

[4]Source: `http://www.cafeconleche.org/books/bible2/chapters/ch17.html`

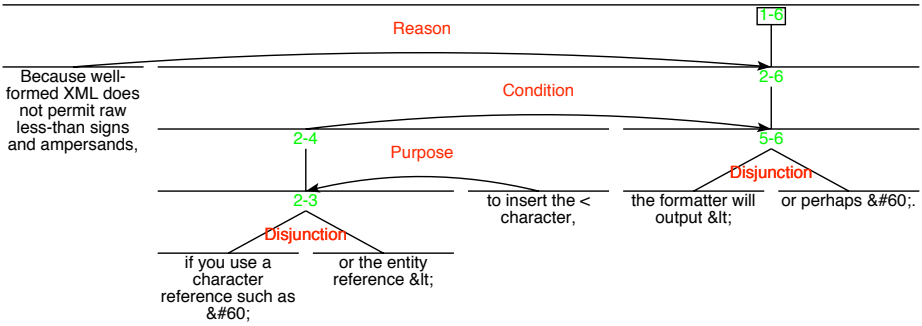

Figure 1: RST-style analysis of sample text

5.2 A Closer Look at Connectives

A closer investigation of the behaviour of connectives in texts reveals a range of complications that can disturb the very parse-friendly situation we encountered in the example above. In particular, we propose twelve phenomena that need to be accounted for; they are associated with the form and syntactic behaviour of connectives, with their basic meaning (in terms of coherence relations) and distinctive pragmatic features, and also with certain discourse-structural implications.

(1) Ambiguity: connective or not. Quite a few German words that can be used as connectives also have other, non-connective readings. For instance, *da* can be a causal subordinator (i.e., connective) but also a locative or temporal anaphor. This type of ambiguity is more widespread than one might think; in Dipper and Stede (2006), we report that 42 out of 135 frequent German connectives also have a non-connective reading, and we point out that many of the problems cannot be handled with off-the-shelf part-of-speech taggers. Hence, local coherence analysis is in need of a disambiguation step.

(2) Ambiguity: connectives can have more than one meaning, i.e., signal more than one coherence relation. For example, *schließlich* can close off a presentational list or enumeration; or it can indicate the ending of a temporal sequence of events; or it can be an argumentative marker conveying that a presented reason is definitive or self-evident. This also relates to the next point (3): Connectives can operate on different types of linguistic objects. A well-known distinction is that between *semantic* and *pragmatic* coherence relations, with the former holding between reported events in the world and the latter between speech acts performed by the interlocutor. In this regard, the temporal reading of *schließlich* conveys a semantic relation, while the argumentative one signals a pragmatic move. Many authors, however, prefer a tripartite distinction along

the lines of Sweetser (1990), who offers *content*, *epistemic* and *speech-act* relations — see, for example, Blühdorn (2005). (4) Some further pragmatic distinctions are usually not modelled as a difference in coherence relation; a well-known case in point is the difference between *because* and *since*, where only the latter has a tendency to mark the following information as hearer-old (not necessarily discourse-old). Also, connectives can convey largely the same information yet differ in terms of stylistic nuances, for instance in degree of formality; thus a concessive relation in English may be signalled in a rather formal way by using a *notwithstanding* construction.

(5) A feature that is somewhat easier to isolate is the (non-)ability of connectives to occur within the scope of focus particles. An example from German is: *Nur weil es regnet, nehme ich das Auto. / ?Nur da es regnet, nehme ich das Auto.* However, a complication is lurking here as well, because this feature borders on (6) the issue of connectives having two parts. Clear cases are *either .. or* and *if .. then*. For the German version *wenn .. dann*, though, a coherence analyzer must account for the possibility of its occurring in reverse order: *Dann nehme ich eben das Auto, wenn Du so bettelst.* Now, looking at highly frequent collocations such as *even though* or *even if*, it is difficult to decide whether we are dealing with a single-word connective and a focus particle, or with a complex connective; one solution is to check in such cases whether the meaning is in fact derived compositionally and in that case to prefer the focus particle analysis. Next, from "regular" two-word connectives it is a small step to (7) the shady area of *phrasal* connectives, which can allow for almost open-ended variation and modification: *aus diesem Grund / aus diesen Gründen / aus all diesen guten Gründen / ...*

