

Automatische Textannotation: Ein SGML- und DSSSL-basierter Ansatz zur angewandten Textlinguistik

Georg Rehm
Institut für Semantische Informationsverarbeitung
Universität Osnabrück

`georg@cl-ki.uni-osnabrueck.de`
<http://www.cl-ki.uni-osnabrueck.de/~georg/>

Minimal überarbeitete Version vom 19. Oktober 1998

Erschienen in: Henning Lobin (Hrsg.): *Text im digitalen Medium – Linguistische Aspekte von Textdesign, Texttechnologie und Hypertext Engineering*, Wiesbaden: Westdeutscher Verlag, 1999. S. 179–195.

1 Einführung

Dieser Beitrag beschreibt ein System zur automatischen Annotation von vorformatierten deutschsprachigen Zeitungstexten. Die Texte werden hierbei zunächst in eine auf der Standard Generalized Markup Language (SGML, siehe ISO8879, 1986; Goldfarb, 1990) basierende Repräsentation überführt und daraufhin mit Hilfe der Document Style Semantics and Specification Language (DSSSL) in Hypertext-Dokumente konvertiert. Ein dynamisch arbeitendes World Wide Web-Frontend, das mittels DSSSL auf den SGML-Quellen der HTML-Dokumente operiert, erlaubt dem Benutzer die Hervorhebung von verschiedenen Informationen, z. B. Satzgrenzen oder rhetorischen Relationen. Der Fokus der Arbeit liegt auf dem Aufbau von rhetorischen Strukturbäumen, die gänzlich auf Oberflächen-Informationen, sogenannten Diskursmarkern, basieren. An Beispielen wird gezeigt, daß es möglich ist, zumindest den groben rhetorischen Aufbau von einzelnen Paragraphen zu bestimmen.

In Abschnitt 2 gehe ich auf rhetorische Relationen und Diskursmarker sowie die Motivation für die vorliegende Arbeit ein. Darauf folgt eine Beschreibung des Diskursmarker-Lexikons, das in dem später vorgestellten Prototypen Verwendung findet. Abschnitt 3 beschäftigt sich mit der Document Style Semantics and Specification Language (DSSSL), die die Grundlage für die transformierenden Operationen auf den SGML-Dokumenten bildet.¹ Nach einer Beschreibung des verwendeten Korpus folgt in Abschnitt 5 ein Überblick über die Funktionalität des Systems. Daraufhin gehe ich auf einige implementatorische Details ein und stelle eine Evaluation des Systems hinsichtlich der syntaktischen Korrektheit der SGML Annotation vor.

2 Rhetorische Relationen und Diskursmarker

Zwischen den verschiedenen Teilen eines Textes oder Diskurses, seien es Nebensätze, Sätze, Abschnitte oder Äußerungen, bestehen rhetorische Relationen, die maßgeblich an der Bildung der Textkohärenz beteiligt sind (vgl. in diesem Zusammenhang etwa Hobbs, 1979; Rickheit, 1991; Hovy, 1993). Eine populäre Theorie zur Beschreibung dieser Relationen ist die Rhetorical Structure Theory (RST, vgl. Mann und Thompson, 1987, 1988; Mann et al., 1989; Lobin, 1998a, in diesem Band, beschreibt die wesentlichen Konzepte, weshalb ich an dieser Stelle auf eine Einführung in die RST verzichten möchte).

Ausgangspunkt meiner Arbeit war der Versuch, die von Marcu (1996, 1997) und Ono et al. (1994) und Miike et al. (1994) vorgeschlagenen Ansätze zur automatischen Erstellung von Zusammenfassungen englischer bzw. japanischer Texte für das Deutsche anzuwenden. Marcu schlägt ein auf der RST basierendes rhetorisches Parsing vor,

¹Für eine Einführung in die (computer)linguistische Arbeit mit SGML siehe Witt (1998), in diesem Band.

das ausschließlich auf speziellen lexikogrammatikalischen Konstruktionen – sog. *cue words*, also diskursmarkierende Wörter und Phrasen – beruht. Auf eine ähnliche Weise verfahren Ono et al. mit japanischen Texten.

Diskursmarker besitzen die Eigenschaft, explizit bestimmte rhetorische Relationen zu signalisieren. Marcu nennt in diesem Zusammenhang beispielsweise *although*, das üblicherweise von der Relation CONCESSION begleitet wird. Insgesamt führt Marcu 461 verschiedene diskursmarkierende Wörter und Phrasen an, die er im Rahmen seines rhetorischen Parsings für den Aufbau von rhetorischen Strukturbäumen ausnutzt. Der Erfolg dieses Ansatzes zeigt, daß sich viele rhetorische Relationen aufgrund von Oberflächeninformationen inferieren lassen, was die Aussagen der Entwickler der RST hinsichtlich dieser Thematik relativiert: „We have found no reliable, unambiguous signals for any of the relations“ (Mann und Thompson, 1988, S. 250) und „The assumption that text structuring relations are lexical is an alternative, not compatible with RST.“ (Mann et al., 1989, S. 9).

Als äußerst problematisch hat sich bei der Anwendung des von Marcu und Ono et al. gewählten Ansatzes für das Deutsche die Tatsache herausgestellt, daß für diese Sprache bislang keine systematischen Untersuchungen oder gar Sammlungen der diskursmarkierenden Wörter und Phrasen existieren. Zwar beschäftigen sich derzeit zwei Projekte mit dieser komplexen Thematik, jedoch liegen hier noch keine empirischen Ergebnisse vor.² Aus diesem Grund habe ich die von Marcu (1997, S. 273–286) angegebene Liste für das Englische mit Hilfe eines Wörterbuchs (Breitsprecher et al., 1988) übersetzt, um auf diese Weise eine rudimentäre Liste von Diskursmarkern für das Deutsche zu erstellen. Von Marcu aufgeführte Wörter, die in der Übersetzung diskontinuierliche Phrasen ergeben hätten, wurden dabei außer Acht gelassen, da diese nicht ohne weiteres mit auf der Oberfläche operierenden Methoden hätten erfaßt werden können. Daraufhin wurde diese Liste mit verschiedenen noch nicht enthaltenen Partikeln (u.a. Fokus-, Modal- und Gradpartikeln), Adverbien und Konjunktionen aus Drosdowski (1995) ergänzt und mittels Synonymwörterbuch (Müller, 1997) erweitert. Einige der aufgenommenen Synonyme stammen aus dem entsprechenden Lexikon des LILOG-Systems, vgl. etwa Emde (1991).

