Universität Hamburg
Fachbereich Informatik

Integration statistischer Methoden in eliminative Verfahren zur Analyse von natürlicher Sprache

Diplomarbeit

abgefaßt von
Ingo Schröder

betreut von
Prof. Dr.–Ing. Wolfgang Menzel
und
Prof. Dr. Christopher Habel

Danksagung

Mein herzlicher Dank geht an die Menschen, die mir bei der Abfassung dieser Arbeit auf die eine oder andere Art und Weise geholfen haben:

- Mein Erstbetreuer Wolfgang Menzel hat mich während dieser Arbeit und davor kontinuierlich unterstützt. Die zahlreichen fachlichen Diskussionen, seine Anregungen und Hinweise haben entscheidend zum Gelingen dieser Arbeit beigetragen.
- Mein Zweitbetreuer Christopher Habel hat meine Vertiefung im Bereich der sprachorientierten Künstlichen Intelligenz über Jahre gefördert und meine Diplomarbeit, obwohl das Thema nicht im Zentrum seines Interesses liegt, mitgetragen.
- Bernhard '*Deutschland gewinnt keine Goldmedaille!*' Lahres hat einige frühe Versionen dieser Arbeit gelesen und mir durch seine konstruktive Kritik geholfen, komplexe Sachverhalte besser zu verstehen und verständlicher darzustellen.
- Dorothea '*Graphik natürlich mit ph*' Kern und Tanja '*Grins*' Naguschewski haben die Arbeit gewissenhaft Korrektur gelesen. Alle verbleibenden Fehler habe ich selbstverständlich nachträglich in die Arbeit eingefügt.
- Uta '*Ja, wo ist sie denn?*' Arnold und Soenke '*Was soll ich da sagen?*' Ziesche haben mich in philosophischen Diskussionen zunächst vom Unsinn und dann vom Sinn des Seins[1] überzeugt.
- Thomas '*heute hier, morgen Wanne*' Will hat in den letzten Monaten dafür gesorgt, daß ich auch an andere Dinge als an diese Arbeit gedacht habe.
- Meine Mutter Ursula '*Oh, wie schön!*' Schröder hat mich während meines ganzen Studiums in jeder Hinsicht unterstützt.

[1] Oder war's umgekehrt?

Inhaltsverzeichnis

1. Einleitung

Vorhaben der Arbeit

Diese Arbeit beschäftigt sich mit statistischen sowie mit eliminativen Methoden zur Analyse von natürlicher Sprache.

Statistische Methoden haben in den letzten Jahren auf dem Gebiet der maschinellen Verarbeitung von natürlicher Sprache einen großen Aufschwung erfahren. Dies läßt sich zum einen darauf zurückführen, daß heutzutage die Voraussetzungen für statistische Verfahren — große natürlichsprachliche Korpora und leistungsfähige Hardware — zur Verfügung stehen. Zum anderen hat sich aber auch gezeigt, daß ein manueller Aufbau einer umfangreichen Grammatik sehr aufwendig und fehleranfällig — wenn nicht unmöglich — ist, so daß nach anderen Methoden zum Aufbau des zur Analyse von natürlichsprachlichen Äußerungen nötigen Wissens gesucht werden muß. Statistische Verfahren stellen eine Möglichkeit dar, dieses Wissen automatisch aus natürlichsprachlichen Korpora zu extrahieren und für die Analyse von unbekannten Äußerungen zur Verfügung zu stellen.[1]

Eliminative Verfahren dagegen spielen in der Computerlinguistik zum gegenwärtigen Zeitpunkt noch eine untergeordnete Rolle. Sie verstehen natürlichsprachliche Analyse als Ausschluß von unplausiblen Strukturmöglichkeiten, bis

[1] An dieser Stelle ist eine Bemerkung zum Sprachgebrauch angebracht. Die Begriffe 'Statistik', 'Stochastik' und 'Probabilismus' sowie die davon abgeleiteten Formen sind weder im normalen noch im wissenschaftlichen Sprachgebrauch synonym (Drosdowski, 1989):

- Statistik: Wissenschaft von der zahlenmäßigen Erfassung, Untersuchung und Auswertung von Massenerscheinungen [...]
- Stochastik: Teilgebiet der Statistik, das sich mit der Untersuchung vom Zufall abhängiger Ereignisse und Prozesse befaßt
- Probabilismus: (Philosophie) Auffassung, daß es in Wissenschaft und Philosophie keine absoluten Wahrheiten, sondern nur Wahrscheinlichkeiten gibt [...]

Da der Sprachgebrauch in der Literatur zu den entsprechenden Verfahren zur Verarbeitung natürlicher Sprache jedoch nicht einheitlich ist, können die Begriffe in dieser Arbeit im Zweifelsfall als synonym angesehen werden.

eine eindeutige Lösung gefunden ist. Damit stehen sie im Gegensatz zu den üblichen konstruktiven Methoden, die Analyse als Aufbau einer die gesamte Äußerung enthaltende Struktur auffassen.

Diese Arbeit enthält eine umfangreiche Zusammenstellung von existierenden stochastischen Methoden zur Analyse von natürlicher Sprache und einige wichtige Erweiterungen des eliminativen Parsings. Der Hauptbeitrag dieser Arbeit liegt jedoch in der Zusammenführung von stochastischen Methoden und eliminativem Vorgehen bei der Analyse von natürlicher Sprache. Es wird gezeigt werden, daß die beiden zueinander orthogonalen Verfahren gut miteinander kombinierbar sind und synergetische Effekte erzielen.

Einordnung

Im letzten Abschnitt sind bereits die beiden grundsätzlichen Herangehensweisen dieser Arbeit vorausgenommen worden: statistische Verfahren und eliminatives Parsing. Hier soll kurz eine funktionale Einordnung der Arbeit vorgenommen werden. Abbildung 1.1 zeigt eine übliche (vereinfachende) Darstellung der an der Verarbeitung von natürlicher Sprache beteiligten Prozesse. Die Generierung ist nur der Vollständigkeit halber in die Abbildung aufgenommen; Sprachproduktion spielt in dieser Arbeit keine Rolle.

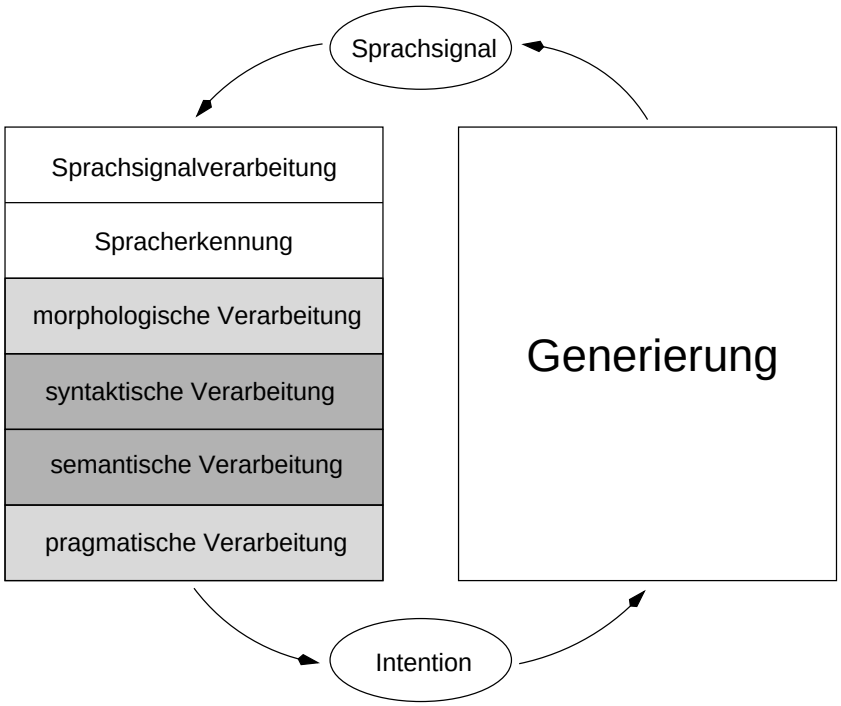


Abbildung 1.1: Sprachverarbeitungsarchitektur

Im Analysezweig sind diejenigen Verarbeitungstufen grau hinterlegt, für die die vorliegende Arbeit einen Beitrag liefert. Obwohl in dieser Arbeit selber nur

syntaktische und semantische Strukturen eine Rolle spielen, ist das Verfahren prinzipiell auch auf die morphologische und (evtl.) pragmatische Verarbeitung ausweitbar.[2]

Darstellungen wie die der Abbildung 1.1 legen den Schluß nahe, daß die einzelnen Verarbeitungsstufen sequentiell durchlaufen werden. Für das zu beschreibende Verfahren ist dieser Schluß falsch. Ein wichtiger Aspekt des Vorschlags besteht im Gegenteil gerade darin, bei der Analyse Einschränkungen von verschiedenen Ebenen gleichzeitig einzusetzen. Beispiele, die die Notwendigkeit für ein solches Vorgehen zeigen, finden sich an zahlreichen Stellen in dieser Arbeit.

Ein weiteres Defizit der Abbildung 1.1 besteht darin, daß sie die verschiedenen Wissensquellen, auf die die einzelnen Module der Analyse zugreifen, nicht enthält. Ein manuelles Erstellen dieser Wissensquellen ist — wie bereits erwähnt — in der Regel fehleranfällig und arbeitsaufwendig. Aus diesem Grund beschäftigt sich ein großer Teil dieser Arbeit mit dem Problem der Wissensakquisition, genauer mit stochastischen Verfahren zur automatischen Extraktion von sprachlichem Wissen aus natürlichsprachlichen Korpora und mit der Anwendung dieses Wissens bei der Analyse.

Gliederung der Arbeit

Diese Arbeit ist in fünf Kapitel und ebenfalls fünf Anhänge eingeteilt.

Dieses Kapitel 1 stellt das Vorhaben vor, ordnet die Arbeit in den Kontext ein und beschreibt die Organisation der Arbeit.

In Kapitel 2 werden eine Reihe von existierenden statistischen Modellen für die Analyse von natürlicher Sprache ausführlich vorgestellt und entlang mehrerer Dimensionen verglichen. Dabei wird sowohl auf die mathematischen Modelle zur Beschreibung der stochastischen Ereignisse bei der Analyse bzw. Generierung von natürlichsprachlichen Äußerungen als auch auf zu erwartenden Ergebnisse und Schwierigkeiten eingegangen.

Kapitel 3 führt in das sogenannte eliminative Parsing durch Constraint–Satisfaction ein und arbeitet einige Erweiterungen heraus; insbesondere werden verschiedene Auswahlfunktionen (siehe Abschnitt 3.5) vorgeschlagen und verglichen. Dabei werden erstmals Experimente zur Beurteilung der vorgeschlagenen Verfahren verwendet.

In Kapitel 4 werden einige stochastische Modelle für das eliminative Parsing vorgestellt. Ausgehend von einem einfachen Modell werden eine Reihe von Verbesserungen vorgeschlagen und anhand einer Reihe von Versuchen bzgl. ihrer Fähigkeit zur Disambiguierung untersucht.

[2] Als Einschränkung sei an dieser Stelle erwähnt, daß die vorzustellenden Verfahren einen (entscheidenden) Beitrag zu den einzelnen Verarbeitungsstufen leisten können. Es ist jedoch nicht ausgeschlossen, daß andere Verfahren eine Prä– oder Postverarbeitung durchführen können.

Schließlich wird in Kapitel 5 eine Zusammenfassung gegeben sowie ein Ausblick auf verbleibende Aufgaben gewagt.

Die Anhänge geben zusätzliche Informationen zum grammatischen System (Anhang A), zum verwendeten Korpus (Anhang B), zum Programmsystem (Anhang C), zur Basisgrammatik (Anhang D) und zu den durchgeführten Experimenten (Anhang E).

2. Vorstellung und Vergleich von statistisch basierten Verfahren

In diesem Kapitel wird zunächst für die Verwendung von statistischen Komponenten bei der Analyse natürlicher Sprache argumentiert. Den Hauptteil bildet dann die Vorstellung von verschiedenen stochastischen Ansätzen sowie ein Vergleich derselben bzgl. der verwendeten Information und der zu erwartenden Ergebnisse.

2.1 Motivation stochastischer Erweiterungen

Eine linguistische Grammatiktheorie legt für eine oder mehrere natürliche Sprachen fest, welche Beziehungen zwischen den einzelnen Teilen der Sprache bestehen und welche Konstruktionen als 'grammatikalisch' angesehen werden. Sie ist nicht auf einen Formalismus festgelegt und macht auch keine Aussagen über Effizienz bzw. Entscheidbarkeit des Wortproblems und über die Erlernbarkeit einer Sprache. So muß auch eine exhaustive Auflistung aller grammatikalischen Sätze als Grammatiktheorie akzeptiert werden. Da aber natürliche Sprachen potentiell unendlich viele grammatikalische Sätze hervorbringen können, bleibt eine solche Auflistung immer unvollständig.

Auch aus Gründen der Erlernbarkeit einer natürlichen Sprache geht man daher davon aus, daß eine Grammatik die Struktur eines Satzes kompositional beschreiben sollte. Im einfachsten Fall müssen dazu die elementaren Einheiten einer Sprache (Laute, Worte, Kategorien, semantische Prädikate u. a. m.) sowie die Regeln, wie man die Elemente zu größeren Einheiten der Sprache kombinieren kann, benannt werden. Mit einer solchen Beschreibung können dann auch potentiell unendlich viele Sätze der Sprache analysiert werden. Es sollte beachtet werden, daß der Begriff der Analyse an dieser Stelle noch keinen Algorithmus für diese Aufgabe meint. Zunächst wird eine Grammatik oft rein deklarativ formuliert, wie es beispielsweise in der HPSG (engl. *head–driven phrase structure grammar*) (Pollard und Sag, 1994) der Fall ist.

Ein solches Schema, das ein Modell für das (unbewußte) Wissen eines idealen Sprechers einer Sprache darstellt, wird von Chomsky (1965) Kompetenzgrammatik oder auch nur Kompetenz genannt. Unter Performanz wird dann die Fähigkeit des Individuums verstanden, aufgrund der erworbenen bzw. angeborenen Kompetenz unendlich viele Sätze der Sprache zu erzeugen und zu verstehen sowie Grammatikalitätsurteile abzugeben.[1]

Sobald für eine Grammatik ein Modell auf einem Rechner erstellt werden soll, ergeben sich (mindestens) zwei Probleme. Zum einen muß die evtl. noch nicht formalisierte Grammatik mit Hilfe von mathematischen Modellen formal erfaßt werden. Dabei wird insbesondere festgelegt, wie die einem Satz zugrundeliegende Struktur repräsentiert wird.[2] Zum anderen müssen Algorithmen gefunden werden, die das neue mathematische Modell umsetzen und für die jeweilige Aufgabenstellung (z. B. Wortproblem, Parsing, Sprachproduktion usw.) angemessen und mit vertretbarem Aufwand an Rechenzeit und Speicher funktionieren.

Im Laufe der letzten Jahre hat sich gezeigt, daß sich für die Grammatikentwicklung, so wie sie in den letzten Absätzen beschrieben wurde, mehrere z. T. gravierende Probleme ergeben.

- Aufwand und Abdeckung: Bisher ist es nicht annähernd gelungen, für eine natürliche Sprache eine Grammatik zu erstellen, die diese vollständig abdeckt, d. h. die in der Lage ist, alle Sätze und Nichtsätze dieser Sprache korrekt zu beurteilen, da es sich als zunehmend schwierig erweist, den Abdeckungsgrad einer Grammatik zu erhöhen. Daraus ergibt sich, daß die Erstellung einer Grammatik eine aufwendige Aufgabe darstellt.

- Fehler und Wartbarkeit: Eine große Grammatik enthält immer auch Fehler. Damit ist nicht die evtl. zu geringe Abdeckung der Sprache gemeint, sondern Effekte, die der Grammatikschreiber nicht vorhergesehen hat und die zu fehlerhaften Analysen führen. Je größer eine Grammatik ist, desto fehleranfälliger gestaltet sich auch eine Wartung bzw. Erweiterung derselben.

- Domänenabhängigkeit: Es können Grammatiken für natürliche Sprachen erstellt werden, die mit relativ hoher Genauigkeit in der Lage sind, Sätze der entsprechenden Domäne zu analysieren. Eine Portierung einer solchen Grammatik auf eine andere Domäne bzw. eine Erweiterung der Domäne auf einen größeren Bereich gestaltet sich dann aber häufig schwierig oder unmöglich, so daß eine Wiederverwendbarkeit der Grammatik nicht gegeben ist.

[1] Siehe (Chomsky, 1965) oder (Bußmann, 1990, → Kompetenz vs. Performanz) für eine etwas ausführlichere Beschreibung sowie weitere Literaturverweise.

[2] Bei der Wahl des formalen Modells gibt es naturgemäß einen hohen Freiheitsgrad. Beispielsweise versucht die HPSG, mittels Merkmalsstrukturen und Unifikation eine komplexe Repräsentation der syntaktischen und semantischen Struktur aufzubauen, während andere Modelle sich auf einfache syntaktische Beziehungen (engl. *shallow syntactic parsing*) beschränken.

- Robustheit: Die Eingaben, mit denen ein System im Einsatz konfrontiert wird, sind im vornherein nicht bekannt. Daher können unbekannte Konstruktionen und unbekannte Worte auftreten, die dem System noch nicht bekannt sind. Jedes (noch so komplexe) System muß mit dieser Situation umgehen können, da der Mensch einerseits Sprache produktiv verwendet, also neue Worte und Konstruktionen erfindet, und andererseits Fehler bei der Produktion von Sprache macht. Aus diesem Grund werden Grammatiken häufig nicht so aufgestellt, wie es das Kompetenzsystem vorschreibt, sondern wie es einem tatsächlichen Sprachgebrauch entspricht. Versucht man das Problem also dadurch in den Griff zu bekommen, daß man mögliche Fehler in der Grammatik modelliert, ergeben sich wiederum Probleme mit der Mehrdeutigkeit der Analysen.

- Präferenzen: Elaborierte Grammatiken weisen einem einzigen Satz oft eine große Anzahl von möglichen Strukturen zu. Im Gegensatz dazu nimmt ein menschlicher Sprecher meistens nur eine oder wenige Lesarten[3] für einen Satz wahr. Andere, weniger präferierte Lesarten werden erst durch intensivere Suche oder auch gar nicht wahrgenommen.

Die beschriebenen Probleme sowie die Tatsache, daß dem Menschen seine Fähigkeit zu sprechen nur zu einem Teil angeboren ist, während ein anderer Teil durch eine Sprachhistorie, also durch sprachliche Reize, erworben wird, legen die Idee nahe, eine Grammatik nicht mehr (ausschließlich) von einem menschlichen Experten erstellen zu lassen. Statt dessen soll ein Teil oder bei einigen Ansätzen die Gesamtheit der Grammatik aus einer (großen) Menge von sprachlichen Eingaben gelernt werden. Dabei bieten sich zwei Herangehensweisen an: statistische Verfahren und Methoden des maschinellen Lernens. Während das maschinelle Lernen z. B. für die Lösung von Kategorisierungsproblemen (Morik, 1995) eingesetzt wird,[4] haben sich statistische Verfahren in der Computerlinguistik bereits bei der Sprachsignalerkennung[5] etabliert. Statistische Verfahren bieten zudem den Vorteil, bereits von Haus aus Präferenzen oder Bewertungen für mögliche Strukturen zu liefern.

Stochastische Verfahren erweitern auf mehr oder weniger direktem Wege die oben beschriebene Kette aus Kompetenzgrammatik, formalem Modell und Parsing–Algorithmus um ein stochastisches Modell, das den einzelnen (Teil–)Strukturen einer Analyse einen Präferenzwert bzw. eine Wahrscheinlichkeit zuweist. Da ein solches stochastisches Modell automatisch durch Trainingsverfahren parametrisiert werden kann, besteht Anlaß zu der Hoffnung, die oben angesprochenen Probleme verringern zu können.

[3] Der Begriff 'Lesart' umfaßt mehr als der der 'Struktur' eines Satzes, kann aber in diesem Kontext als vergleichbar aufgefaßt werden.

[4] Lernverfahren werden auch im Bereich der Sprachverarbeitung benutzt. So wendet Brill (1995) das fehlergesteuerte Lernen von Transformationsregeln auf die Kategoriebestimmung an; Samuelsson, Tapanainen und Voutilainen (1996) lassen Regeln für Constraint–Grammatiken (Karlsson, 1990) lernen.

[5] Siehe etwa (Eppinger und Herter, 1993) oder (O'Shaughnessy, 1990) für eine Einführung.

Nachdem im nächsten Abschnitt einige Kriterien für die Beurteilung entwickelt worden sind, sollen verschiedene statistische Verfahren zur Analyse natürlicher Sprache vorgestellt werden. Dabei wird klar werden, daß das stochastische Modell z. T. sehr direkt, d. h. zum Beispiel deklarativ auf der Ebene der elementaren Einheiten und Kombinationsoperationen des Sprachmodells, etabliert wird. In diesen Fällen ist es meist einfach, die den Strukturen zugewiesenen Präferenzwerte deklarativ anzugeben; die Algorithmen müssen dann so erweitert werden, daß sie diese Werte korrekt berechnen. Bei anderen Verfahren wird das stochastische Modell operationell durch Angabe eines Parsing–Algorithmus festgelegt.

2.2 Kriterien

Für die Beurteilung von stochastischen Modellen kann man verschiedene Kriterien verwenden. Dazu zählen etwa experimentelle Ergebnisse, Einfachheit einer Realisierung, Effizienz bzgl. des Zeit– und Platzbedarfs einer Implementation und die kognitive Adäquatheit der Modellierung. Neben diesen soll aber hier im Vordergrund stehen, welche Informationen im Kontext einer Äußerung sich das stochastische Modell zunutze macht. Insbesondere sind deshalb die folgenden Kriterien bei einer Beurteilung eines Modells von Interesse:

- Lexikalische Integration: Ist der Formalismus in der Lage, Abhängigkeiten zwischen lexikalischen Einheiten (Lexeme bzw. Lemmata) zu berücksichtigen? Die Beispielsätze 2.1 bis 2.4 unterscheiden sich in der Abfolge der Kategorien nicht; trotzdem erwartet man aufgrund der unterschiedlichen lexikalischen Affinitäten unterschiedliche präferierte Lesarten.

  (2.1) Er bemalt die Wand mit dem Pinsel.
  (2.2) Er bemalt die Wand mit dem Stuck.
  (2.3) Er bemalt die Wand in der Laube.
  (2.4) Er bemalt das Geschenk in der Laube.

- Strukturelle Abhängigkeiten: Inwieweit zieht der Formalismus strukturelle Abhängigkeiten ins Kalkül? So sind die Lexeme ‘wirft’ und ‘Ball’ in den beiden Sätzen (2.5) und (2.6) strukturell gleich weit, bzgl. der linearen Abfolge aber offensichtlich unterschiedlich weit voneinander entfernt.

  (2.5) Peter wirft den Ball in den Korb.
  (2.6) Peter wirft den großen roten Ball in den Korb.

- Semantische Klassenbildung: Erlaubt das Verfahren eine Integration von semantischen Klassen, um selektionale Restriktionen zu berücksichtigen? Als Abstraktion über den konkreten lexikalischen Einheiten sollten allein die semantischen Eigenschaften wie Belebtheit dafür sorgen, daß die beiden Sätze 2.7 und 2.8 jeweils mit dem Wort ‘Bauarbeiter’ als Subjekt analysiert werden.

(2.7) Die Bauarbeiter errichten Kuhställe.

(2.8) Kuhställe errichten die Bauarbeiter.

- Diskurs und Pragmatik: Sind dem Verfahren Informationen über die Grenze einer Äußerung hinweg zugänglich? Inwieweit kann Wissen zu der Gesprächssituation z. B. über Dialogpartner und Intentionen der Sprecher verwendet werden? Die Forderung nach einer Einflußnahme von pragmatischen Aspekten ist für stochastische Verfahren eine weitgehende, die Wichtigkeit für das menschliche Sprachverstehen ist jedoch unbestritten.[6]

- Mächtigkeit: Inwieweit schränkt das Verfahren die Auswahl der statistischen Einheiten ein bzw. welche Informationen können von der Statistik berücksichtigt werden? Ist es möglich, Abhängigkeiten jeweils auf der angemessenen Ebene (lexikalische oder strukturelle Einheiten, Kategorien, semantische Klassen u. a. m.) auszudrücken?

Wie in den folgenden Abschnitten deutlich werden wird, steht der Forderung nach Berücksichtigung von möglichst umfangreichen und vielseitigen Informationen das Problem gegenüber, die sich durch komplexe Modelle ergebende große Anzahl an Parametern abschätzen zu können. In allen Ansätzen werden deshalb Vereinfachungen durchgeführt, und wichtige Informationen bleiben unberücksichtigt.

2.3 Wortfolgen

Die einfachste und zugleich eine äußerst erfolgreiche statistische Modellierung von natürlicher Sprache sind die sogenannten n–Gramm–Modelle (Jelinek, 1990; Charniak, 1993; Merialdo, 1994; Dermatas und Kokkinakis, 1995).

Was macht diese Modelle aus? Als Hauptannahme hinter diesen Modellen steht, daß nur die $n-1$ einem Wort vorausgehenden Worte einen Einfluß auf dieses Wort haben. Dies stellt natürlich eine starke Vereinfachung dar, da ein Wort auf der einen Seite aufgrund von beliebig langen Abhängigkeiten (engl. *unbounded dependencies*) auch von Worten abhängig sein kann, die weiter als n Lexeme entfernt auftreten. Auf der anderen Seite bleiben eine Vielzahl weiterer — z. T. schwer erfaßbarer — Größen wie syntaktische Struktur, Semantik, Kontext u. a. m. unberücksichtigt. Je größer die Zahl n ist, desto größer ist die Anzahl der zu schätzenden stochastischen Parameter.[7] Aus diesem Grund wird n meist relativ klein gewählt, z. B. $n = 2$ für Bigramme oder $n = 3$ für Trigramme. Gegeben sei eine Wortfolge $\Omega = \omega_1 \ldots \omega_m$. Aus formalen Gründen sei diese Wortfolge um eine prinzipiell unbegrenzte Folge eines ausgezeichneten Symbols $\tilde{\omega}$ nach vorne ergänzt, so daß $\forall_{i \leq 0} \omega_i = \tilde{\omega}$ gilt. Für Trigramme erhält man dann:

[6] Zum Beispiel wird für das Verständnis der elliptischen Äußerung 'Pils?' in einer Kneipe fast ausschließlich pragmatisches Wissen verwendet. Nur weil eine Auflösung aufgrund von pragmatischem Wissen möglich ist, kann die sprachliche Form derart reduziert werden.

[7] Die Anzahl der zu schätzenden Parameter ergibt sich bei einer Lexikongröße L zu L^n.

$$P(\omega_i|\omega_1,\ldots,\omega_{i-1}) \approx P(\omega_i|\omega_{i-1},\omega_{i-2}) \tag{2.1}$$

Für die Wortfolge Ω ergibt sich dann aufgrund der angenommenen Unabhängigkeit zwischen den Vorkommen der Worte:

$$P(\Omega) \approx \prod_{i=1}^{m} P(\omega_i|\omega_{i-1},\ldots,\omega_1) \tag{2.2}$$

$$\approx \prod_{i=1}^{m} P(\omega_i|\omega_{i-1},\omega_{i-2}) \tag{2.3}$$

In Gleichung 2.2 geht die obige Grundannahme der n–Gramme ein, während Gleichung 2.3 die Vereinfachung auf Trigramme beschreibt.

n–Gramme finden bei der Kategoriebestimmung (engl. *part–of–speech tagging*) und bei der Erkennung gesprochener Sprache eine große Rolle. Eine syntaktische Struktur wird einer Äußerung allerdings nicht ohne eine Erweiterung des Schemas zugewiesen.

Die Abschätzung der Parameter $P(\omega_i \,|\, \omega_{i-1},\omega_{i-2})$ erfolgt durch einfaches Abzählen in einem großen Korpus, also durch Verwendung des Maximum–Likelihood–Schätzer. Diese Parameter werden dann meist mit Glättungsverfahren (siehe Abschnitt 4.9) verbessert.

Die n–Gramme stellen bei den stochastischen Modellen ein Extrem dar. Sie berücksichtigen ausschließlich Abfolgeabhängigkeiten von lexikalischen Einheiten, strukturelle Eigenschaften bleiben völlig unbeachtet. Eine Abstraktion zu semantischen Klassen oder gar Intentionen findet ebenfalls nicht statt. Ihren Erfolg verdanken diese Modelle ihrer Einfachheit bzgl. ihrer Realisierbarkeit und der geringen Voraussetzungen an ein Korpus, der lediglich als Abfolge von Äußerungen vorliegen muß.

Lafferty, Sleator und Temperly (1992) stellen zusammenfassend fest:

> *... one can hope to contest the unreasonable dominion that the insipid trigram holds over probabilistic models of natural language.*

2.4 Probabilistische kontextfreie Grammatiken

Die erste statistische Erweiterung im Sinne der Einführung in Abschnitt 2.1 stellt die probabilistische kontextfreie Grammatik (engl. *probabilistic context–free grammar,* kurz *PCFG*) dar. Dieser liegt eine kontextfreie Grammatik (Hopcroft und Ullman, 1979) zugrunde, bei der die produktiven Einheiten — die kontextfreien Regeln — mit Wahrscheinlichkeiten angereichert werden. Auf diese Weise lassen sich dann auch Wahrscheinlichkeiten für Zeichenketten, Strukturbäume bzw. Ableitungen angeben. Die Darstellung orientiert sich an denen von Krenn und Samuelsson (1996) sowie von Charniak (1993).

2.4.1 Definition

Eine probabilistische kontextfreie Grammatik besteht aus einem Quintupel $G = \langle V_N, V_T, S, R, P\rangle$, wobei den einzelnen Elementen folgende Bedeutung zukommt:

V_N	endliche Menge an Nichtterminalsymbolen
V_T	endliche Menge an Terminalsymbolen; $V = V_N \cup V_T$
$S \in V_N$	ausgezeichnetes Startsymbol
R	endliche Menge von Produktionen der Form $X \to \beta$ mit $X \in V_N$ und $\beta \in V^*$
P	Funktion $R \mapsto [0,1]$ mit

$$(\forall X \in V_N) \sum_{\beta \in V^*} P(X \to \beta) = 1 \qquad (2.4)$$

Die Funktion P ordnet also jeder Produktion aus R eine Wahrscheinlichkeit[8] zu. Die Forderung, daß sich die Summe aller Wahrscheinlichkeiten für ein bestimmtes Nichtterminal zu Eins ergibt, ergibt sich aus der Beobachtung, daß jedes Nichtterminal in jeder gültigen Ableitung durch eine Produktionenanwendung ersetzt werden muß.

Im folgenden werden einige Begriffe eingeführt, die sowohl für kontextfreie Grammatiken als auch für ihre probabilistische Version existieren und die Einführung von weiteren Wahrscheinlichkeiten vereinfachen.

Die Ableitungsrelation $\Rightarrow$ ist wie gewohnt definiert. Eine Zeichenkette $\alpha X \gamma \in V^*$ mit $X \in V_N$ läßt sich nur unter der Bedingung $X \to \beta \in R$ in die Zeichenkette $\alpha\beta\gamma \in V^*$ umschreiben (formal $\alpha X\gamma \Rightarrow \alpha\beta\gamma$). Sei $\Rightarrow^*$ die transitive, reflexive Hülle der direkten Ableitungsrelation $\Rightarrow$. Die Sprache der von der kontextfreien Grammatik G akzeptierten Worte[9] ergibt sich dann wie folgt:

$$L(G) = \{\Omega \in V_T^* \mid S \Rightarrow^* \Omega\} \qquad (2.5)$$

Für $\Omega \in L(G)$ gibt es also eine Folge von Symbolketten, Ableitung genannt, $\phi_0, \ldots, \phi_n$ mit $\phi_i \in V^*$ und

$$S = \phi_0 \Rightarrow \phi_1 \Rightarrow \ldots \Rightarrow \phi_n = \Omega \qquad (2.6)$$

sowie eine Folge von Produktionen $\xi_1, \ldots, \xi_n$ mit $\xi_i \in R$, so daß

[8] Die Notation $P(X \to \beta)$ ist irreführend, da es sich um eine *bedingte* Wahrscheinlichkeit $P(X \to \beta \mid X)$ handelt.

[9] Die Bezeichnung der Nichtterminalfolgen variiert je nach Betrachtungsstandpunkt. Aus der Sicht der formalen Sprachen erscheint der Begriff 'Wort' gerechtfertigt, für einen Computerlinguisten ist der Begriff 'Satz' geläufiger.

$$\phi_{i-1} = \alpha X \gamma \quad \text{und} \quad \phi_i = \alpha\beta\gamma \quad \text{sowie} \quad \xi_i = X \to \beta \quad \text{für alle i} \tag{2.7}$$

gilt. Einige dieser Ableitungen unterscheiden sich nur in der Reihenfolge der Anwendung der Produktionen. Dagegen wird während einer Linksableitung immer dasjenige Nichtterminal X zuerst ersetzt, das am weitesten links steht, d. h. für jede Produktionenanwendung ξ_i mit ϕ_{i-1} und ϕ_i wie oben gilt $\alpha \in V_T^*$.[10] Ein Schritt in der Linksableitung wird mit $\phi \Rightarrow_l \pi$ bezeichnet, die transitive, reflexive Hülle analog mit $\Rightarrow_l^*$. Eine Linksableitung ist eindeutig durch die Folge der Produktionen $\xi_1, \ldots, \xi_n$ definiert, da festgelegt ist, an welcher Stelle die Produktionenanwendung stattfindet.[11] Weiterhin ist jeder Linksableitung eineindeutig ein Strukturbaum zugeordnet (Schöning, 1992).