Turning to structural questions, one well-known complication is (8) the embedding of discourse units into one another, which is problematic for straightforward tree representations of text structure: *Gestern habe ich, weil ich etwas krank war, keinen Spaziergang gemacht.* Besides embedding, connectives can (9) occasionally link text segments that are non-adjacent — a phenomenon that has been studied intensively by Wolf and Gibson (2005) and also by Webber et al. (2003). An example from Webber et al.: *John loves Barolo. So he ordered three cases of the '97. But he had to cancel the order because then he discovered he was broke.* Here, the *then* is to be understood as linking the discovery event back to the ordering event rather than to the (adjacent) cancelling. Non-adjacency also leads to the issue of crossing dependencies, which is discussed, *inter alia*, by the two teams of authors mentioned above. It correlates with the problem (10) of two connectives occurring in the same clause, as also exemplified in the *Barolo* example *(because then*), which renders the parsing task significantly more complex than in our "ideal" example of the previous subsection. A slightly different problem is to be found in situations where (11) a coherence relations is signalled twice, by two different connectives, where one typically is to be read cataphorically: *Ich nehme deshalb das Auto, weil Du so bettelst.* This is not quite the same as the two-word connectives in (6), and a coherence analyzer will have to be very careful not to hypothesize two separate causal relationships in such examples.

Finally, (12) certain connectives convey information about the discourse structure *beyond* the local relation between two segments. A case in point is the first word of this paragraph, which not only makes a 'List' or 'Enumeration' relation explicit, but also

provides the information that this very list is now coming to an end. A smart coherence analyzer could thus reduce the search space for linking the subsequent text segment — knowing that it will definitely *not* be part of the same 'List' configuration.

5.3 A Declarative Resource: Discourse Marker Lexicon

The catalogue of potential difficulties given in the previous subsection has demonstrated that local coherence analysis based on connectives can be a quite complicated task. Depending on the specific goals of the overall intended application, not all of the problems will always be relevant or critical, but in general it seems advisable to design an approach that is in principle prepared to deal with such issues. The proposal here is to make use of a declarative resource — a lexicon — for assembling all kinds of information about connectives, and then to have a coherence analysis module peruse as much of this information as it possibly can (or needs). The first version of our *Discourse Marker Lexicon* (DIMLEX) was described in (Stede and Umbach, 1998); it was used at the time for relatively simple "rhetorical parsing" as outlined in Section 5.1 and also for a language generation application, where the task is to select for a given coherence relation (as determined by the text planner) a connective that can suitably express that relation in the linguistic context generated so far. To deal with such diverse applications, we followed the basic idea to encode the information in the XML-based DIMLEX in a rather abstract fashion, and to use XSLT scripts to transform it into application-specific versions. During this step, some information from DIMLEX will be ignored, other will be converted to a specific format. For instance, our first rhetorical parser needed the information in Prolog format, while the generator needed Lisp notation; the XSLT scripts produced both of these (and, in addition, HTML versions for visualization purposes) from the "master lexicon".

More recently, Dipper and Stede (2006) worked on the problem of disambiguating (non-)connective uses (complication (1) in the previous subsection), and found that to a large extent, it can be solved by inspecting the local part-of-speech context; we have begun to add corresponding disambiguation rules to DIMLEX. Furthermore, in an ongoing joint project with a group from IDS Mannheim, we are analyzing especially the *causal* connectives in greater detail than we had done before. The results are also being incorporated into the lexicon.

At the moment, a DIMLEX entry holds the following information: (1) ortographic variants of the connective; (2) non-/connective disambiguation rules in the shape of weighted part-of-speech sequences that either favour the connective reading (positive weights) or the non-connective reading (negative weights); (3) connective can be in the scope of a focus particle (yes/no); (4) connective can function as a "correlate" to a different connective (yes/no, and which kinds); (5) syntactic category and features, esp. constraints on positioning within the sentence (represented along the lines of Pasch et al. (2003)) and constraints on the linear order of the *internal* segment (the one containing the connective) and the *external* one; (6) semantic readings: coherence relations that the connective can signal, along with disambiguation information (see below); (7) argument

linking: mapping between internal/external arguments and the 'thematic roles' (example: for *although*, the internal argument is the 'Conceded', the external argument the 'Anyway' participant); (8) semantic or pragmatic features to make fine-grained distinctions between similar connectives.