Ein Teil der aufgebauten Liste von Diskursmarkern stammt auch aus eigenen Korpusanalysen: Da der in den Abschnitten 5 und 6 vorgestellte Prototyp mit einer sehr hohen Genauigkeit die Satzgrenzen der im Korpus enthaltenen Texte bestimmen kann, war eine Auflistung der häufig am Satzanfang und direkt nach einem Interpunktionszeichen auftauchenden Wörter – an diesen Positionen befinden sich häufig Diskurs-

²Das Institut für deutsche Sprache erarbeitet im Projekt „Handbuch der deutschen Konnektoren“ eine umfangreiche Liste von Diskursmarkern (vgl. <http://www.ids-mannheim.de/gra/konhome.html>), und das Projekt „KIT-Marker“ der Technischen Universität Berlin beschäftigt sich mit dem Aufbau eines Diskursmarker-Lexikons, das vor allem in der automatischen Textgenerierung Einsatz finden soll, vgl. <http://flp.cs.tu-berlin.de/kit/markerD.html> und Stede und Umbach (1998).

marker – mit einfachen Mitteln zu realisieren.³

Als heikel hat sich daraufhin die Zuordnung von rhetorischen Relationen zu Diskursmarkern erwiesen. Diese konnte natürlich in vielen Fällen nicht eindeutig vorgenommen werden, da vielerlei Ambiguitäten vorhanden sind, beispielsweise kann *als* die Relation COMPARISON begleiten oder auch als temporale Konjunktion fungieren. Kritisch waren auch viele Zuordnungen, bei denen entweder SEQUENCE oder ELABORATION als mögliche Relationen in Frage kamen. Die Unterscheidung dieser ähnlichen Relationen spielt für die Funktionalität des Systems keine Rolle, aber bei verschiedenen Diskursmarkern war die Zuordnung in eine der beiden Relationen eindeutig zu treffen, weshalb trotzdem beide Relationen unterschieden werden. Aus den genannten Gründen darf die von mir vorgenommene eindeutige Zuordnung (zu jedem Marker existiert genau eine mögliche Relation) von Diskursmarkern zu rhetorischen Relationen keinesfalls als „in jeder Hinsicht plausibel“ oder gar „endgültig“ mißverstanden werden, vielmehr handelt es sich um eine ad hoc durchgeführte Abbildung, die meinen Anforderungen größtenteils genügt.

Auch die Wahl der Relationen, die erkannt werden sollten, war problematisch. Nach einer Analyse der Liste von Diskursmarkern hat sich eine Menge von 17 verschiedenen Relationen ergeben, die größtenteils auf den Vorschlägen von Mann und Thompson (1987) basieren, z. B. CONDITION, EVIDENCE oder CONSEQUENCE. Fokus-, Modal- und Gradpartikeln sind bei der Erkennung von rhetorischen Relationen sehr hilfreich, konnten jedoch oftmals nicht eindeutig zugeordnet werden, weshalb diese Wortgruppen explizit als Partikeln gekennzeichnet werden. Tabelle 1 zeigt die Anzahl der für die verschiedenen Relationen verwendeten Diskursmarker sowie die Vorkommen der durch diskursmarkierende Wörter bzw. Phrasen erkannten Relationen im Korpus; In Rehm (1998) befindet sich eine vollständige Auflistung der benutzten Diskursmarker sowie weitere Details zur Implementation.

3 Die Document Style Semantics and Specification Language

Die Document Style Semantics and Specification Language (DSSSL, siehe ISO10179, 1996) definiert verschiedene Prozesse zur Manipulation und Formatierung von SGML-Dokumenten. In einem ersten Schritt können SGML-Dokumente hierbei transformiert werden, um sie daraufhin mit Hilfe einer Stil-Spezifikation zu formatieren. Die Norm ISO 10179 definiert für diese Vorgänge zwei auf dem Lisp-Dialekt Scheme basierende Programmiersprachen, die Transformation Language und die Style Language. Beide Sprachen teilen sich eine Menge von Grundfunktionen, die in der sog. Expression

³Hierzu wurde das von Jani Jaakkola und Pekka Kilpelainen entwickelte Werkzeug `sgrep` – eine SGML unterstützende Variante des UNIX-Tools `grep` – in Verbindung mit einem Shell Skript eingesetzt, siehe <http://www.cs.helsinki.fi/~jjaakkol/sgrep.html>.

Relation	Diskursmarkierende Wörter		Diskursmarkierende Phrasen	
	Anzahl	Vorkommen	Anzahl	Vorkommen
CAUSE	19	5073	12	28
COMPARISON	10	423	7	88
CONCESSION	18	1832	3	92
CONDITION	11	2222	6	13
CONSEQUENCE	9	423	16	61
CONTRAST	14	1900	9	97
ELABORATION	69	12672	54	444
EVIDENCE	9	78	3	3
EXAMPLE	2	21	2	54
PRESENTATIONAL SEQUENCE	5	42	8	280
RESTATEMENT	5	11	4	5
SEQUENCE	88	9850	24	1
SUMMARY	4	545	10	25
FOKUSPARTIKEL	46	1919	5	6
GRADPARTIKEL	54	1043	8	345
MODALPARTIKEL	19	1693	1	13
TEMPORAL	119	6489	18	156
Σ	492	45813	173	1650

Tabelle 1: Anzahl der im Prototyp verwendeten rhetorischen Relationen sowie die jeweiligen Vorkommen im Korpus

Language definiert sind. Weiterhin erlaubt DSSSL keine Seiteneffekte, d. h. es existieren keine globalen Variablen, keine Routinen zur Ein- und Ausgabe etc.

Die Transformation Language gestattet, neue SGML-Strukturen zu erzeugen, Strukturen zu tilgen oder existente Strukturen neu anzuordnen. Es findet also eine Transformation einer zu einer gegebenen Dokumenttyp-Definition (DTD) konformen Dokument-Instanz statt, so daß sie nach Beendigung der Transformation einer anderen DTD zugehörig ist. Über die Style Language kann man, vereinfacht gesagt, SGML-Elementen typographische Eigenschaften zuweisen, um beispielsweise ein Element `<UEBERSCHRIFT>` in einem serifenlosen, 18 Punkte großen Font zentriert anzuordnen.⁴ Da die Style Language für das Verständnis des in den Abschnitten 5 und 6 vorgestellten Prototyps nicht von Belang ist, möchte ich auf eine weitere Erläuterung der relevanten Mechanismen an dieser Stelle verzichten und stattdessen auf die Literatur verweisen: ISO10179 (1996), Macherius (1997), Prescod (1997) und Behme und Mintert (1998a,b).