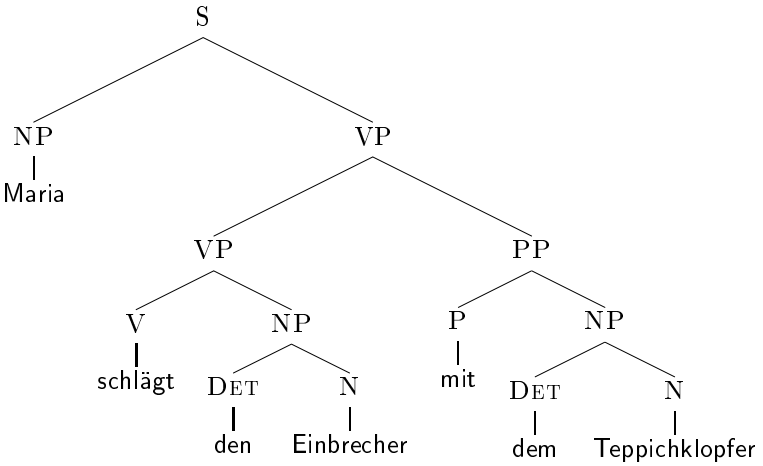


Abbildung 2.1: Möglichkeit für einen Phrasenstrukturbaum

Beispielsweise entspricht dem Strukturbaum in Abbildung 2.1 die Linksableitung in Abbildung 2.2.

(1.) S → NP VP
(2.) NP → Maria
(3.) VP → VP PP
(4.) VP → V NP
(5.) V → schlägt
(6.) NP → DET N
(7.) DET → den
(8.) N → Einbrecher
(9.) PP → P NP
(10.) P → mit
(11.) NP → DET N
(12.) DET → dem
(13.) N → Teppichklopfer

Abbildung 2.2: Mögliche Linksableitung

Durch den Übergang zur Linksableitung hat man aber keinesfalls jede Mehrdeutigkeit beseitigt. Ein Wort $\Omega \in L(G)$ kann mehr als eine Linksableitung und

[10] Eine Rechtsableitung ist dann analog definiert.
[11] Im folgenden wird deshalb auch eine Folge von Produktionenanwendungen als Ableitung bezeichnet.

damit mehr als einen Strukturbaum besitzen. Beispielsweise kann eine Grammatik dem Wort bzw. Satz[12] in Abbildung 2.1 zusätzlich den Strukturbaum in Abbildung 2.3 zuweisen. Eine solche Grammatik heißt dann *mehrdeutig*.

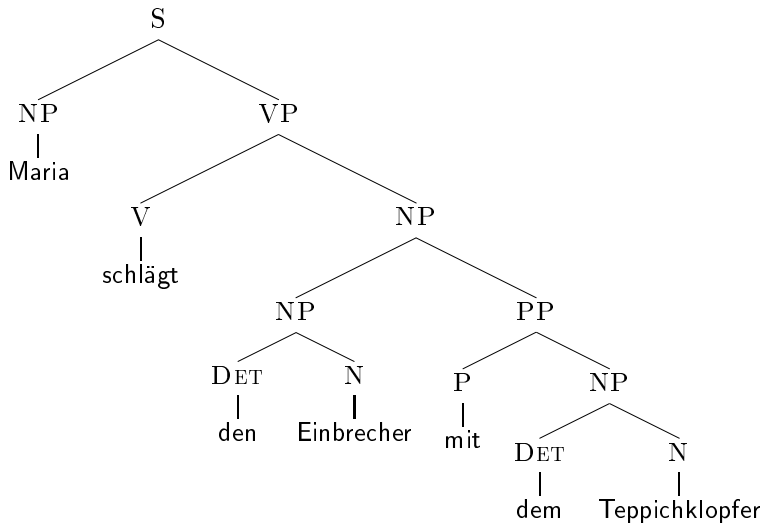


Abbildung 2.3: Alternative Möglichkeit für einen Phrasenstrukturbaum

Die wahrscheinlichkeitszuweisende Funktion P kann nun von einzelnen Produktionen auf Linksableitungen bzw. Strukturbäume und Sätze erweitert werden. Sei $\Xi = \xi_1, \ldots, \xi_n$ die Folge der Produktionen einer Linksableitung. Dann berechnet sich die Wahrscheinlichkeit der Linksableitung wie folgt:

$$
\begin{aligned}
P(\Xi) &= P(\xi_1, \ldots, \xi_n) && (2.8)\\
&= P(\xi_n | \xi_1, \ldots, \xi_{n-1}) \cdot P(\xi_1, \ldots, \xi_{n-1}) && (2.9)\\
&= \prod_{i=1}^{n} P(\xi_i | \xi_1, \ldots, \xi_{i-1}) && (2.10)\\
&\approx \prod_{i=1}^{n} P(\xi_i) && (2.11)
\end{aligned}
$$

Die Gleichung 2.11 beruht auf der wichtigsten Grundannahme für probabilistische kontextfreie Grammatiken, die besagt, daß die Anwendungen zweier Produktionen unabhängig voneinander sind, d. h. die Anwendung geschieht kontextfrei. Es ist klar sein, daß dies eine erhebliche Vereinfachung darstellt. Deutlich wird dies beispielsweise bei der Produktion NP $\rightarrow$ NP PP, da mehrere bereits von einer Nominalphrase abhängige Präpositionalphrasen die Angliederung einer weiteren solchen intuitiv unwahrscheinlicher machen, dies aber durch eine PCFG nicht berücksichtigt wird.

[12] Vergleiche Fußnote 9 auf Seite 11.

Auch einem einzelnen Satz läßt sich nun eine Wahrscheinlichkeit zuweisen, indem man die Wahrscheinlichkeiten aller seiner Linksableitungen aufsummiert. Damit ergibt sich für probabilistische kontextfreie Grammatiken folgende Wahrscheinlichkeitsfunktion, die Regeln, Ableitungen und Sätzen eine Wahrscheinlichkeit zuordnet.

- Wahrscheinlichkeiten für Produktionen $P : R \mapsto [0, 1]$
- Wahrscheinlichkeiten für Ableitungen $P : R^* \mapsto [0, 1]$ mit

$$P(\Xi \in R^*) = \begin{cases} \prod_{i=1}^{n} P(\xi_i) & : \quad \Xi \text{ ist Linksableitung} \\ 0 & : \quad \text{sonst} \end{cases} \tag{2.12}$$

  Die Wahrscheinlichkeit einer Linksableitung ist das Produkt der Wahrscheinlichkeiten aller angewendeten Produktionen; Nicht–Linksableitungen wird die Wahrscheinlichkeit Null zugewiesen.

- Wahrscheinlichkeiten für Terminalfolgen $P : V_T^* \mapsto [0, 1]$ mit

$$P(\Omega \in V_T^*) = \sum_{\Xi : S \overset{\Xi}{\Rightarrow}_l \Omega} P(\Xi) \tag{2.13}$$

  Die Wahrscheinlichkeit einer Terminalfolge ergibt sich als Summe der Wahrscheinlichkeiten aller sie erzeugenden (Links–)Ableitungen.

2.4.2 Parsing–Algorithmen

Nach der Definition der PCFG sollen im folgenden Verfahren vorgestellt werden, wie mit solchen Grammatiken für einen Eingabesatz $\Omega = \omega_1, \ldots, \omega_n$ der wahrscheinlichste Strukturbaum bzw. die wahrscheinlichste Ableitung $\Xi_{\text{Ziel}} \in R^*$ gemäß der Gleichungen 2.12 und 2.14 gefunden werden kann.

$$\Xi_{\text{Ziel}} = \arg \max_{\Xi \in R^*} P(\Xi) \tag{2.14}$$

Die einfachsten Parsing–Algorithmen für kontextfreie Grammatiken sind diejenigen der nach den Autoren Cocke, Kasami und Younger benannten CKY–Familie (Naumann und Langer, 1994). Diese fordern allerdings, daß die Grammatik in Chomsky–Normalform (kurz CNF)[13] vorliegt. Eine beliebige probabilistische kontextfreie Grammatik läßt sich nach einfachen Ersetzungsregeln in Chomsky–Normalform bringen (Krenn und Samuelsson, 1996). Aus Sicht der formalen Sprachen bedeutet dies eine Äquivalenz zwischen der allgemeinen Form der PCFG und der CNF; für computerlinguistische Anwendungen

[13] Eine Grammatik in Chomsky–Normalform enthält nur Regeln der Form $X \to YZ$ mit $X, Y, Z \in V_N$ oder der Form $X \to x$ mit $X \in V_N$ und $x \in V_T$.

stellt eine solche Umwandlung allerdings ein Problem dar, da sich die Strukturbeschreibungen stark verändern. Hier wird die CNF nur verwendet, um die Darstellung der Algorithmen einfach zu halten.

Der Algorithmus basiert auf einfachen rekursiven Gleichungen für eine Anzahl von Variablen. Für jedes Nichtterminal X_r und jedes Intervall $\omega_i, \ldots, \omega_k$ mit $1 \leq i \leq k \leq n$ wird in der Variablen $\delta_{i,k+1}^{X_r}$ die maximale Wahrscheinlichkeit vermerkt, daß das Nichtterminal X_r die Wurzel des partiellen Strukturbaumes von $\omega_i, \ldots, \omega_k$ darstellt. Diese Variablen können bei bekanntem Satz $\Omega = \omega_1, \ldots, \omega_n$ wie folgt berechnet werden:

- Für die Variablen der Intervalle der Länge eins $\delta_{i,i+1}^{X_r}$ mit $X_r \in V_N, 1 \leq i \leq n$ gilt:

$$\delta_{i,i+1}^{X_r} = P(X_r \to \omega_i) \tag{2.15}$$

- Alle anderen Variablen mit $X_r, X_s, X_t \in V_N$ und $1 \leq i < k \leq n$ kann man durch

$$\delta_{i,k}^{X_r} = \max_{\substack{i<j<k \\ X_s, X_t \in V_N}} P(X_r \to X_s X_t) \cdot \delta_{i,j}^{X_s} \cdot \delta_{j,k}^{X_t} \tag{2.16}$$

rekursiv berechnen.

Oh, und meine Diplomarbeit handelt von Mahagonitischen. — U. Arnold

Diese Vorgehensweise entspricht direkt dem ursprünglichen CKY–Algorithmus mit dem Unterschied, daß die Variablen zusätzlich mit der entsprechenden Wahrscheinlichkeit versehen werden. Die Wahrscheinlichkeit des letztendlich gesuchten Phrasenstrukturbaumes für den gesamten Satz ist dann $\delta_{1,n+1}^{S}$. Der gesuchte Strukturbaum läßt sich finden, wenn man an den Variablen $\delta_{i,k}^{X_r}$ zusätzlich notiert, welche anderen Variablen $\delta_{i,j}^{X_s}$ und $\delta_{j,k}^{X_t}$ zu dieser beigetragen haben. Die Zeitkomplexität ergibt sich für diesen Algorithmus zu $O(|V_N|^3 \cdot n^3)$, die Platzkomplexität zu $O(|V_N| \cdot n^2)$.

Das CYK–Parsing–Verfahren zeigt starke Ähnlichkeiten mit der Viterbi–Suche bei Hidden–Markov–Modellen (kurz HMM). Siehe z. B. Charniak (1993, S. 87ff.) für eine Gegenüberstellung oder den folgenden Abschnitt 2.4.3 für eine Anwendung des ebenfalls von den HMM her bekannten Baum–Welch–Algorithmus, der Parallelen zur Viterbi–Suche aufweist.

Die Einschränkung auf Grammatiken in CNF ist keinesfalls zwingend. Parsing–Algorithmen, die ein sogenanntes Chart verwenden und auf den nach seinem Erfinder benannten Early–Algorithmus (Earley, 1970) zurückgehen, arbeiten mit (fast) beliebigen Grammatikformen und können leicht an PCFG angepaßt werden. In (Gazdar und Mellish, 1989, S. 195ff) werden für einen Bottom–Up–Parser die sogenannte Fundamentalregel und Bottom–Up–Regel angegeben, die hier in einer an PCFG angepaßten Version wiedergegeben werden.[14]

[14] Ähnlich einfach kann die Regel für das Top–Down–Parsing angepaßt werden.

Fundamental rule

If the chart contains edges $\langle i, j, X \rightarrow \alpha.Y\gamma, p_1\rangle$ and $\langle j, k, Y \rightarrow \beta., p_2\rangle$, where $X, Y \in V_N$ and $\alpha, \beta, \gamma \in V^*$, then add edge $\langle i, k, X \rightarrow \alpha Y.\gamma, p_1 \cdot p_2\rangle$ to the chart or adjust the probability of edge ϵ to $\max(p_1 \cdot p_2, p_3)$ if an edge $\epsilon = \langle i, k, X \rightarrow \alpha Y.\gamma, p_3\rangle$ already exists in the chart.

Bottom–up rule

If you are adding edge $\langle i, j, Y \rightarrow \alpha., p'\rangle$ to the chart, then for every rule in the grammar of the form $X \rightarrow Y\beta \quad (p)$, add an edge $\langle i, i, X \rightarrow .Y\beta, p\rangle$ to the chart.

Die wahrscheinlichste Ableitung findet sich im Chart als Kante $\langle 0, n, S \rightarrow \alpha., p\rangle$. Mit einer Zeitkomplexität von $O(n^3)$ und einer Platzkomplexität $O(n^2)$ fallen beide Algorithmen in dieselbe Komplexitätsklasse. Charniak (1993) gibt Hinweise zu Beweisen für die auf allgemeine kontextfreie Grammatiken erweiterten Gleichungen 2.15 und 2.16.

2.4.3 Schätzung von Parametern

Die zu schätzenden Parameter einer PCFG sind die Wahrscheinlichkeiten der Produktionen $X \rightarrow \alpha$ mit $X \in V_N$ und $\alpha \in V^*$. Im folgenden werden zwei Möglichkeiten vorgestellt, diese Parameter mit Hilfe eines Korpus für eine Grammatik zu schätzen.

Am einfachsten ist eine Schätzung, wenn ein syntaktisch annotiertes Korpus zur Verfügung steht, d. h. die Sätze als Strukturbäume vorliegen. Nach Krenn und Samuelsson (1996, Abschnitt 1.7.3) sind die relativen Häufigkeiten die wichtigsten Maximum–Likelihood–Schätzer für zu schätzende Wahrscheinlichkeiten. Deshalb kann man die Parameter wie folgt abschätzen:

$$P(X \rightarrow \alpha) \approx \frac{\text{Anzahl Vorkommen } X \rightarrow \alpha \text{ im Korpus}}{\text{Anzahl Vorkommen } X \text{ im Korpus}} \tag{2.17}$$

Beachtenswert ist im Fall des Vorliegens eines auf diese Weise annotierten Korpus, daß die gesamte PCFG — also nicht nur die Wahrscheinlichkeiten — aus dem Korpus abgeleitet werden kann, da jeder innerer Knoten in einem Strukturbaum genau einer kontextfreien Regel entspricht. Unberücksichtigt bleiben dabei Aspekte des Datenmangels.

Wesentlich schwieriger gestaltet sich die Schätzung, wenn kein annotiertes Korpus vorliegt. In diesem Fall ist man auf Verfahren angewiesen, die zwar iterativ die Parameter verbessern, jedoch nicht garantieren können, daß das globale Maximum[15] gefunden wird, sondern evtl. nur lokale Maxima liefern.

[15] Als Qualitätsmaß für die zu findenden Parameter kann die Wahrscheinlichkeit, die einem annotierten Testkorpus zugewiesen wird, verwendet werden; dieser Testkorpus dient dann nicht zum Training der Parameter.

Hier sollen die Grundlagen des sogenannten Inside–Outside–Algorithmus vorgestellt werden, der starke Ähnlichkeit mit dem in der Sprachsignalverarbeitung verwendeten Forward–Backward–Algorithmus besitzt und als eine Variante des Parsing–Algorithmus aus Abschnitt 2.4.2 aufgefaßt werden kann. Der Algorithmus verwendet sogenannte Inside– und Outside–Variablen (vgl. mit den Variablen $\delta_{i,k}^{X_r}$ in Abschnitt 2.4.2), die die folgende Bedeutung und Beziehung zu den zu schätzenden Parametern haben:

- Die Inside–Wahrscheinlichkeit $\beta_{i,k+1}^{X_r}$ ist die Wahrscheinlichkeit, daß das Nichtterminal X_r während einer Ableitung zur Terminalfolge des Intervalls $\omega_i, \ldots, \omega_k$ expandiert wird. Formal gilt:

$$\beta_{i,k+1}^{X_r} \stackrel{\text{def}}{=} P(X_r \Rightarrow^* \omega_i, \ldots, \omega_k \mid X_r) \tag{2.18}$$

  Für alle Intervalle der Länge Eins und alle Nichtterminale $X_r \in V_N$ gilt:

$$\beta_{i,i+1}^{X_r} = P(X_r \to \omega_i) \tag{2.19}$$

  Der Rest der Inside–Variablen läßt sich dann mit $X_r, X_s, X_t \in V_N$ und $1 \leq i < k \leq n$ wie folgt berechnen:

$$\beta_{i,k}^{X_r} = \sum_{X_s, X_t \in V_N} \sum_{i<j<k} P(X_r \to X_s X_t) \cdot \beta_{i,j}^{X_s} \cdot \beta_{j,k}^{X_t} \tag{2.20}$$

- Die Outside–Wahrscheinlichkeit stellt insofern das Gegenstück zur Inside–Wahrscheinlichkeit dar, als daß sie die Wahrscheinlichkeit angibt, mit der die Terminale 'außerhalb' des Nichtterminals X_r erzeugt werden. Formal gilt:

$$\alpha_{i,k}^{X_r} \stackrel{\text{def}}{=} P(S \Rightarrow^* \omega_1, \ldots, \omega_{i-1}, X_r, \omega_{k+1}, \ldots, \omega_n \mid S) \tag{2.21}$$

  Für das Intervall, das den gesamten Satz überspannt, gilt:

$$\alpha_{1,n}^{X_r} = \begin{cases} 1.0 & \text{für } X_r = S \\ 0.0 & \text{sonst} \end{cases} \tag{2.22}$$

  Die Variablen $\alpha_{i,k}^{X_r}$ berechnen sich für $X_r, X_s, X_t \in V_N$ und $1 \leq i \leq k \leq n$ wie folgt:

$$\begin{aligned} \alpha_{i,k}^{X_r} &= \sum_{X_s, X_t \in V_N} \sum_{j=1}^{k-1} \alpha_{j,k}^{X_s} \cdot P(X_s \to X_t X_r) \cdot \beta_{j,i}^{X_t} \\ &+ \sum_{X_s, X_t \in V_N} \sum_{j=k+1}^{n} \alpha_{i,j}^{X_s} \cdot P(X_s \to X_r X_t) \cdot \beta_{k+1,j}^{X_t} \end{aligned} \tag{2.23}$$

Mit Hilfe der Inside– und Outside–Variablen werden ausgehend von einer (beliebigen) Initialisierung der Wahrscheinlichkeiten $P(X \to \omega)$ und $P(X_r \to X_s X_t)$ in einem iterativen Prozeß diese neu geschätzt, bis ein Konvergenzkriterium erfüllt ist.[16]

Dazu werden die folgenden Zählvariablen[17] eingeführt:

$$C(X \to \omega) = \frac{\sum_{1 \le k \le n} \alpha^X_{k,k} \cdot P(X \to \omega)}{P(\omega_1 \ldots \omega_n)} \tag{2.24}$$

$$C(X_r \to X_s X_t) = \frac{\sum_{1 \le k < l < m \le n} \alpha^{X_r}_{k,m} \cdot P(X_r \to X_s X_t) \cdot \beta^{X_s}_{k,l} \cdot \beta^{X_t}_{l,m+1}}{P(\omega_1 \ldots \omega_n)} \tag{2.25}$$

Diese Gleichungen können nun im iterativen Prozeß verwendet werden, um von den Inside– und Outside–Variablen zu neuen, verbesserten Schätzungen $\tilde{P}$ für die Parameter $P(X \to \alpha)$ mit $\alpha \in V^*$ zu gelangen.

$$\tilde{P}(X \to \alpha) = \frac{C(X \to \alpha)}{\sum_{\beta \in V^*} C(X \to \beta)} \tag{2.26}$$

2.4.4 Erweiterungen

Das Schema der PCFG ist mehrfach erweitert worden. Im Abschnitt 2.8 werden ableitungsbasierte Grammatiken und das Paring als Mustererkennung vorgestellt, die ebenso wie der Parser $\mathcal{P}$earl (Magerman und Marcus, 1991) eine Erweiterung der PCFG darstellen.

2.4.5 Beurteilung

In gewisser Weise stellen die PCFG ein Gegenstück zu den n–Grammen (siehe Abschnitt 2.3) dar. Während die letzteren nur Abfolgeabhängigkeiten auf *lexikalischer* Ebene berücksichtigen, verwenden die PCFG ausschließlich *strukturelle* Informationen. PCFG können nur die stochastischen Abhängigkeiten nutzen, die bei *einer* Regelanwendung auftreten. Am schwersten wiegt, daß keinerlei lexikalische Abhängigkeiten erfaßt werden können. Von PCFG in ihrer einfachen Ausführung ist deshalb kaum zu erwarten, daß sie für die Beispielsätze 2.1 bis 2.4 auf S. 8 den intuitiv richtigen Strukturen die größten Wahrscheinlichkeiten zuweisen.

[16] Die Werte bei Erreichen des Konvergenzkriteriums können vom globalen Maximum abweichen. Siehe Bemerkung weiter oben.

[17] Das Verfahren geht auf ein Trainingsverfahren für Markov–Ketten zurück. Dabei wurde die Anzahl der Übergänge von einem Zustand in einen anderen in diesen Variablen gezählt. Hier handelt es sich um ein 'gewichtetes Zählen'. Vergleiche (Charniak, 1993, Kapitel 4 und 6) für eine eingängige Gegenüberstellung.

2.5 Stochastische Baumsubstitutionsgrammatiken

Stochastische Baumsubstitutionsgrammatiken (engl. *stochastic tree substitution grammar,* kurz *STSG*) gehen auf (Bod, 1995; Bod, 1996) zurück, der bemerkt (1995, S. 14f.):

> *If application probabilities are assigned to the single productive units of an underlying competence model, it is tacitly assumed that the statistical units coincide with the linguistic units. [...] This assumption is wrong.*

Als Konsequenz aus dieser Beobachtung (vgl. Abschnitt 2.2) stellt er als Hypothese seiner Arbeit folgende Absichtserklärung voran:

> *Since we do not know beforehand which dependencies may be statistically interesting, we should not constrain or predefine the statistical units, but* take all, arbitrarily large (previously experienced) structures and interpretations as possible statistical units.

Die Arbeiten von Bod (1995) konzentrieren sich aus praktischen Gründen jedoch auf syntaktische Baumfragmente als linguistische Strukturen;[18] das gleiche gilt für die Darstellung hier.

2.5.1 Definition

Eine stochastische Baumsubstitutionsgrammatik besteht aus einem Quintupel $G = \langle V_N, V_T, S, T, P \rangle$, wobei den einzelnen Komponenten folgende Bedeutung zukommt (vgl. Abschnitt 2.4.1):

V_N	endliche Menge an Nichtterminalsymbolen
V_T	endliche Menge an Terminalsymbolen; $V = V_N \cup V_T$
$S \in V_N$	ausgezeichnetes Startsymbol
E	endliche Menge von elementaren Bäumen, deren innere Knoten Nichtterminale aus V_N und deren Blattknoten Terminale oder Nichtterminale aus V sind; $X^{(\theta)} \in V_N$ sei der Wurzelknoten für alle Bäume $\theta \in E$; $\Phi_{(\theta)} \in V^*$ sei die Folge $\phi_1, \ldots, \phi_m$ ihrer Blattknoten
P	Funktion $P : E \mapsto [0, 1]$, die den elementaren Bäumen eine Wahrscheinlichkeit zuweist, mit

$$(\forall X \in V_N) \sum_{\substack{\theta \in E \\ X^{(\theta)} = X}} P(\theta) = 1 \qquad (2.27)$$

[18] Vergleiche Kapitel 8 in (Bod, 1995) für einige Erweiterungen.

Die Menge aller Bäume mit V als Knotenmenge sei mit T bezeichnet, so daß im allgemeinen $E \subseteq T$ gilt. Über diesen Bäumen ist die partielle Operation der *Linkssubstitution*[19] $\circ : T \times T \mapsto T$ wie folgt definiert. Seien θ_1 und θ_2 Bäume, so daß $\Phi_{(\theta_1)} = \alpha X^{(\theta_2)} \beta$ mit $\alpha \in V_T^*$ und $\beta \in V^*$. In anderen Worten, der am weitesten links stehende nichtterminale Blattknoten in θ_1 stimmt mit der Wurzel von θ_2 überein. Dann erhält man $\theta_1 \circ \theta_2$, indem man eben diesen am weitesten links stehenden nichtterminalen Blattknoten $X^{(\theta_2)}$ in θ_1 durch θ_2 ersetzt. Erfüllen θ_1 und θ_2 diese Voraussetzungen nicht, so ist die Substitution nicht definiert.[20]

Eine *Linksableitung*[21] einer STSG ist als Folge von Elementarbäumen $\theta_1, \ldots, \theta_n$ definiert, so daß $X^{(\theta_1)} = S$ gilt sowie $\theta_1 \circ \ldots \circ \theta_n$ definiert und $\Phi_{(\theta_1 \circ \ldots \circ \theta_n)} \in V_T^*$ erfüllt sind. Gilt $\Phi_{(\theta_1 \circ \ldots \circ \theta_n)} = \Omega \in V_T^*$, so heißt die Folge Ableitung der Zeichenkette Ω. Die Wahrscheinlichkeitsfunktion P läßt sich aufgrund der Unabhängigkeitsannahme der Substitutionen auf Ableitungen erweitern.

$$P(\theta_1, \ldots, \theta_n) = \begin{cases} \prod_{i=1}^{n} P(\theta_i) & : \quad \theta_1, \ldots, \theta_n \text{ ist Linksableitung} \\ 0 & : \quad \text{sonst} \end{cases} \tag{2.28}$$

Ein *Strukturbaum* für einen Satz $\Omega \in V_T^*$ ist ein Baum $\theta = \theta_1 \circ \ldots \circ \theta_n$, wenn $\theta_1, \ldots, \theta_n$ eine Ableitung von Ω ist. Die Wahrscheinlichkeitsfunktion P läßt sich nun auf Bäume erweitern.

$$P(\theta) = \sum_{\substack{\Theta \in E^+ \\ \circ(\Theta) = \theta}} P(\Theta) \tag{2.29}$$

Anders als bei den PCFG entspricht bei den STSG zwar einer Linksableitung exakt ein Strukturbaum, allerdings kann es für einen Strukturbaum mehr als eine Linksableitung geben. Sind die beiden in Abbildung 2.4 gezeigten Ableitungen die einzigen für den Strukturbaum, so ergibt sich die Wahrscheinlichkeit für diesen zu $p_1 \cdot p_2 \cdot p_3 + p_4 \cdot p_5$.

Die von einer STSG G generierte Sprache ergibt sich dann wie folgt:

$$L(G) = \{\Omega \in V_T^* \mid \Theta \in E^+ \wedge \Phi_{(\circ(\Theta))} = \Omega\} \tag{2.30}$$

2.5.2 Parsing–Algorithmen

Aus der Definition der STSG ergibt sich, daß Algorithmen für kontextfreie Grammatiken auch für STSG angewendet werden können, wenn man für jeden

[19] Im folgenden wird auch nur Substitution geschrieben.

[20] Es sei $\theta_1 \circ \theta_2 \circ \theta_3$ eine abkürzende Schreibweise für $(\theta_1 \circ \theta_2) \circ \theta_3$ und $\circ(\theta_1, \ldots, \theta_n)$ eine Präfixschreibweise für $\theta_1 \circ \ldots \circ \theta_n$.

[21] Hier wird im folgenden der Begriff Ableitung verwendet.

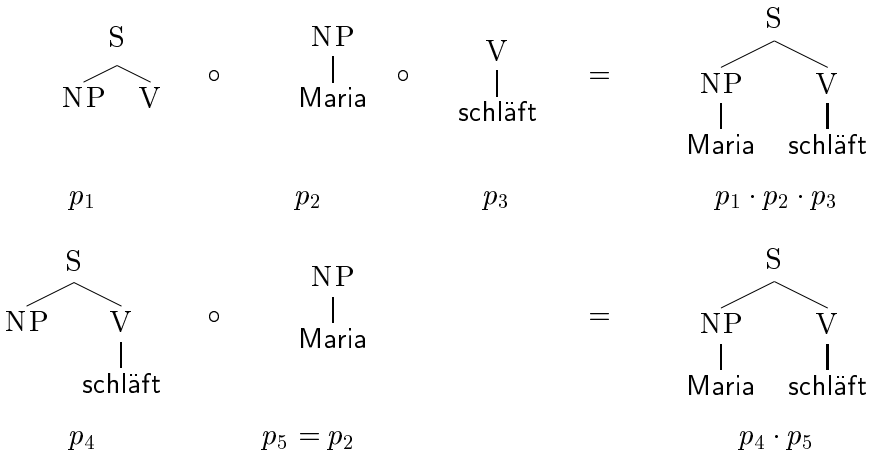


Abbildung 2.4: Zwei Linksableitungen einer STSG für einen Strukturbaum

elementaren Baum θ eine kontextfreie Regel $X^{(\theta)} \rightarrow \Phi_{(\theta)}$ einfügt. Insbesondere die in Abschnitt 2.4.2 erwähnten Algorithmen, die ein sogenanntes Chart verwenden, scheinen geeignet, da sie eine polynomielle Laufzeit von $O(n^3)$ besitzen.[22] Um solche Algorithmen für das Parsing mit STSG zu verwenden, müssen die Kanten in dem Chart statt mit einfachen Kategoriesymbolen mit den vollen elementaren Bäumen sowie mit Verweisen auf die Kanten, die mit ihnen kombiniert wurden, versehen werden.

Das kubische Laufzeitverhalten des Parsing bedeutet keinesfalls, daß in polynomieller Zeit der wahrscheinlichste Strukturbaum (vgl. Gleichung 2.29) ermittelt werden kann. Für einen gegebenen Satz kann es exponentiell viele Strukturbäume geben, die in kompakter Form in dem Chart repräsentiert sind. Zusätzlich kann es für einen Strukturbaum bei STSG exponentiell viele Ableitungen geben.[23] Aus diesem Grund bleibt die direkte Anwendung der Formeln aus Abschnitt 2.5.1 zur Ermittlung des wahrscheinlichsten Strukturbaumes ineffizient bzw. undurchführbar. Bod (1995) verwendet deshalb das Monte–Carlo–Verfahren (bzw. Simulation), einen probabilistischen Algorithmus[24], um den wahrscheinlichsten Strukturbaum abzuschätzen. Dazu wird eine (große) Anzahl von Zufallsableitungen aus dem Chart extrahiert, indem Teilableitungen in dem Chart gemäß ihrer Wahrscheinlichkeit ausgewählt werden. Nach dem Gesetz der großen Zahlen (Stahel, 1995, Abschnitt 5.8) werden wahrscheinliche, sich aus den Ableitungen ergebende Strukturbäume schließlich häufiger als unwahrscheinliche Strukturbäume auftreten. Bod (1995) gibt eine obere Grenze E für die Wahrscheinlichkeit an, daß der häufigste Strukturbaum nicht der wahrscheinlichste ist.

[22] Die Länge der Zeichenkette wird wieder mit n bezeichnet.

[23] Die Situation bei PCFG stellt sich ähnlich dar.

[24] Probabilistische Algorithmen garantieren nicht das korrekte Ergebnis, erlauben aber, die Fehlerbehaftung abzuschätzen und den Fehler beliebig klein zu machen. Den wohl bekanntesten probabilistischen Algorithmus stellt Rabins Algorithmus (Gruska, 1992, Abschnitt 4.2) zur Überprüfung, ob eine gegebene Zahl prim ist, dar.

$$E = \sum_i \left(1 - \left(\sqrt{P_{mpp}} - \sqrt{P_i}\right)^2\right)^N \tag{2.31}$$

Dabei bezeichnet P_{mpp} die Wahrscheinlichkeit des häufigsten Strukturbaumes, P_i die der anderen und N die Anzahl der durchgeführten Simulationen. Zusammen mit einer Abschätzung der Standardabweichung wird diese Grenze genutzt, um die Anzahl der zu erzeugenden Zufallsableitungen N klein zu halten.

2.5.3 Schätzung von Parametern

Bod (1995) wendet das Schema der STSG auf verschiedene Parsing–Probleme an, wobei er aber immer die elementaren Bäume zunächst aus einem Baumkorpus[25] extrahiert. Dabei wird jeder verbundene Teilgraph θ eines Baumes θ' aus dem Korpus als elementarer Baum aufgenommen, wenn er die folgenden beiden Eigenschaften erfüllt:

- θ enthält mindestens zwei Knoten.
- Jeder Knoten des Teilbaumes θ besitzt entweder Entsprechungen aller oder keiner Töchter des ihm entsprechenden Knotens aus dem Baum θ'.

Bei einer Trainingsmenge von 675 Strukturbäumen werden etwa $4 \cdot 10^5$ verschiedene elementare Bäume extrahiert. Neben den elementaren Bäumen werden auch die zu schätzenden Parameter $P(\theta)$ für $\theta \in E$ zunächst aus dem Korpus gewonnen, indem der Maximum–Likelihood–Schätzer, die relativen Häufigkeiten, zur Verwendung kommen.

$$P(\theta) = \frac{\text{Anzahl Vorkommen } \theta \text{ im Korpus}}{\text{Anzahl Vorkommen } \theta' \in E \text{ im Korpus mit } X^{(\theta)} = X^{(\theta')}} \tag{2.32}$$

Ausgehend von diesen Annahmen entwickelt Bod (1995) verschiedene Modelle für jeweils verschiedene Probleme und schlägt Lösungsansätze vor.

- DOP–1: Das erste Modell analysiert Ketten von Kategoriensymbolen anstelle der im allgemeinen üblichen Wortketten. Dabei werden exakt die elementaren Bäume und Wahrscheinlichkeiten wie oben aufgeführt verwendet. Im Abschnitt 2.5.4 werden einige Ergebnisse dieses Modells diskutiert (siehe auch Abbildung 2.5).

[25] Hier findet das ATIS–Korpus (engl. *air travel information system*) aus der Pennsylvania Treebank (Marcus, Santorini und Marcinkiewicz, 1993) Verwendung, das besonders geeignet ist, da es nur Fragen und Imperative enthält, die nicht zwingend Bezug auf einen Kontext nehmen.