The information for disambiguating between different coherence relations (6) is, similar to (2), represented as weighted rules. The features used here largely pertain to position, tense and aspect of the clause, mood and modality, or lexical collocations. Weights are derived by corpus analyses and thus reflect the probability that a certain feature co-occurs (or does not co-occur) with a certain relation.

Furthermore, several areas in DIMLEX contain linguistic examples to illustrate the relevant distinctions. The information in DIMLEX is formalized to different degrees: Areas under development consist of natural language descriptions that are gradually being turned into interpretable attribute/value representations when the descriptions become stable. Areas that have reached this stage can be translated, e.g., via XSLT, to a specific application-oriented lexical resource, such as one supporting a local coherence analyzer.

5.4 Local Coherence Analysis in a Multi-Level Approach

Finally, we describe our ongoing re-implementation of the coherence analysis module mentioned earlier, now couched in the multi-level framework, using PAULA representations and the Java API (see Section 4.2). For reasons that should have become clear, we regard automatic coherence analysis as a task that can only be partially solved by current language technology, and accordingly, we view connective-based analysis as one contribution to this effort. Thus our module will generate hypotheses of coherence relations and related spans, solely on the basis of connectives occuring in the text. This information is represented in two PAULA layers and may later be combined with the results of other modules contributing to the task — for instance a module checking for lexical cohesion in order to hypothesize Elaboration relationships, which typically are not signalled by connectives. Modules following in the processing chain may combine the various hypotheses into the most likely overall relational tree structure for the paragraph (or a set of such tree structures, see (Reitter and Stede, 2003)), or they may use the hypotheses directly for some application purpose such as text summarization or question-answering, without relying on a spanning tree.

The first step consists in identifying the connectives: Words listed in DIMLEX are isolated in the text (including a check for complex connectives, i.e., two corresponding words in adjacent clauses), and the disambiguation patterns are checked against the PoS layer. A new layer is created, holding those words that were recognized as connectives. Next, the segmentation module tries to identify the minimal discourse units. It uses three layers: the results of sentence splitting and those of dependency parsing in order to identify clauses functioning as separate units; furthermore, prepositional phrases are isolated as minimal units when their head corresponds to a connective as recorded on the newly created connective layer. The minimal units are in turn represented as a separate layer in PAULA.

Next, the connective layer is extended with information on relations and scopes: Every connective is associated with one or more atrribute-value structures listing possible coherence relations along with probabilities, as well as one or more scope assignments, that is mappings between the relation's thematic roles and lists of minimal unit identifiers. All relations stored with the connective in DIMLEX are recorded as hypotheses, and probabilities added as the result of evaluating the associated disambiguation rules, which largely operate on the syntax layer. For example, for the connective *schließlich* we found that with the main verb of the clause elided, the Reason reading is very unlikely; on the other hand, if the verb is in present tense and the Aktionsart is state, it very likely signals Reason. All rules of this kind are being checked for each connective and a ranking of the associated coherence relations is determined by accumulating the weights.

Finally, for each relation we also hypothesize its scope: The thematic roles are associated with lists of minimal units. Also, in this step we check for the possibilities of correlates: when two connectives appear in the same sentence and can signal the same relation, and (according to the DIMLEX entry) one could be a correlate of the other, it is marked as such. Scope determination is usually straightforward for coordinating and subordinating conjunctions. For adverbials, we typically hypothesize different solutions and rank them according to size: The most narrow interpretation is taken as most likely. In this step, we consider the analysis layer of logical document structure in order to disprefer segments that would stretch across paragraphs or other kinds of boundaries. Similarly, a layer with the results of "text tiling" (breakdown of the text in terms of thematic units, in the tradition of Hearst (1994)) can be used for this purpose, as well as as an "attribution" layer that identifies those modal contexts that attribute a span of text to a particular source (as in indirect speech).