Der in den Abschnitten 5 und 6 dargestellte Prototyp generiert aus vorformatierten Texten eigenständig SGML Dokument-Instanzen, die anschließend in HTML-Dokumente transformiert werden. Diese Transformationen werden nicht mit Hilfe der im DSSSL-Standard definierten Transformation Language durchgeführt, da für die Spezifikation dieser Programmiersprache noch keine Implementation vorliegt. Die Style Language hingegen ist weitestgehend in dem von James Clark entwickelten System `jade` (James' DSSSL Engine) implementiert.⁵ Dieser DSSSL-Prozessor enthält auch den SGML-Parser `nsqmls`, so daß man in einem Verarbeitungsschritt die Gültigkeit einer Dokument-Instanz überprüfen und diese anschließend formatieren kann. Damit jedoch trotz der bislang fehlenden Implementation der Transformation Language Manipulationen an der Dokumentstruktur möglich sind, hat Clark einige proprietäre, nicht zum Standard konforme, Erweiterungen für eben diesen Zweck in das SGML Transformationsmodul von `jade` integriert (vgl. Clark, 1997).

Üblicherweise werden in einem DSSSL-Skript für die verschiedenen Elemente einer Dokumenttyp-Definition Regeln definiert, die die Verarbeitung dieser Elemente und ihres Inhalts festlegen. Der in Abbildung 1 dargestellte Auszug entstammt dem von mir implementierten Prototypen, in dem ein SGML-Element `<CUE>` (siehe hierzu Abschnitt 6) benutzt wird, um Diskursmarker zu kennzeichnen. Dieses Element verfügt über ein Attribut `TYPE`, das bestimmt, ob es sich bei dem erkannten Diskursmarker um ein Wort (`WORD`) oder eine Phrase (`PHRASE`) handelt. In einem zweiten Attribut namens `REL` ist die rhetorische Relation vermerkt, die von dem jeweiligen Diskursmarker

⁴Elemente werden in DSSSL sogenannten Flow Objects zugeordnet. Hierbei handelt es sich um typographische Objekte, die im Standard wie folgt definiert werden: „A specification of a task to be performed by the formatter. A flow object has a class, which specifies the kind of task, and characteristics which further parameterize the task.“ (ISO10179, 1996, S. 5).

⁵Die aktuelle Version von `jade` ist 1.1.1, erhältlich ist das System für UNIX- und Windows NT-Systeme unter <http://www.jclark.com/jade/>.

```

1 (element cue ; Element <CUE>
2 (let ((type (attribute-string "TYPE"))); Variable 'type'
3 (let ((rel (attribute-string "REL"))) ; Relation merken
4 (case type
5 ("WORD") (make sequence ; Falls TYPE=WORD
6 (make element gi: "FONT" ; Rot
7 attributes: '("COLOR" "#FF0000"))
8 (make element gi: "STRONG"
9 (literal " [")
10 (process-children)
11 (literal "] ")))
12 (literal " (" ; Relationsnamen
13 (make element gi: "EM" ; ausgeben
14 (literal rel))
15 (literal ") ")))
16 ("PHRASE") (make sequence ; Falls TYPE=PHRASE
17 (make element gi: "FONT" ; Blau
18 attributes: '("COLOR" "#0000FF"))
19 (make element gi: "STRONG"
20 (literal " [")
21 (process-children)
22 (literal "] ")))
23 (literal " (" ; Relationsnamen
24 (make element gi: "EM" ; ausgeben
25 (literal rel))
26 (literal ") ")))))) ; restlichen
27 (process-children) ; Inhalt bearbeiten

```

Abbildung 1: Auszug eines DSSSL-Skripts zur HTML-seitigen Einfärbung von in der SGML-Quelle annotierten Diskursmarkern und der Ausgabe der entsprechenden Relations-Namen

signalisiert wird. Nachdem nun der durch das Element `<CUE>` markierte Satz bzw. Nebensatz in dem resultierenden HTML-Dokument Rot oder Blau eingefärbt wurde, wird in runden Klammern zusätzlich der Name der signalisierten Relation ausgegeben (ein Beispiel hierfür zeigt Abbildung 3). Die Einfärbung des Textes, den die rhetorische Relation umschließt, erfolgt im Zieldokument über das HTML-Element ``. Dieses Element gestattet, über das Attribut `COLOR` eine Farbe zu definieren, die nur für einen bestimmten Textbereich gilt.

Über die Anweisung `make element` (Abbildung 1, Zeilen 6, 8, 13, etc.) kann man im Zieldokument neue Tags erzeugen, deren Namen über das Argument `gi` festgelegt werden.⁶ Mit Hilfe des Arguments `attributes` (Zeilen 7 und 18) kann man Attribute für das soeben erzeugte Element spezifizieren. Die Anweisung `process-children` veranlaßt die DSSSL-Maschine zur Abarbeitung derjenigen Elemente, die sich in dem von dem aktuellen Element aufgespannten Teilbaum befinden.

4 Das verwendete Korpus

Das Testkorpus für den vorliegenden Prototypen bilden 840 Texte, die aus der CD ROM „11 Jahre taz“ exportiert wurden und aus der *tageszeitung* und *Le Monde diplomatique* stammen. Die verwendeten Texte gehören der *tageszeitung*-spezifischen Textsorte *Dokumentation* an und sind mit einer Länge von 8.250 Zeichen oder mehr relativ umfangreich; insgesamt ergibt sich eine Korpusgröße von ca. 13 Megabyte. Die Dokumente liegen in einem erweiterten ASCII-Format – Latin1, definiert in der Norm ISO 8859 – vor, d. h. Umlaute und sonstige Sonderzeichen, die die auf 7 Bit basierende ASCII Norm ISO 646 (der Basiszeichensatz von SGML) nicht repräsentieren kann, sind direkt im Text als solche kodiert.