- DOP–2: Beim Übergang zu Wortketten tritt unmittelbar das Problem auf, daß einige Worte der Testmenge nicht in der Trainingsmenge auftauchen. Sätze mit unbekannten Worten können nicht direkt analysiert werden[26], so daß beim Auftreten von solchen Worten zusätzliche minimale elementare Bäume angenommen werden müssen, die das betreffende Wort enthalten und so eine Analyse ermöglichen. Als Nachteil ergibt sich, daß den nun möglichen Ableitungen keine Wahrscheinlichkeit mehr zugewiesen werden kann. Um dennoch eine Bewertung zu ermöglichen, wird deshalb im Rahmen einer Teilanalyse (engl. *partial parsing*) auf die Wahrscheinlichkeit der Fragmente außerhalb des unbekannten Wortes zurückgegriffen. Anders ausgedrückt, das unbekannte Wort wird auf eine solche Weise interpretiert, daß die Wahrscheinlichkeit für den umgebenden Teil maximal wird.

- DOP–3: Das aus stochastischer Sicht unbefriedigende Verfahren von DOP–2 und die Einsicht, daß nicht nur unbekannte Worte das Parsing erschweren, sondern auch von der graphemischen Form her bekannte Worte, die aber einer anderen Kategorie angehören als die bekannte Form, führen zu der Annahme, daß die elementaren Bäume eines Korpus nur als eine Stichprobe einer größeren Menge aufzufassen sind. Bod (1995) schätzt die Wahrscheinlichkeit für nicht beobachtete elementare Bäume mit der Good–Turing–Methode (Samuelsson, 1996; Krenn und Samuelsson, 1996, Abschnitt 2.4) ab.

- DOP–4: Als eine Variante von DOP–3 wird hier die sogenannte Add–k–Methode mit verschiedenen Parametern verwendet, um die Wahrscheinlichkeit für nicht beobachtete elementare Bäume abzuschätzen.

- DOP–5: Für das letzte tatsächlich implementierte Modell wird die Variante DOP–3 mit einem externen Lexikon angereichert, so daß nun nur noch diejenigen Worte als unbekannt oder unbekannt bzgl. der Kategorie eingestuft werden müssen, die nicht im Lexikon vorkommen.

2.5.4 Beurteilung

Die Abbildung 2.5 zeigt die Ergebnisse für das Modell DOP–1 aus Bod (1995). Als Genauigkeitsmaß wurde der Anteil der exakt übereinstimmenden Strukturbäume verwendet. Die Daten werden jeweils für Experimente gezeigt, bei denen nur elementare Bäume bis zu einer bestimmten Tiefe verwendet werden.[27] Gegenübergestellt sind die Genauigkeiten für die wahrscheinlichste Ableitung, den wahrscheinlichsten Strukturbaum und den wahrscheinlichsten Strukturbaum, wenn die Testmenge mit der Trainingsmenge identifiziert wird. Bei einer Tiefe von Eins entsprechen die STSG exakt den PCFG, d. h. die Ergebnisse für Ableitung und Strukturbaum müssen dort identisch sein. Es zeigt

[26] Dies ist sofort einsichtig, wenn man bedenkt, daß für solche Sätze ohne Sonderbehandlung kein einziger Strukturbaum erstellt werden kann.

[27] Die Angabe der verbrauchten CPU–Zeit als Indikator für die Zeitkomplexität wurde für die unbegrenzte Tiefe auf 100% gesetzt, um einen Vergleich zu ermöglichen.

sich jedoch bei den anderen Tiefen, daß sich die Ergebnisse für die Ableitung und den Strukturbaum tatsächlich stark unterscheiden (vgl. Abschnitt 2.5.1); die aufwendige Verwendung der Simulation scheint dementsprechend gerechtfertigt.

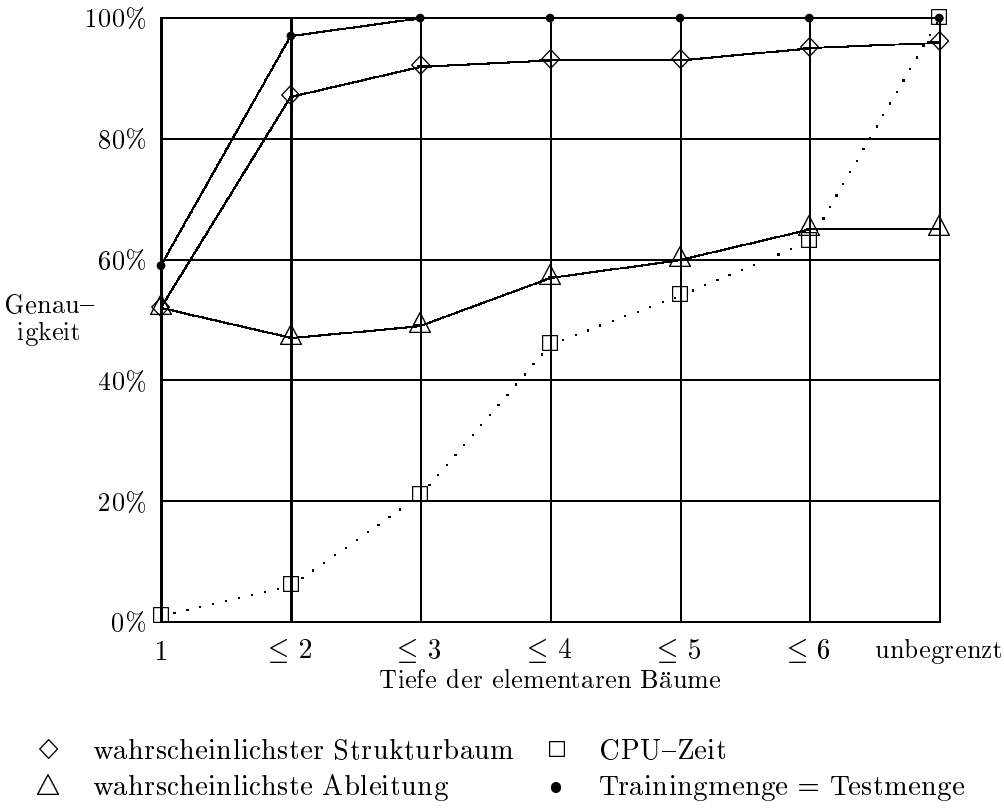


Abbildung 2.5: Ergebnisse des STSG–Modells DOP–1

Insgesamt sind die Ergebnisse — insbesondere bei Beachtung des strengen Genauigkeitsmaßes — als hervorragend zu beurteilen. Abbildung 2.6 zeigt die erzielten Ergebnisse aller fünf Modelle im Vergleich. Dabei wird deutlich, daß das Problem der unbekannten bzw. bezüglich der Kategorie unbekannten Worte schwer wiegt. Erst das Modell DOP–5, das mit Schätzungen für unbeobachtete elementare Bäume und einem externen Lexikon arbeitet, erreicht gleich gute Ergebnisse wie das einfache Modell DOP–1.

Wie ist das gute Abschneiden des Ansatzes zu erklären? Das Verfahren vereinigt die Vorteile der n–Gramme mit denen der PCFG. Auf der einen Seite ist das Verfahren (zumindest in den Varianten DOP–2 bis DOP–5) lexikalisch sensitiv, und auf der anderen Seite spielen strukturelle Abhängigkeiten natürlich eine herausragende Rolle, da das Verfahren als elementare Einheiten bereits Bäume einsetzt. Damit sind zwei wichtige Kriterien für eine erfolgreiche stochastische Modellierung erfüllt. Darüber hinausgehend werden sogar semantische Abhängigkeiten mit in die Überlegungen einbezogen (Bod, 1995, Kapitel 8).

Als Nachteil ist der enorme Bedarf an Speicherplatz und Rechenzeit zu nen-

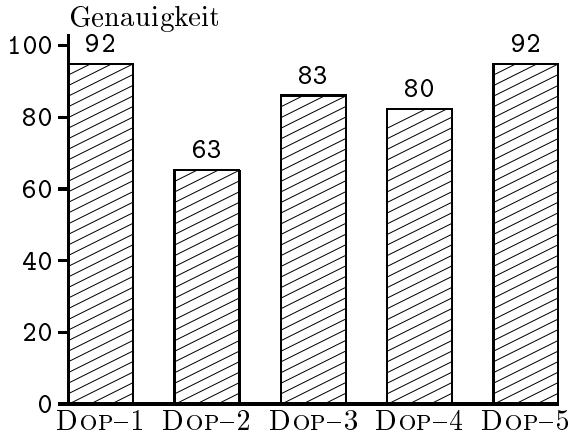


Abbildung 2.6: Vergleich der DOP–Modelle für elementare Bäume bis zu einer Tiefe von 3

nen, der allerdings durch das Verfahren der Simulation z. T. kompensiert wird. Anzumerken bleibt, daß Goodman (1996) schwerwiegende Vorwürfe gegen die Arbeit erhebt, die allerdings von Bod (1996) direkt beantwortet werden.

2.6 Stochastische Verkettungsgrammatiken

Verkettungsgrammatiken (engl. *link grammar*) stellen ein formales System für die Kodierung von Grammatiken dar. Sie fallen in die Klasse der kontextfreien Grammatiken (Sleator und Temperley, 1991), besitzen aber eine Reihe von Eigenschaften, die sie für die Grammatikerstellung und das Parsing von natürlichsprachlichen Äußerungen geeignet erscheinen lassen. Nach (Sleator und Temperly, 1993) zählen dazu die folgenden:

- Verkettungsgrammatiken sind lexikalisch, d. h. durch die Terminale oder Worte der Sprache definiert. Dadurch ergibt sich eine leichtere Wartbarkeit sowie eine generelle Eignung als Grundlage für ein stochastisches Modell.

- Syntaktisch oder semantisch zusammengehörende Worte werden durch Verkettungsgrammatiken direkt miteinander verbunden.

- Verkettungsgrammatiken erlauben eine orthogonale Beschreibung von natürlicher Sprache, d. h. Einflußfaktoren für bestimmte Unterscheidungen werden nur dort berücksichtigt, wo sie eine Rolle spielen. Beispielsweise spielt es für die Konstruktion einer Nominalphrase im Deutschen keine Rolle, ob die Nominalphrase als Subjekt, Objekt oder Präpositionalergänzung fungiert.[28]

[28] In diesem Beispiel wird von Kasus und anderen Attributen abgesehen.

2.6.1 Definition

Im folgenden werden zunächst Verkettungsgrammatiken informal eingeführt, dann formalisiert und schließlich die probabilistische Erweiterung vorgestellt.

Eine Verkettungsgrammatik besteht aus einer Beschreibung der Verkettungsmöglichkeiten (engl. *link requirement*) aller Worte der Sprache. Tabelle 2.1 zeigt ein kleines, leicht angepaßtes Beispiel aus (Sleator und Temperly, 1993).

Worte	Verkettungsmöglichkeiten
a the	`D+`
snake cat	`D- and (O- xor S+)`
Mary	`O- xor S+`
ran	`S-`
chased	`S- and O+`

Tabelle 2.1: Einfache Verkettungsgrammatik

Die Junktoren '`and`' und '`xor`' und die Klammern in den Formeln sind dabei wie bei aussagenlogischen Formeln üblich zu lesen. Eine einzelne Verkettungsmöglichkeit, z. B. '`D+`', besteht aus einem Namen (hier '`D`') und einer Ausrichtung (hier '`+`'). Eine Ausrichtung '`+`' bedeutet, daß das Wort mit einem anderen Wort auf der rechten Seite durch eine Verkettung mit dem entsprechenden Namen verbunden werden muß; die Ausrichtung '`-`' bedeutet das Analoge für Worte auf der linken Seite.[29]

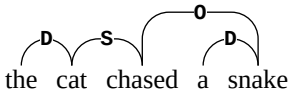


Abbildung 2.7: Beispielverkettung

Ein Satz gehört dann zur durch die Verkettungsgrammatik definierten Sprache, wenn sich ein Satz von Verkettungen wie in Abbildung 2.7 finden läßt, der die folgenden Bedingungen erfüllt:

- Planarität (engl. *planarity*): Die (oberhalb des Satzes gezeichneten) Verkettungen dürfen sich nicht kreuzen.[30]
- Verbundenheit (engl. *connectivity*): Die Verkettungen müssen alle Worte des Satzes (direkt oder indirekt) miteinander verbinden.

[29] Eine größere Flexibilität läßt sich erreichen, wenn man statt der Gleichheit der Namen der Verkettungsmöglichkeiten von zwei verketteten Worten eine allgemeinere 'Verträglichkeit' annimmt. Beispielsweise kann man die Namen durch Merkmalsstrukturen ersetzen und als Verträglichkeitsoperation die Unifizierbarkeit nutzen. Weitere Erweiterungen betreffen die Optionalität von Formelteilen und die Einführung einer Mehrfachverkettung (Sleator und Temperly, 1993).

[30] Für diese Eigenschaft wird auch der Begriff 'Projektivität' verwendet.

- Lineare Ordnung (engl. *ordering*): Je weiter links eine Verkettung in der Formel der Verkettungsmöglichkeiten für ein Wort steht, desto näher befindet sich das verbundene Wort im Satz.
- Ausschluß (engl. *exclusion*): Keine zwei Verkettungen dürfen dasselbe Paar Worte verbinden.
- Erfüllung (engl. *satisfaction*): Die Formeln der Verkettungsmöglichkeiten für alle Worte werden erfüllt.

Unter anderen können auch die Sätze 2.9 bis 2.11, nicht jedoch die Sätze 2.12 bis 2.15 durch die Beispielgrammatik erzeugt werden.[31]

(2.9) Mary chased the cat

(2.10) the snake chased a cat

(2.11) the cat ran

(2.12) * Mary chased

(2.13) * a snake the cat chased

(2.14) * cat the chased a snake

(2.15) * cat ran

Abbildung 2.8 zeigt als weiteres Beispiel Möglichkeiten für Verkettungssätze, die den Strukturbäumen der Abbildungen 2.1 und 2.3 entsprechen.

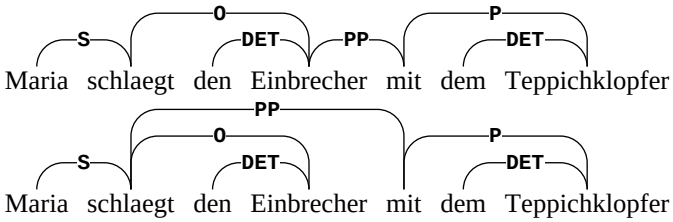


Abbildung 2.8: Zwei mögliche Verkettungssätze

Formal besteht eine Verkettungsgrammatik aus einem Tripel $G = \langle V_C, V_T, \Delta \rangle$, wobei den einzelnen Komponenten folgende Bedeutung zukommt:

V_C	endliche Menge an Verkettungsnamen (engl. *connector*)
V_T	endliche Menge an Terminalsymbolen
Δ	Funktion $\Delta : V_T \mapsto 2^{V_C^* \times V_C^*}$

[31] Hier soll der Stern vor einem Satz nicht dessen Ungrammatikalität in der englischen Sprache signalisieren, sondern anzeigen, daß dieser Satz nicht von der Grammatik erzeugt werden kann. Der Satz 2.13 ist beispielsweise in gewissen Kontexten akzeptabel (evtl. elliptisch für 'It was a snake the cat chased.').

Die Funktion Δ weist jedem Terminal bzw. Wort die Menge seiner möglichen Verkettungen, die sogenannten Disjunkte, zu. Diese Menge an Disjunkten stellt dabei die oben eingeführten Formeln in disjunktiver Normalform dar, wobei die Terme innerhalb eines Disjunkts durch den Junktor '`and`' und die verschiedenen Disjunkte durch den Junktor '`xor`' verknüpft sind; die Terme l_{ij} erhalten die Ausrichtung '`-`' und die Terme r_{ij} die Ausrichtung '`+`' (siehe Gleichung 2.33).

$$\Delta(\tilde{\omega}) = \left\{ \begin{array}{l} \langle l_{11} \ldots l_{1m_1}, r_{1n_1} \ldots r_{11} \rangle, \\ \langle l_{21} \ldots l_{2m_2}, r_{2n_2} \ldots r_{21} \rangle, \\ \vdots \\ \langle l_{o1} \ldots l_{om_o}, r_{on_o} \ldots r_{o1} \rangle \end{array} \right\} \tag{2.33}$$

Ein Satz $\Omega = \omega_1 \ldots \omega_p$ gehört zur durch eine Verkettungsgrammatik G definierten Sprache $L(G)$, wenn es zu jedem Wort ω_i ein Disjunkt $d_i \in \Delta(\omega_i)$ gibt, so daß die Disjunkte eine gültige Verkettung bilden können.[32]

$$d_i = \Delta(\omega_i)_{j_i} = \langle l_{j_i 1} \ldots l_{j_i m_{j_i}}, r_{j_i n_{j_i}} \ldots r_{j_i 1} \rangle \tag{2.34}$$

$$= \langle l^i_1 \ldots l^i_{m_i}, r^i_{n_i} \ldots r^i_1 \rangle \tag{2.35}$$

Für die Disjunkte $d_1, \ldots, d_p$ muß es einen geeigneten Satz von Verkettungen $L \subseteq \{\langle i, j, k, l \rangle \mid 1 \leq i, j \leq p, 1 \leq k \leq n_i, 1 \leq l \leq m_j\}$ mit $r^i_k = l^j_l$ für alle i, j, k und l geben, der die folgenden Bedingungen erfüllt (vgl. informale Kriterien oben):[33]

- Planarität: Es gibt keine Verkettungen $\langle i, j, \square, \square \rangle \in L$ und $\langle i', j', \square, \square \rangle \in L$ mit $i < i' < j < j'$.
- Verbundenheit: Für alle $1 \leq i_0, i_n \leq p$ gibt es eine Folge $1 \leq i_1, \ldots, i_{n-1} \leq p$, so daß für alle $1 \leq j \leq n$ die Worte an den Positionen i_{j-1} und i_j verkettet sind, also entweder $\langle i_{j-1}, i_j, \square, \square \rangle \in L$ oder $\langle i_j, i_{j-1}, \square, \square \rangle \in L$ gilt.
- Lineare Ordnung: Es gibt keine zwei Verkettungen $\langle i, j, k, l \rangle \in L$ und $\langle i', j', k', l' \rangle \in L$ mit $i = i'$ und $j < j'$ sowie $k < k'$. Analoges gilt für $i < i'$ und $j = j'$ sowie $l < l'$.

[32] Im folgenden wird angenommen, daß die Disjunkte in $\Delta(\omega_i)$ durchnumeriert sind, so daß über einen Index auf sie zugegriffen werden kann. Die Variablen i und j stehen für Indices von Worten, k und l bezeichnen die zur Verwendung kommenden Verkettungsmöglichkeiten der Worte.

[33] Mit $\square$ gekennzeichnete Variablen sind in dem jeweiligen Kontext irrelevant; gleichzeitiges mehrfaches Auftreten des Symbols $\square$ bedeutet keine Gleichheit zwischen den betroffenen Variablen.

- Ausschluß: Es gibt keine zwei Verkettungen $\langle i, j, k, l\rangle \in L$ und $\langle i, j, k', l'\rangle \in L$ mit $k \neq k'$ oder $l \neq l'$.
- Erfüllung: Für alle $1 \leq i \leq p$, alle $1 \leq k \leq n_i$ und alle $1 \leq l \leq m_i$ gilt $|\{\langle i, \Box, k, \Box\rangle \in L\}| = 1$ und $|\{\langle \Box, i, \Box, l\rangle \in L\}| = 1$.

2.6.2 Parsing–Algorithmen

Sleator und Temperly (1993) berichten von einem Algorithmus, der dynamisches Programmieren mit Memotechniken nutzt, um bei einer Komplexität von $O(c^2 \cdot d)$ bzw. $O(n^3)$ effizient ausführbar zu sein (c Anzahl Konnektoren, d Anzahl Disjunkte).

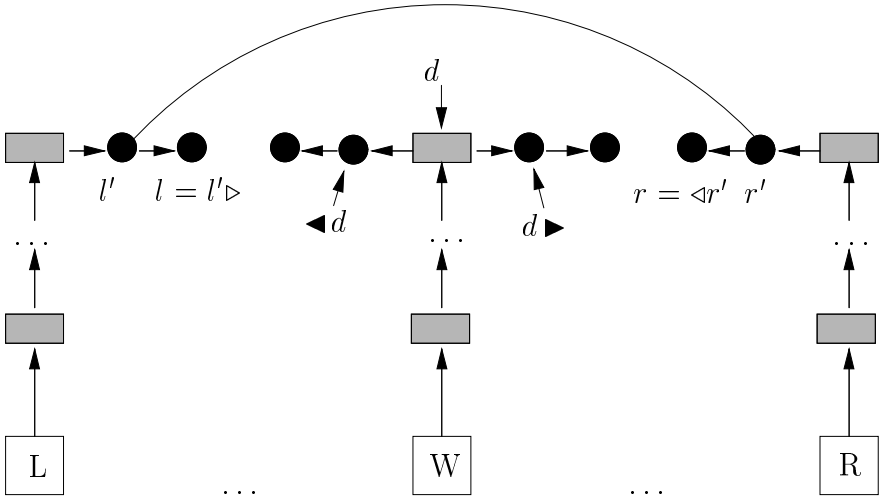


Abbildung 2.9: Parsing bei Verkettungsgrammatiken

Ein Disjunkt kann als Paar von linker und rechter Liste aufgefaßt werden; $\blacktriangleleft d$ bezeichne das erste Element der linken Liste des Disjunkts d, $d \blacktriangleright$ das erste Element der rechten Liste. Mit $\triangleleft l$ wird das nächste Element bzgl. des Verkettungsnamens l in der linken Liste bezeichnet, mit $r\triangleright$ das nächste Element bzgl. des Verkettungsnamens r in der rechten Liste (vgl. Abbildung 2.9). Stehen diese Elemente nicht zur Verfügung — z. B. wenn das Ende einer Liste erreicht ist — nehmen die Variablen den Wert Nil an.

Der Algorithmus speichert für je zwei Worte L und R und zwei zugehörige Konnektoren l und r in der Variablen $\gamma(L, R, l, r)$ die Anzahl der möglichen Vervollständigungen der Verkettungen 'innerhalb' der Konnektoren l und r.

$$\gamma(L, L+1, l, r) = 0 \quad \text{für } l \neq \textsc{Nil} \text{ oder } r \neq \textsc{Nil} \tag{2.36}$$

$$\gamma(L, L+1, l, r) = 1 \quad \text{für } l = \textsc{Nil} \text{ und } r = \textsc{Nil} \tag{2.37}$$

$$\gamma(L, R, l, r) = \sum_{L<W<R} \sum_{d \in \Delta(W)} \begin{pmatrix} {}^{\triangleleft}\gamma(L, R, l, r, W, d) & + \\ {}^{\triangleleft}\gamma^{\triangleright}(L, R, l, r, W, d) & + \\ \gamma^{\triangleright}(L, R, l, r, W, d) & \end{pmatrix} \tag{2.38}$$

Dabei bezeichnen die Hilfsvariablen ${}^{\triangleleft}\gamma$, ${}^{\triangleleft}\gamma^{\triangleright}$ und $\gamma^{\triangleright}$ die drei zu unterscheidenden Fälle, daß entweder der Verkettungsname l mit dem Verkettungsnamen $\blacktriangleleft d$ verkettet wird oder der Verkettungsname r mit $d \blacktriangleright$ oder sowohl l als auch r gleichzeitig.[34]

$$
\begin{aligned}
{}^{\triangleleft}\gamma(L, R, l, r, W, d) &= \mathbf{1}(l = \blacktriangleleft d) \cdot \\
&\quad \gamma(L, W, l\triangleright, \triangleleft \blacktriangleleft d) \cdot \gamma(W, R, d\blacktriangleright, r) && (2.39)\\
{}^{\triangleleft}\gamma^{\triangleright}(L, R, l, r, W, d) &= \mathbf{1}(l = \blacktriangleleft d) \cdot \mathbf{1}(r = d\blacktriangleright) \cdot \\
&\quad \gamma(L, W, l\triangleright, \triangleleft \blacktriangleleft d) \cdot \gamma(W, R, d\blacktriangleright \triangleright, \triangleleft r) && (2.40)\\
\gamma^{\triangleright}(L, R, l, r, W, d) &= \delta_{\text{NIL}}(l) \cdot \mathbf{1}(r = d\blacktriangleright) \cdot \\
&\quad \gamma(L, W, l, \blacktriangleleft d) \cdot \gamma(W, R, d\blacktriangleright \triangleright, \triangleleft r) && (2.41)
\end{aligned}
$$

Vielleicht noch ein oder zwei erklärende Worte? —B. Lahres

Sleator und Temperly (1993, Abschnitt 5) beschreiben einige Techniken zur Effizienzsteigerung des Verfahrens.

Die stochastischen Einheiten der probabilistischen Variante betreffen die Aktion der Verkettung (Lafferty, Sleator und Temperly, 1992). Abhängig von zwei Worten L und R sowie den verwendeten zu diesen Worten gehörenden Verkettungsnamen l und r erhält man als Parameter $P(W, d, O \mid L, R, l, r)$. Dabei ist W das Wort für die Verkettung, $d \in \Delta(W)$ das verwendete Disjunkt; O bestimmt, ob W nach links, nach rechts oder in beide Richtungen verbunden wird (vgl. Gleichung 2.38). Diese Parameter lassen sich dann wie folgt approximieren:

$$
\begin{aligned}
P(W, d, O \mid L, R, l, r) &= P(W \mid L, R, l, r) \cdot P(d \mid W, L, R, l, r) \cdot \\
&\quad P(O \mid d, W, L, R, l, r) && (2.42)\\
&\approx P(W \mid L, R, l, r) \cdot P(d \mid W, l, r) \cdot \\
&\quad P(O \mid d, l, r) && (2.43)
\end{aligned}
$$

Für einen bestimmten Verkettungssatz $\mathcal{L}$ eines Satzes Ω als Menge von Einzelverkettungen erhält man die in Gleichung 2.44 angegebene Wahrscheinlichkeit. Dabei stellt $P(W_0, d_0)$ die Wahrscheinlichkeit für ein initiales Wort und ein initiales Disjunkt (als weiterer stochastischer Parameter) dar.

$$
P(\Omega, \mathcal{L}) = P(W_0, d_0) \cdot \prod_{\langle W, d, O, L, R, l, r\rangle \in \mathcal{L}} P(W, d, O \mid L, R, l, r) \qquad (2.44)
$$

Analog zur Unabhängigkeitsannahme bei der Anwendung von kontextfreien Regeln werden auch hier die einzelnen Verkettungen als unabhängige stochastische Ereignisse betrachtet.

[34] Der Ausdruck $\mathbf{1}(\ldots)$ bezeichnet die Boole'sche Indikatorfunktion. Der δ–Ausdruck wird in der Quelle nur als 'übliche' δ–Funktion bezeichnet, wahrscheinlich ist $\mathbf{1}(l = \text{NIL})$ gemeint.

2.6.3 Schätzung von Parametern

In (Lafferty, Sleator und Temperly, 1992) geben die Autoren Gleichungen für das Training der Parameter aus einem Korpus an. Diese Gleichungen gehen auf die des Inside–Outside–Algorithmus zurück (vgl. Abschnitt 2.4.3).

2.6.4 Beurteilung

(Stochastische) Verkettungsgrammatiken verfolgen einen lexikalischen Ansatz, d. h., das grammatische Wissen wird direkt an den lexikalischen Einheiten der Sprache notiert. Die stochastische Variante berücksichtigt sowohl Wort– als auch Strukturinformationen. Allerdings werden bei der stochastischen Bewertung der Verkettungsoperation nur die jeweils direkt betroffenen Worte sowie die Art der Unterordnung zwischen diesen beachtet.[35] Ob eine solche im Vergleich etwa zu den STSG reduzierte Sensitivität für die Verarbeitung relevanter Konfigurationen ausreicht, bleibt offen.

Grinberg, Lafferty und Sleator (1995) diskutieren eine Variante der Verkettungsgrammatiken für robustes Parsing mit einem zweistufigen Verfahren. Erst wenn keine korrekte Lösung gefunden wird, werden in einem zweiten Durchgang Varianten betrachtet, die bzgl. der ursprünglichen Grammatik Defizite aufweisen. Mellish (1989) schlägt ein solches zweistufiges Vorgehen für Chart–Parser vor. Die Verwendung der stochastischen Variante führt bei den Verkettungsgrammatiken also nicht automatisch zu einem robusten Systemverhalten.

2.7 Stochastische lexikalische Kopfgrammatik

Collins (1996) beschreibt einen neuen statistischen Parser, der auf den Wahrscheinlichkeiten für Abhängigkeiten zwischen lexikalischen Köpfen basiert und hier 'stochastische lexikalische Kopfgrammatik' (engl. *stochastic lexical head grammar,* kurz *SLHG*) genannt wird.

Es findet ein normaler Bottom–Up–Chart–Parser Verwendung, der eine Folge von Wort–Kategoriesymbol–Paaren als Eingabe in eine kompakte Repräsentation von Strukturbäumen als Ausgabe überführt. Die Kombinationsoperation entspricht also genau derjenigen der PCFG (vgl. Abschnitt 2.4); die Verfahren unterscheiden sich lediglich in der Berechnung der Wahrscheinlichkeiten.

2.7.1 Vorverarbeitung

Die Worte des Satzes werden vor dem Parsing von einem stochastischen Kategoriebestimmungsverfahren (engl. *tagger*) in einer Vorverarbeitung mit Kate-

[35] Tatsächlich können die Parameter $P(W, d, O \mid L, R, l, r)$ maximal drei Worte und zwei Unterordnungen in Beziehung setzen, wenn das Wort W mit beiden Randworten L und R verkettet wird.

goriesymbolen (engl. *part–of–speech symbol,* kurz *POS*) versehen. Weiterverarbeitet wird in dem Grundmodell nur die wahrscheinlichste Sequenz von Wort–Kategoriesymbol–Paaren.

Weiterhin werden nicht–rekursive Nominalgruppen (von Collins (1996) als 'baseNP' bezeichnet) zusammengefaßt und im folgenden durch ihren lexikalischen Kopf vertreten. Im Beipiel 2.16 sind die zusammengefaßten Nominalgruppen geklammert, die lexikalischen Köpfe hervorgehoben sowie die Kategoriesymbole als Indices notiert.

(2.16) $(\textit{Maria})_{\mathrm{NP}}$ $\text{schlägt}_{\mathrm{V}}$ (den *Einbrecher*)$_{\mathrm{NP}}$ mit_{P} (dem *Teppichklopfer*)$_{\mathrm{NP}}$.

Da sowohl die Etikettierung mit Kategoriesymbolen als auch die Gruppierung zu Nominalgruppen durch probabilistische Komponenten geschieht, existieren (geschätzte) Wahrscheinlichkeiten für die im folgenden verwendeten kleinsten Einheiten.

2.7.2 Definition

Die folgende Definition entfernt sich von der Beschreibung aus Collins (1996), um eine bessere Vergleichbarkeit mit den anderen Ansätzen zu ermöglichen. Eine SLHG besteht aus einem Tupel $G = \langle V_N, V_T, R, P\rangle$, wobei den einzelnen Komponenten folgende Bedeutung zukommt:

V_N	endliche Menge an Nichtterminalsymbolen
V_T	endliche Menge an Terminalsymbolen
R	endliche Menge an Unterordnungsmöglichkeiten der Form $\omega \overset{K}{\underset{I,\ J}{\rightarrow}} \omega'$ mit $\omega, \omega' \in V_T$ und $I, J, K \in V_N$ oder der Form $\omega \rightarrow \top$ mit $\omega \in V_T$.
P	Funktion $P : R \mapsto [0,1]$ mit

$$(\forall \omega \in V_T)\, P(\omega \rightarrow \top) + \sum_{\substack{\omega' \in V_T \\ I,J,K \in V_N}} P(\omega \overset{K}{\underset{I,\ J}{\longrightarrow}} \omega') = 1 \tag{2.45}$$

Die Unterordnungsmöglichkeiten lassen sich am einfachsten durch Rückgriff auf Abbildung 2.10 in Bezug auf Abbildung 2.3 veranschaulichen. Jeder Vater–Sohn–Beziehung im Strukturbaum — außer der des lexikalischen Kopfes des Satzes — entspricht genau eine Unterordnungsbeziehung.

Dabei entspricht ein Pfeil mit der Beschriftung 'I K J' von einem Wort ω zu einem anderen Wort ω' genau der Unterordnungsmöglichkeit $\omega \overset{K}{\underset{I,\ J}{\rightarrow}} \omega'$ aus einer SLHG. Das Tripel von Nichtterminalsymbolen charakterisiert die Art der Unterordnungsbeziehung zwischen den Worten.

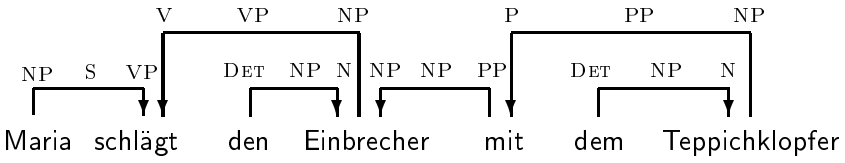
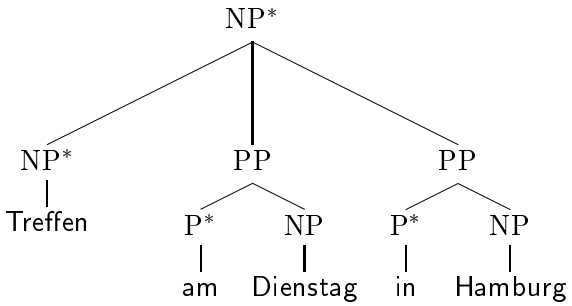


Abbildung 2.10: Abhängigkeitsbeziehungen bei SLHG

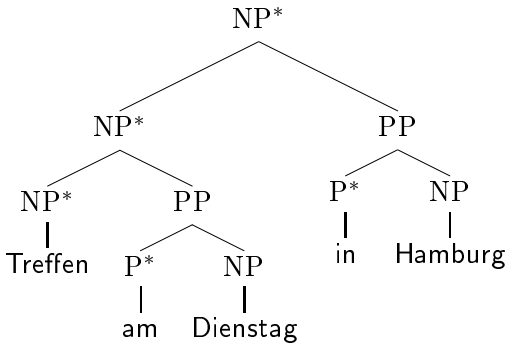
Gibt es dieses Wort 'eineindeutig' etwa?
—T. Naguschewski

Es sollte beachtet werden, daß es nicht möglich ist, zwischen diesen Unterordnungsmöglichkeiten und Strukturbäumen eineindeutig abzubilden. Für die Beispielsphrase 2.17 beispielsweise weisen die beiden Strukturbäume aus Abbildung 2.11 identische Unterordnungen auf.