For illustration, we consider the first half of a text from the *Potsdam Commentary Corpus* (Stede, 2004), which argues against preserving a disputed building in Berlin, the so-called "Steglitzer Kreisel" (see Figure 2). Connective identification would isolate the words that are set in italics, resulting in the "scaffold" for the text:

```
(1). Selbst wenn (2). (3). Aber (4). (5). Nicht nur (6a),
sondern (6b). Zwar (7). Aber (8).
```

Selbst wenn is analyzed as a connective with focus particle; since no *dann* is present in the subsequent clause, the (infrequent) linear order *"A. Selbst wenn B."* can be recognized as such. The text contains two examples of complex connectives: *nicht nur . . . sondern* in (6), and *zwar . . . aber* in (7/8). The former suggests only one scope assignment; as for the latter, one segment is trivially the *zwar*-sentence, while the other probably ends with (8), but it might also extend beyond that sentence. Thus we need to represent alternative scope assignments. The same holds for the *aber* in (4): The left segment obviously includes (3) but can extend to (2) and (1). Similarly, the right segment can consist of merely (4) or more — the connective analysis procedure cannot make a decision here but has to represent the (weighted) alternatives. As for the relations, we (depending of course on the inventory we use) encounter ambiguities for *aber* and *zwar aber*; in terms of Mann and Thompson (1988), the former can signal Contrast, Antithesis or Concession; the latter only Antithesis and Concession (*zwar* clearly marks a satellite in RST terms, and thus the

(1) Alles spricht gegen den Steglitzer Kreisel. (2) *Selbst wenn* man vergisst, dass der olle Schuhkarton in bester Lage einst ein privates Prestigeobjekt war, das der öffentlichen Hand für teures Geld aufgenötigt wurde. (3) Ein Symbol der West-Berliner Filzwirtschaft in den späten sechziger Jahren. (4) *Aber* lassen wir das ruhig beiseite. (5) Der Kreisel ist Asbest verseucht. (6) *Nicht nur* hier und da, *sondern* durch und durch. (7) *Zwar* könnte man, wie beim Palast der Republik, den Bau bis aufs wackelige Stahlskelett entkleiden und neu aufbauen. (8) *Aber* das würde mindestens 84 Millionen Euro, vielleicht auch das Doppelte kosten. (…)

Figure 2: Excerpt from sample text (Source: *Tagesspiegel*). Sentence numbers added for reference.

multinuclear Contrast does not apply). In these cases, the disambiguation procedure will not be able to distinguish between these (very similar) relations.

6 Looking back and forth – Conclusion

Having described the multi-level approach as a text-technological view on the overall problem of text analysis, we can now compare it to the 'historical' perspectives characterized in Sections 2 and 3. First, notice that the MLA approach is not in conflict with knowledge-intensive processes: ontologies, inference engines, and/or rich lexical resources can of course contribute to modules analyzing phenomena like coreference between definite NPs, coherence relations, etc. There is, however, a sharp contrast to the "knowledge-only" approaches in the early Conceptual Dependency tradition: MLA is based on the assumptions that text analysis should not be an all-or-nothing step, and that it should not *necessarily* rely on pre-coded knowledge of the domain or the world-at-large. The radical modularity and emphasis on re-combinability of modules also differentiates MLA from LILOG-style pipeline architectures, where interfaces between subsequent modules were fine-tuned in order to achieve a smooth interaction of, for instance, sentence syntax and subsequent semantic interpretation, so that certain interesting linguistic phenomena could be handled. MLA instead emphasizes the independence of modules, which entails that more difficult work needs to be done by the "high-level" modules when combining the, possibly partial or even conflicting, results of "lower-level" modules. An approach that, like LILOG, focuses on the linguistic (sentence-based) analysis but shares many of MLA's goals is the 'Heart of Gold' architecture (Schäfer, 2007), which integrates tagging, chunking, parsing and other modules, and represents the — possibly underspecified — results using minimal-recursion semantics (Copestake et al., 2005).