Alle Texte folgen gewissen Formatierungs-Konventionen: Am Kopf jedes Dokuments befindet sich ein Block von Meta-Informationen, der Auskunft gibt über die Namen von Verlag, Autor und Zeitung, über das Datum, an dem dieser Artikel in der entsprechenden Zeitung erschienen ist, über die Seitenzahl und die Länge des Artikels in Zeilen. Darauf folgt der eigentliche Text. Überschriften, Unterüberschriften und Abschnitte mit Fließtext werden durch Leerzeilen voneinander abgetrennt. Viele Texte verfügen zusätzlich über Endnoten. Diese werden ebenfalls durch Leerzeilen getrennt und mit einer Zahl eingeleitet.

5 Funktionalität des Prototypen

Der im folgenden vorgestellte Prototyp zeichnet die im Korpus enthaltenen Dokumente automatisch hinsichtlich verschiedener Eigenschaften mit SGML-Elementen aus

⁶Der SGML Standard (vgl. ISO8879, 1986; Goldfarb, 1990) bezeichnet den Namen eines Tags als Generic Identifier (GI).

(Näheres zur Implementation findet sich in Abschnitt 6). Zunächst werden Satzgrenzen erkannt und als solche markiert. Im Zuge dieser Analyse werden Datums- und Zeitangaben, Abkürzungen, verschiedene Notationsvarianten von Zahlenangaben, geklammerte Ausdrücke, Zitate und Texteschübe ebenfalls explizit gekennzeichnet. Auch die im Kopf jedes Textes enthaltenen Meta-Informationen und die Grenzen der einzelnen Abschnitte werden markiert. Über eine Heuristik, die auf statistischen Informationen wie der Anzahl Worte pro Abschnitt, Anzahl Sätze pro Abschnitt, Anzahl Worte des vorangegangenen Abschnitts etc. beruht, werden die Funktionen der einzelnen Abschnitte ermittelt; so kann es sich bei einem Abschnitt entweder um eine Überschrift, eine Unterüberschrift, Fließtext, eine Zwischenüberschrift oder um eine Endnote handeln. Im nächsten Verarbeitungsschritt werden an drei verschiedenen Positionen die diskursmarkierenden Wörter und Phrasen annotiert. Erkannt werden diese an satzinitialer Stellung, direkt nach einem Interpunktionszeichen und an beliebiger Position in einem Satz. Der Skopus der durch die Diskursmarker signalisierten Relationen kann aufgrund der nur oberflächlich durchgeführten Analyse nicht genau bestimmt werden und endet daher vor demjenigen Interpunktionszeichen, das als nächstes im Text auftaucht, wobei geklammerte Ausdrücke und Einschübe nicht berücksichtigt werden.

Konforme SGML Dokument-Instanzen erfordern eine spezielle Art der Kennzeichnung von Sonderzeichen, die der ASCII Zeichensatz nicht darstellen kann. Diese Konvertierung, die im wesentlichen Umlaute auf ihre jeweiligen Ersatzdarstellungen (aus „ä“ wird beispielsweise „ä ;“) abbildet, findet im letzten Verarbeitungsschritt statt.

Die im Zuge der verschiedenen Analysestufen gewonnenen Informationen erlauben nun eine Konvertierung der erzeugten SGML-Dokumente in das HTML (Hypertext Markup Language) Format, das Auszeichnungsschema für Dokumente des World Wide Web. Hierzu wurde ein DSSSL-Skript implementiert, das die verschiedenen makrostrukturellen Funktionen, die Paragraphen in einem Text haben können (siehe oben), in HTML-Elemente umsetzt. Die Funktion „Überschrift“ wird beispielsweise auf das Element <H1> – Überschrift erster Stufe – abgebildet, Endnoten werden in eine nicht-numerierete Liste konvertiert (, *unordered list* und , *list item*). Weiterhin werden verschiedene Hyperlinks in das zu erzeugende Dokument integriert, die eine einfache Navigation auf dem Korpus und einen effizienten Zugriff auf verschiedene Diagnose-Ausgaben und die SGML-Quelle des jeweiligen Dokuments erlauben.

Darüber hinaus wird in die Hypertext-Dokumente ein Menü aufgenommen, das eine dynamisch generierte farbliche Hervorhebung verschiedener annotierter Textelemente gestattet. Diese sind im einzelnen: rhetorische Relationen und Diskursmarker, geklammerte Ausdrücke, Einschübe, Satzgrenzen und Zitate. Diese Funktionalität gewährleistet eine einfache Überprüfung der automatisch durchgeführten SGML-Annotation der Rohdokumente, um z. B. Fehler in der Satzgrenzenbestimmung zu lokalisieren. Abbildung 2 zeigt ein aus der SGML-Quelle konvertiertes HTML-Dokument mit dem oben erwähnten Auswahlmenü in einem World Wide Web Browser.