(2.17) ein Treffen am Dienstag in Hamburg



(a) flache Strukturierung



(b) tiefe Strukturierung

Abbildung 2.11: Mögliche Strukturbäume

Eine Ableitung $\Xi = \xi_1, \ldots, \xi_n$ eines Satzes $\Omega = \omega_1 \ldots \omega_n$ legt für jedes Wort ω_i des Satzes fest, von welchem anderen Wort es in welcher Art abhängt bzw. welches Wort es wie modifiziert. Dabei gilt $\xi_i = \langle I_i, J_i, K_i, m_i \rangle$ mit $I_i, J_i, K_i \in V_N$ und $1 \leq m_i \leq n$, wenn das Wort ω_i das Wort ω_{m_i} mit der Relation, die durch das Tripel 'I K J' charakterisiert wird, modifiziert.

Das Wort ω_r, das den lexikalischen Kopf des Satzes bildet, modifiziert kein anderes Wort; aus formalen Gründen soll dann $\xi_r = \langle \top, \top, \top, 0 \rangle$ gelten. Die folgenden Bedingungen an eine Ableitung müssen erfüllt sein:

- Planarität: Es gibt keine $1 \leq i < j \leq n, i \neq j \neq r$ mit $i < j < m_i < m_j$, $m_i < m_j < i < j$ oder $m_j < i < j < m_i$.
- Projektion: Für alle $\xi_i = \langle I_i, J_i, K_i, m_i \rangle$ mit $i \neq r$ gilt, daß entweder ω_i die Kategorie I_i besitzt oder ein $\xi_j = \langle I_j, J_j, K_j, i \rangle$ mit $j \neq i$ existiert, so daß $J_j = I_i$ gilt.

Die Gleichung 2.45 bewirkt, daß der gesamten von einer SLHG G erzeugten Sprache $L(G)$ eine Wahrscheinlichkeit von 1 zugewiesen wird.[36]

2.7.3 Schätzung von Parametern

Die zu schätzenden Parameter sind die Wahrscheinlichkeiten $P(\omega \xrightarrow[I,\ J]{K} \omega' \,|\, \Omega)$. Collins (1996) definiert zunächst den folgenden Maximum–Likelihood–Schätzer für die Wahrscheinlichkeit, daß das Wort ω das Wort ω' mit der Beziehung 'I, J, K' modifiziert, vorausgesetzt beide Worte treten in einem Satz auf.

$$\hat{F}(I, J, K \,|\, \omega, \omega') = \frac{\text{Anzahl Vorkommen } \omega \text{ modifiziert } \omega' \text{ mit 'I J K'}}{\text{Anzahl Vorkommen } \omega \text{ und } \omega' \text{ in einem Satz}} \tag{2.46}$$

Die gesuchten Wahrscheinlichkeiten werden dann durch den folgenden Ausdruck approximiert. Die folgenden Parameter können wegen der Abhängigkeit vom Satz Ω erst während des Parsings berechnet werden.

$$P(\omega \xrightarrow[I,\ J]{K} \omega') \approx \frac{\hat{F}(I, J, K \,|\, \omega, \omega')}{\sum\limits_{\omega'', I', J', K'} \hat{F}(I', J', K' \,|\, \omega, \omega'')} \tag{2.47}$$

2.7.4 Beurteilung

Collins (1996) erzielt mit seinem Parsingverfahren sehr gute Ergebnisse bei Versuchen mit dem Korpus des Wall Street Journals der Penn Treebank (Marcus, Santorini und Marcinkiewicz, 1993). Die Methode der stochastischen lexikalischen Kopfgrammatik ist sowohl lexikalisch, da direkt Abhängigkeiten zwischen Wortformen verwendet werden, als auch syntaktisch strukturell sensitiv, da Dependenzinformationen berücksichtigt werden. Allerdings werden nur Abhängigkeitsbeziehungen zwischen je zwei Worten berücksichtigt.

In Kapitel 4 wird sich zeigen, daß dieses Verfahren gewisse Ähnlichkeiten mit der dort vorgeschlagenen Klasse von stochastischen Modellen aufweist.

[36] Lafferty, Sleator und Temperly (1992) nennen diese Eigenschaft, die auch den SLG und den PCFG zukommt, generativ (engl. *generative*).

2.8 Weitere stochastische Modelle

Neben den vorgestellten stochastischen Verfahren existieren natürlich etliche weitere. In diesem Abschnitt sollen noch einige kurz vorgestellt werden, ohne daß detailliert auf sie eingegangen wird.

So sind für die Gruppe der Baumadjunktionsgrammatiken (engl. *tree adjoining grammar,* kurz *TAG*) (Vijay-Shanker, 1987; Joshi, Levy und Takahashi, 1975; Schabes und Waters, 1995) stochastische Erweiterungen vorgeschlagen worden (Schabes, 1992; Resnik, 1992), die jedoch nicht evaluiert worden sind. Eisner (1996) stellt gleich drei stochastische Modelle für eine Dependenzmodellierung vor, ohne jedoch in der erzwungenen Knappheit eine formale und verständliche Definition zu geben.

2.8.1 Ableitungsbasiertes probabilistisches Parsing

Das ableitungsbasierte Parsing (engl. *history–based grammar,* kurz *HBG*) (Black et al., 1992) ist eine Erweiterung der kontextfreien Analyse, bei der die Rolle des Kontextes einer Regelanwendung betont wird. In jedem Schritt einer Linksableitung einer kontextfreien Grammatik (vgl. Abschnitt 2.4) wird nun die gesamte bisherige Ableitung als zu verwendende Information betrachtet. Wenn $Xi = \xi_1 \dots \xi_m$ eine Linksableitung darstellt, dann ergibt sich die Wahrscheinlichkeit wie in Gleichung 2.48.

$$P(\Xi) = \prod_{i=1}^{m} P(\xi_i \,|\, \xi_1 \dots \xi_{i-1}) \tag{2.48}$$

$$\approx \prod_{i=1}^{m} P(\xi_i \,|\, E[\xi_1 \dots \xi_{i-1}]) \tag{2.49}$$

Da die Abschätzung der Parameter $P(\xi_i \,|\, \xi_1 \dots \xi_{i-1})$ im allgemeinen aufgrund des Datenmangels nicht möglich ist,[37] werden Äquivalenzklassen $E[\xi_1 \dots \xi_{i-1}]$ über den Ableitungen gebildet (Gleichung 2.49). Alle Parameter einer Äquivalenzklasse erhalten dann die gleiche Wahrscheinlichkeit. Im Falle der probabilistischen kontextfreien Grammatiken gibt es nur eine einzige Äquivalenzklasse, d. h., die bisherige Ableitung spielt keine Rolle für die Parameter, so daß sich die Berechnung wie in Gleichung 2.50 vereinfacht (vgl. Gleichung 2.12).

$$P(\Xi) = \prod_{i=1}^{m} P(\xi_i) \tag{2.50}$$

In einem Vergleich zwischen einer PCFG und einem HBG–Modell, das Äquivalenzklassen für die Parameterabschätzung aufgrund von syntaktischen und

[37] Eine Ableitung spezifiziert neben der Struktur auch die Abfolge der Worte.

semantischen Kategorien, Kopfkonstituenten sowie einem Index im übergeordneten Knoten bildet, erzielt das HBG–Modell eine Übereinstimmung der Konstituentenstruktur ohne Beschriftungen mit der handannotierten Struktur von 75% gegenüber 60% für das PCFG–Modell.

Die HBG stellen eine Verallgemeinerung der PCFG, die sie als Spezialfall einschließen, dar, indem sie die Parameter für eine Regelanwendung sensitiv bzgl. des Kontextes machen. Die besseren Ergebnisse für das HBG–Modell sind insofern zu erwarten. Allerdings sind die berücksichtigten Abhängigkeiten wenig plausibel. Zwar erlaubt das Modell, strukturelle und lexikalische Abhängigkeiten zu verwenden, allerdings immer nur bzgl. in der Linksableitung früher getroffenen Entscheidungen, also immer 'nach links'. Diese Eigenschaft stellt eine nicht linguistisch motivierte Einschränkung dar. Obwohl der verwendete Prozeß bei den STSG ein anderer als bei den HBG ist,[38] kann man die HBG bzgl. der verwendeten Information zwischen den PCFG und den STSG einordnen. Die HBG sind grundsätzlich für eine inkrementelle Arbeitsweise geeignet, da eine Regelanwendung nur von den in der Linksableitung früher auftauchenden Anwendungen abhängig ist.

2.8.2 Parsing als Mustererkennung

Magerman (1994; 1995b) beschreibt ein Analyseverfahren für natürlich–sprachliche Äußerungen als geometrische Mustererkennung. Dabei werden statistische Entscheidungsbäume (Magerman, 1994) verwendet, um für die Wortknoten eine Kategorie, für die inneren Knoten ein Nichtterminalsymbol und für alle Knoten eine Ausrichtung auszuwählen.

Die Abbildung 2.12 aus Magerman (1995b) illustriert den Analysevorgang. Von links nach rechts werden den Worten die Kategorie und die Ausrichtung durch statistische Entscheidungsbäume zugewiesen. Wenn eine Konstituente vollständig ist (Ausrichtung '↖'), wird ein neuer Knoten erzeugt, das Kopfwort deterministisch bestimmt sowie dessen Ausrichtung und das Nichtterminal durch Entscheidungsbäume ausgewählt. Durch die Zuweisung der Attribute Kategorie bzw. Nichtterminal und Ausrichtung wird der Strukturbaum vollständig bestimmt.

Die Wahrscheinlichkeit für eine Auswahl eines Entscheidungsbaumes ergibt sich als Produkt der Einzelentscheidungen, die Wahrscheinlichkeit eines Strukturbaumes als Produkt der Auswahlentscheidungen (Magerman, 1994). Die möglichen Fragen beziehen sich auf die Knoten in einem Fenster der Breite 5 und auf Töchterknoten.

Magerman (1995b) berichtet von sehr guten Ergebnissen bei Experimenten, sowohl im Vergleich mit einer handkodierten Grammatik als auch im Vergleich

[38] Die Informationen über strukturelle und lexikalische Abhängigkeiten finden sich bei den STSG in den elementaren Einheiten, den elementaren Bäumen, während bei den HBG diese Information erst bei der Regelanwendung verwendet wird.

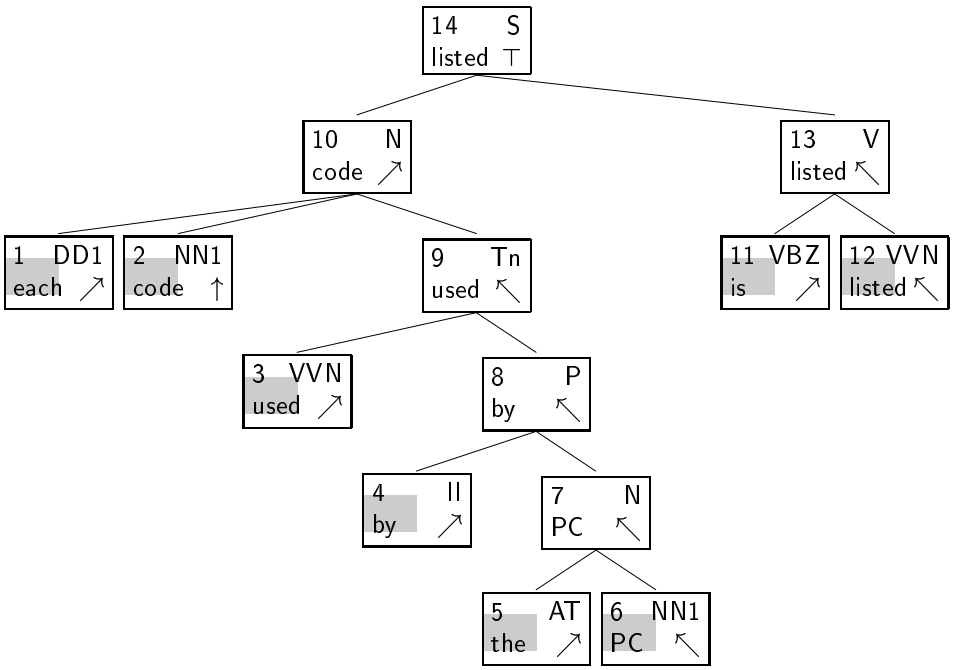


Abbildung 2.12: Struktur anhand von zugewiesenen Attributen; Angaben (im Uhrzeigersinn) jeweils zum Schritt des Knotenaufbaus, Kategorie bzw. Nichtterminal, Ausrichtung und Kopfwort

mit anderen stochastischen Verfahren. Das Modell ist in der Lage, sowohl lexikalische als auch strukturelle Informationen aus den benachbarten Knoten für eine Entscheidung heranzuziehen. Die Beschränkung auf Informationen an benachbarten Knoten gemäß der für den Entscheidungsbaum erlaubten Fragen sorgt für eine Einordnung des Verfahren zwischen den STSG auf der einen Seite und den PCFG bzw. den HBG auf der anderen Seite. Ebenso wie die HBG ist das Parsing als Mustererkennung prinzipell auf ein inkrementelles Vorgehen erweiterbar, da Konstituenten sofort aufgebaut werden, sobald alle untergeordneten Knoten vorhanden sind.

2.9 Zusammenfassung

In diesem Kapitel wird eine Anzahl von stochastischen Modellen für die Verarbeitung von natürlicher Sprache vorgestellt. Ein wichtiges Unterscheidungskriterium stellt der Umfang der bei Etablierung einer (Teil–)Struktur berücksichtigten Information dar. Das Verfahren der n–Gramme und die probabilistischen kontextfreien Grammatiken beschränken sich dabei auf lexikalische bzw. strukturelle Abhängigkeiten. Die stochastischen Baumsubstitutionsgrammatiken dagegen berücksichtigen — zumindest in der Theorie — die vollständige zur Verfügung stehende und an einem Strukturbaum annotierbare Information und kommen damit der Forderung nach Datenorientiertheit nach. Die Verkettungs-

grammatiken und die stochastische lexikalische Kopfgrammatik verwenden als einzige keine Phrasenkonstituentenstruktur, sondern eine Dependenzmodellierung. Beide sind lexikalisch sensitiv und für beide ist die strukturelle Sensitivität auf jeweils eine Unterordnungsbeziehung reduziert. Im Kapitel 4 wird eine stochastische Modellierung für die in Kapitel 3 eingeführte Constraint–Dependenz–Grammatik vorgestellt, die zunächst den gleichen Beschränkungen wie die hier vorgestellten stochastischen Dependenzmodelle unterliegt.

3. Eliminatives Parsing

Der Begriff des 'eliminativen Parsings' wird als Gegensatz zum konstruktiven Charakter anderer Verfahren verwendet. Viele Parsing–Ansätze führen im Verlauf einer Analyse einer natürlichsprachlichen Eingabe neue Objekte ein, z. B. Chart–Einträge, vollständige Phrasenkonstituenten, HPSG–Merkmalsstrukturen u. a. m., und konstruieren aus diesen Teilen schließlich die gewünschte Zielstruktur. Im Gegensatz dazu gehen eliminative Verfahren von einer maximal unterspezifizierten Repräsentation der Lösungsmenge aus.[1] Aus dieser Repräsentation werden dann im Verlauf der Analyse unplausible Varianten herausgestrichen oder eliminiert, bis schließlich eine oder wenige Strukturen übrigbleiben. Dieses Vorgehen bietet prinzipiell einige vorteilhafte Eigenschaften:

- Auch für Eingaben, denen normalerweise aufgrund mangelnder Abdekkung der Grammatik, Performanzerscheinungen o. ä. keine Struktur zugewiesen werden kann, wird eine Lösung gefunden. Diese wird vermutlich zumindest in Teilen plausibel sein, so daß das Verfahren einen gewissen Grad an Robustheit zeigt.
- Die Anzahl der noch nicht eliminierten Varianten in der Lösungsmenge dient als einfaches Maß für die noch zu leistende Disambiguierung, welches zur Steuerung des weiteren Verfahrens eingesetzt werden kann.[2]

Als Nachteil erweist sich, daß nicht jede Art der (syntaktischen) Strukturrepräsentation für eliminative Verfahren geeignet ist. Zum Beispiel gibt es ohne Berücksichtigung der Nichtterminalsymbole bei einer Satzlänge von n bereits exponentiell viele binäre Phrasenstrukturbäume; eine unterspezifizierte Lösungsmenge einer solchen Größe läßt sich nicht mehr erstellen und verarbeiten. Für die in dieser Arbeit gewählte Strukturrepräsentation der Dependenz gilt diese Einschränkung jedoch nicht.

[1] Die einfachste und am häufigsten verwendete Form einer solchen unterspezifizierten Lösungsmenge stellt eine Aufzählung aller lokalen Lösungsmöglichkeiten dar. Diese spezielle Repräsentation wird auch in dieser Arbeit verwendet.

[2] Siehe (Menzel, 1994) für eine ausführlichere Darstellung.

Maruyama (1990a; 1990b) formuliert das eliminative Parsing–Problem erstmalig als Constraint–Satisfaction–Problem (engl. *constraint satisfaction problem,* kurz *CSP*) und nutzt zur Lösung dessen bekannte Algorithmen. Die Klasse der Constraint–Satisfaction–Probleme wird im folgenden Abschnitt formal eingeführt. Daraufhin wird gezeigt, wie eliminatives Parsing in diese Klasse paßt, und schließlich werden Erweiterungen diskutiert, die für die Neuerungen in Kapitel 4 notwendig sind.

3.1 Constraint–Satisfaction–Probleme

CSP gehen auf (Waltz, 1975) zurück und können bei Beschränkung auf binäre CSP[3] als Tripel $\langle V, D, C\rangle$ mit der folgenden Bedeutung für die Elemente definiert werden:

V	endliche Menge an Variable $V = \{v_1, \ldots, v_n\}$
D	endliche Menge $D = \{D_1, \ldots, D_n\}$ der möglichen Werte der Variablen $v_1, \ldots, v_n$, d. h., eine Variable v_i kann Werte aus der Menge $D_i = \{d_i^1, \ldots, d_i^{m_i}\}$ annehmen
C	endliche Menge an Constraints $\langle v_i, a_i, v_j, a_j\rangle \in C$ mit $v_i, v_j \in V, a_i \in D_i, a_j \in D_j$ über den Variablen; eine Belegung der Variablen v_i mit dem Wert a_i ist unvereinbar mit einem Wert a_j für die Variable v_j

Eine Lösung eines CSP besteht aus einer Belegung der Variablen $v_1 = a_1, \ldots, v_n = a_n, a_i \in D_i$, so daß kein Constraint verletzt wird.

$$(\forall 1 \leq i, j \leq n)\langle v_i, a_i, v_j, a_j\rangle \notin C \tag{3.1}$$

Das allgemeine CSP fällt in die Klasse der NP–vollständigen Probleme[4]; ein allgemeines effizientes Lösungsverfahren kann es deshalb nicht geben. Allerdings sind eine Reihe von Algorithmen entwickelt worden, die spezielle oder allgemeine CSP in akzeptabler Zeit lösen können. Meseguer (1989), Kumar (1992) sowie Nadel (1988) geben Überblicke über diverse Techniken. Zu unterscheiden sind retrospektive Verfahren wie Backtracking, Backmarking, Backjumping u. a. m. und prospektive Verfahren[5] wie Konsistenzpropagation u. a. m. In konkreten Anwendungen werden meistens beide Arten gemeinsam — etwa bei der Vorwärtsprüfung (engl. *forward checking*) — unter Berücksichtigung von problemspezifischen Heuristiken wie Ordnen der Variablen benutzt. In den Abschnitten 3.3.2 und 3.4.3 wird weiter auf spezielle Algorithmen eingegangen werden.

[3] Binäre CSP besitzen die Stelligkeit 2, d. h. seine Constraints setzen maximal zwei Variablen in Beziehung. CSP anderer Stelligkeit werden hier nicht betrachtet.

[4] Siehe z. B. (Schröder, 1995).

[5] Algorithmen zu den prospektiven Techniken finden sich z. B. in (Mackworth, 1977), (Freuder, 1982), (Nudel, 1983), (Freuder, 1985), (Mackworth und Freuder, 1985), (Mohr und Henderson, 1986) und (Mackworth, 1992).

3.2 Constraint–Dependenz–Grammatik

Im folgenden wird das Basisverfahren für sogenannte Constraint–Dependenz–Grammatiken vorgestellt. Erweiterungen werden in den Abschnitten 3.3 und 3.4 diskutiert.

3.2.1 Definition

Von Maruyama (1990a; 1990b) wird eine Constraint–Dependenz–Grammatik (engl. *constraint dependency grammar,* kurz *CDG*), die einer natürlichsprachlichen Äußerung Abhängigkeitsstrukturen zuweist, als Quadrupel $\langle V_T, L, R, C\rangle$ definiert, wobei den einzelnen Elementen die folgende Bedeutung zukommt:[6]

V_T	endliche Menge an Terminalsymbolen[7]
L	endliche Menge an Beschriftungen (engl. *label*)
R	endliche Menge an Rollen (engl. *role*),
C	endliche Menge der unären und binären Beschränkungen oder Constraints

Die Terminalsymbole stellen die Worte der Sprache — gegebenenfalls angereichert um lexikalische Information — dar. Unterschiedliche Rollen stehen für unterschiedliche Repräsentationsebenen, z. B. syntaktische, semantische oder referentielle Dependenz. Für jeweils eine solche Ebene geben die Beschriftungen die Funktion oder Art der Unterordnung an, z. B. Subjekt oder direktes Objekt für die syntaktische Repräsentation.

Das Parsing–Problem für eine CDG $G = \langle V_T, L, R, C\rangle$ und einen Satz $\omega = \omega_1 \ldots \omega_n$ mit $\omega_i \in V_T$ kann man nun wie folgt als CSP–Problem formulieren. Pro Wort ω_i und Rolle $r_j \in R$ wird eine Variable $v_{\omega_i}^{r_j}$[8] eingeführt, die Werte $\langle l, m\rangle \in L \times \{nil, 1, \ldots, n\}$ annehmen kann. Eine Belegung der Variablen, also eine mögliche Lösung des CSP, entspricht dann einer Dependenzstruktur des Satzes.

Ein Beispiel soll das Gesagte veranschaulichen. Dafür seien die Rollen $r_1 =$ SYN(TAX), $r_2 =$ SEM(ANTIK) $\in R$ sowie die Beschriftungen $L = \{$*nil, subj, amod, pn, pmod, agent, theme*$\}$ definiert.

(3.1) sagen wir lieber am mittwoch (n001k.013.2, modifiziert)[9]

Die Belegung der zehn Variablen für den Beispielsatz 3.1 in Abbildung 3.1 kann als Paar von Dependenzbäumen wie in Abbildung 3.2 visualisiert werden.

[6] In (Maruyama, Watanabe und Ogino, 1990) wird eine Anwendung der CDG vorgestellt.
[7] Maruyama (1990a; 1990b) verwendet hier das Symbol Σ.
[8] Statt $v_{\omega_i}^{r_j}$ wird auch abkürzend v_i^j geschrieben.
[9] Die verwendeten Sätze stammen aus der VERBMOBIL–Domäne der Terminabsprachen. Siehe Anhang B.

$$
\begin{array}{ll}
v_1^{\text{SEM}} = \langle nil, 0\rangle & v_1^{\text{SYN}} = \langle nil, 0\rangle \\
v_2^{\text{SEM}} = \langle agent, 1\rangle & v_2^{\text{SYN}} = \langle subj, 1\rangle \\
v_3^{\text{SEM}} = \langle nil, 0\rangle & v_3^{\text{SYN}} = \langle amod, 1\rangle \\
v_4^{\text{SEM}} = \langle nil, 0\rangle & v_4^{\text{SYN}} = \langle pn, 5\rangle \\
v_5^{\text{SEM}} = \langle theme, 1\rangle & v_5^{\text{SYN}} = \langle pmod, 1\rangle
\end{array}
$$

Abbildung 3.1: Variablenbelegung für CDG

Nimmt man einige spezielle Constraints hinzu, die für die Wohlgeformtheit der Dependenzbäume[10] verantwortlich sind, so repräsentiert eine Lösung des CSP eine Analyse der natürlichsprachlichen Äußerung.

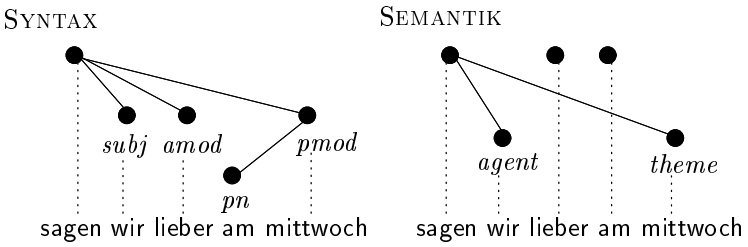


Abbildung 3.2: Variablenbelegung für CDG als Dependenzbäume

Wie sehen nun die Constraints der Grammatik aus?[11] Unäre Constraints betreffen genau eine Variable v_i^j, die bekanntlich für ein Wort und eine Rolle steht, und einen möglichen Wert $\langle l, m\rangle$, der eine Beschriftung l und das übergeordnete Wort ω_m repräsentiert; eben diese Informationen stehen auch bei der Formulierung der Constraints zur Verfügung. Die folgenden beiden Constraints — das erste unär, das zweite binär — zeigen exemplarisch, daß diese als logische Formeln aufgefaßt werden, die bei gegebener Variablenbindung zu einem Wahrheitswert evaluiert werden können.[12]

SubjNum: X.role=Syn ∧ X.label=subj → X↓num=X↑num
'Das Subjekt stimmt mit dem Verb im Numerus überein.'

OneSubj: X.role=Syn ∧ X.label=subj ∧ Y.role=Syn → Y.label≠subj
'Es gibt ein eindeutiges Subjekt.'

Variablen werden in Constraints mit Großbuchstaben (X und Y) bezeichnet; die Rolle und die Beschriftungen werden über passende Selektoren (role und

[10] Kriterien, die zur Wohlgeformtheit herangezogen werden können, betreffen z. B. Zyklenfreiheit, Eindeutigkeit der Wurzel, Überschneidungsfreiheit der Kanten u. a. m.

[11] Die Constraints einer CDG sind nicht mit denjenigen anderer sogenannter constraint–basierter Grammatiken (z. B. GPSG, HPSG u. a. m.) zu verwechseln. Siehe (Hess, 1996, Abschnitt 2.3) für eine knappe Gegenüberstellung der beiden Arten von Constraints.

[12] Die Schreibung als (aussagen–)logische Formel unterstützt die Sichtweise auf Constraints als Prädikate über Strukturkonfigurationen.

label) abgefragt; Informationen zu dem untergeordneten und dem übergeordneten Wort sind über spezielle Auszeichnungen (↓ und ↑) verfügbar. Unäre Constraints enthalten genau eine Variable, binäre dagegen zwei. Anhang A.2 enthält eine vollständige Beschreibung der Syntax und Semantik von Constraints.

3.2.2 Ausdruckskraft von Constraints

Im folgenden werden einige durch den Formalismus der CDG gegebenen Eigenschaften für Constraints explizit beschrieben, da diese oftmals für Schwierigkeiten bei der Grammatikerstellung sorgen.

Wann immer Constraints angewendet werden, zielen sie auf die Existenz von *negativer Evidenz* ab, d. h., sie beurteilen lokale Konfigurationen danach, ob sie der Grammatik genügen oder nicht. Im folgenden Beispiel wird erzwungen, daß ein durch eine Präposition modifiziertes Nomen selbst als Präpositionalgruppe modifiziert wird; oder anders ausgedrückt, für alle anderen Konfigurationen wird die Ungrammatikalität festgestellt.

$$\text{Prep} : \text{X.label} = \text{pn} \wedge \text{X}\uparrow\text{id} = \text{Y}\downarrow\text{id} \quad \rightarrow \quad \text{Y.label} = \text{pmod}$$

Ein Grammatikschreiber muß das gesamte grammatische System auf solche negative Evidenzen zurückführen. Dieses Vorgehen unterscheidet sich wesentlich von dem anderer Formalismen wie z. B. Phrasenstrukturgrammatiken, bei denen erst die Existenz einer positiven Evidenz eine bestimmte Struktur ermöglicht.

Es ist unmöglich, eine Anbindung eines Nomen als Präpositionalgruppe zu unterbinden, wenn sich keine passende Präposition findet, da es Constraints nicht möglich ist, die *Nicht–Existenz* einer Konfiguration zu überprüfen. Eine Erweiterung des Formalismus um ein solches Existenzprädikat ist nicht unproblematisch, da ein solches Prädikat bei gegebener Konfiguration nicht in konstanter Zeit berechenbar wäre, so daß sich die Gesamtkomplexität des Analyseverfahrens erhöhen würde (siehe Abschnitt 3.2.3).

$$\text{Prep} : \text{X.label} = \text{pmod} \wedge \text{existsModifier}(\text{pn}, \text{X}\downarrow\text{id})$$

Ein weiteres Problem für binäre Constraints stellen strukturelle Konfigurationen dar, die zu ihrer Überprüfung ternäre Constraints benötigen. Lahres (1995) beschreibt solche Fälle und schlägt eine Erweiterung des Formalismus der Constraints um ein Evidenzprädikat vor. Dadurch ist es möglich, ternäre Constraints auf binäre zurückzuführen, indem man eine Variable in dem Evidenzprädikat aufgehen läßt.

Solche Erweiterungen sind jedoch nicht unproblematisch, da sich dadurch u. U. eine Reigenfolgeabhängigkeit der Constraintanwendungen ergibt.

3.2.3 Verarbeitungsaspekte

Als Lösungsverfahren wird ein Algorithmus verwendet (Maruyama, 1990a; Maruyama, 1990b), der zunächst einen gewissen Grad an lokaler Konsistenz (zwischen je zwei Variablen) herstellt und daraufhin Backtracking zum Extrahieren der Lösungen anwendet. Schröder (1995) führt Schritt für Schritt durch ein Beispiel.

Bei gegebener CD–Grammatik G läuft die Analyse eines Satzes beim Basisverfahren in den folgenden Schritten ab. Die Zeitkomplexität des jeweiligen Schrittes ist am Schluß angegeben; n bezeichnet die Länge des Satzes, u die Anzahl der unären Constraints und b die Anzahl der binären Constraints. Eine ausführlichere Darstellung findet sich z. B. in (Maruyama, 1990a; Maruyama, 1990b; Schröder, 1995).[13, 14]

1. Initialisierung der CSP–Variablen: Für alle Worte ω_i des Satzes und alle Rollen r_j der Grammatik G wird eine Variable v_i^j erzeugt und mit ihren möglichen Werten initialisiert. $O(n^2)$

2. Anwendung der unären Constraints: Alle Constraints der Grammatik G werden für alle Variablen für alle ihre möglichen Werte angewendet. Wird ein Constraint für einen Wert verletzt, so wird dieser Wert aus der Domäne der Variablen gelöscht. $O(u \cdot n^2)$

3. Initialisierung des Constraint–Netzwerkes: Alle Variablen werden durch Kanten im Constraint–Graphen miteinander verbunden. An diesen Kanten wird eine Tabelle notiert, die die Werte der ersten Variablen in den Spalten und die Werte der zweiten Variable in den Reihen enthält; die Zellen der Tabelle werden mit dem Wahrheitswert wahr initialisiert. $O(n^4)$

4. Anwendung der binären Constraints: Für alle Quadrupel aus zwei Variablen und zwei zu diesen Variablen passenden Werten — diese werden gerade durch die Zellen der Tabellen aus dem vorherigen Schritt repräsentiert — werden alle binären Constraints angewendet. Wird ein Constraint für solch ein Quadrupel verletzt, so wird der Wahrheitswert in der Tabellenzelle auf falsch gesetzt. $O(b \cdot n^4)$

5. Herstellen von lokaler Konsistenz: Wann immer eine komplette Spalte oder eine komplette Reihe in einer der Tabellen an den Kanten des Constraint–Graphen den Wahrheitswert falsch enthält, bedeutet dies, daß es für den zugehörigen Wert der entsprechenden Variablen keinen anderen

[13] Helzerman und Harper (1992) beschreiben eine Implementation des Parsings mit CDG auf einer massiv parallelen Rechnerarchitektur mit $O(n^4)$ Prozessoren und einer Zeitkomplexität von $O(u + b + \log(n))$. Die gute Parallelisierbarkeit des Parsings mit CDG ist auch beim Vergleich mit der menschlichen Informationsverarbeitung beim Sprachverstehen interessant.

[14] Bei Vergleichen der Zeitkomplexität von Analyseverfahren wird insbesondere die Äußerungslänge n berücksichtigt. Das Parsing mit CDG schneidet bei Beachtung der Grammatikgröße, die oft sehr viel größer als die Äußerungslänge ist, im Vergleich zu kontextfreien Verfahren besser ab (Harper et al., 1994).

Wert der anderen Variablen gibt, so daß kein Constraint verletzt ist. Ein solcher Wert kann dann aus der Domäne der Variablen und aus allen zugehörigen Tabellen gelöscht werden. Enthält jede Spalte und jede Zeile der Tabellen mindestens einmal den Wert wahr, so ist der Constraint–Graph kantenkonsistent.[15] $O(n^4)$

6. Extrahieren der Lösung: Durch andere Standardsuchverfahren, z. B. Backtracking, werden mögliche Lösungen, also Belegungen der Variablen, aus dem Constraint–Graphen ermittelt. Diese Suche ist notwendig, da ein kantenkonsistenter Constraint–Graph im allgemeinen durchaus global inkonsistente Wertbelegungen enthält (Schröder, 1995, Abschnitt 2.1.7). Die exponentielle Laufzeit stellt natürlich den schlechtesten Fall dar; durch die bereits eingestellte lokale Konsistenz ist oft eine effiziente Abarbeitung möglich bzw. gar keine Suche mehr erforderlich. $O(2^n)$

3.3 Wortgraphen und lexikalische Ambiguität

Das Verfahren, wie es bisher vorgestellt wurde, setzt voraus, daß die Folge der Worte mit ihren lexikalischen Eigenschaften beim Parsing bekannt ist. Da dies im allgemeinen nicht der Fall ist, wird in diesem Abschnitt eine Erweiterung zur Verarbeitung von Wortgraphen vorgestellt.