However, MLA (being a text-technological conception) aims to cover text documents as a whole and thus to also account for (logical and content-based) document structure and for phenomena of discourse structure. As a case in point, we took the task of

local coherence analysis, which on the one hand builds upon other analysis levels (part-of-speech tagging, syntactic analysis, logical document structure, text tiling) and then contributes a new level consisting of two technical layers: One dividing the text into a sequence of minimal units, the other identifying the connectives and associating them with (possibly sets of) coherence relations and scope assignments. Thus we do not aim at constructing a full "discourse tree" in this step. For this more ambitious task, other modules could contribute their share of information, such as a lexical-cohesion module delivering hypotheses on Elaboration relations. Then, one can join the accumulated information into the most likely overall tree for a paragraph, or alternatively peruse the individual pieces of information in isolation.

The main advantage of MLA as described here is that of "radical modularity": A framework such as that of PAULA makes it very simple to add a new module or to replace an existing one with a better one fulfilling the same function. Or, for that matter, to employ several alternative modules with the same function (e.g., part-of-speech taggers) in order to evaluate them in the context of a specific application, or to implement voting schemes for a particular level of analysis. At the same time, this great flexibility comes with a price tag: Processing many levels of standoff-XML annotations is computationally expensive. For specific applications, effective programming-language-specific APIs thus play an important role in reducing the need for traversing the graph structures with generic XML processing tools. As pointed out above, we have developed a Java API for the PAULA framework; further APIs for script languages are in preparation.

References

Asher, N. and Lascarides, A. (2003). *Logics of Conversation*. Cambridge University Press, Cambridge.

Blühdorn, H. (2005). Zur Semantik kausaler Satzverbindungen: Integration, Fokussierung, Definitheit und modale Umgebung. *Studi Linguistici e Filologici Online*, 3(2):311–338.

Chiarcos, C., Dipper, S., Götze, M., Ritz, J., and Stede, M. (2008). A flexible framework for integrating annotations from different tools and tagsets. In *Proc. of the First International Conference on Global Interoperability for Language Resources*, Hongkong.

Copestake, A., Flickinger, D., Sag, I., and Pollard, C. (2005). Minimal recursion semantics: An introduction. *Journal of Research on Language and Computation*, 3(3):281–332.

Corston-Oliver, S. (1998). *Computing of Representations of the Structure of Written Discourse*. PhD thesis, University of California at Santa Barbara.

Cunningham, H., Maynard, D., Bontcheva, K., and Tablan, V. (2002). GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*.

DeJong, G. (1982). An overview of the FRUMP system. In Lehnert, W. and Ringle, M., editors, *Strategies for Natural Language Processing*, pages 149–176. Lawrence Erlbaum Associates, Hillsdale/NJ.

Dipper, S. (2005). XML-based stand-off representation and exploitation of multi-level linguistic annotation. In Eckstein, R. and Tolksdorf, R., editors, *Proceedings of Berliner XML Tage*, pages 39–50.

Dipper, S. and Stede, M. (2006). Disambiguating potential connectives. In Butt, M., editor, *Proceedings of KONVENS '06*, pages 167–173, Konstanz.

Endriss, U., Küssner, U., and Stede, M. (1998). Repräsentation zeitlicher Ausdrücke: Die Temporal Expression Language. Verbmobil Memo 133, Technical University Berlin, Department of Computer Science.

Goecke, D., Naber, D., and Witt, A. (2003). Query von Multiebenen-annotierten XML-dokumenten mit Prolog. In Seewald-Heeg, U., editor, *Sprachtechnologie für die multilinguale Kommunikation*. Gardez! Verlag, Sankt Augustin.

Götz, T. and Suhre, O. (2004). Design and implementation of the UIMA common analysis system. *IBM Systems Journal*, 43(3).

Grishman, R. and Sundheim, B. (1996). Message understanding conference - 6: A brief history. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pages 466–471, Kopenhagen.

Hearst, M. A. (1994). Multi-paragraph segmentation of expository text. In *Proceedings of the 32nd Meeting of the Association for Computational Linguistics*, Las Cruces/NM.