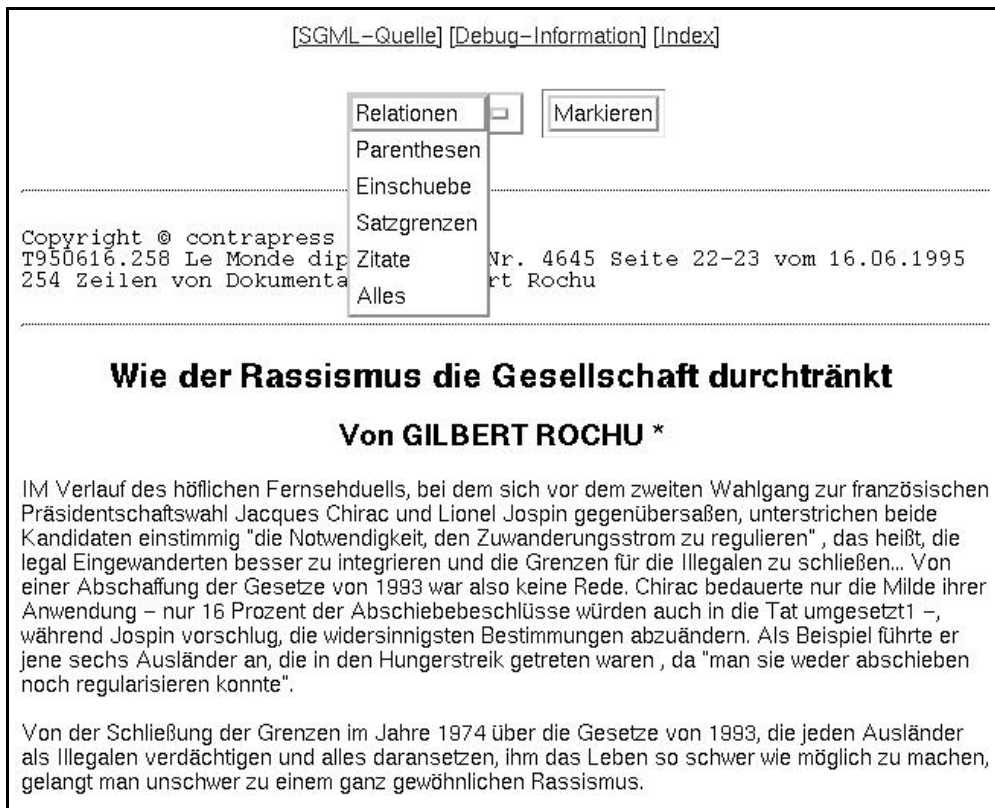


Abbildung 2: Die konvertierte HTML-Version eines SGML-Dokuments

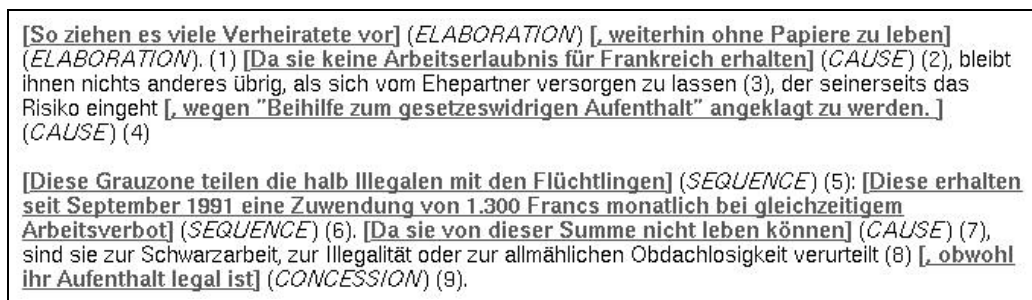


Abbildung 3: Auszug aus der Hypertext-Version eines Textes, bei der die rhetorischen Relationen markiert wurden

Wählt man einen Menüpunkt aus und betätigt den Knopf „Markieren“, wird auf dem Rechner, der das Korpus mittels einer Webserver-Software zur Verfügung stellt, für den Benutzer unsichtbar ein CGI-Skript⁷ aktiviert, das mittels der in Abschnitt 3 erwähnten DSSSL Maschine *jade* die gewünschten Textbereiche bei einer erneuten Konvertierung der zugrundeliegenden SGML-Quelle farblich markiert und an den Browser zurückliefert. Abbildung 3 zeigt zwei Abschnitte aus einem im Korpus enthaltenen Text⁸, wobei die erkannten rhetorischen Relationen in eckigen Klammern markiert wurden. Der Relationsname befindet sich kursiv gesetzt in runden Klammern hinter der jeweiligen Relation. Zur Veranschaulichung wurden lediglich die Durchnummerierungen der relevanten Nebensätze manuell in das der Abbildung zugrundeliegende HTML-Dokument eingefügt, damit in den Strukturbäumen (s.u.) auf sie Bezug genommen werden kann.

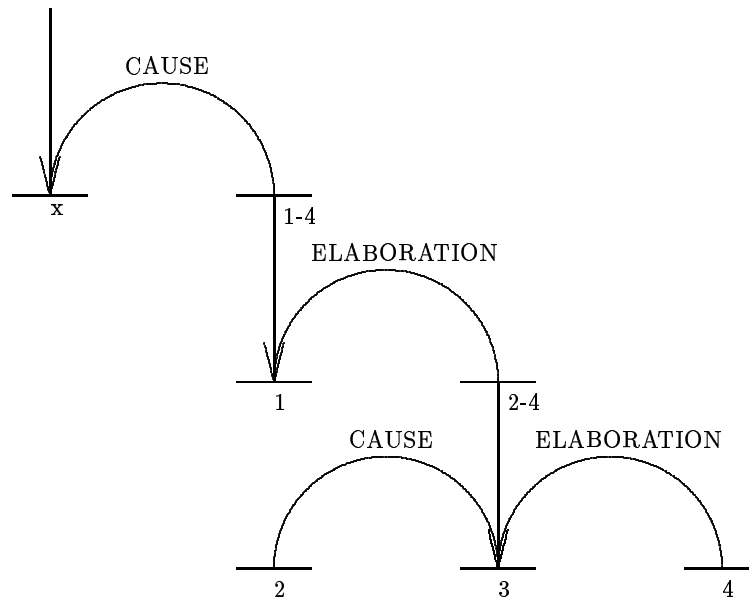


Abbildung 4: RST-Baum für den ersten Abschnitt aus Abbildung 3

Die beiden in den Abbildungen 4 und 5 dargestellten rhetorischen Strukturbäume

⁷CGI (Common Gateway Interface) ist ein Standard, mit dessen Hilfe man die Funktionalität von Webserver-Software durch eigene Programme erweitern kann. Siehe hierzu beispielsweise <http://www.w3.org/CGI/>.

⁸*Wie der Rassismus die Gesellschaft durchtränkt*, Gilbert Rochu, In: *Le Monde diplomatique* Nr. 4645 vom 16.06.1995.