3.3.1 Warum Wortgraphen?

Aktuelle Spracherkenner können als Ergebnis entweder wahrscheinlichste Wortketten oder sogenannte Wortgraphen, die eine kompakte Repräsentation der möglichen Wortfolgen darstellen, ausgeben.[16]

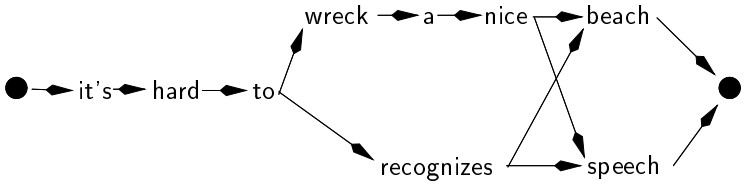


Abbildung 3.3: Wortgraph als Ausgabe eines Spracherkenners

Die (Harper und Helzerman, 1994) entnommene Abbildung 3.3 zeigt einen Wortgraphen, der die Wortfolgen der Beispiele 3.2 bis 3.5 kompakt repräsentiert.

(3.2) it's hard to wreck a nice beach

[15] Algorithmen zu verschiedenen Techniken zur Herstellung von lokaler Konsistenz finden sich z. B. in (Mackworth, 1977), (Freuder, 1982), (Nudel, 1983), (Freuder, 1985), (Mackworth und Freuder, 1985), (Mohr und Henderson, 1986) und (Mackworth, 1992).

[16] Die wahrscheinlichste Wortkette kann auch als degenerierter linearer Graph aufgefaßt werden.

(3.3) it's hard to wreck a nice speech

(3.4) * it's hard to recognizes speech

(3.5) * it's hard to recognizes beach

Wortgraphen enthalten die tatsächlich gesprochene Wortfolge häufiger als Satzhypothese und werden aus diesem Grund oft als angemessenere Ausgabe der Spracherkenner betrachtet.

Auch wenn man auf Wortgraphen als reichhaltigere Ausgabestruktur eines Worterkenners verzichten kann,[17] bleibt die direkte Anwendung des Basisverfahrens außer für Testfälle undurchführbar. Die lexikalische Ambiguität eines Wortes erlaubt keine eindeutige Identifikation des Lemmas bei gegebener graphemischer Form. Für den Beispielsatz 3.6 kann nicht a priori entschieden werden, ob mit dem Graphem 'Bank' die Sitzgelegenheit, das Gebäude oder die Institution gemeint ist oder auch nur, in welchem Kasus (hier möglicherweise Nominativ, Akkusativ oder Genitiv) das Graphem 'Männer' steht. Die Verwendung von mehreren lexikalischen Varianten für eine Wortform führt wieder auf einen Wortgraphen (einer speziellen Klasse).

(3.6) Die beiden Männer trafen sich an der Bank.

Eine Erweiterung des Verfahrens zur Verarbeitung von Wortgraphen an Stelle von Wortfolgen wird in (Harper et al., 1992) unter dem Namen 'Constraint–Netz für gesprochene Sprache', kurz SLCN (engl. *spoken language constraint network*), vorgestellt. Für eine detailliertere Darstellung sei auf (Harper et al., 1994) verwiesen.

3.3.2 Modifikation des Algorithmus

Die beiden wichtigsten Modifikationen des Algorithmus betreffen zum einen die Anwendbarkeit von binären Constraints und zum anderen die Bedingungen, bei denen eine mögliche Lösung gelöscht werden kann.

- Beim Basisverfahren müssen alle binären Constraints zwischen allen Variablen des CSP gelten, da vorausgesetzt wird, daß alle Worte Teil der Lösung sind. In SLCN dagegen muß zusätzlich die Struktur des Wortgraphen berücksichtigt werden. Zum Beispiel brauchen keine Constraints zwischen den Variablen zu gelten, die für die Worte 'wreck' und 'recognizes' in Abbildung 3.3 stehen, da diese Worte in keiner Lösung gemeinsam auftreten können.
- Ohne Berücksichtigung des Wortgraphen könnte im Beispielsatz 3.5 für das Wort 'to' die Variante, in der 'to' einen Infinitiv einleitet, gelöscht

[17] Bei geschriebener Eingabe ist dies z. B. sicher der Fall.

werden, da das folgende Wort 'recognizes' keine infinite Form ist. Unter Berücksichtigung des Wortgraphen dagegen darf dies nicht geschehen, solange es alternative Satzhypothesen gibt, in denen die infinitiveinleitende Lesart korrekt ist (hier alle Hypothesen, die das Wort 'wreck' enthalten). Während also für das Basisverfahren gilt, daß eine Unterordnungsvariante ausgeschlossen werden darf, wenn mindestens ein Constraint mit *einer* anderen Variable verletzt wird, muß bei der Verarbeitung von Wortgraphen mindestens ein Constraint für mindestens eine Variable *jeder* Satzhypothese verletzt werden.

Die Algorithmen zum Parsing von Wortgraphen sind (inklusive Pseudokode) ausführlich beschrieben (Harper und Helzerman, 1994), so daß auf eine Darstellung hier verzichtet werden kann. Abweichend von dem dort vorgestellten Algorithmus wird in dieser Arbeit das Kriterium, unter welchen Bedingungen ein Wert b eines Knoten j einen anderen Wert a in einem anderen Knoten i stützen kann, anders gewählt. Dazu wird die Bedingung im Algorithmus an geeigneter Stelle[18] durch die allgemeinere Bedingung ersetzt, daß einerseits $\langle j, b\rangle$ gemäß einer Auswahlfunktion (siehe Abschnitt 3.5) geeignet sein muß, $\langle i, a\rangle$ zu stützen, und andererseits die zu i, a, j und b gehörenden Worte in *einem* Pfad durch den Wortgraphen vorkommen müssen.

3.4 Präferenzen durch gewichtete Constraints

In der Einleitung wird auf Seite 39 argumentiert, daß eliminative Parsing–Verfahren prinzipiell geeignet seien, robustes Systemverhalten hervorzurufen. Dies bedeutet, daß auch für Eingaben, die einen Teil der Grammatik verletzen, eine (Teil–)Struktur gefunden werden kann. Die Algorithmen des Constraint–Satisfaction liefern dagegen bei Verletzung von auch nur einem Constraint keine Lösung mehr, so daß kein robustes Systemverhalten erreicht wird. In dem Fall, daß keine mit allen Constraints konsistente Belegung der Variablen des CSP gefunden werden kann, spricht man von einem überbestimmten (engl. *over-constrained*) CSP. Um dieses Problem zu überwinden, schlägt Menzel (1995) vor, von der strikt binären Sichtweise auf die Gültigkeit von Constraints zu einem bewertenden Charakter überzugehen. CSP, bei denen die Belegung von Variablen, die möglichst wenige und möglichst schwach bewertete Constraints verletzt, gesucht wird, werden in der Literatur als partielle CSP, kurz PCSP (engl. *partial constraint satisfaction problem*), bezeichnet.

Nach einer Definition der PCSP in Abschnitt 3.4.1 und einem illustrierenden Beispiel in Abschnitt 3.4.2 werden in Abschnitt 3.4.3 einige Vorschläge zur Lösung von PCSP vorgestellt. Dazu gehören die Generalisierung einer Reihe von Verfahren zur Lösung von CSP auf PCSP (Freuder und Wallace, 1992), die Constraint–Abschwächung (engl. *constraint relaxtion*) (Padró, 1995) und das

[18] Harper und Helzerman (1994) beschreiben den entsprechenden Algorithmus in der Abbildung 23 ihrer Veröffentlichung. Die angesprochende Bedingung findet sich dort in Zeile 16.

hierarchische Constraint–Lösen (engl. *hierarchical constraint solving*) (Wilson und Borning, 1993). Im Abschnitt 3.5 wird dafür argumentiert, für das Parsing als PCSP auf andere — problemspezifische — Verfahren zurückzugreifen.

3.4.1 Partielle Constraint–Satisfaction–Probleme

Ein PCSP ist ein Quadrupel $\langle V, D, C, W\rangle$, wobei den Elementen V, D und C die gleiche Bedeutung wie beim CSP zukommt. W ist eine Funktion $W : C \mapsto [0,1]$, die jedem Constraint eine Bewertung zuweist.[19] Die Lösung eines PCSP besteht aus einer Belegung der Variablen $v_1 = a_1, \ldots, v_n = a_n$, so daß eine globale Bewertungsfunktion optimiert wird. Welche Bewertungsfunktion für eine konkrete Anwendung ausgewählt wird, ist problemabhängig. Verschiedene Heuristiken für die Auswahl der Funktion existieren, jedoch keine Entscheidungsanweisung; im Zweifelsfall werden empirische Ergebnisse zur Entscheidungsfindung herangezogen.

Hier soll eine hohe Bewertung eines Constraints bedeuten, daß das Constraint weniger wichtig ist bzw. daß seine Verletzung tolerierbar ist. Constraints $c \in C$ mit $W(c) = 0$ entsprechen näherungsweise den unbewerteten Constraints eines CSP, wenn man z. B. die globale Bewertungsfunktion in Gleichung 3.2 zur Auswahl der Lösung heranzieht.

$$A = \underset{\langle a_1,\ldots,a_n\rangle \in D_1\times\ldots\times D_n}{\arg\max} \prod_{\substack{1\le i,j\le n \\ c=\langle v_i,a_i,v_j,a_j\rangle\in C}} W(c) \tag{3.2}$$

$$A = \underset{\langle a_1,\ldots,a_n\rangle \in D_1\times\ldots\times D_n}{\arg\min} \sum_{\substack{1\le i,j\le n \\ c=\langle v_i,a_i,v_j,a_j\rangle\in C}} (1 - W(c)) \tag{3.3}$$

Eine andere Sichtweise auf ein PCSP ergibt sich mit einer anderen Bewertungsfunktion wie in Gleichung 3.3. Durch die Summenbildung kann kein Constraint mehr als unverletzbar deklariert werden; auch ein mit Null bewertetes Constraint kann in einer Lösung verletzt sein, wenn andere (möglicherweise schwächer bewertete) Constraints den Beitrag aufwiegen. Unter der Annahme, daß es für das Parsing möglich sein sollte, Constraints zu formulieren, die auf jeden Fall eingehalten werden müssen,[20] scheidet die auf der Summation basierende Bewertungsfunktion für die Analyse natürlichsprachlicher Äußerungen aus.

Oftmals werden die Lösungen eines PCSP nur näherungsweise berechnet, um eine effiziente Verarbeitung zu ermöglichen (siehe Abschnitte 3.4.3 und 3.5).

[19] Wie üblich gibt es zahlreiche Möglichkeiten für eine Definition; Brewka, Guesgen und Hertzberg (1992) geben einige Varianten für die Definition des PCSP und dessen Lösungen unter Rückgriff auf Ergebnisse des nicht monotonen Schließens an.

[20] Einige Wohlgeformtheitsbedingungen für Bäume sowie Grundannahmen im grammatischen System gehören in diese Klasse.

3.4.2 Beispiel für eine Analyse mit bewerteten Constraints

Menzel (1995) zeigt an dem folgenden Beispiel exemplarisch, wie die Annahme von zwei relativ unabhängigen Repräsentationsebenen (vgl. Bemerkungen dazu auf Seite 61) und die Bewertung von Constraints zu einem robusten Verhalten führen. Als Bewertungsfunktion kommt dabei ein lokal operierendes Kriterium auf der Basis der Summe der Quadrate der Bewertungen zum Einsatz (siehe Abschnitt 3.4.3).

(3.7) Pferde fressen Gras.

Die gewünschte Struktur für den Beispielsatz 3.7 sei die in Abbildung 3.4 anhand von zwei Bäumen dargestellte Dependenz.

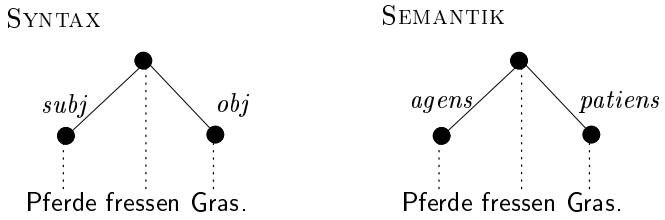


Abbildung 3.4: Zielstrukturen für Beispielsatz 3.7

In (Menzel, 1995; Menzel, 1996) werden die in Tabelle 3.1[21] textuell angegebenen Constraints mit durch Plausibilitätsüberlegungen manuell erzeugten Bewertungen benutzt, um anhand von zu verschiedenen Graden syntaktisch und semantisch abweichenden Eingaben zu überprüfen, ob das Verfahren in der Lage ist, Erwartungsverletzungen auf einer Ebene durch starke Evidenz für bestimmte Strukturen auf der anderen Ebene auszugleichen.

Die im wesentlichen aus der Originalquelle (Menzel, 1996) entnommene Tabelle 3.2 faßt das Ergebnis der Experimente zusammen. Erfolgreich analysierte Sätze sind grau hinterlegt. In Abschnitt 3.5 wird erneut auf dieses Beispiel eingegangen.

Die Ergebnisse zeigen, daß Erwartungsverletzungen durch erfolgreiche Analysen auf einer anderen Ebene kompensiert werden können. Erst wenn sich mehrfach und auf mehreren Ebenen gleichzeitig Abweichungen von der Grammatik einstellen, wird die gewünschte Struktur nicht mehr gefunden.

3.4.3 Verarbeitungsaspekte

Es existieren eine Reihe von Verfahren zur Lösung von PCSP. Dabei ist zwischen exakten Algorithmen und heuristischen bzw. approximativen Verfahren

[21] Die Bewertungen der Constraints 6 und 7 sind in den Quellen (Menzel, 1995; Menzel, 1996) versehentlich vertauscht. Die Constraints 11 und 12 sind ergänzt worden, um die strukturelle Anbindung des Verbs eindeutig zu machen.

(1.)	Ein Nomen hängt syntaktisch entweder als Subjekt oder als Objekt vom Verb ab.	(0.0)
(2.)	Das Subjekt stimmt mit dem Verb im Numerus überein.	(0.1)
(3.)	Das Subjekt steht vor dem Verb.	(0.3)
(4.)	Kein Wort kann von zwei verschiedenen Worten mit derselben Beschriftung syntaktisch modifiziert werden.	(0.0)
(5.)	Ein Nomen hängt semantisch entweder als Agens oder als Patiens vom Verb ab.	(0.0)
(6.)	'Tiere' sind Agens in Bezug auf das Ereignis 'fressen'.	(0.7)
(7.)	'Pflanzen' sind Patiens in Bezug auf das Ereignis 'fressen'.	(0.1)
(8.)	Kein Wort kann von zwei verschiedenen Worten mit derselben Beschriftung semantisch modifiziert werden.	(0.5)
(9.)	Das syntaktische Subjekt ist das semantische Agens.	(0.2)
(10.)	Das direkte syntaktische Objekt ist das semantische Patiens.	(0.3)
(11.)	Das Verb ist die Wurzel des syntaktischen Dependenzbaumes.	(0.0)
(12.)	Das Verb ist die Wurzel des semantischen Dependenzbaumes.	(0.0)

Tabelle 3.1: Beispielgrammatik mit Bewertungen

Syntax	Semantik		
	unterstützend	neutral	kontraintuitiv
unterstützend	Pferde fressen Gras.	Autos fressen Geld.	Gräser fressen Pferd.
markierte Abfolge	Gras fressen Pferde.	Geld fressen Autos.	Pferd fressen Gräser.
Numerusfehler	Pferd fressen Gras.	Auto fressen Geld.	Gras fressen Pferd.
kombiniert	Gras fressen Pferd.	Geld fressen Auto.	Pferd fressen Gras.

Tabelle 3.2: Parsing–Ergebnisse bei von Erwartungen abweichenden Eingaben

zu unterscheiden. Während erstere garantieren, daß die 'beste' bzw. die 'besten' Variablenbelegungen gefunden werden, berechnen die letzteren diese nur näherungsweise; bei ihnen ist also nicht ausgeschlossen, daß sie das globale Maximum verfehlen.

Direkte Entsprechungen der CSP–Algorithmen

Verfahren zur Lösung von CSP werden auf PCSP übertragen und bzgl. ihrer Effizienz evaluiert (Freuder und Wallace, 1992). Aus dem sogenannten Back-

tracking, das sukzessive die Variablen mit Werten belegt und bei einer Verletzung eines Constraints die letzte Variablenbelegung revidiert, wird z. B. der sogenannte Verzweige–&–Begrenze–Algorithmus (engl. *branch and bound*) abgeleitet, der ebenfalls eine sukzessive Variablenbelegung durchführt, jedoch erst dann eine Variablenbelegung revidiert, wenn die Belegung das bisherige Fehlerminimum überschreitet. Auch für prospektive Verfahren werden Entsprechungen vorgeschlagen (vgl. Abschnitt 3.1).

Die in (Freuder und Wallace, 1992; Wallace und Freuder, 1995b) vom CSP übernommenen Methoden berücksichtigen meist keine Bewertung der Constraints, sondern minimieren die Anzahl der verletzten Constraints. Das heißt aber nicht, daß diese Algorithmen nicht entsprechend erweitert werden könnten. Die Verfahren in (Freuder und Wallace, 1992) liefern in der Regel das optimale Ergebnis, während in (Wallace und Freuder, 1995b) ein heuristisches Verfahren überprüft wird.

Wallace und Freuder (1995a) beschreiben die Jederzeiteigenschaft (engl. *anytime property*) (Russell und Zilberstein, 1991; Hertzberg, 1995; Menzel, 1994) für Lösungsverfahren des PCSP.

Iterative Constraint–Abschwächung

Iterative Constraint–Abschwächung (engl. *iterative constraint relaxation*) ist eine Optimierungsmethode zur Lösung von PCSP. Sie umfaßt eine Klasse von iterativen Algorithmen, die lokale Informationen zur Optimierung einer globalen Funktion nutzen.[22] Wie bei anderen lokal operierenden Verfahren kann auch bei der iterativen Constraint–Abschwächung nicht garantiert werden, daß das global beste Ergebnis gefunden wird, da die Methode u. U. nicht in der Lage ist, globale Maxima von lokalen zu unterscheiden.

Padró (1995) wendet erstmalig die Technik der iterativen Constraint–Abschwächung auf Probleme der Verarbeitung natürlicher Sprache, im besonderen der Kategoriebestimmung (engl. *part–of–speech tagging*) und der Wortbedeutungsaufklärung (engl. *word sense disambiguation*), an.

Gegeben sei ein PCSP $\langle V, D, C, W\rangle$ (siehe Abschnitt 3.4.1) mit dem Variablenvektor $V = \langle v_1, \ldots, v_n\rangle$, dem Domänenvektor $D = \langle D_1, \ldots, D_n\rangle$ mit $D_i = \langle d_i^1, \ldots, d_i^{m_i}\rangle$, der Constraintmenge C und der Bewertungsfunktion W. Eine *gewichtete Variablenbelegung* ist dann ein Vektor $P = \langle P_1, \ldots, P_n\rangle$, der für jede Variable v_i für jeden möglichen Wert ein Gewicht enthält: $P_i = \langle p_i^1, \ldots, p_i^{m_i}\rangle$ mit $\sum_{j=i}^{m_i} p_i^j = 1, 0 \leq p_i^j \leq 1$. Eine Variablenbelegung im Sinne des Abschnitts 3.1 stellt dann einen Spezialfall der gewichteten Variablenbelegung dar, bei dem genau ein p_i^j den Wert Eins hat, und alle anderen den Wert Null annehmen.

Beginnend mit einer initialen gewichteten Belegung der Variablen werden die Gewichte der Werte in dem iterativen Prozeß der folgenden beiden Schritte verändert, bis ein Konvergenzkriterium erreicht ist.

[22] Padró (1995) sieht sie aus diesem Grund in enger Verwandtschaft zu Boltzmann–Maschinen, Gradientenabstiegsverfahren und Neuronalen Netzen.

1. Durch eine Unterstützungsfunktion S_{ij} (engl. *support function*) wird berechnet, wie kompatibel der Wert d_j einer Variable v_i mit der aktuellen gewichteten Belegung ist.

2. Das Gewicht jedes Variablenwertes wird anhand einer Aktualisierungsfunktion (engl. *update function*) gemäß der dem Variablenwert entgegengebrachten Unterstützung auf den neuen Stand gebracht. Für Variablenwerte mit hoher Unterstützung wächst das Gewicht, für solche mit niedriger Unterstützung fällt es.

Verschiedene Unterstützungsfunktionen und Aktualisierungsfunktionen sind in (Padró, 1995) angegeben. Die folgenden sind als Beispiele aufgeführt; welche Funktionen für eine bestimmte Anwendung angemessen sind, ist stark vom Problem und von der Definition der globalen Bewertungsfunktion (vgl. Abschnitt 3.4.1) abhängig. Auch in diesem Fall muß zur Klärung der Güte der Funktionen evtl. auf empirische Daten zurückgegriffen werden.

$$S_{ij} = p_i^j \cdot \sum_{\substack{c=\langle v_i,a_j,v_k,a_l\rangle \in C \vee \\ c=\langle v_k,a_l,v_i,a_j\rangle \in C}} W(c) \cdot p_k^l \tag{3.4}$$

$$\tilde{p}_i^j = \frac{p_i^j \cdot S_{ij}}{\sum_{k=1}^{m_i} p_i^k \cdot S_{ij}} \tag{3.5}$$

Um aus einer gewichteten Belegung eine Lösung für ein PCSP zu erhalten, muß aus jedem Vektor P_i ein Wert als Belegung bestimmt werden. Im einfachsten Fall wird der Wert mit dem größten Gewicht ausgewählt. Ob damit eine gute Auswahl getroffen wird, welche Verbesserungen es gibt und wie erfolgreich eine Anwendung im Kontext der CDG sein könnte, bleiben offene, aber interessante Fragen für die Zukunft.

Die iterative Constraint–Abschwächung besitzt die Jederzeiteigenschaft (Russell und Zilberstein, 1991; Hertzberg, 1995; Menzel, 1994), da bereits nach der ersten Iteration eine Lösung vorgewiesen werden kann. Weitere Berechnungsschleifen verbessern daraufhin das Ergebnis bei zur Verfügung stehender Zeit. Es arbeitet robust, da sich Constraints natürlich — bis zu einem gewissen Grad — widersprechen können. Auch mehrstellige Constraints stellen kein Problem dar. Das Verfahren ist sehr gut parallelisierbar, da die Funktionen für alle Werte parallel berechnet werden können, wodurch sich die polynomielle Zeitkomplexität bei serieller Implementation auf einen konstanten Zeitbedarf reduzieren läßt. Als Nachteil läßt sich der approximative Charakter des Verfahrens nennen; weiterhin ist die Wahl der Unterstützungs– und der Aktualisierungsfunktionen keine triviale Aufgabe.

Hierarchisches Constraint–Lösen

Wilson und Borning (1993) schlagen eine Erweiterung des CLP–Schemas (Frühwirth et al., 1993) namens HCLP (engl. *hierarchical constraint logic programming*) vor, die eine Variante des PCSP lösen kann. Bei der Formulierung eines HCLP werden Constraints in einer Hierarchie ihrer Verletzbarkeit H angeordnet. Mit H_0 wird der Vektor der um jeden Preis zu erfüllenden Constraints bezeichnet; H_1 ist der Vektor der nächst schwächeren Constraints usw. Für jede dieser Ebenen wird für eine Belegung der Variablen überprüft, welche Constraints diese erfüllt bzw. verletzt. Anhand einer zu wählenden Fehlerfunktion[23] und einer geeigneten Vergleichsfunktion (engl. *comparator*) können nun verschiedene Belegungen bzgl. ihrer Eignung, das Problem zu lösen, verglichen werden. Als wichtiger Unterschied zum allgemeinen PCSP ist zu nennen, daß eine Belegung als besser beurteilt wird, sobald sie auf einer Constraint–Ebene H_i von der Vergleichsoperation besser beurteilt wird als eine konkurrierende Belegung, ohne die höheren Ebenen H_j mit $j > i$ zu berücksichtigen. Insgesamt scheint das Schema der HCLP eher für eine kleine Menge von diskreten Hierarchiestufen[24] ausgelegt zu sein, die manuell erstellt werden.

Eine genauere Überprüfung auf Eignung für eine Übertragung der CDG auf HCLP scheint trotz der Unterschiede in der Ausrichtung lohnenswert und stellt eine Aufgabe für die Zukunft dar.

3.5 Spezielle Verfahren für präferenzbasierte CDG

Die bis hier vorgestellten Verfahren zur Lösung des PCSP — Erweiterungen der CSP–Algorithmen, iterative Constraint–Abschwächung und hierarchisches Constraint–Lösen — stellen allgemeine Schemata dar und sind nicht auf ein spezielles Problem angepaßt. Obwohl eine genaue Untersuchung aller Verfahren zur Lösung des durch eine präferenzbasierte CDG definierten PCSP aussteht, wird in dieser Arbeit mit einem problemspezifischen Lösungsansatz gearbeitet. Dafür lassen sich im wesentlichen zwei Gründe anführen.

- Der Übergang zu präferenzbasierten Constraints sollte nicht zu einem völlig neuen Verfahren gegenüber den wohlbekannten und –verstandenen Verfahren für CDG führen. Statt dessen ist es vorteilhaft, existierende Algorithmen für die Verarbeitung von Präferenzen zu modifizieren, um z. B. die Erweiterung auf Wortgraphen (siehe Abschnitt 3.3) weiterhin nutzen zu können.

- Das spezielle Problem des Parsing läßt einige Annahmen darüber sinnvoll erscheinen, welche Eigenschaften eine Wertbelegung haben sollte, um als Lösung ausgeschlossen zu werden. Solche problemspezifischen Kriterien

[23] Die Fehlerfunktionen können zusätzlich einzelne Gewichte für Constraints einbeziehen.

[24] Die Ebenen in den Beispielen in (Wilson und Borning, 1993) werden konsequenterweise auch mit englischen Begriffen wie 'required', 'strong' usw. bezeichnet.

			Syn			Sem				
			fressen	gras		pferde		fressen	gras	
			root	*subj*	*obj*	*ag*	*pat*	*root*	*ag*	*pat*
Syn	pferde	*subj*	1.0	.0	1.0	1.0	.042	1.0	.1	1.0
		obj	1.0	.03	.0	.06	.7	1.0	.1	1.0
	fressen	*root*		.03	1.0	1.0	.7	1.0	.1	1.0
	gras	*subj*				.03	.021	.03	.003	.0018
		obj				1.0	.7	1.0	.006	1.0
Sem	pferde	*ag*						1.0	.05	1.0
		pat						.7	.07	.35
	fressen	*root*							.1	1.0

Tabelle 3.3: Bewertungen im Constraint–Netzwerk nach Anwendung der unären und binären Constraints für die Äußerung 'Pferde fressen Gras.'

sind in die allgemeinen Methoden nicht ohne weiteres integrierbar. In den folgenden Abschnitten wird versucht, solch problemspezifisches Wissen anhand von sogenannten Auswahlfunktionen in das Verfahren einzubringen.

3.5.1 Problemstellung

Zunächst stellt sich die Frage, was sich durch den Übergang zu bewerteten Constraints eigentlich ändert. Bisher wurde eine mögliche Abhängigkeitsbeziehung zunächst durch die unären Constraints überprüft (Schritt 2 in Abschnitt 3.2.3); anschließend stellten die binären Constraints die Wohlgeformtheit für jeweils ein Paar solcher Dependenzbeziehungen fest (Schritt 4 in Abschnitt 3.2.3). Im aufgebauten Constraint–Netzwerk wurden je nach Ergebnis bei der Constraint–Auswertung die Wahrheitswerte wahr bzw. falsch eingetragen.

Constraints mit Bewertungen liefern bei ihrer Verletzung durch eine Konfiguration nicht mehr den Wert falsch zurück, sondern eine reelle Zahl als Bewertung. Die Tabellen an den Kanten des Constraint–Graphen enthalten deswegen statt der Wahrheitswerte nun reelle Zahlen im Intervall $[0, 1]$. Beginnend bei einer Bewertung von Eins wird für jedes verletzte Constraint dessen Bewertung mit der aktuellen Bewertung in der Tabelle multipliziert. Tabelle 3.3 zeigt diese Bewertungen in dem Constraint–Netzwerk für die Analyse der Äußerung 'Pferde fressen Gras.', das aus Abschnitt 3.4.2 übernommen wurde und auf (Menzel, 1995) zurückgeht. Dabei sind die einzelnen Tabellen durch Doppellinien voneinander abgegrenzt.

Um nun den entscheidenden Schritt der Constraint–Propagation, das Filtern (Schritt 5 in Abschnitt 3.2.3), durchführen zu können, muß entschieden werden, welche Tabelleneinträge dem bisherigen Wert falsch entsprechen, also evtl. Anlaß für die Löschung einer Lösung geben. Genau genommen muß nicht zwangsweise für jeden Tabelleneintrag einzeln geklärt werden, ob er als Wert falsch

angesehen wird; statt dessen müssen ganze Reihen bzw. Spalten der Tabellen ausgemacht werden, die komplett als falsch bewertet werden, da erst vollständige Reihen bzw. Spalten den Filterprozeß in Gang setzen. Genau für diese Auswahl kann Wissen über das Problem, hier die Analyse natürlichsprachlicher Äußerungen, verwendet werden. Die Diskussion verschiedener Auswahlfunktionen für diese Aufgabe ist Gegenstand der folgenden Abschnitte.

Menzel (1996) führt die folgenden Kriterien für eine solche Auswahl an:[25]

- Im globalen Vergleich über alle Tabellen des Constraint–Netzwerks sollte eine Bewertung nahe des Minimums erreicht werden, um überhaupt einen quantitativen Vergleich zu ermöglichen.
- Der Kontrast zu alternativen Modifikationen sollte möglichst stark sein, da dann offensichtlich eine starke Dispräferenz gegen diesen Wert vorliegt.
- Die einzelnen Werte für eine bestimmte Unterordnung sollten einen möglichst geringen Kontrast aufweisen, da in diesem Fall davon ausgegangen werden kann, daß die schlechte Bewertung tatsächlich auf die zu löschende Beziehung zurückzuführen ist.

Bei diesen Kriterien wurde absichtlich offen gelassen, für welche Art von Bewertung diese Eigenschaften gültig sein sollen, da einerseits das Objekt der Beurteilung die Tabelleneinträge bzw. –reihen und –spalten darstellen, sich andererseits aber (linguistisch und verarbeitungsorientiert) gute Heuristiken nur für ganze Unterordnungsbeziehungen angeben lassen. Die Bewertungen für diese sind aber leider über das gesamte Constraint–Netz verteilt. So finden sich z. B. Bewertungen für die Unterordnung des Wortes 'Pferd' als Agens unter das Verb an sechs verschiedenen Stellen in der Tabelle 3.3, so daß eine Beurteilung von Unterordnungsrelationen aufgrund von lokalen Informationen unmöglich ist.

Im folgenden werden verschiedene Auswahlfunktionen vorgestellt und daraufhin überprüft, ob sie die oben genannten Kriterien erfüllen.

3.5.2 Auswahl auf der Basis der Quadrate der Bewertungen

In (Menzel, 1995; Menzel, 1996) wird zu experimentellen Zwecken eine Auswahlfunktion auf der Basis der Quadrate der Bewertungen verwendet. Sei $T = \langle T_1, \ldots, T_n \rangle$ die Liste aller Tabellen an den Kanten im Constraint–Graph. Die einzelnen Tabellen T_i nehmen die Gestalt wie in Gleichung 3.6 an.

$$T_i = \begin{pmatrix} t_i^{1,1} & \ldots & t_i^{x_i,1} \\ \vdots & \ddots & \vdots \\ t_i^{1,y_i} & \ldots & t_i^{x_i,y_i} \end{pmatrix} \tag{3.6}$$

[25] Die Begründungen sind aus Gründen der Verständlichkeit hinzugefügt und stammen nicht aus der Quelle.

Die 'schlechteste' Zeile in einer Tabelle T_i wird dann gemäß Gleichung 3.7 ermittelt; für Spalten gilt eine analoge Gleichung.[26]

$$A_i^Z = \underset{1 \leq k \leq y_i}{\arg\min} \frac{\sum_{j=1}^{x_i} (t_i^{j,k})^2}{x_i} \tag{3.7}$$

Die Auswahlfunktion berechnet anhand dieser Gleichungen zunächst die 'schlechteste' Zeile A_i^Z bzw. Spalte A_i^S für jede Tabelle T_i und daraufhin für das gesamte Netzwerk. Nur diese eine Zeile bzw. Spalte wird dann gestrichen und der Filterungsprozeß in Gang gesetzt. Nach Abschluß wird die nächste Zeile bzw. Spalte gewählt, bis schließlich nur noch eine Lösung übrigbleibt.

Dieses Vorgehen zeigt für die Beispieläußerungen aus (Menzel, 1996) sehr gute Ergebnisse (siehe Abschnitt 3.4.2). Aber erfüllt es die gestellten Kriterien? Zunächst werden durch das Verfahren nicht Bewertungen für Unterordnungen verglichen, sondern nur Teilaspekte derselben. Weiterhin wird nur das (absolute) Minimum, nicht aber der Kontrast berechnet. Dagegen wird ein schwacher Kontrast innerhalb einer Zeile bzw. Spalte betont, da durch die Bildung der Quadrate diese verstärkt werden. Zusammenfassend kann das Verfahren also nur als Annäherung an eine effektive Auswahlfunktion angesehen werden; eine Evaluation wird in Abschnitt 3.5.6 beschrieben.

Es bleibt zu fragen, welche Anteile die Auswahlfunktion und der Filterungsprozeß an der Löschung der Möglichkeiten haben. Die Beschreibung des Ablaufs der Analyse anhand der Tabelle 3.3 gibt interessante Antworten auf diese Frage. Als erste zu löschende Zeile wird die Subjektlesart des Wortes 'Gras' aufgrund des Zusammenspiels mit der Semantik desselben Wortes identifiziert (Werte 0.003 und 0.0018). Das Filtern sorgt dann für das Löschen der Objektlesart des Wortes 'Pferde', da sich in der zugehörigen Spalte für die Syntax von 'Gras' nur noch eine Null befindet. Jeder weitere Aufruf des Filterungsprozesses führt von nun an zu *keiner* weiteren Löschung, d. h. die Auswahlfunktion sorgt allein für die weitere Reduzierung der Lesarten.