Herzog, O. and Rollinger, C.-R., editors (1991). *Text Understanding in LILOG: Integrating Computational Linguistics and Artificial Intelligence*. Springer, Berlin/Heidelberg.

Hirst, G. and Ryan, M. (1992). Mixed-depth representations for natural language text. In Jacobs, P., editor, *Text-Based Intelligent Systems*, pages 59–82. Lawrence Erlbaum Associates, Hillsdale/NJ.

Hovy, E., Hermjakob, U., Lin, C.-Y., and Ravichandran, D. (2002). Using knowledge to facilitate pinpointing of factoid answers. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*, Taipei.

Lobin, H. (2000). *Informationsmodellierung in XML und SGML*. Springer, Berlin/Heidelberg.

Luft, A. (2007). Automatische erkennung und annotierung der temporalen struktur in texten. Diplomarbeit, Hochschule Mittweida, Fachbereich Mathematik/Physik/Informatik.

Lüngen, H., Lobin, H., Bärenfänger, M., Hilbert, M., and Puskas, C. (2006). Text parsing of a complex genre. In Martens, B. and Dobreva, M., editors, *Proc. of the Conference on Electronic Publishing (ELPUB 2006)*, Bansko, Bulgaria.

Mann, W. and Thompson, S. (1988). Rhetorical structure theory: Towards a functional theory of text organization. *TEXT*, 8:243–281.

Marcu, D. (2000). *The theory and practice of discourse parsing and summarization*. MIT Press, Cambridge/MA.

Pasch, R., Braußе, U., Breindl, E., and Waßner, U. H. (2003). *Handbuch der deutschen Konnektoren*. Walter de Gruyter, Berlin/New York.

Polanyi, L. (1988). A formal model of the structure of discourse. *Journal of Pragmatics*, 12:601–638.

Reiter, E. (2007). Last word: The shrinking horizons of computational linguistics. *Computational Linguistics*, 33(2):283–287.

Reitter, D. and Stede, M. (2003). Step by step: underspecified markup in incremental rhetorical analysis. In *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC)*, Budapest.

Schäfer, U. (2007). *Integrating Deep and Shallow Natural Language Processing Components - Representations and Hybrid Architectures*. PhD thesis, Saarland University. Vol. 22 of the Saarbrücken Dissertation Series in Computational Linguistics and Language Technology.

Schank, R. C. and Riesbeck, C. K. (1981). *Inside Computer Understanding: Five Programs Plus Miniatures*. Lawrence Erlbaum Associates, Hillsdale/NJ.

Stede, M. (1992). The search for robustness in natural language understanding. *Artificial Intelligence Review*, 6(4):383–414.

Stede, M. (2004). The Potsdam commentary corpus. In *Proceedings of the ACL Workshop on Discourse Annotation*, pages 96–102, Barcelona.

Stede, M., Bieler, H., Dipper, S., and Suryiawongkul, A. (2006). SUMMaR: Combining linguistics and statistics for text summarization. In *Proceedings of the European Conference on Artificial Intelligence*, Riva del Garda.

Stede, M. and Umbach, C. (1998). DiMLex: A lexicon of discourse markers for text generation and understanding. In *Proceedings of the 17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics (COLING/ACL'98)*, pages 1238–1242, Montreal, Canada.

Sumita, K., Ono, K., Chino, T., Ukita, T., and Amano, S. (1992). A discourse structure analyzer for Japanese text. In *Proceedings of the International Conference on Fifth Generation Computer Systems*, pages 1133–1140.

Sweetser, E. (1990). *From etymology to pragmatics*. Cambridge University Press, Cambridge.

Webber, B., Stone, M., Joshi, A., and Knott, A. (2003). Anaphora and discourse structure. *Computational Linguistics*, 29(4):545–587.

Wiebe, J. (1994). Tracking point of view in narrative. *Computational Linguistics*, 20(2):233–287.

Wolf, F. and Gibson, E. (2005). Representing discourse coherence: a corpus-based study. *Computational Linguistics*, 31(2):249–287.