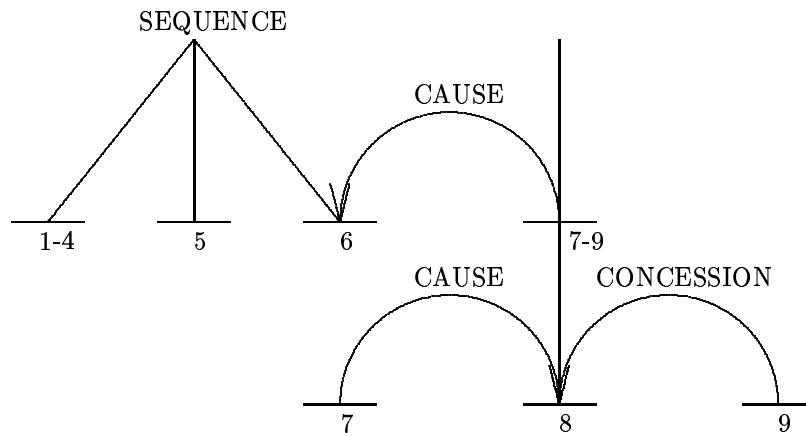


Abbildung 5: RST-Baum für den zweiten Abschnitt aus Abbildung 3

für den Text aus Abbildung 3 kann man mit Hilfe der erkannten Diskursmarker sowie den sie begleitenden rhetorischen Relationen aufbauen. Da hierfür bislang weder ein algorithmisches Verfahren noch eine Implementierung vorliegt, sollten die Struktur­bäume sowie das im folgenden grob angerissene Verfahren zu ihrem Aufbau als erste Näherungen an diese Problematik verstanden werden. Eine zu Beginn eines Abschnittes auftauchende Relation *ELABORATION* bezieht sich auf vorangegangene Teile des Textes (x). Nebensatz (3) aus Abbildung 3 verfügt über keine explizit gekennzeichnete rhetorische Relation, in einem solchen Fall könnte man etwa *ELABORATION* als eine Art Default-Relation annehmen. (2) kennzeichnet aufgrund des satzinitial vorkommenden Diskursmarkers *Da* eine *CAUSE* Relation, die sich in beinahe allen Fällen auf den darauffolgenden Textabschnitt bezieht. Eine sinnvolle RST-Analyse würde sicherlich (4) als *ELABORATION* von Abschnitt (3) annehmen. Abbildung 3 zeigt in diesem Zusammenhang fälschlicherweise eine *CAUSE* Relation, die aufgrund des Diskursmarkers *wegen* annotiert wurde. Es bleibt offen, ob und auf welche Weise ein solches Problem mit auf der Oberfläche arbeitenden Methoden adäquat zu behandeln ist.

6 Implementation

Der im vorangegangenen Abschnitt beschriebene Prototyp besteht im wesentlichen aus verschiedenen Automaten, die auf einer UNIX Plattform in C implementiert wur-

den, wobei `flex`⁹ eingesetzt wurde. Die einzelnen Module fungieren als Filter, d. h. sie erwarten einen Strom von Zeichen (in diesem Fall ein Textdokument) als Eingabe, erkennen gewisse Regularitäten, annotieren diese und geben das Dokument wieder aus. In einer Kaskade (einer sog. UNIX Pipeline) werden in dem Prototyp insgesamt sechs solcher Filter hintereinandergeschaltet. Als Eingabe erwartet diese Kaskade Texte des in Abschnitt 4 beschriebenen Korpus, Resultat der Verarbeitung sind SGML Dokumentinstanzen.

Die Erkennung der Dokumentstruktur erfolgt über Pattern Matching Mechanismen. `flex` bietet dem Benutzer die Möglichkeit, in einem Eingabestrom reguläre Ausdrücke (vgl. etwa Friedl, 1997) zu erkennen und daraufhin verschiedene Aktionen in Form von C Funktionen auszuführen. Auf diese Weise kann man z. B. Muster angeben, die Grenzen zwischen Abschnitten – in diesem Falle Leerzeilen – erkennen und im Falle eines Vorkommens ein SGML Element `<PARAGRAPH>` in den Text einfügen. Die meisten Module verfügen darüberhinaus auch über verschiedene Zustände (im automaten-theoretischen Sinn), so ist es beispielsweise nur dann möglich, das schließende Element `</SATZ>` zu annotieren, wenn hierzu bereits ein korrespondierendes öffnendes Element vorliegt.

Zur korrekten Bestimmung von Satzgrenzen ist eine möglichst genaue Disambiguierung des Zeichens „.“ notwendig. Diese erfolgt mittels eines Lexikons, das ca. 150 Abkürzungen enthält. Darüberhinaus verfügt das entsprechende Modul über verschiedene reguläre Ausdrücke, die Zahlenangaben oder Datumsausdrücke (z. B. 1.500 oder 3. September 1998) erkennen und annotieren. Auf diese Weise werden all diejenigen Vorkommen des Punktes, die Bestandteil von Abkürzungen oder verschiedenen Zahlenangaben sind, erkannt, so daß die verbleibenden Punkte höchstwahrscheinlich als Satzdemarkierung fungieren. Hoffmann (1994) beschreibt ein ähnliches auf `flex` basierendes Verfahren zur Satzgrenzenbestimmung, wobei sie ebenfalls mit Texten arbeitet, die in der *tageszeitung* erschienen sind. Für grundlegende Probleme, die in Verbindung mit der Disambiguierung von Satzzeichen auftreten siehe z. B. Nunberg (1990), Grefenstette und Tapanainen (1994), Palmer (1994) und Palmer und Hearst (1997).

Die benutzte Dokumenttyp-Definition ist mit ca. 200 Zeilen relativ umfangreich. Sie beschreibt die benutzte Textsorte sehr detailliert und erwartet nach dem Basiselement `<TEXT>` zunächst den in jedem Dokument aus dem Korpus vorhandenen Block mit Meta-Informationen, dessen einzelne Bestandteile ebenfalls explizit markiert werden. Darauf folgen verschiedene Abschnitte, die durch `<PARAGRAPH>` angezeigt werden. Ein Attribut dieses Elements beschreibt die ermittelte Funktion des jeweiligen Abschnitts, also ob es sich beispielsweise um eine Überschrift oder Fließtext handelt.

⁹Hierbei handelt es sich um die von Vern Paxson implementierte frei verfügbare Variante von `lex`, einem Werkzeug zur lexikalischen Analyse, das vor allem im Compiler-Bau aber auch in der Computerlinguistik Verwendung findet. Siehe hierzu beispielsweise Levine et al. (1992) und `ftp://ftp.informatik.tu-muenchen.de/pub/comp/os/unix/gnu/flex/`.

Paragrafen enthalten Sätze, die durch das Element `<SATZ>` markiert werden. Sätze können wiederum u.a. durch Diskursmarker signalisierte rhetorische Relationen enthalten. Diese werden durch das Element `<CUE>` markiert. Das öffnende Element wird vor dem erkannten Diskursmarker und das schließende Element vor dem nächsten im Strom auftauchenden Interpunktionszeichen gesetzt. In Attributen wird weiterhin die zu dem Diskursmarker korrespondierende rhetorische Relation vermerkt und ob es sich um ein diskursmarkierendes Wort oder eine Phrase gehandelt hat (siehe hierzu auch Abschnitt 3, insbes. Abbildung 1).