Dieses Beispiel läßt sich verallgemeinern. Da die Auswahlfunktion immer genau eine Zeile bzw. Spalte zur Löschung freigibt, kann der Filterungsprozeß nur dann für eine weitere Löschung sorgen, wenn es Tabelleneinträge mit Nullen gibt, da nur solche Tabelleneinträge eindeutig, also unabhängig von der Auswahlfunktion, als dem Wahrheitswert falsch entsprechend angenommen werden können. Diese sind im allgemeinen aber äußerst selten, da sie nur von mit Null bewerteten binären Constraints eingeführt werden können.[27] Die Auswahlfunktion auf der Basis der Summe der Quadrate der Bewertungen führt im schlimmsten Fall also dazu, daß sukzessive die 'schlechtesten' Zeilen bzw. Spalten gestrichen werden, ohne daß der Filterungsprozeß überhaupt einen Beitrag liefert.

[26] Der zu minimierende Ausdruck wird hier bzgl. der Zeilen– bzw. der Spaltenlänge zu normalisierend angenommen und deshalb als Bruch geschrieben.

[27] Bei den Experimenten für diese Arbeit finden sich überhaupt nur vier solche Constraints; alle stammen aus der Basisgrammatik (siehe Anhang D).

3.5.3 Auswahl auf der Basis eines absoluten Schwellwertes

Die wohl einfachste Auswahlfunktion besteht darin, einen absoluten Schwellwert festzulegen, den Tabelleneinträge nicht unterschreiten dürfen, ohne zur Löschung freigegeben werden zu dürfen. Dieser Schwellwert kann im Verlauf der Analyse sukzessive angehoben werden, bis schließlich keine weitere Reduzierung der Lesarten mehr möglich ist.

Diese Art der Auswahlfunktion erfüllt das erste Kriterium nur näherungsweise, die anderen beiden überhaupt nicht. Auch diese Funktion wird in Abschnitt 3.5.6 evaluiert.

3.5.4 Auswahl eines Anteils der Tabelleneinträge

Um zu betonen, daß eine Löschung eines Wertes nur *relativ* zu Alternativen geschehen soll, wird als weitere einfache Auswahlfunktion die Löschung eines Anteils der Einträge pro Tabelle probiert. Für jede Tabelle werden dabei diejenigen Einträge gelöscht, die in einer nach absteigender Bewertung sortierten Liste die hinteren Plätze einnehmen, also etwa das letzte Drittel. Welcher Anteil dabei jeweils gelöscht wird, kann im Verlauf der Analyse variiert werden; zunächst wird man einen kleinen Anteil löschen, später dann ein größeres Risiko eingehen und einen größeren Teil löschen, falls nicht schon eine Disambiguierung erreicht wurde.

3.5.5 Auswahl von Anteilen in einer Zeile und Spalte

Die unbefriedigenden Eigenschaften der bisher gefundenen Auswahlfunktionen führen zu einer weiteren, die versucht, verschiedene Aspekte der gewünschten Eigenschaften miteinander zu kombinieren. Vor der Vorstellung soll aber das erste Kriterium für eine gute Auswahlfunktion aus (Menzel, 1996) erneut kritisch betrachtet werden. Es wird darin eine Bewertung nahe des globalen Minimums für eine zu löschende Zeile bzw. Spalte gefordert. Implizit wird dabei die Annahme gemacht, daß die verschiedenen Zeilen und Spalten (als eine Annäherung an bestimmte Unterordnungsbeziehungen) überhaupt vergleichbar sind. Diese Annahme ist nicht in jedem Fall gerechtfertigt. Die Tabelle 3.4 gibt den Zustand des Constraint–Netzwerkes für die Äußerung 'Gras fressen Pferd.' unmittelbar nach Anwendung der Constraints wieder.[28]

Da u. a. kein Nomen mit dem Verb im Numerus übereinstimmt, findet sich keine Subjektlesart für eine syntaktische Anbindung, die global gut bewertet ist. Sowohl das gewünschte Subjekt 'Pferd' als auch das designierte Objekt 'Gras' weisen in der Objektlesart global bessere Bewertungen auf als in den

[28] Eine gute Auswahlfunktion soll nicht nur für gutmütige Constraint–Netze wie in Tabelle 3.3 gute Hypothesen liefern, sondern auch für schwierigere Fälle wie diesen.

			Syn			Sem				
			fressen	gras		pferd		fressen	gras	
			root	*subj*	*obj*	*ag*	*pat*	*root*	*ag*	*pat*
Syn	pferd	*subj*	.03	.0	.03	.03	.00126	.03	.003	.03
		obj	1.0	.1	.0	.06	.7	1.0	.1	1.0
	fressen	*root*		.1	1.0	1.0	.7	1.0	.1	1.0
	gras	*subj*				.1	.07	.1	.01	.006
		obj				1.0	.7	1.0	.006	1.0
Sem	pferd	*ag*						1.0	.05	1.0
		pat						.7	.07	.35
	fressen	*root*							.1	1.0

Tabelle 3.4: Bewertungen im Constraint–Netzwerk nach Anwendung der unären und binären Constraints für die Äußerung 'Gras fressen Pferd.'

Subjektlesarten. So stellt die Subjektlesart des Wortes 'Pferd' global gesehen die zweitschlechteste Bewertung.[29]

Anhand dieses kleinen Beispiels sollte klar werden, daß der globale Vergleich von Lesarten nicht immer möglich ist; die nächste Auswahlfunktion verzichtet daher darauf.

Sie basiert auf der Annahme, daß ein Tabelleneintrag bzgl. seiner Bewertung sowohl mit Alternativen in der Zeile als auch mit Alternativen in der Spalte verglichen werden kann. Wenn er in mindestens einem dieser Vergleiche sehr schlecht abschneidet, kann man annehmen, daß bessere Wertekombinationen für die beiden betroffenen Constraint–Knoten existieren. Das Zeile–Spalte–Verfahren markiert alle diejenigen Tabelleneinträge als dem Wahrheitswert falsch bzgl. des Filterns äquivalent, die entweder in ihrer Zeile oder in ihrer Spalte einen bestimmten Prozentsatz f des Maximums der Zeile bzw. Spalte nicht erreichen. Formal beschreibt Gleichung 3.8 für einen Tabelleneintrag $t_i^{x,y}$ das Vorgehen.

$$A_i^{x,y} = t_i^{x,y} < f \cdot \max_{1 \le j \le x_i} t_i^{j,y} \qquad \vee \qquad t_i^{x,y} < f \cdot \max_{1 \le j \le y_i} t_i^{x,j} \tag{3.8}$$

Der verwendete Parameter f mit $f \in [0,1]$ kann im Verlauf der Analyse variiert werden, falls keine frühe, vollständige Disambiguierung möglich ist. Bei einem Wert von $p = 0,04$ werden z. B. in Tabelle 3.3 acht Werte als falsch markiert,

[29] Bei Verwendung der normalisierten Summe der Quadrate der Bewertungen ergibt sich die zweitschlechteste Bewertung zu $4,5 \cdot 10^{-4}$ im Vergleich zu $6,8 \cdot 10^{-5}$ für die Subjektlesart des Wortes 'Gras'. Nur die Löschung der einzigen Alternative durch den Filterprozeß verhindert eine Löschung im zweiten Durchgang. Hier führt diese Auswahlfunktion zu exakt dem gewünschten Verhalten; das Beispiel soll vielmehr zeigen, daß ein globaler Vergleich problematisch sein kann. Wäre z. B. das Constraint 4 aus Abschnitt 3.4.2 statt mit Null mit einem Wert größer als Null bewertet — was bei der allgemeinen Formulierung des Constraints (nicht aber in dem hier betrachteten speziellen Fall) linguistisch durchaus sinnvoll wäre — so würde das Verfahren die beiden einzigen möglichen Subjekte des Satzes gleich zu Beginn löschen und so eine erfolgreiche Analyse verhindern.

was wiederum zur Löschung von drei Unterordnungsbeziehungen durch das Filtern führt. Erreicht p den Wert $0,05$, wird schließlich auch die letzte falsche Unterordnungsbeziehung gelöscht.

Syntax	Semantik		
	unterstützend	neutral	kontraintuitiv
unterstützend	Pferde fressen Gras.	Autos fressen Geld.	Gräser fressen Pferd.
markierte Abfolge	Gras fressen Pferde.	Geld fressen Autos.	Pferd fressen Gräser.[30]
Numerusfehler	Pferd fressen Gras.	Auto fressen Geld.	Gras fressen Pferd.
kombiniert	Gras fressen Pferd.	Geld fressen Auto.	Pferd fressen Gras.

Tabelle 3.5: Parsing–Ergebnisse für die Zeile–Spalte–Auswahlfunktion bei von Erwartungen abweichenden Eingaben

Tabelle 3.5 zeigt das Abschneiden dieser Auswahlfunktion für die Eingaben aus Abschnitt 3.4.2. Zusammen mit Tabelle 3.2 ergibt sich eine Gegenüberstellung zwischen dem Zeile–Spalte–Verfahren und der Methode auf der Basis der Summe der kleinsten Quadrate. Dabei wird deutlich, daß die neue Auswahlfunktion bzgl. der Anzahl der korrekt analysierten Eingaben nicht schlechter abschneidet; allerdings kann ein zweifacher Fehler auf der Ebene der Syntax nun auf keinen Fall mehr durch eine korrekte Semantik aufgewogen werden. Die Quadrateauswahlfunktion zeigt hier ein 'ausgeglicheneres' Bild. Inwieweit ein systematischer Zusammenhang der Grund für das unterschiedliche Verhalten darstellt oder ob etwa eine etwas andere Bewertung der Constraints zugunsten der Semantik den Unterschied egalisiert, bleibt eine offene Frage.

3.5.6 Zusammenfassung und Evaluation

Die in den letzten Abschnitten vorgestellten Auswahlfunktionen wurden experimentell evaluiert. In Kapitel 4 wird ausführlich auf die Art der Experimente sowie die Bewertungsmaßstäbe eingegangen, und in Anhang E sind die Ergebnisse aller relevanten Experimente aufgeführt. Hier werden nur die Endergebnisse zu Vergleichszwecken aufgeführt; dabei sind ein kleiner Fehler und eine hohe Genauigkeit vorteilhaft. Abbildung 3.5 zeigt die Ergebnisse für das Modell M_{clCd}^{2-19} (siehe Abschnitt 4.3 für eine Erklärung des Modells).

Es fällt auf, daß mit Ausnahme der einfachsten Auswahlfunktion des absoluten Schwellwertes die Unterschiede zwischen den verschiedenen Varianten relativ

[30] Für die Äußerung 'Pferd fressen Gräser.' führt das Verfahren zu keiner eindeutigen Lösung. Die verbleibende Ambiguität kann als Anzeichen für mehrere mögliche Lesarten interpretiert werden.

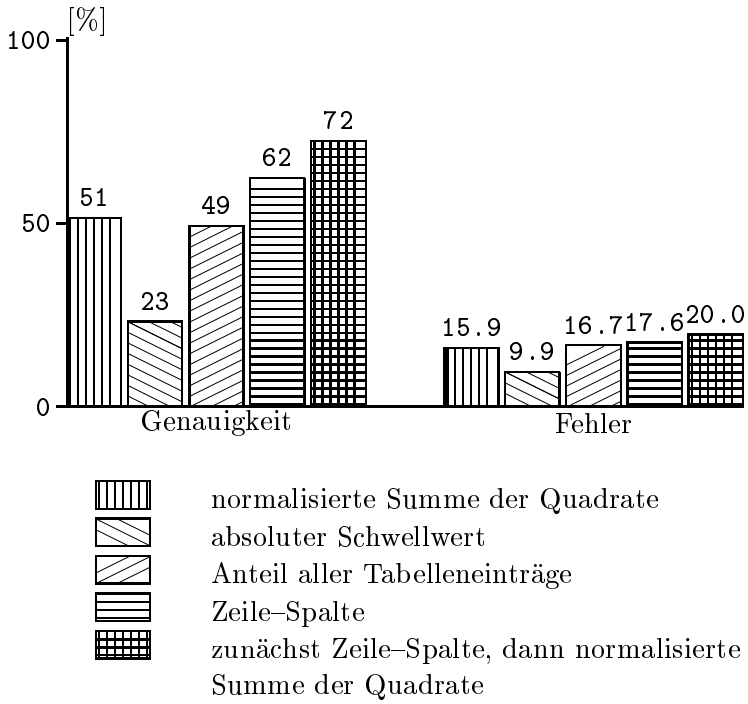


Abbildung 3.5: Parsing–Ergebnisse für verschiedene Auswahlfunktionen

gering sind. Ein Grund dafür könnte sein, daß sämtliche Auswahlfunktionen lokal zu einer Tabelle arbeiten, also die zu einer Unterordnungsrelation gehörigen Bewertungen nicht vollständig berücksichtigen. Eine Erweiterung der Auswahlfunktionen ist in weiteren Arbeiten vorgesehen.

Alle Auswahlfunktionen führen u. U. zu einer Restambiguität und damit zu einer verminderten Genauigkeit. Restambiguitäten entstehen, wenn die Auswahlfunktion solche Werte selektiert, die im nächsten Filterschritt zur Löschung aller Lösungen führt. In einem solchen Fall ist es natürlich günstiger, auf den Stand vor der letzten Auswahl zurückzugehen, um wenigstens eine Teillösung vorweisen zu können (siehe Abschnitt 4.4). Bei allen Verfahren außer dem auf den Quadraten der Bewertungen basierenden kann es auch zu Konstellationen kommen, bei denen der Filterprozeß keine Löschung und die Auswahlfunktion keine weitere Auswahl mehr durchführen können. Um zu überprüfen, welchen Anteil solche Verklemmungen an der Restambiguität haben, wird ein Experiment mit einer zweistufigen Auswahlfunktion durchgeführt, bei der *nach Beendigung* des Zeile–Spalte–Verfahrens mit dem auf der normalisierten Summe der Quadrate basierenden Verfahren weitere Zeilen ausgewählt werden. Diese Maßnahme führt zu einer um zehn Prozentpunkte verbesserten Genauigkeit bei einer Erhöhung des Fehlers um 2,4 Prozentpunkte und erweist sich damit als eine wirksame Maßnahme zur Reduzierung der Restambiguität. Zweifelsohne bedarf aber auch die damit erreichte Genauigkeit von 72% weiterer Maßnahmen, die jedoch nicht Gegenstand dieser Arbeit sind.

In Kapitel 4 wird der Einfachheit halber mit dem Zeile–Spalte–Verfahren gearbeitet und auf die zweistufige Auswahlfunktion verzichtet.

3.6 Weitere Erweiterungen

Als weitere interessante — aber in diesem Kontext weniger relevante — Erweiterungen sind die folgenden zu nennen:

- Adaptive Constraint–Anwendung: Schon Maruyama (1990b, S. 35) schlägt vor, bei verbleibender Mehrdeutigkeit nach einem ersten Analysedurchlauf weitere z. B. semantische Constraints hinzuzunehmen und so zusätzliche Informationsquellen für die Disambiguierung zu erschließen. (Harper et al., 1992; Harper et al., 1994; Harper und Helzerman, 1994) führen diese Idee konsequent auf den Ebenen Semantik, Pragmatik, Kontext u. a. m. aus. Eine relative Autonomie der Repräsentationsebenen, die nur schwach durch ebenenübergreifende Constraints gekoppelt sind, schlägt erstmals Menzel (1995) vor. Dadurch sollen Erwartungsverletzungen auf einer Ebene durch gelungene Disambiguierung auf anderen Ebenen ausgeglichen werden (vgl. Bemerkungen dazu in Abschnitt 3.4).

  Schließlich argumentiert (Menzel, 1994; Menzel, 1996) für eine differenzierte Planung der Constraint–Anwendungen unter Zeitbeschränkungen im Sinne eines ressourcen–adaptiven Verhaltens.[31]

- Ankopplung von externen Wissensbasen: Lahres (1996) untersucht u. a. die Möglichkeit, den Formalismus durch Ankopplung von externen Wissensbasen, z. B. in Form von terminologischen Logiken, zu erweitern.

- Inkrementalität: Für ein interaktives Sprachverstehenssystem ergibt sich die Inkrementalität als wichtige Eigenschaft (Menzel, 1994; Menzel, 1996). Leider sind die notwendigen Änderungen umfangreich und bisher nicht weiter untersucht.

[31] Zu ressourcen–adaptivem Verhalten siehe z. B. (Wahlster et al., 1995) und die darin angegebenen Verweise.

4. Eliminatives Parsing mit probabilistischem Wissen

Zu Beginn des Kapitels 2 wurde für eine stochastische Erweiterung von Grammatikmodellen argumentiert. In diesem Kapitel sollen einige stochastische Modelle für das Parsing mit dem in Kapitel 3 eingeführten Grammatiktyp Constraint–Dependenz–Grammatik mit Präferenzen entwickelt und evaluiert werden. Dabei wird die Grammatik selbst aus stochastischen Daten gewonnen. In Abschnitt 4.12 wird als weitere Möglichkeit, statistisches Wissen für CDG nutzbar zu machen, ein Verfahren untersucht, das nicht die Grammatik selbst automatisch extrahiert, sondern nur die Bewertungen für manuell erstellte Constraints findet.

Zu Beginn soll betont werden, daß ein Grammatikmodell nicht zwangsläufig zu einem stochastischen Modell desselben führt; vielmehr sind im allgemeinen eine Vielzahl verschiedener Modelle denkbar. Zum Beispiel führen sowohl Eisner (1996) als auch Collins (1996) stochastische Modelle für Varianten von Abhängigkeitsgrammatiken ein. Insbesondere das letztere der beiden entwickelten Modelle weist einige Ähnlichkeiten zu den hier vorgestellten Modellen auf, die wiederum nur einen Ausschnitt der möglichen Modellvarianten darstellen. Einige der Freiheitsgrade beim Entwurf eines stochastischen Modell sind im folgenden aufgeführt:

- Auswahl der stochastischen Parameter: Welche Informationen werden bei der Definition der Parameter (kleinste Wahrscheinlichkeit tragende Einheiten) herangezogen?
- Annahmen über Vereinfachungen: Welche Voraussetzungen z. B. über die Unabhängigkeit zwischen stochastischen Ereignissen werden durch das Modell gemacht?
- Approximationen: Welche Abschätzungen werden bei der Bestimmung der Parameter und im weiteren Verlauf der Analyse gemacht?

4.1 Motivation von stochastischen Erweiterungen für CDG

In Abschnitt 2.1 wurden einige Probleme bei der Grammatikerstellung, etwa mit dem Aufwand und der Angemessenheit der zugewiesenen Struktur, angesprochen. Stochastische Modelle vermeiden einen Teil dieser Probleme, da sie zum einen — nachdem das Modell erst einmal implementiert ist — nur noch einen sehr geringen Aufwand für den Grammatikschreiber bedeuten; Aufwand an Rechnerzeit und Speicherplatz können schon jetzt (und in Zukunft immer stärker) im Vergleich zu menschlicher Arbeitszeit vernachlässigt werden. Zum anderen weisen stochastische Modelle Strukturen für natürlichsprachliche Äußerungen Präferenzwerte aufgrund von beobachteten Sprachdaten zu. Daraus ergibt sich einerseits, daß stochastische Modelle in der Lage sind, mit unbekannten (oder selten verwendeten) Strukturen und Worten umzugehen und andererseits dennoch die häufiger beobachteten Strukturen[1] zu präferieren. Mit anderen Worten, stochastische Modelle führen zu Robustheit sowohl gegenüber Untergenerierung (aufgrund von unbekannten Strukturen) als auch gegenüber Übergenerierung (aufgrund von hoher Ambiguität).

Für CDG gilt besonders, daß Fehler in der Grammatik schwer erkennbar und damit schwer vermeidbar sind.[2] Für einen Grammatikschreiber ist nur schwer durchschaubar, welches Constraint bestimmte Analysen (u. U. fälschlicherweise) ausschließt, da während der Analyse kaum auf aufschlußreiche Teilergebnisse zugegriffen werden kann. Noch einmal zusätzlich undurchschaubar wird die Situation, wenn die Bewertungen von Constraints zusammenwirken. Stochastische Ansätze können hier helfen, Constraints mit einem Präferenzwert zu versehen.

Als weitere, kognitionswissenschaftliche Motivation ist zu nennen, daß stochastische Erweiterungen auf natürliche Weise die Eigenschaft eines menschlichen Hörers modellieren, die präferierte Analyse einer Äußerung aufgrund der Häufigkeiten von früher wahrgenommenen Analysen auszuwählen. Diese Eigenschaft kommt allen stochastischen Modellen zu, da sie als Grundprinzip die Häufigkeiten in einem Korpus nutzen, das in diesem Fall der Sprachhistorie entspricht, um (Teil–)Analysen eine Bewertung bzw. Wahrscheinlichkeit zuzuweisen. Schömann (1995) gibt einen Überblick über menschliche Verarbeitungsprozesse bei der Disambiguierung. Psycholinguistische Experimente beziehen sich aus Gründen der Praktikabilität jedoch größten Teils auf lexikalische Ambiguitäten. Experimente, die ein constraint–basiertes[3] psycholinguistisches Modell zur Auflösung von syntaktischen Ambiguitäten stützen, werden z. B. von MacDonald (1993), Trueswell, Tanenhaus und Garnsey (1994) und Tabossi und

[1] Es wird implizit bei der Verwendung von stochastischen Modellen die Annahme gemacht, daß häufig auftretende Strukturen mit den 'plausiblen' übereinstimmen.

[2] Die mangelnde Erfahrung mit CDG ist sicher ein Grund, daß hier bisher wenige geeignete Hilfsmittel und Arbeitsweisen entwickelt wurden.

[3] Das Attribut 'constraint–basiert' ist hier allgemeiner als sonst in dieser Arbeit zu verstehen.

Zardon (1993) vorgestellt. Narayanan und Jurafsky (1996) stellen ein probabilistisches Modell des Sprachverstehens vor, das Ergebnisse einiger psycholinguistischer Experimente zu erklären versucht; auf allgemeineren (jedoch nicht allgemein akzeptierten) psycholinguistischen Prinzipien basiert der Vorschlag von Li (1996) zur syntaktischen Disambiguierung. Ein alternatives — nicht auf Wahrscheinlichkeiten beruhendes — Modell, das sogenannte Holzwegmodell (engl. *garden path model*), vertritt Franzier und Rayner (1982).

4.2 Entwurf eines ersten stochastischen Modells

In diesem sowie in den folgenden Abschnitten werden ausschließlich Modelle für syntaktische Abhängigkeitsstrukturen vorgestellt; umfangreichere Modelle, die weitere Repräsentationsebenen wie Semantik, Diskurs u. a. berücksichtigen, sind für die Zukunft vorgesehen.

Wie in Kapitel 2 mehrfach vorgestellt, sind bei der Analyse natürlichsprachlicher Äußerungen mit Hilfe von statistischem Wissen mehrere Komponenten zu unterscheiden.

- Grammatikmodell: Das verwendete Grammatikmodell bestimmt die Art der verwendeten Strukturen wie z. B. Dependenz– sowie Phrasenstrukturbäume oder auch Merkmalsstrukturen mitsamt der verwendeten kleinsten Einheiten wie etwa Beschriftungen, Nichtterminalsymbole oder Attribute. In dieser Arbeit wird aus den oben genannten Gründen ein Dependenzmodell verwendet; Anhang B beschreibt den genauen Aufbau des grammatischen Systems.

- Stochastisches Modell: Das stochastische Modell weist den innerhalb des festgelegten Grammatiksystems möglichen Strukturen eines Satzes eine Wahrscheinlichkeit zu. Bei der Berechnung dieser Wahrscheinlichkeit werden zumeist vereinfachende Annahmen gemacht. In diesem Kapitel werden — von einem einfachen Modell in diesem Abschnitt ausgehend — einige auf Erweiterungen basierende stochastische Modelle beschrieben.

- Bestimmung der Parameter: Die vom stochastischen Modell verwendeten kleinsten Einheiten, die sogenannten Parameter, müssen vor der ersten Analyse bestimmt werden. Da ein Ermitteln dieser Parameter meist nicht möglich ist, weil es entweder keinen global 'richtigen' Wert gibt oder dieser aus praktischen Gründen nicht berechenbar ist, werden diese häufig mit Hilfe eines Korpus abgeschätzt.

- Parser: Der Parser stellt ein Verfahren zur praktischen Berechnung der durch das stochastische Modell und die abgeschätzten Parameter definierten Wahrscheinlichkeiten komplexerer Einheiten sowie zur Bestimmung derjenigen Struktur mit der größten Wahrscheinlichkeit dar. Auch hier kann man zwischen Verfahren unterscheiden, die die Wahrscheinlichkeiten und wahrscheinlichsten Strukturen exakt berechnen, und solchen, die Approximationen derselben finden (siehe z. B. Abschnitte 2.5.2 und 3.5).

Im folgenden wird das erste stochastische Modell mit dem Namen M_{clC} entwickelt.[4] Gegeben ein Satz $\Omega = \omega_1 \ldots \omega_n$ und die durch das grammatische System der CDG gegebene Menge der gültigen Beschriftungen L (vgl. Abschnitt 3.2.1), weist ein stochastisches Modell jedem möglichen Dependenzbaum D des Satzes eine Wahrscheinlichkeit $P(D \mid \Omega)$ zu. Die gesuchte Struktur des Satzes ist derjenige Dependenzbaum mit maximaler Wahrscheinlichkeit.

$$D_{\text{Ziel}} = \arg\max_{D} P(D \mid \Omega) \tag{4.1}$$

Ein Dependenzbaum D eines Satzes $\Omega = \omega_1 \ldots \omega_n$ läßt sich angeben, indem man für jedes Wort ω_i festlegt, welches andere Wort ω_{m_i} es mit welcher Beschriftung l modifiziert.[5] In allen Modellen dieser Arbeit wird davon ausgegangen, daß diese einzelnen Modifikationen unabhängig voneinander sind (vgl. Unabhängigkeitsannahmen in den Abschnitten 2.4.1 und 2.5.1). Hier bedeutet die Unabhängigkeitsannahme keinesfalls, daß das Verfahren generell nicht lexikalisch oder strukturell sensitiv (vgl. Abschnitt 2.2) ist. Vielmehr wird die paarweise Unabhängigkeit zwischen zwei (möglicherweise lexikalisch sensitiven) Unterordnungsbeziehungen angenommen. Auch dies stellt eine starke Vereinfachung dar.

(4.1) viertel vor drei ist auch fuer mich okay (n001k.014.1)

Zum Beispiel sind im Satz 4.1 die Unterordnungen der Worte 'viertel' und 'vor' unter das Wort 'drei' stark korrelliert; diese wechselseitige Information wird durch die Unabhängigkeitsannahme verworfen. Andererseits vereinfacht diese die Berechnung wie folgt:

$$P(D \mid \Omega) \approx \prod_{1 \le i \le n} P(\omega_i, l_i, \omega_{m_i} \mid \Omega) \tag{4.2}$$

Nun wird weiterhin vereinfachend angenommen, daß es ausreicht, statt der Wortformen die Kategorien der betroffenen Worte zu berücksichtigen (vgl. Abschnitt 4.7). Die Berechnung ergibt sich dann wie folgt:

$$P(\omega_i, l_i, \omega_{m_i} \mid \Omega) \approx P(c_i, l_i, c_{m_i} \mid \Omega) \tag{4.3}$$

Damit ist das stochastische Modell M_{clC} angegeben. Die Parameter des Modells stellen die Wahrscheinlichkeiten $P(c_i, l_i, c_{m_i} \mid \Omega)$ dar, die kaum exakt berechnet werden können, da es sich um bedingte Wahrscheinlichkeiten handelt, die nur jeweils für einen speziellen Satz gelten. Für eine Berechnung müßte man alle zu analysierenden Sätze bereits in der Trainingsmenge beobachten können; in diesem hypothetischen Fall wäre eine einfache Speicherung der Strukturen zu allen

[4] Abschnitt 4.3 erklärt die Bedeutung der verwendeten Notation.

[5] Der Einfachhalt halber wird hier der Fall, daß ein Wort die Wurzel des Dependenzbaumes darstellt, nicht betrachtet; siehe Abschnitt 3.2.1.

Sätzen die einfachere und bessere (wenn auch undurchführbare) Lösung. Aus diesem Grund müssen die Parameter (aus einem annotierten Korpus) geschätzt werden. Der Maximum–Likelihood–Schätzer[6] (vgl. Abschnitt 2.4.3) der Wahrscheinlichkeit, daß ein Wort mit der Kategorie c_i ein Wort mit der Kategorie c_j unter der Voraussetzung, daß Worte der Kategorien c_i und c_j in einem Satz vorkommen, mit der Beschriftung l modifiziert, ergibt sich wie folgt:

$$P(l \mid c_i, c_j) \approx L(l \mid c_i, c_j) = \frac{\text{Anzahl Modifikationen von } c_j \text{ durch } c_i \text{ mit } l}{\text{Anzahl Vorkommen } c_i \text{ und } c_j \text{ in einem Satz}} \quad (4.4)$$

Dieser Schätzer kann nun verwendet werden, um die Parameter des Modells zu approximieren.

$$\begin{aligned} P(c_i, l_i, c_{m_i} \mid \Omega) &\approx \frac{P(l_i \mid c_i, c_{m_i})}{\sum\limits_{\substack{1 \le k \le n \\ l \in L}} P(l \mid c_i, c_k)} && (4.5) \\ &\approx \frac{L(l_i \mid c_i, c_{m_i})}{\sum\limits_{\substack{1 \le k \le n \\ l \in L}} L(l \mid c_i, c_k)} && (4.6) \\ &= L(c_i, l_i, c_{m_i} \mid \Omega) && (4.7) \end{aligned}$$

Der Nenner in der Gleichung 4.7 normalisiert die Parameter, so daß sich die Wahrscheinlichkeiten, daß ein Wort einer bestimmten Kategorie überhaupt von einem Wort im Satz modifiziert wird, zu Eins addieren (siehe Gleichung 4.8 und Bemerkungen zu der Realisierung in Abschnitt 4.3).

$$\sum_{\substack{1 \le j \le n \\ l \in L}} P(c_i, l, c_j \mid \Omega) = 1 \quad (4.8)$$

Ein Parser nutzt das stochastische Modell und die abgeschätzten Parameter zur Analyse von natürlichsprachlichen Äußerungen. Dabei findet das in Abschnitt 3.4 vorgestellte Verfahren Verwendung. Dieses Verfahren hat den großen Vorteil, daß es hierbei ohne weitere Probleme möglich ist, neben dem stochastischen Sprachmodell auch noch eine Grammatik zu verwenden, die manuell erstellt wurde; ein solches Vorgehen wurde für die Experimente dieser Arbeit gewählt (siehe Abschnitt 4.3). Es sollte beachtet werden, daß dieses Verfahren nur eine Approximation der wahrscheinlichsten Struktur liefert. Da jedoch eine Anzahl von Vereinfachungen für das gesamte Verfahren gilt, kann dies ohne weitere Einschränkungen hingenommen werden.

[6] Im folgenden werden Maximum–Likelihood–Schätzer durch das Funktionssymbol L von Wahrscheinlichkeiten mit dem Symbol P unterschieden.

4.3 Experimente zur Evaluation von Modellen

Zur Überprüfung der Fähigkeit stochastischer Modelle, natürlichsprachlichen Äußerungen eine bewertete Struktur zuzuweisen, wird eine Reihe von Experimenten durchgeführt. Für die den Experimenten zugrunde liegenden stochastischen Modelle und die Experimente selbst wird eine spezielle Notation, wie z. B. M_{clCd}^{2-19}, eingeführt. Die Bedeutung der Super– und Subskripte wird in diesem Abschnitt erklärt. Als Basis für sprachliches Material findet das in Anhang B beschriebene Korpus Verwendung, das jeweils in Test– und Trainingsmenge unterteilt wird. Um die einzelnen Ergebnisse möglichst einfach vergleichen zu können, wird für sämtliche Experimente derselbe Dialog `n001k` (siehe Anhang B) als Testmenge gewählt. Die Modelle, Versuche und schließlich Ergebnisse unterscheiden sich anhand von drei Kriterien:

- Trainingsmenge: Zur Überprüfung der Hypothese, daß bereits vorgekommene Äußerungen besser analysiert werden können (siehe Abschnitt 4.6), und des Einflusses der Korpusgröße auf das Ergebnis (siehe Abschnitt 4.8) werden die Untersuchungen mit verschiedenen Trainingsmengen durchgeführt. Dabei kommen folgende Teilmengen des Korpus zum Einsatz: das gesamte Korpus (1–19), das Korpus ohne die Testmenge (2–19) sowie zwei echte Teilmengen der letztgenannten (2–9 und 2). Bei der Bezeichnung der Experimente wird die verwendete Trainingsmenge als Superskript vermerkt.

- Stochastische Parameter: Ein wesentliches Kriterium eines stochastischen Modells stellt die Auswahl der stochastischen Parameter dar. Die Parameter des ersten Modells in Abschnitt 4.2 lassen sich als Antwort auf die folgende Frage verstehen:

  > Gegeben die syntaktischen Kategorien des modifizierenden und des modifizierten Wortes, wie groß ist die Wahrscheinlichkeit einer Modifikation mit einer bestimmten Beschriftung?

  Dabei wird die Wahrscheinlichkeit des Parameters von den Bedingungen Kategorie des untergeordneten Wortes, Kategorie des übergeordneten Wortes und Beschriftung abhängig gemacht. Als weitere Bedingungen sind auch die Wortformen, die Anordnung sowie der Abstand der beiden beteiligten Worte u. a. m. möglich. Diese werden in verschiedenen Versuchen variiert und bei der Notation als Subskript vermerkt.[7]

- Bewertungsmaßstäbe: Je nach Sichtweise auf den Parsingprozeß kann man verschiedene Bewertungsmaßstäbe für die Ergebnisse definieren. Im Abschnitt 4.4 werden zwei solche eingeführt und verglichen..

[7] Das Subskript in der Bezeichnung der Modelle setzt sich als Folge von Buchstaben zusammen. Die einzelnen Buchstaben geben an, welche Informationen für die Auswahl des Parameters herangezogen werden. Die Minuskeln 'c' und 'w' stehen für die Kategorie und die Wortform des untergeordneten Wortes, die Majuskeln 'C' und 'W' für die des übergeordneten Wortes, 'l' für die Beschriftung der Modifikationsbeziehung und 'd' für Abstand und Anordnung der beiden Worte.

In den Abschnitten 3.4.3 und insbesondere 3.5 werden Varianten vorgestellt, wie die durch die stochastischen Modelle formulierten partiellen Constraint–Satisfaction–Probleme gelöst werden können. In allen Versuchen dieses Kapitels wird als Auswahlfunktion die in Abschnitt 3.5 als Reihe–Spalte–Verfahren vorgestellte Methode verwendet.[8] In den Experimenten wird das Fortschreiten der Analyse durch eine Erhöhung des Parameters f (siehe Abschnitt 3.5.5) dieser Auswahlfunktion modelliert. Um eine Analyse unter Zeitdruck (Menzel, 1994) zu simulieren, kann dann die Analyse mit einem Wert $f \ll 1$ abgebrochen werden oder — als bessere Varianten — die Analyse mit einem Wert $f \gg 0$ begonnen bzw. mit weniger Zwischenschritten für den Parameter f durchgeführt werden.

In den Experimenten wird zusätzlich zum stochastischen Modell eine sehr einfache Basisgrammatik verwendet, die insbesondere die Wohlgeformtheit der Strukturen sicherstellen (siehe auch Abschnitt 3.2.1 und die dort enthaltene Fußnote 10) sowie nicht durch das stochastische Modell erfaßte Abhängigkeiten abdecken soll. In Abschnitt 4.10 wird näher auf die Basisgrammatik eingegangen; in Anhang D ist sie aufgeführt. Es sollte klar sein, daß durch die Verwendung dieser Basisgrammatik jeder wahrscheinlichkeitstheoretische Anspruch, reale Wahrscheinlichkeiten zu berechnen, zunichte gemacht wird. Da der Status der Bewertungen aufgrund der Abschätzungen, Vereinfachungen und Heuristiken aber ohnehin fragwürdig ist, kann diese Einschränkung hingenommen werden; die zusätzliche Verwendung der Basisgrammatik stellt ein Mittel zur Verbesserung der Resultate — nicht zur Modellierung des Analyseprozesses — dar.

Der Nenner der Gleichung 4.7 für die Abschätzung der Parameter dient der Normalisierung. Durch dieses Vorgehen sind die Schätzungen der Parameter erst zur Analysezeit (und nicht schon zum Zeitpunkt des Trainings) berechenbar, da der Satz Ω bekannt sein muß, bevor der Nenner berechnet werden kann. Aus Gründen der Einfachheit wird deshalb die Normalisierung aufgegeben (siehe (Collins, 1996) für ein ähnliches Vorgehen). Dies stellt eine Vereinfachung dar, da zusätzliche Informationen über den Satzkontext nun nicht mehr berücksichtigt werden. Zur Einschätzung der Maßnahme wurden semiautomatisch Vorexperimente auf den realen Versuchsdaten durchgeführt; dabei ergaben sich bei den Ergebnissen nur minimale Abweichungen.[9] Die kleinen zu erwartenden Abweichungen und die sich ergebende Vereinfachung im Parser lassen das Vorgehen gerechtfertigt erscheinen; aufwendigere Untersuchungen sind für die Zukunft geplant.

[8] Bei der Vorstellung der Methoden in Abschnitt 3.4 werden die Auswahlfunktionen variiert, während die übrigen Parameter konstant gehalten werden.

[9] Diese Vorexperimente erlauben natürlich keine allgemeine Aussage; offensichtlich sind für das gegebene Verfahren die relativen Werte aber wichtiger als die absoluten.

4.4 Bewertungsmaßstäbe

Zur Beurteilung der Leistungsfähigkeit der verschiedenen Modelle können verschiedene Maßstäbe herangezogen werden, vgl. etwa (Krenn und Samuelsson, 1996, Abschnitt 3.1). In dieser Arbeit werden die durch die Gleichungen 4.9 und 4.10 definierten Maße Fehler und Genauigkeit verwendet.[10]

$$\text{Fehler} = \frac{\text{Anzahl falsch eliminierter Unterordnungen}}{\text{Anzahl Worte im Satz}} \tag{4.9}$$

$$\text{Genauigkeit} = \frac{\text{Anzahl Worte im Satz}}{\text{Anzahl nicht eliminierter Unterordnungen}} \tag{4.10}$$

Sowohl der Fehler als auch die Genauigkeit liegen im Normalfall im Intervall $[0, 1]$; im pathologischen Fall der Löschung aller Unterordnungsmöglichkeiten ist die Genauigkeit nicht definiert. Während ein möglichst kleiner Fehler erstrebenswert ist, ist eine große Genauigkeit wünschenswert. Die beiden verwendeten Maße verhalten sich gegenläufig. Ein Fehler von Null läßt sich bei sehr kleiner Genauigkeit ebenso leicht herstellen wie eine große Genauigkeit mit dann allerdings sehr großem Fehler. Realistische Wertepaare liegen zwischen diesen Extremen.

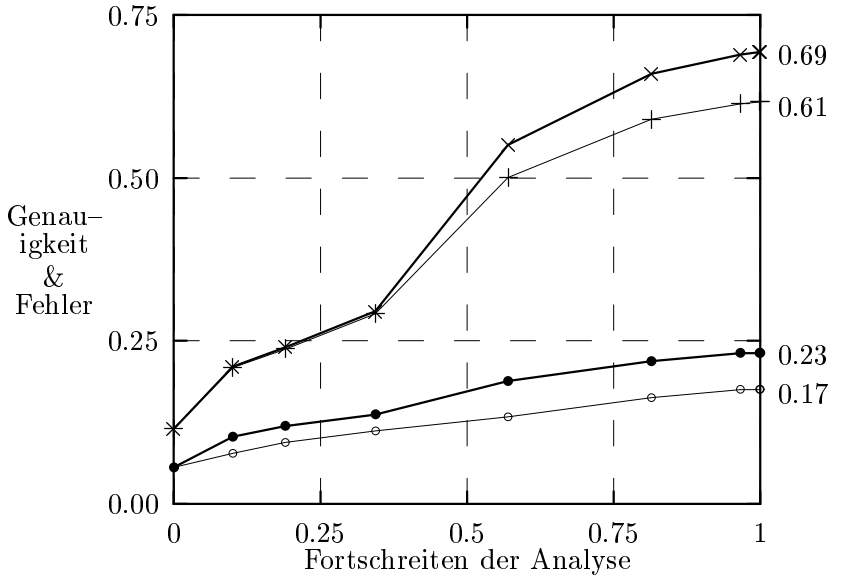


Abbildung 4.1: Bewertungsmaßstäbe mit und ohne Rückbewertung

[10] Das hier mit Genauigkeit bezeichnete Maß wird in der englischsprachigen Literatur auch als 'recall' geführt.

Eine weitere Freiheit für die Maße bestehen bei PCDG bzgl. des Zeitpunktes, an dem die Zahlen für die Berechnung der Maße bestimmt werden. Da die Analyse in mehreren Zyklen abläuft, kann u. U. nicht nur der aktuelle Zustand der Analyse zugegriffen werden, sondern auch die Zustände der letzten Zyklen. Dieser Umstand macht sich insbesondere dann bemerkbar, wenn im Laufe der Analyse alle Unterordnungsmöglichkeiten eliminiert wurden, d. h. alle Lösungen ausgeschlossen wurden. Offensichtlich ist es in diesem Fall günstiger, auf den Stand der Berechnung vor dem aktuellen Zyklus zurückzugreifen, um wenigstens eine Teillösung vorweisen zu können. Dabei ist ein solches Verhalten nicht als Versagen des Systems zu werten: In gewisser Weise wurde das Verfahren gezwungen, die verbleibende Mehrdeutigkeit aufzulösen, ohne daß es das dafür notwendige Wissen besitzt oder ohne daß es sogar Anlaß dafür gibt. Ein Bewerten einer solchen Analyse aufgrund des letzten gültigen Zustandes wird mit *Rückbewertung* bezeichnet. Abbildung 4.1 zeigt exemplarisch jeweils Fehler und Genauigkeit mit und ohne Rückbewertung für das Modell aus Abschnitt 4.5. Da eine Rückbewertung im allgemeinen zum ersten einen kleineren Fehler, zum zweiten eine adäquatere (aber schlechtere) Genauigkeit und zum dritten eine natürlichere Beurteilung der Ergebnisse liefert, wird in folgenden Abbildungen nur der Fehler und die Genauigkeit mit Rückbewertung angegeben.

4.5 Informationen über Abstand und Anordnung

Das einfache stochastische Modell in Abschnitt 4.2 hält für alle möglichen Kategorien von unter– und übergeordnetem Wort und alle möglichen Beschriftungen einen Parameter bereit, der angibt, wie eine solche Unterordnung bewertet werden soll. Dazu sei der folgende Beispielsatz angegeben:

(4.2) in der zweiten oktoberwoche geht es bei mir nur dienstag denn ab mittwoch bin ich auf einer konferenz in berlin (n003k.002.1)

Der Beispielsatz 4.2 enthält vier Präpositionen, die syntaktisch von einem Nomen mit der Beschriftung '*pn*' abhängen:[11] 'in' zu 'oktoberwoche', 'ab' zu 'mittwoch', 'auf' zu 'konferenz' und 'in' zu 'berlin'.[12] Diesen vier Unterordnungsbeziehungen wird — da sie sich nicht in den für die Parameter relevanten Attributen unterscheiden — dieselbe (hohe) Bewertung zugeordnet. Neben diesen intuitiv durchaus in eine Klasse fallenden Wortpaaren werden jedoch auch sämtliche andere Kombinationen aus einer der vier Präpositionen und einem der vier Nomen gleich bewertet. Zum Beispiel erscheint dem stochastischen Modell eine Unterordnung der Präposition 'in' unter das Nomen 'berlin' genauso wahrscheinlich wie Unterordnungen unter die Nomen 'oktoberwoche', 'dienstag', 'mittwoch' und 'konferenz'.

Offensichtlich mangelt es dem Modell an Anordnungs– und Abstandswissen: Daß sich Präpositionen oft auf Nomen beziehen, die in unmittelbarer Nähe

[11] Zum syntaktischen System siehe Anhang B.4.
[12] Die Präposition 'bei' modifiziert das Pronomen 'mir'.

rechts von ihnen stehen, kann nicht berücksichtigt werden. Um diesen Mangel zu beseitigen, wird das Modell aus Abschnitt 4.2 um das diskrete, im folgenden definierte Distanzmaß angereichert. Die neuen Parameter beziehen neben den Kategorien und der Beschriftung auch dieses Distanzmaß mit ein. Die *Distanz* $d(i,j)$ zweier Worte ω_i und ω_j eines Satzes $\Omega = \omega_1 \ldots \omega_n$ wird durch die Gleichung 4.11 bestimmt.

$$d(i,j) = \begin{cases} -2 & \text{gdw.} \quad n < i-j \leq -3 \\ -1 & \text{gdw.} \quad -2 \leq i-j \leq -1 \\ 1 & \text{gdw.} \quad 1 \leq i-j \leq 2 \\ 2 & \text{gdw.} \quad 3 \leq i-j < n \end{cases} \tag{4.11}$$

Theoretisch vervierfacht sich die Anzahl der Parameter durch die Einführung des Distanzmaßes; in der Praxis ist damit zu rechnen, daß dieser schlimmste Fall nicht erreicht wird, da einige Kombinationen gar nicht vorkommen und deshalb unberücksichtigt bleiben.

Abbildung 4.2 stellt die Ergebnisse der Versuche mit und ohne Berücksichtigung des Distanzmaßes gegenüber.

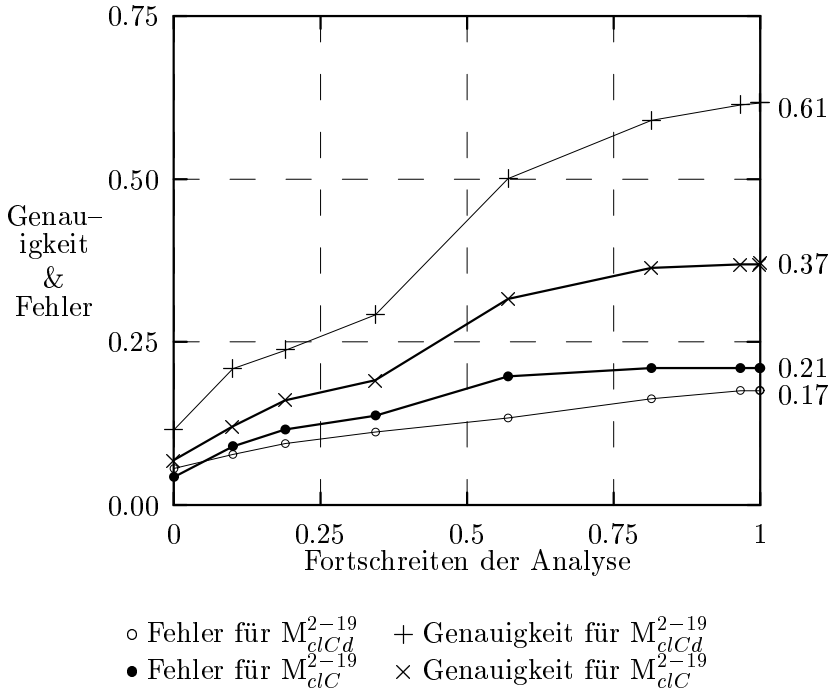


Abbildung 4.2: Einfluß eines Distanzmaßes (E_2 vs. E_7)

How impressive!
—D. Kern

Die Ergebnisse belegen eindrucksvoll, wie wichtig ein Distanzmaß für stochastische Modelle ist: Die Genauigkeit verbessert sich um 40% bei gleichzeitiger Verringerung des Fehlers um 20%.

4.6 Erkennt das Modell seine Trainingsmenge?

Obwohl damit kaum die wirkliche Leistungsfähigkeit eines Modells nachgewiesen werden kann, ist es doch interessant zu überprüfen, wie sich das Modell verhält, wenn man die Trainings– und die Testmenge identifiziert. Auf diese Weise läßt sich feststellen, ob das System die Forderung, bereits beobachtete Daten besser analysieren zu können, erfüllt (vgl. Abschnitt 4.1).

Zu diesem Zweck wurde das Modell M_{clCd} mit dem Dialog `n001k` trainiert und — wie bei allen anderen Experimenten — auch auf diesem getestet. Abbildung 4.3 zeigt die Ergebnisse sowie die Ergebnisse für disjunkte Test– und Trainingsmengen im Vergleich.

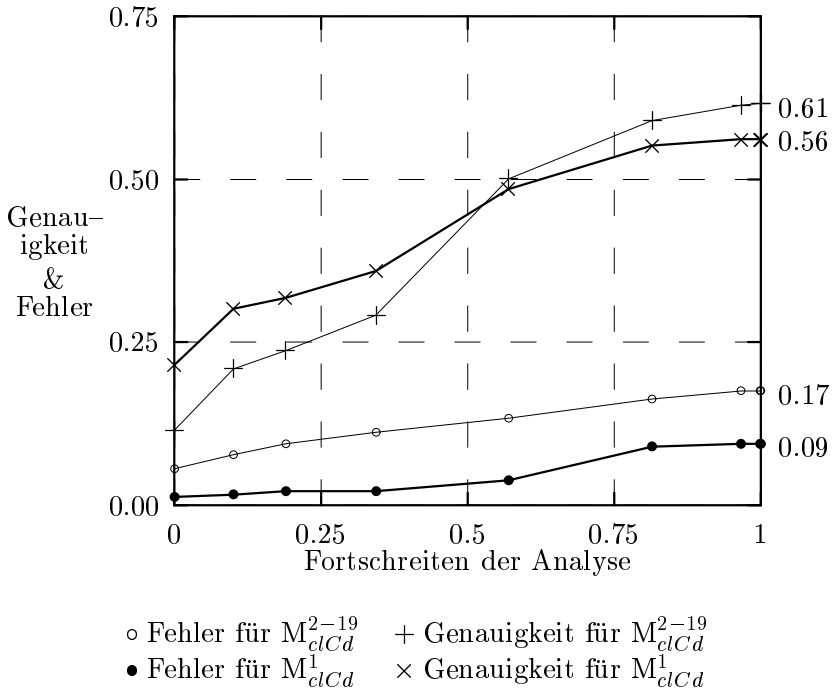


Abbildung 4.3: Identifikation von Test– und Trainingsmenge (E_2 vs. E_3)

Während sich die Genauigkeit sogar etwas verschlechtert, zeigt der Fehler die erwarteten wesentlich besseren Werte. Das Modell analysiert also bekannte Äußerungen besser als unbekannte. Aber wie kommt es dabei zu dem nicht zu vernachlässigenden Fehler von etwa 9%? Die Klasseneinteilung, die das Modell vornimmt, ist trotz des Distanzmaßes relativ grob. Das heißt, das Modell kann feinere Unterschiede zwischen zwei ähnlichen, aber nicht identischen Fällen nicht berücksichtigen. Diese Eigenschaft verhindert ein Memorieren der Testmenge (vgl. ähnliche Betrachtungen bei einem feineren Modell am Ende des Abschnitts 4.7) und führt schließlich zu Fehlern.

4.7 Lexikalische Sensitivität

Die Berücksichtigung von lexikalischer Information bzw. genauer von Lexeminformation wird als für stochastische Ansätze essentiell angesehen (Eisner, 1996; Lafferty, Sleator und Temperly, 1992; Black et al., 1992; Collins, 1996; Magerman, 1995b). Diese Forderung sei an einem Beispiel veranschaulicht:

(4.3) dann sehen wir uns am mittwoch in ihrem buero (n017k.012.1, gekürzt)

(4.4) dann sehen wir uns am mittwoch in dieser woche (konstruiert)

Die Sätze 4.3 und 4.4 unterscheiden sich lediglich durch die Lexeme (nicht aber durch die Kategorien) der satzfinalen Nominalgruppe. Dieser Unterschied reicht aber offensichtlich aus, um die präferierte strukturelle Angliederung der Nominalgruppe vom Verb im Satz 4.3 zum vorangehenden Nomen im Satz 4.4 zu ändern (siehe Abbildung 4.4).

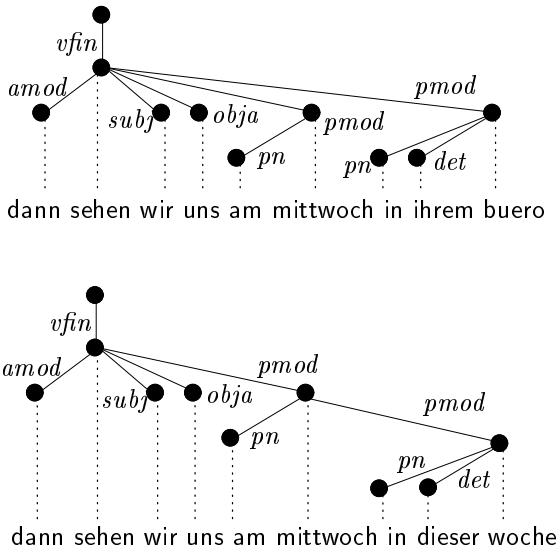


Abbildung 4.4: Verschiedene präferierte Strukturen für sich nur in einzelnen Lexemen unterscheidenden Sätzen

Eine Entscheidung über die Anbindung der Nominalgruppe erfordert demnach mehr als Informationen über die betroffenen Kategorien, in diesem Beispiel etwa Wissen über die beteiligten Lexeme. Relevantes Wissen läßt sich natürlichsprachlich z. B. wie folgt formulieren:

- Ein 'Mittwoch' befindet sich normalerweise nicht in einem 'Büro'.
- Ein genauer 'Mittwoch' läßt sich hervorragend durch die Angabe der entsprechenden 'Woche' spezifizieren.[13]

[13] Die Beschreibung in diesem Absatz ist natürlich vereinfachend. Zwar ist es richtig, daß

Um diesem Sachverhalt Rechnung zu tragen, wird das stochastische Modell aus Abschnitt 4.5 so abgewandelt, daß es bei der Bewertung einer Unterordnungsmöglichkeit nicht auf die Kategorien, sondern auf die Lexeme zurückgreift. Es erstellt also je einen Parameter pro Lexempaar, Beschriftung und Distanz. Abbildung 4.5 stellt die Ergebnisse des alten Modells dem neuen, abgewandelten Modell gegenüber.

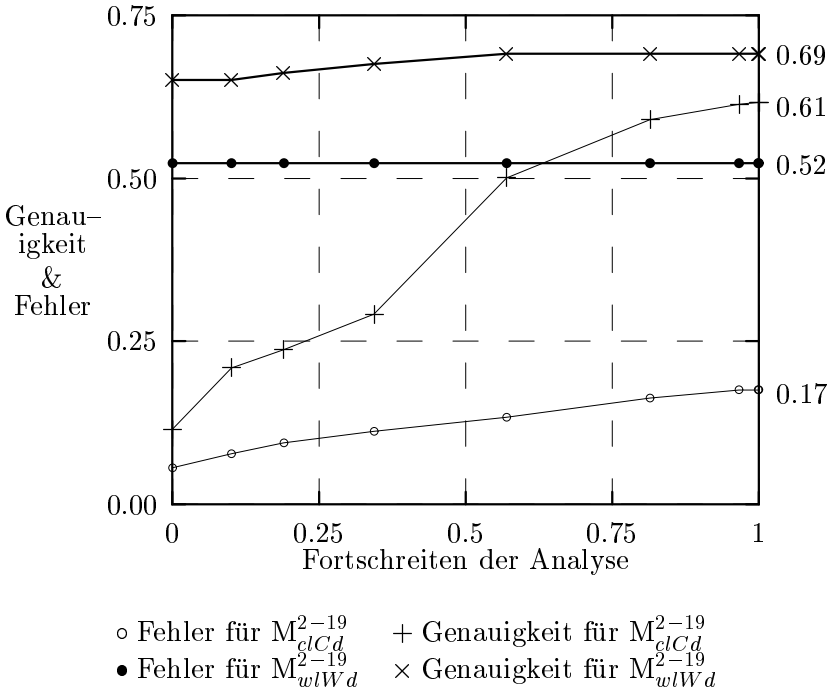


Abbildung 4.5: Ergebnisse eines 'feineren' Modells (E_2 vs. E_9)

Das Resultat ist ernüchternd. Die Ergebnisse des neuen Modells sind mit einer Fehlerrate von über 50% unbrauchbar. Warum erreicht das an sich bessere Modell derartig schlechte Resultate? Einen Hinweis findet man, wenn man sich anschaut, wie sich Fehler und Genauigkeit im Verlauf der Analyse entwickeln. Abbildung 4.5 zeigt für das neue Modell, daß der Fehler während der gesamten Analyse konstant (hoch) bleibt und die Genauigkeit nur um vier Prozentpunkte steigt. Das bedeutet, daß die eigentliche Analyse fast nicht zum Ergebnis

die Lexeme (genaugenommen auch die Aussprache, Intonation und der Kontext) einem Hörer die besten zur Verfügung stehenden Hinweise geben kann, wie ein Wort zu interpretieren ist und wie es strukturell in einem Satz eingebunden ist. Allerdings verbietet der damit verbundene Aufwand eine Berücksichtigung aller Informationen. Vielmehr wird man durch z. B. semantische Klassenbildung analog zur syntaktischen Kategoriebildung versuchen, Verallgemeinerungen über den Wortformen zu finden, und diese dann zur Disambiguierung einsetzen. Den präferierten Kontext eines Wortes nennt man dann selektionale Restriktion. Das oben angegebene Wissen könnte man auch allgemeiner wie folgt formulieren:

- Temporale Angaben werden normalerweise nicht bzgl. ihrer Lokation spezifiziert.
- Ein temporaler Ausdruck läßt sich durch weitere temporale Angaben weiter einschränken.

beiträgt.

Diese paradoxe Situation läßt sich durch einen Blick auf die geschätzten Parameter leicht auflösen. Die Trainingsmenge enthält genügend Daten, daß ca. 1000 Parameter des Modells wenigstens einmal auftauchen. Die Schätzung von 94% dieser Parameter beruht allerdings auf nur einer Beobachtung. Schätzungen, die auf einer geringen Anzahl von Beobachtungen basieren, müssen eigentlich zurückgewiesen werden; das Modell würde allerdings noch schlechter abschneiden, da es nicht geschätzte Parameter mit Null bewertet (siehe auch Abschnitt 4.9). Als Folgerung aus dieser Beobachtung läßt sich feststellen, daß die Datenbasis offensichtlich für das neue Modell zu klein ist.[14]

Wenn die Gründe für das schlechte Abschneiden des neuen Modells tatsächlich in der für die große Anzahl an Parametern ungeeigneten Datenbasis liegen, stellen sich folgende Fragen:

- Müßten sich die Ergebnisse nicht wesentlich verbessern, wenn man dafür sorgen könnte, daß die erforderlichen Parameter (zumindest näherungsweise) existieren?
- Wenn das Modell aus Abschnitt 4.5 im Gegensatz zum neuen, feineren Modell gute Ergebnisse liefert, lassen sich Modelle finden, die die Parameter feiner über den Raum der möglichen Unterordnungsbeziehungen verteilen und gleichzeitig noch genügend gut abschätzen?

Im folgenden werden modifizierte Experimente zur Überprüfung der Hypothesen durchgeführt; die Ergebnisse sind in Abbildung 4.6 gegenübergestellt.

Die Qualität der Parameter läßt sich — wie in der ersten Frage gefordert — für eine bestimmte Testmenge verbessern, wenn man diese Testmenge in die Trainingsmenge einbezieht (siehe Abschnitt 4.6).[15] Das Modell M_{wlWd}^{1-19} zeigt das erwartete Verhalten: nahezu fehlerlose Disambiguierung bei akzeptabler Genauigkeit. Bei diesem hervorragenden Ergebnis darf nicht vergessen werden, daß durch das kleine Korpus einerseits und das Trainieren des Modells mit der Testmenge andererseits ein Effekt erreicht wird, den es bei realer Evaluation möglichst zu vermeiden gilt: das Memorieren der Testmenge durch das System. Hier wirkt sich das kleine Trainingskorpus verstärkend aus, da die nivellierenden Effekte gering sind.

Die zweite Frage oben führt zu einem modifizierten Modell bei konstant gehaltener Trainingsmenge. Da das Modell M_{wlWd} offensichtlich eine für das Korpus zu feine Klassenbildung durchführt, bietet es sich an, nach einem Modell zu

[14] Eine Auszählung von sowohl in der Trainings– als auch in der Testmenge vorkommenden Wortpaaren ergibt, daß nur etwas mehr als die Hälfte der Paare aus der Testmenge auch in der Trainingsmenge vorkommen. Bei diesen Voraussetzungen muß der Fehler bei mindestens knapp 50% liegen. Der erreichte Fehler von 'nur' 56% zeigt, daß ein Großteil der Fehler tatsächlich auf das beschriebene Problem zurückgeht.

[15] Wie bereits in Abschnitt 4.6 bemerkt, läßt sich mit einem derartig trainierten Modell nicht dessen wirkliche Leistungsfähigkeit überprüfen.

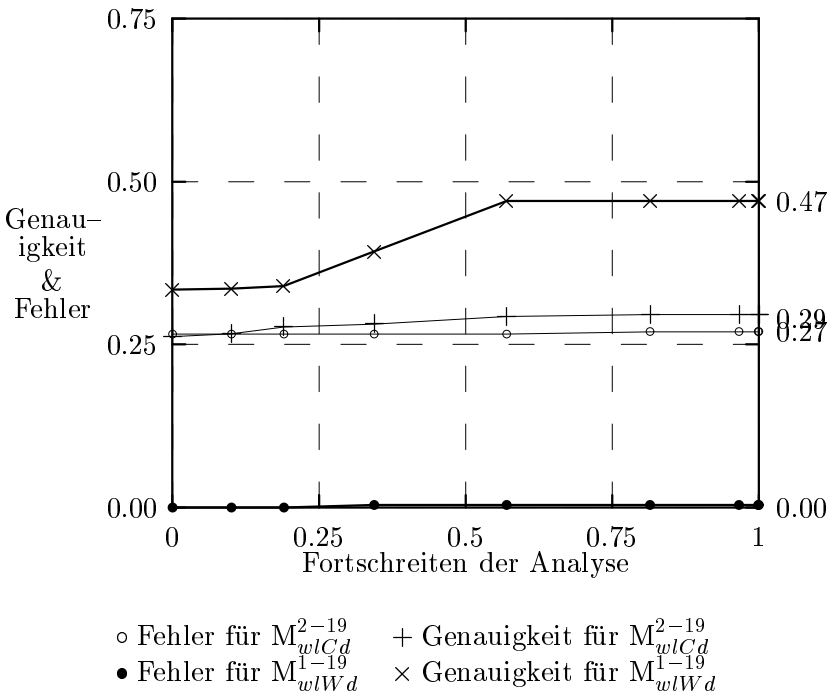


Abbildung 4.6: Einfluß der Anzahl der Parameter auf Fehler und Genauigkeit (E_8 vs. E_{14})

suchen, das eine gröbere Einteilung als dieses und eine feinere als das Modell M_{clCd} vornimmt. Die beiden Modelle M_{wlCd} und M_{clWd} erfüllen diese Forderung; für das erste der beiden sind die Ergebnisse ebenfalls in Abbildung 4.6 aufgeführt. Dabei wird deutlich, daß die Ergebnisse schon besser sind als die des Modells M_{wlWd}; absolut betrachtet sind sie allerdings immer noch unbefriedigend.

Evaluationen mit größeren Korpora sollten für dieses Modell ebenfalls wesentlich bessere Ergebnisse liefern.

4.8 Abhängigkeit von der Korpusgröße

Das Modell M_{clCd} verwendet bei c Kategorien und l Beschriftungen $c^2 \cdot l \cdot 4$ Parameter. Für das verwendete syntaktische System (vgl. Anhang B.4) ergeben sich also $41^2 \cdot 17 \cdot 4 = 114308$ Parameter. Auch unter der Annahme, daß nicht alle diese Parameter relevant sind, also auch nicht ermittelt werden müssen, ist es leicht einsehbar, daß es ein großes Problem darstellt, diese große Anzahl an Werten abzuschätzen.[16] Wie fast alle anderen stochastischen Ansätze im Bereich der Verarbeitung von natürlicher Sprache hat auch der in dieser Arbeit verfolgte Ansatz das Problem, das die zur Verfügung stehenden Korpora kaum ausreichen, die Parameter gut abzuschätzen.

[16] Siehe hierzu insbesondere auch die Betrachtungen im Abschnitt 4.7.

Um zu untersuchen, welchen Einfluß die Korpusgröße auf die Ergebnisse hat, werden zwei Experimente für das Modell M_{clCd} mit jeweils verkleinerter Trainingsmenge durchgeführt. Einmal wird die Trainingsmenge auf die Dialoge n002k bis n009k eingeschränkt, ein zweites Mal nur der Dialog n002k als Trainingsmaterial verwendet. Die Ergebnisse in Abbildung 4.7 sind direkt mit denen des Modells M^{2-19}_{clCd} aus Abbildung 4.3 vergleichbar, da sich die Versuche, wie hier beabsichtigt, nur in der verwendeten Trainingsmenge unterscheiden.

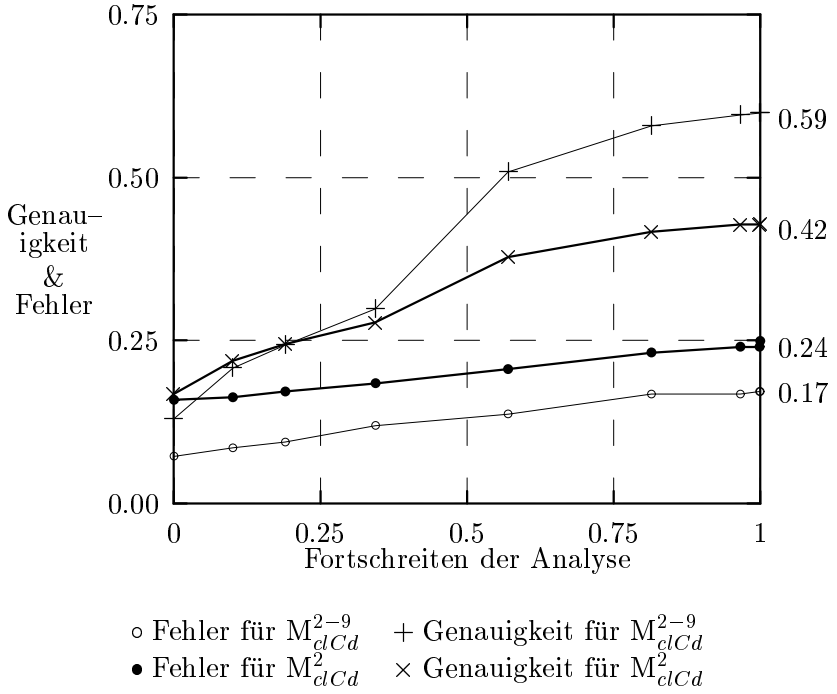


Abbildung 4.7: Einfluß der Korpusgröße auf Fehler und Genauigkeit (E_{17} vs. E_{18})

Als wichtiges Ergebnis läßt sich feststellen, daß die beiden Versuche mit den Trainingsmengen 2–19 und 2–9 nahezu identische Ergebnisse liefern. Die drei zusätzlichen Dialoge n011k, n017k und n019k in der Trainingsmenge führen zu keinem großen Gewinn bzgl. des Fehlers oder der Genauigkeit. Welches die genauen Gründe dafür sind, kann nur durch zusätzlich durchzuführende Experimente[17] überprüft werden. Ohne zusätzliche Evidenzen kann weder gefolgert werden, daß tatsächlich alle relevanten (und durch dieses spezielle Korpus abschätzbaren) Parameter bereits durch die kleinere Trainingsmenge ausreichend genau bestimmt werden, noch kann ausgeschlossen werden, daß es sich bei dem Ergebnis um eine nicht repräsentative Koinzidenz handelt.

Der Vergleich mit den Ergebnissen des Experiments mit dem Dialog n002k als Trainingsmenge zeigt die erwarteten Resultate: Sowohl Fehler als auch Genauigkeit verschlechtern sich (um etwa ein Drittel). Der einzelne Dialog n002k

[17] Oder aber durch einen manuellen Vergleich der beiden Statistiken.

ist offensichtlich nicht geeignet, die Parameter des Modells genau genug abzuschätzen.