Anzahl (n) erkannter Fehler	Anzahl Dokumente
$n = 0$	335
$0 < n \leq 5$	336
$5 < n < 10$	68
$n \geq 10$	101
	$\Sigma = 840$

Tabelle 2: Evaluation des Prototypen mittels eines SGML Parsers

Tabelle 2 zeigt eine Evaluation des Prototypen hinsichtlich der syntaktischen Korrektheit der generierten SGML Annotation. Mittels eines SGML Parsers kann man überprüfen, ob ein SGML Dokument zu einer zugrundeliegenden Dokumenttyp-Definition (DTD) konform ist oder ob Fehler enthalten sind. Ein möglicher Fehler ist beispielsweise, wenn ein Element `<CUE>` außerhalb eines Satzes auftaucht, da die DTD eine solche Konstruktion nicht gestattet. Die Tabelle zeigt, daß mehr als ein Drittel der Texte (39,9%) korrekt ausgezeichnet wird und daß weitere 306 Dokumente (40%) weniger als sechs Fehler aufweisen. Die Fehler resultieren zum größten Teil aus schon in der Quelle fehlerhaft formatierten Dokumenten. Ausschlaggebend für viele Fehler ist auch die Tatsache, daß das Korpus nicht nur Texte enthält, die aus mehreren Abschnitten von Fließtext bestehen – ausschließlich für diese wurde das System entwickelt –, sondern auch Gedichte, Dialoge und Briefe. Bei diesen Textsorten kommt es zwangsläufig zu Fehlern in der Erkennung der Dokumentstruktur, da die entsprechenden Dokumente nach den Konventionen ihrer jeweiligen Textsorte formatiert wurden.

7 Ausblick

Der beschriebene Ansatz zeigt, daß die automatische Überführung von nicht explizit strukturierten Texten in eine SGML Repräsentation mit einfachen Mitteln zu realisie-

ren ist. Die Document Style Semantics and Specification Language dient daraufhin zur Transformation der SGML Dokumente, beispielsweise um sie für das World Wide Web aufzubereiten. Weiterhin scheint es möglich, mittels einer Textanalyse, die gänzlich auf Oberflächeninformationen beruht, zumindest grobe rhetorische Strukturen aufzubauen. Fraglich ist, ob die in Abschnitt 5 skizzierte Weise zur Gewinnung dieser Baumstrukturen in einen Algorithmus umsetzbar ist, um auf diese Weise vollständige Bäume für ganze Texte und nicht nur für einzelne Abschnitte zu produzieren. Einen solchen vollständigen rhetorischen Strukturbaum kann man, wie von vielen Autoren (siehe beispielsweise Sparck Jones, 1993) vorgeschlagen, als Grundgerüst einer Zusammenfassung der wesentlichen Aussagen eines Textes ansehen, wenn man die in dem Baum enthaltenen informationsarmen Satelliten tilgt (siehe hierzu auch Marcu, 1998).

Danksagung

Der Autor bedankt sich bei Alexander Krumeich, Martin Müller und Wilfried Teiken für hilfreiche Anmerkungen.

Literatur

- BEHME, HENNING UND MINTERT, STEFAN (1998a): "Klammern gehört zum Handwerk – DSSSL: XML-Dokumente fürs Web formatieren". *iX* 1998 (3): S. 156–161.
- BEHME, HENNING UND MINTERT, STEFAN (1998b): *XML in der Praxis – Professionelles Web-Publishing mit der Extensible Markup Language*. Bonn, Reading, Menlo Park etc.: Addison-Wesley.
- BREITSPRECHER, ROLAND; TERRELL, PETER; CALDERWOOD-SCHNORR, VERONIKA UND MORRIS, WENDY V. A. (Herausgeber) (1988): *Pons Globalwörterbuch Englisch – Deutsch*. Stuttgart: Klett. Nachdruck.
- CLARK, JAMES (1997): "Using Jade for SGML Transformations". Online erhältlich unter <http://www.jclark.com/jade/transform.htm>.
- DROSDOWSKI, GÜNTHER (Herausgeber) (1995): *Grammatik der deutschen Gegenwartssprache*. Nummer 4 in Der Duden in 12 Bänden. Mannheim, Leipzig, Wien, Zürich: Dudenverlag, 5. Auflage.
- EMDE, WERNER (1991): "Managing Lexical Knowledge in LEU/2". In: *Text Understanding in LILOG*, herausgegeben von Herzog, Otthein und Rollinger, Claus-Rainer, Berlin, Heidelberg: Springer, Nummer 546 in Lecture Notes in Artificial Intelligence, S. 167–179.

- FRIEDL, JEFFREY E. F. (1997): *Mastering Regular Expressions*. Sebastopol: O'Reilly & Associates.
- GOLDFARB, CHARLES F. (1990): *The SGML Handbook*. Oxford: Oxford University Press.
- GRFENSTETTE, GREGORY UND TAPANAINEN, PASI (1994): "What is a Word, What is a Sentence? – Problems of Tokenization". In: *Proceedings of the Third Conference on Computational Lexicography and Text Research (COMPLEX '94)*. Research Institute for Linguistics, Hungarian Academy of Sciences, Budapest, S. 79–87. Online erhältlich unter <http://www.xerox.fr/grenoble/mltt/articles/home.html>.
- HOBBS, JERRY R. (1979): "Coherence and Coreference". *Cognitive Science* 3: S. 67–90.
- HOFFMANN, CHRISTIANE (1994): *Automatische Disambiguierung von Satzgrenzen in einem maschinenlesbaren deutschen Korpus*. Hausarbeit im Studiengang Linguistische Datenverarbeitung/Computerlinguistik, Universität Trier. Online erhältlich unter <ftp://ftp.uni-trier.de/ftp/pub/local/ldv/>.
- HOVY, EDUARD H. (1993): "Automated Discourse Generation using Discourse Structure Relations". *Artificial Intelligence* 63: S. 341–385.
- ISO10179 (1996): "Information Processing – Processing Languages – Document Style Semantics and Specification Language (DSSSL)". Internationale Norm, International Organization for Standardization, Genf. Online erhältlich unter <ftp://ftp.ornl.gov/pub/sgml/WG8/DSSSL/>.
- ISO8879 (1986): "Information Processing – Text and Office Information Systems – Standard Generalized Markup Language". Internationale Norm, International Organization for Standardization, Genf.
- LEVINE, JOHN R.; MASON, TONY UND BROWN, DOUG (1992): *lex & yacc*. A Nutshell Handbook. Cambridge, Köln, Paris etc.: O'Reilly & Associates, 2. Auflage.
- LOBIN, HENNING (1998a): "Intelligente Dokumente – Linguistische Repräsentation komplexer Inhalte für die hypermediale Wissensvermittlung". In: Lobin (1998b), S. 155–178.
- LOBIN, HENNING (Herausgeber) (1998b): *Intelligente Dokumente – Linguistische Repräsentation komplexer Inhalte für die hypermediale Wissensvermittlung*. Wiesbaden: Westdeutscher Verlag.
- MACHERIUS, INGO (1997): *Ein Handhabungssystem für HTML Dokumente*. Diplomarbeit, Technische Universität Clausthal. Online erhältlich unter http://home.tu-clausthal.de/~inim/thesis/thesis_im.zip.

- MANN, WILLIAM C.; MATTHIESSEN, CHRISTIAN M. I. M. UND THOMPSON, SANDRA A. (1989): "Rhetorical Structure Theory and Text Analysis". Information Sciences Institute Research Report ISI/RR-89-242, University of Southern California.
- MANN, WILLIAM C. UND THOMPSON, SANDRA A. (1987): "Rhetorical Structure Theory: Description and Construction of Text Structures". In: *New Results in Artificial Intelligence, Psychology and Linguistics*, herausgegeben von Kempen, Gerhard, Dordrecht, Boston, Lancaster: Martinus Nijhoff Publishers, S. 85–96.
- MANN, WILLIAM C. UND THOMPSON, SANDRA A. (1988): "Rhetorical Structure Theory: Toward a Functional Theory of Text Organization". *Text* 8: S. 243–281.
- MARCU, DANIEL (1996): "Building Up Rhetorical Structure Trees". In: *The Proceedings of the Thirteenth National Conference on Artificial Intelligence*. Portland: American Association for Artificial Intelligence, Band 2, S. 1069–1074. Online erhältlich unter <http://www.isi.edu/~marcu/papers.html>.
- MARCU, DANIEL (1997): *The Rhetorical Parsing, Summarization, and Generation of Natural Language Texts (= Technical Report CSRG-371, Computer Systems Research Group, University of Toronto)*. Dissertation, Department of Computer Science, University of Toronto, Toronto. Online erhältlich unter <http://www.isi.edu/~marcu/papers.html>.
- MARCU, DANIEL (1998): "To build Text Summaries of High Quality, Nuclearity is not sufficient". In: *The Working Notes of the the AAAI-98 Spring Symposium on Intelligent Text Summarization*, Stanford: American Association for Artificial Intelligence. Online erhältlich unter <http://www.isi.edu/~marcu/papers.html>.
- MIIKE, SEIJI; ITOH, ETSUO; ONO, KENJI UND SUMITA, KAZUO (1994): "A Full-Text Retrieval System with a Dynamic Abstract Generation Function". In: *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, herausgegeben von Croft, W. Bruce und van Rijsbergen, C. J. Berlin, Heidelberg, New York etc.: Springer, S. 152–161.
- MÜLLER, WOLFGANG (Herausgeber) (1997): *Die sinn- und sachverwandten Wörter – Synonymwörterbuch der deutschen Sprache*. Nummer 8 in Der Duden in 12 Bänden. Mannheim, Leipzig, Wien, Zürich: Dudenverlag, 2. Auflage.
- NUNBERG, GEOFFREY (1990): *The Linguistics of Punctuation*. Nummer 18 in CSLI Lecture Notes. Stanford: Center for the Study of Language and Information.
- ONO, KENJI; SUMITA, KAZUO UND MIIKE, SEIJI (1994): "Abstract Generation Based on Rhetorical Structure Extraction". In: *COLING 94 – The 15th In-*

- ternational Conference on Computational Linguistics*. Association for Computational Linguistics, Kyoto, Japan, Band 2, S. 344–348. Online erhältlich unter <http://xxx.lanl.gov/ps/cmp-lg/9411023>.
- PALMER, DAVID D. (1994): “SATZ – An Adaptive Sentence Segmentation System”. Technischer Bericht CSD-94-846, Computer Science Division, University of California, Berkeley. Online erhältlich unter <http://sunsite.berkeley.edu/TR/UCB:CSD-94-846>.
- PALMER, DAVID D. UND HEARST, MARTI A. (1997): “Adaptive Multilingual Sentence Boundary Disambiguation”. *Computational Linguistics* 23 (2): S. 241–267.
- PRESCOD, PAUL (1997): “Introduction to DSSSL”. Online erhältlich unter <http://itrc.uwaterloo.ca/~papresco/dsssl/tutorial.html>.
- REHM, GEORG (1998): *Vorüberlegungen zur automatischen Zusammenfassung deutschsprachiger Texte mittels einer SGML- und DSSSL-basierten Repräsentation von RST-Relationen*. Magisterarbeit, Studiengang Computerlinguistik und Künstliche Intelligenz, Universität Osnabrück. In Vorbereitung.
- RICKHEIT, GERT (Herausgeber) (1991): *Kohärenzprozesse. Modellierung von Sprachverarbeitung in Texten und Diskursen*. Opladen: Westdeutscher Verlag.
- SPARCK JONES, KAREN (1993): “What might be in a Summary?” In: *Information Retrieval 93: Von der Modellierung zur Anwendung*, herausgegeben von Knorz, Gerhard; Krause, Jürgen und Womser-Hacker, Christa. Universitätsverlag Konstanz, Nummer 12 in Schriften zur Informationswissenschaft, S. 9–26. Online erhältlich unter <ftp://ftp.cl.cam.ac.uk/public/clanonftp/papers/ksj/>.
- STEDE, MANFRED UND UMBACH, CARLA (1998): “DiMLex: A Lexicon of Discourse Markers for Text Generation and Understanding”. In: *Proceedings of the COLING-ACL '98 Conference*. Association for Computational Linguistics, Montréal. Online erhältlich unter <http://flp.cs.tu-berlin.de/~marker/inlg98.ps>.
- WITT, ANDREAS (1998): “SGML und Linguistik”. In: Lobin (1998b), S. 121–154.