4.9 Rückfall– und Glättungstechniken

In den letzten Abschnitten ist deutlich geworden, daß der Datenmangel bei der Schätzung der Parameter ein (allgemeines) Problem darstellt. Verschiedene Standardtechniken sind entwickelt worden, die die Auswirkungen dieses Problems gering zu halten versuchen. Die Hauptidee bei diesen Verfahren ist es, auf gröber verteilte und deshalb besser schätzbare Parameter zurückzugreifen, ohne die gut abschätzbaren feineren Parameter zu verwerfen.[18] Im folgenden werden die beiden wichtigsten Verfahren beschrieben; eines wird experimentell auf das Parsing mit PCDG angewendet.

4.9.1 Rückfall auf gröbere Parameter

Die verschiedenen Modelle vereinfachen die Berechnung der Parameter auf verschiedene Weisen. Die wichtigste Vereinfachung stellt die Auswahl der stochastischen Parameter, die Gleichung 4.12 für einige Modelle zeigt (vgl. Abschnitt 4.2 und 4.3), selbst dar.

$$P(\omega_i, l_i, \omega_{m_i} \,|\, \Omega) \approx \begin{cases} L(c_i, l_i, c_{m_i} \,|\, \Omega) & \text{für } \mathrm{M}_{clC} \\ L(c_i, l_i, c_{m_i}, d(i, m_i) \,|\, \Omega) & \text{für } \mathrm{M}_{clCd} \\ L(c_i, l_i, \omega_{m_i}, d(i, m_i) \,|\, \Omega) & \text{für } \mathrm{M}_{clWd} \\ L(\omega_i, l_i, c_{m_i}, d(i, m_i) \,|\, \Omega) & \text{für } \mathrm{M}_{wlCd} \\ L(\omega_i, l_i, \omega_{m_i}, d(i, m_i) \,|\, \Omega) & \text{für } \mathrm{M}_{wlWd} \end{cases} \tag{4.12}$$

Dabei wird die Verteilung der Parameter über den Raum der Elementarwahrscheinlichkeiten beim Übergang vom Modell M_{clC} zu Modell M_{wlWd} immer feiner, d. h., es sind auch zunehmend mehr Parameter abzuschätzen. Für das Modell M_{wlWd} kumulierte die Schwierigkeit, diese große Anzahl an Parametern abzuschätzen, in der Feststellung (siehe Abschnitt 4.7), daß der Datenmangel eine brauchbare Abschätzung unmöglich macht.

Die Technik des Rückfalls (engl. *back–off*) basiert auf der simplen Idee, die feineren Parameter nur dann zu verwenden, wenn genügend Daten für eine Abschätzung vorhanden sind, und sonst auf gröbere Parameter zurückzugreifen. Die letzten Abschnitte haben z. B. gezeigt, daß das Modell M_{clCd} relativ stabile Ergebnisse erreicht, während das Modell M_{wlWd} den Raum der Elementarwahrscheinlichkeiten sehr fein einteilt, so daß sich ein Rückfall auf die gröberen Parameter für ein Experiment anbietet. Gleichung 4.13 formalisiert dieses Vorgehen.

[18] Siehe etwa (Collins und Brooks, 1995; Collins, 1996) für Beispiele.

$$P(\omega_i, l_i, \omega_{m_i} \mid \Omega) \approx \begin{cases} L(\omega_i, l_i, \omega_{m_i}, d(i, m_i) \mid \Omega) & \text{falls existent} \\ L(c_i, l_i, c_{m_i}, d(i, m_i) \mid \Omega) & \text{sonst} \end{cases} \tag{4.13}$$

Selbstverständlich sind auch umfangreichere Rückfalltechniken denkbar, etwa ein kaskadierender Rückfall mit den fünf in Gleichung 4.12 angegebenen Modellen.

Die Ergebnisse des Experiments E_{20} (siehe Abbildung 4.8) sind unbefriedigend. Zwar ist der erreichte Fehler etwas kleiner als die Fehler der Experimente für die einzelnen stochastischen Modelle, die Genauigkeit jedoch signifikant schlechter. Um zu überprüfen, ob die verbleibende Ambiguität durch andere Auswahlfunktionen weiter vermindert werden kann, wird ein weiteres Experiment E_{22} durchgeführt. Dabei wird zunächst nach dem Reihe/Spalte–Kriterium disambiguiert, solange Ambiguität mit dieser Auswahlfunktion aufgelöst werden kann. Daraufhin wird das Kriterium der kleinsten Quadrate verwendet, um evtl. verbleibende Ungenauigkeiten aufzulösen (siehe Abschnitt 3.4.3). Abbildung 4.8 zeigt aber, daß auch mit dieser erweiterten Auswahlfunktion nicht die Ergebnisse des einfachsten Modells erreicht werden.

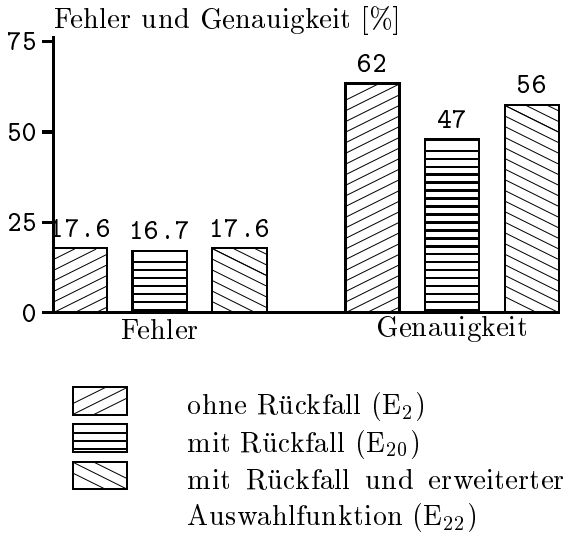


Abbildung 4.8: Ergebnisse mit und ohne Rückfalltechniken

Es lassen sich einige Vermutungen über die Gründe für das schlechte Abschneiden anstellen. Zunächst einmal findet eine einfache Variante des Rückfalls Verwendung, da zum einen nur ein Rückfall auf *eine* weitere Ebene, also keine kaskadierte Berechnung, stattfindet. Zum anderen ist das Kriterium für den Rückfall relativ schwach; der Maximum–Likelihood–Schätzer gilt bereits als existent, wenn nur eine einzige Beobachtung stattgefunden hat. Normalerweise wird diese Grenze wesentlich höher angesetzt. Hier hätte eine höhere Grenze den Rückfall in praktisch allen Fällen erzwungen (siehe Bemerkungen zum Datenmangel in Abschnitt 4.7) und damit dem Experiment jede Aussagekraft genommen.

4.9.2 Glättung von Parametern

Das Rückfallverfahren basiert auf der Idee, daß gröbere Parameter, wenn sie denn gut geschätzt werden können, eine annehmbare Annährung an feinere Parameter darstellen. Glättungsverfahren (engl. *smoothing, deleted interpolation*) (Charniak, 1993; Magerman, 1995a) gehen ebenfalls auf diesen Zusammenhang zurück, setzen ihn jedoch konsequenter um. Beim Glätten werden zur Approximation eines Parameters *immer* die Schätzungen für feinere und gröbere Parameter kombiniert. Gleichung 4.15 formalisiert diesen Gedanken exemplarisch für die hier vorgestellten Modelle.[19]

$$\begin{aligned} P(\omega_i, l_i, \omega_{m_i} \,|\, \Omega) \quad \approx \quad & \lambda_1 \cdot L(\omega_i, l_i, \omega_{m_i}, d(i, m_i) \,|\, \Omega) + \\ & \lambda_2 \cdot L(\omega_i, l_i, c_{m_i}, d(i, m_i) \,|\, \Omega) + \\ & \lambda_3 \cdot L(c_i, l_i, \omega_{m_i}, d(i, m_i) \,|\, \Omega) + \\ & \lambda_4 \cdot L(c_i, l_i, c_{m_i}, d(i, m_i) \,|\, \Omega) + \\ & \lambda_5 \cdot L(c_i, l_i, c_{m_i} \,|\, \Omega) \end{aligned} \tag{4.14}$$

$$\text{mit} \sum_{i=1}^{5} \lambda_i = 1 \tag{4.15}$$

Weitere Experimente mit ausgefeilteren Rückfall– und Glättungstechniken sind für die Zukunft besonders erstrebenswert.

4.10 Parsing ohne Statistik oder ohne Grammatik

In Abschnitt 4.3 wird beschrieben, daß für die durchgeführten Experimente sowohl eine Statistik (für das stochastische Modell) als auch eine Basisgrammatik (für Wohlgeformtheit und nicht durch das stochastische Modell erfaßte Abhängigkeiten) verwendet werden. Dabei stellt sich die Frage, welchen Anteil die beiden Komponenten jeweils einzeln beitragen. Aus diesem Grunde wird untersucht, was eine Statistik ohne Basisgrammatik leistet. Abbildung 4.9 stellt die Ergebnisse für das Modell M_{clCd}^{2-19} mit und ohne Basisgrammatik gegenüber.

Es fällt auf, daß sich die Fehlerraten nicht stark unterscheiden, während die Genauigkeit ohne die Basisgrammatik einen Einbruch erfährt. Dieses Verhalten ist bei der gegebenen Versuchsanordnung zu erwarten gewesen. Die Basisgrammatik enthält nur Constraints, die für das gesamte Korpus (von Ausnahmen einmal abgesehen) gültig sind; deshalb sind die aufgetretenen Fehler ausschließlich auf die Statistik zurückzuführen. Entweder sind die Parameter ungenau oder das stochastische Modell ist nicht fein genug. Der trotzdem vorhandene Unterschied

[19] Die einzelnen Gewichte λ_i können in einer erweiterten Variante auch Funktionen sein, so daß die Gewichte für Worte, Kategorien etc. individuell verteilt sein können.

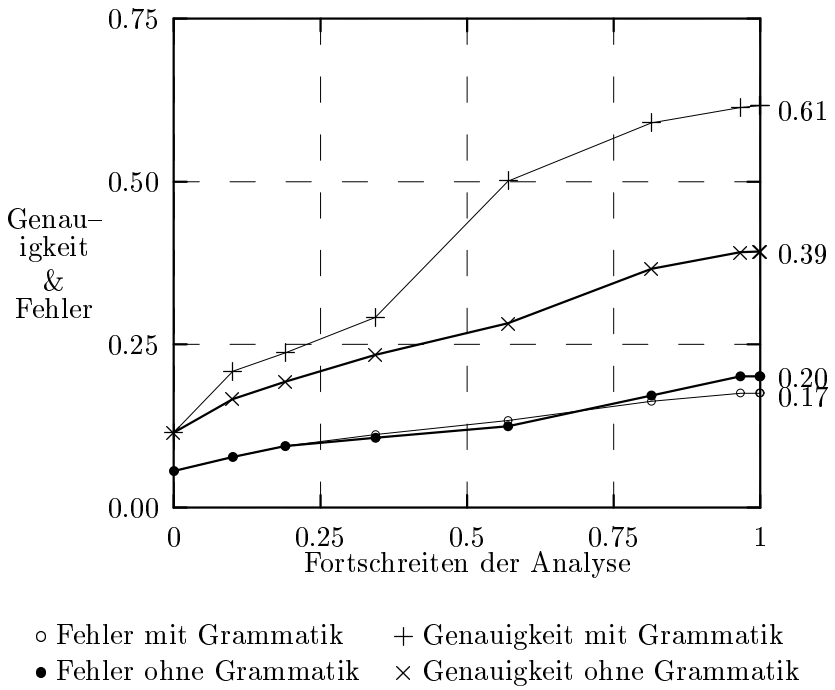


Abbildung 4.9: Einfluß der Basisgrammatik (E_2 vs. E_{16})

in der Fehlerrate läßt sich auf das (bei den obigen Überlegungen nicht berücksichtigte) Zusammenspiel von Statistik und Basisgrammatik zurückführen, das in diesem Fall offensichtlich für eine weitere Verringerung des Fehlers sorgt.

Im Gegensatz dazu wird die Genauigkeit durch die Basisgrammatik wirklich verbessert, da die Basisgrammatik grammatisches Wissen kodiert, das durch das stochastische Modell nicht abgedeckt wird. Durch die Basisgrammatik wird eine große Anzahl an Varianten ausgeschlossen, über die das stochastische Modell keine Aussage treffen kann.

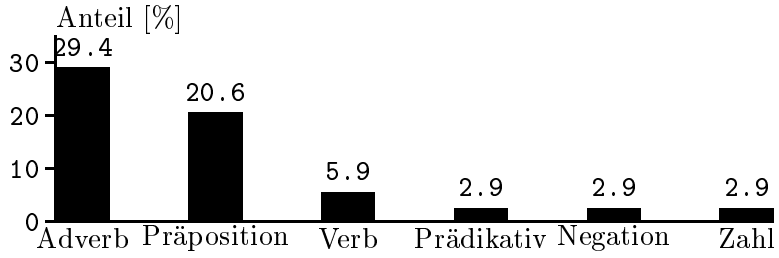
Wenn die Basisgrammatik offensichtlich geeignet ist, einen Teil der Disambiguierung durchzuführen, könnte man vermuten, die Grammatik allein sei zur Analyse fähig. Dies ist nicht der Fall. Experimente mit der Basisgrammatik und ohne eine Statistik führen zu keinem Resultat, da die Basisgrammatik allein nicht genügend disambiguiert, um überhaupt von einem sinnvollen Ergebnis sprechen zu können. In gewisser Weise entspricht dies dem in Abschnitt 4.4 erwähnten wertlosen Extremum mit einer sehr geringen Genauigkeit und einem kleinen Fehler. Natürlich bedeutet dies nicht, daß es keine manuell erstellte, stark disambiguierende Grammatik geben kann.[20] Vielmehr sollte in diesem Abschnitt der Einfluß der in den Experimenten verwendeten Grammatik überprüft werden.

[20] Eine gute Grammatik bedürfte aber wahrscheinlich wieder eines hohen Aufwandes und besäße die in Abschnitt 2.1 diskutierten Nachteile.

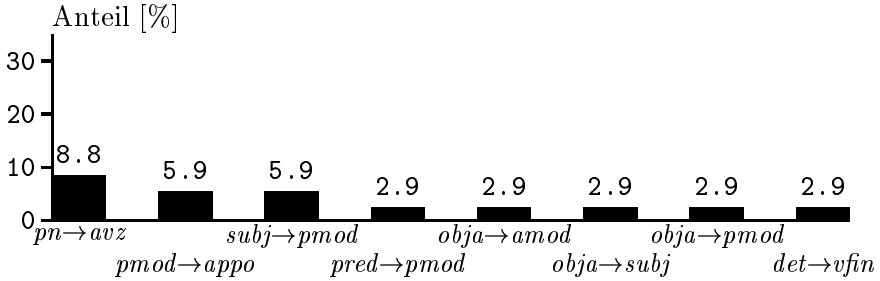
4.11 Fehleranalyse

Welche Art von Fehler werden von den Modellen gemacht? In Abschnitt 4.7 wird zwar ein Beispiel für einen realen Fehler gegeben, eine systematische Fehleranalyse ist aber grundsätzlich äußerst schwierig, da es im allgemeinen nicht möglich ist, einzelne Constraints, Constraint–Bewertungen oder Heuristiken für ein bestimmtes Resultat verantwortlich zu machen. Es liegt in der Grundidee der CDG mit Präferenzen begründet, daß eben das Zusammenspiel der einzelnen Elemente ein Verhalten hervorruft.

Beobachten lassen sich allerdings die verschiedenen Fehlerklassen und deren Häufigkeiten. Für einige solche Klassen lassen sich Vermutungen über ihre Ursachen anstellen, und diese sind z. T. — nach eingehender Überprüfung — zu beseitigen. Für das Experiment M_{clCd}^{2-19} verteilen sich die Fehler gemäß Abbildung 4.10. Im oberen Teil sind die Fehler aufgeführt, bei denen lediglich das übergeordnete Wort, d. h. die strukturelle Anbindung, falsch bestimmt wurde; im unteren Teil stimmt dagegen schon die gefundene Beschriftung, also die Funktion, nicht.



(a) Falsche Zuordnungen (insgesamt 64.7%)



(b) Falsche Beschriftungen und z. T. Zuordnungen (insgesamt 35.3%)

Abbildung 4.10: Fehlerverteilung

Die falsche Unterordnung von Adverben ist für etwa ein Drittel der Fehler verantwortlich. Dafür lassen sich eine Anzahl von Gründen finden, die im folgenden angesprochen werden sollen. Zunächst kann auch der menschliche Hörer die Adverben schwierig eindeutig unter ein bestimmtes Wort einordnen; oft fungieren sie als eine Art Satzpartikel und nicht als Modifikation eines Verbs (siehe auch Bemerkungen zu dieser Problematik auf Seite 104 im Anhang B.4).

Diese Zuordnungsschwierigkeit bedeutet eine prinzipielle Unsicherheit für statistische Verfahren. Da es keine regelhafte Beziehung gibt, die trainiert werden könnte, müssen sämtliche statistische Verfahren, die dieses inkonsistente Korpus verwenden, fehlerhaft arbeiten. Eine andere Sichtweise auf das Problem würde mehr als eine Struktur als korrekt akzeptieren, da auch ein menschlicher Hörer kaum in der Lage ist, die Struktur eindeutig zu bestimmen.[21] Diese Art der Fehlerbetrachtung ist in dem Kontext dieser Arbeit schwierig zu integrieren und auch in der Literatur nicht zu finden. Ein weiterer Grund für den hohen Fehleranteil ist, daß oft nicht nur syntaktische (sondern z. B. auch semantische) Kriterien über die korrekte Unterordnung entscheiden.

(4.5) am dienstag hätte ich noch einen termin frei (n001k.002.1, gekürzt)

Im Beispielsatz 4.5 kann das Adverb 'noch' sowohl das finite Verb — im Sinne einer temporalen Einschränkung — als auch das Nomen 'Termin' — im aufzählenden Sinne — modifizieren.[22] Solche Unterschiede kann das einfache stochastische Modell natürlich nicht erfassen; für die Zukunft läßt sich aber erwarten, daß komplexere Modelle mit solchen Dependenzen besser umgehen können werden.

Eine weitere große Fehlerquelle stellen die falsch zugeordneten Präpositionalgruppen dar. Für die Auflösung dieses schon klassischen Ambiguitätsproblems wird lexikalisches Wissen benötigt; ein schlechtes Abschneiden des einfachen Modells ist zu erwarten (siehe auch Abschnitt 4.7). Aufwendigere Modelle sollten in der Lage sein, auch diesen Fehler zu reduzieren.

Die beiden oben angesprochenen Fehlerklassen decken bereits über die Hälfte der Fehler ab. Die übrigen zwölf Klassen lassen sich z. T. auf Unzulänglichkeiten im grammatischen System bzw. der Basisgrammatik[23] oder wieder auf eine zu geringe Sensitivität des Modells zurückführen. Aufgrund der geringen absoluten Anzahl an Fehlern in diesen Klassen sind allgemein gültige Aussagen kaum möglich.

4.12 Lernen von Bewertungen für manuell erstellte Constraints

Einem menschlichen Grammatikschreiber fällt es aufgrund seiner Ausbildung und seiner sprachlichen Kompetenz meist relativ leicht, für einen bestimmten

[21] Ein menschlicher Hörer löst bestimmte Abhängigkeiten überhaupt nicht auf, da diese für ein (Grund–)Verständnis der Äußerung nicht relevant sind. Normalerweise wird diese verbleibende Ambiguität nicht bewußt wahrgenommen und nicht als störend empfunden; nur ein kleiner Teil dieser unaufgelösten Ambiguitäten führt tatsächlich zu Mißverständnissen.

[22] Menschliche Hörer nutzen in diesem Fall zur Disambiguierung z. B. Kontextinformation oder prosodische Indikatoren.

[23] So scheint die Lesart von Worten, die sowohl abtrennbares Verbpräfix als auch Präposition sein können, als Verbpräfix stark präferiert zu sein (Fehlerklasse $pn \rightarrow avz$). Hier liesse sich durch Änderungen in der Basisgrammatik wahrscheinlich Abhilfe schaffen.

Grammatiktyp die 'Enscheidungsregeln' für Grammatikalität aufzustellen; bei dieser Aufgabe kann er auf seine Intuition und auf Regelwerke zurückgreifen. Bei der Bewertung von solchen Regeln, also etwa von Constraints, dagegen versagt die Intuition recht schnell, während Grammatikwerke überhaupt keine Aussage machen.[24] In diesem Abschnitt wird — anders als in den vorangehenden — die Möglichkeit diskutiert, die Bewertungen von manuell erstellten Constraints durch ein annotiertes Korpus, wie es auch zum Training der stochastischen Modelle verwendet wurde, schätzen zu lassen.

4.12.1 Bewertung als Anteil der verletzten Constraints

Bei gegebenem annotierten Korpus als Menge von Unterordnungskonfigurationen und gegebenem Constraint kann der Anteil der das Constraint verletzenden Konfigurationen an allen Konfigurationen als Näherung für eine Bewertung betrachtet werden (siehe Gleichung 4.16).

$$W(c) \approx \frac{\text{Anzahl der Konfigurationen, die das Constraint } c \text{ verletzen}}{\text{Anzahl aller Konfigurationen}} \tag{4.16}$$

Ein starkes Constraint wird selten oder überhaupt nicht verletzt werden und deshalb eine Bewertung nahe Null erhalten, während gegen ein schwaches Constraint öfter verstossen werden wird, woraus sich eine höhere Bewertung ergibt.

Diese Abschätzung der Bewertungen erweist sich jedoch als ungeeignet, wenn man bedenkt, daß sich Constraints stark darin unterscheiden, über welche Konfigurationen sie Aussagen machen. Einige Constraints sind sehr allgemein und schränken dadurch eine große Anzahl an Konfigurationen ein. Andere beziehen sich nur auf einige wenige spezielle Konfigurationen; sie machen über die anderen Konfigurationen keine Aussage. Daß der letzte Abschnitt keinen Widerspruch zu der Aussage (Abschnitt 3.2.1), Constraints bezögen sich immer auf alle Konfigurationen, darstellt, zeigt das folgende (bewußt falsche) Beispiel.

$\mathsf{SubjCaseGen} : \mathsf{X.label} = \mathsf{subj} \rightarrow \mathsf{X{\downarrow}case} = \mathsf{gen}$
'Subjekte besitzen den Kasus Genitiv.'

$\mathsf{ArtNomCase} : \mathsf{X{\downarrow}cat} = \mathsf{ART} \wedge \mathsf{X.label} = \mathsf{det} \rightarrow \mathsf{X{\downarrow}case} = \mathsf{X{\uparrow}case}$
'Artikel und Nomen stimmen im Kasus überein.'

Das Constraint SubjCaseGen fordert (als Beispiel für ein 'schlechtes' Constraint), daß ein Subjekt den Kasus Genitiv besitzt, während das Constraint ArtNomCase die Kongruenz zwischen Artikel und Nomen bzgl. des Kasus herstellt. Der

[24] Nicht repräsentative (Selbst–)Versuche haben ergeben, daß oft eine Intuition darüber vorhanden ist, ob ein Constraint relativ zu einem anderen eher schlechter, besser oder gleich bewertet werden soll; eine konkrete Bewertung wird dagegen als nicht machbar empfunden. Es ist auch unklar, inwieweit solche Ad–hoc–Bewertungen hilfreich sein können. Manchmal finden sich in Regelwerken zur Grammatik Hinweise über die Angemessenheit oder Häufigkeit von sprachlichen Elementen in Form von sehr allgemeinen Kategorien wie *selten* oder *veraltet*.

Einfachheit halber sei angenommen, daß das gesamte Korpus aus dem Satz 4.6 besteht, wobei 'das buero' als Subjekt und 'in des Anbau' als Präpositionalphrase mit Kasusfehler analysiert werden.

(4.6) das buero ist in des Anbau des neuen infobaus (n011k.011.1, angepaßt)

Beide Constraints SubjCaseGen und ArtNomCase werden genau einmal durch das Korpus verletzt, erhalten also dieselbe Bewertung (in beiden Fällen $W(c) = 1/9$) zugesprochen. Diese Gleichbehandlung entspricht nicht der Intuition: Während kein einziges Subjekt des Korpus den Kasus Genitiv annimmt (das Constraint also nahe Eins richtig bewertet wäre), stimmen doch in wenigstens zwei von drei Nominalphrasen der Kasus des Artikels mit dem des Nomens überein, wodurch eine Bewertung von etwa einem Drittel gerechtfertigt wäre.

Das (wenn auch hier konstruierte) Beispiel macht deutlich, daß eine Abschätzung der Bewertungen nach Gleichung 4.16 ungeeignet ist, da nicht beachtet wird, daß verschiedene Constraints sich auf verschiedene und unterschiedlich viele Konfigurationen beziehen. Im obigen Beispiel müßte beachtet werden, daß im Korpus ein einziges Subjekt und drei Nominalphrasen vorkommen.

4.12.2 Bewertung als Anteil der verletzten an den relevanten Constraints

In Abschnitt 4.12.1 wurde deutlich, daß eine automatische Bewertung miteinbeziehen muß, für wieviele Konfigurationen im Korpus ein Constraint überhaupt relevant ist. Dies führt unmittelbar zu der verbesserten Gleichung 4.17.

$$W(c) \approx \frac{\text{Anzahl der Konfigurationen, die das Constraint } c \text{ verletzen}}{\text{Anzahl der für das Constraint } c \text{ relevanten Konfigurationen}} \quad (4.17)$$

Bei dieser Überlegung ergibt sich die Frage, welche Konfigurationen für ein bestimmtes Constraint relevant sind. Da sich die Relevanz eines Constraints — wie weiter unten deutlich wird — nicht automatisch aus dem Constraint ergibt, erscheint es sinnvoll, den Grammatikschreiber darüber entscheiden zu lassen. Dazu werden die Constraints mit einem zusätzlichen Prädikat versehen, das nur dann erfüllt ist, wenn das Constraint für eine Konfiguration relevant ist. Wenn das Relevanzprädikat mit p und das Wohlgeformtheitsprädikat mit k bezeichnet wird, so läßt sich die Gleichung 4.17 formaler als Gleichung 4.18 umschreiben.

$$W(\langle p, k\rangle) \approx \frac{\text{Anzahl der Konfigurationen, die } k \text{ verletzen}}{\text{Anzahl der Konfigurationen, die } p \text{ erfüllen}} \quad (4.18)$$

Wie findet sich der Relevanzbegriff eigentlich in den Constraints, wie sie bisher formuliert wurden? Häufig werden Constraints in Form von Implikationen geschrieben. Dabei überprüft die Prämisse p die Relevanz des Constraints und die

Konklusio k macht eine Aussage über die Wohlgeformtheit einer Konfiguration. Zum Beispiel ist das Constraint SubjCaseGen aus Abschnitt 4.12.1 so formuliert, daß nur Subjektkonfigurationen relevant sind und für diese gefordert wird, daß das Subjekt im Genitiv steht. Wenn sich aus bisherigen Constraints häufig das Relevanz– und das Wohlgeformtheitsprädikat extrahieren lassen, indem man die Formel des Constraints in eine Implikation mit Prämisse– und Konklusioteil umformt, stellt sich die Frage, ob man nicht auf eine Auszeichnung durch den Grammatikschreiber völlig verzichten kann. Dazu sei wiederum ein Beispiel angegeben.

SubjNumOrig: X.role=Syn $\wedge$ X.label=subj $\rightarrow$ X$\downarrow$num=X$\uparrow$num
'Das Subjekt stimmt mit dem Verb im Numerus überein.'

SubjNumAlt1: X.role=Syn $\wedge$ $\sim$X$\downarrow$num=X$\uparrow$num $\rightarrow$ $\sim$X.label=subj

SubjNumAlt2: X.label=subj $\wedge$ $\sim$X$\downarrow$num=X$\uparrow$num $\rightarrow$ $\sim$X.role=Syn

Das hier angegebene Constraint SubjNumOrig stammt aus Abschnitt 3.2.1, während SubjNumAlt1 und SubjNumAlt2 logische Umformungen von SubjNumOrig darstellen. Daraus ergibt sich, daß alle drei für alle Konfigurationen den gleichen Wahrheitswert ergeben, also auch im Sinne des Parsings mit PCDG äquivalent sind. Ein Unterschied ergibt sich in dem Moment, in dem man aus diesen Implikationen die Relevanz– und Wohlgeformtheitsprädikate extrahieren möchte, etwa um eine Bewertung abzuschätzen. Während sich bei der Formulierung SubjNumOrig die Prämisse der Implikation ungefähr mit dem intuitiven Relevanzbegriff deckt, führt eine Verwendung der Formulierung SubjNumAlt2 möglicherweise zu einer falschen Bewertung, da es unberücksichtigte Abhängigkeiten zwischen der Prämisse und der Konklusio gibt. In dem konkreten Fall kann man annehmen, daß die Verwendung einer Beschriftung subj nur syntaktisch interpretiert werden kann, also eine Rolle Syn impliziert. Dann wäre die Konklusio immer falsch, sobald die Prämisse wahr ist, so daß sich eine Bewertung von Eins ergebe, obwohl eine Bewertung nahe Null angemessen wäre. Eine unterschiedliche Bewertung des Constraints abhängig von der Formulierung wäre die Folge. Es sei an dieser Stelle noch einmal betont, daß die Formulierungen logisch äquivalent sind und nur die Extraktion von Relevanz– und Wohlgeformtheitsprädikaten aufgrund von Abhängigkeiten zwischen diesen zu unterschiedlichen Resultaten führt.

Das Beispiel macht deutlich, daß eine automatische Umformung der Constraints in Implikationen nicht unproblematisch ist. Aber auch ein Grammatikschreiber muß bei der Formulierung von Relevanz– und Wohlgeformtheitsprädikaten auf eventuelle Abhängigkeiten achtgeben. Der folgende Abschnitt deutet ein weiteres Problem mit der hier vorgestellten Bewertung anhand des Anteiles der verletzten an den relevanten Konfigurationen an.

4.12.3 Wartbarkeit von Constraint–Mengen

Constraints sind in ihren Wirkweisen im Analyseverlauf nur schwer durchschaubar. Diese Aussage gilt umso mehr, wenn es sich um bewertete Constraints handelt. Aus diesem Grund ist es schwierig, Abhängigkeiten und Inkonsistenzen in der Constraint–Menge zu entdecken und zu entschärfen. Das automatische Lernen von Bewertungen nach Abschnitt 4.12.2 vereinfacht die Grammatikerstellung auf der einen Seite, da die Bewertungen nicht mehr manuell erstellt werden müssen. Auf der anderen Seite muß der Grammatikschreiber während der Grammatikerstellung immer im Gedächtnis behalten, nach welchen Prinzipien diese Constraints später bewertet werden.

Dazu sei ein weiteres Beispiel gegeben. Grammatiken enthalten häufig solche wie die im folgenden angedeuteten Constraint–Paare:

Super : Bedingung A

Sub : Bedingung A $\wedge$ Bedingung B

Das Constraint Sub stellt also eine Spezialisierung des Constraints Super dar. Es sei weiterhin angenommen, daß beide Bedingungen A und B einen gewissen Grad an Ungrammatikalität anzeigen. Dem Constraint B wird automatisch eine mindestens so hohe Bewertung wie dem Constraints A zugewiesen, da jede Konfiguration, die A verletzt, auch B verletzt. Eine tatsächlich zu bewertende Konfiguration, die A verletzt, wird aber auf diese Weise für die Verletzung doppelt 'bestraft', da sich die Bewertungen von verletzten Constraints aufmultiplizieren.

Wie sind solche — kaum kontrollierbaren — Effekte zu vermeiden? Gelingt es, die Mengen an Konfigurationen, die durch einzelne Constraints verletzt werden, disjunkt zu halten, so wird eine angemessenere Bewertung erzeugt. Für das gegebene Beispiel wären die Constraints z. B. wie folgt umzuschreiben:

Super : Bedingung A $\wedge \sim$Bedingung B

Sub : Bedingung A $\wedge$ Bedingung B

Eine solche Maßnahme erzeugt eine Partition der durch einzelne Constraints verletzten Konfigurationsmengen; eine multiplikative Vergrößerung der Constraintmenge wäre die Folge. Damit lassen sich keine Verallgemeinerungen mehr über Konfigurationen ausdrücken, wodurch das Verfahren inpraktikabel wird.

Zusammenfassend läßt sich sagen, daß beim automatischen Lernen von Bewertungen Abhängigkeiten zwischen einzelnen Constraints nicht automatisch erkannt werden können. Ein Grammatikschreiber muß aus diesem Grund mit der zusätzlichen Einschränkung arbeiten, daß Abhängigkeiten in der Constraint–Menge zu fehlerhaften Bewertungen von Konfigurationen führen können.

4.12.4 Bewertung

Das Lernen von Bewertungen aus einem annotierten Korpus stellt eine reizvolle Alternative zu manuellen Beurteilungen dar. Allerdings ist es für einen Grammatikschreiber nicht immer einfach, die Constraint–Menge konsistent zu halten, so daß keine Mehrfachbewertungen stattfinden können. Eine Evaluation setzt eine angemessen große Grammatik voraus und stellt eine Aufgabe für die Zukunft dar.

5. Abschließende Bemerkungen

5.1 Zusammenfassung

Im ersten Teil dieser Arbeit wurden einige stochastische Verfahren zur Analyse von natürlicher Sprache vorgestellt und bzgl. ihrer prinzipiellen Fähigkeit zur Disambiguierung untersucht. Dabei wurde insbesondere festgestellt, daß leistungsfähige stochastische Modelle sowohl lexikalisch als auch strukturell sensitiv sein müssen, um die Regelmäßigkeiten der natürlichen Sprache auch nur annähernd beschreiben zu können. 'Strukturell' kann dabei so verstanden werden, daß mindestens syntaktische Affinitäten, besser auch semantische Abhängigkeiten erfaßt werden. Während die n–Gramme und die probabilistischen kontextfreien Grammatiken als 'klassische' Vertreter der stochastischen Modelle entweder den strukturellen oder den lexikalischen Aspekt völlig außer acht lassen, sind neuere Ansätze wenigstens lexikalisch und syntaktisch strukturell sensitiv.

Im zweiten Teil wurde dann eine stochastische Modellklasse für das eliminative Parsing entwickelt. Prinzipiell können über diesen Mechanismus beliebige stochastische Abhängigkeiten durch die Formulierung als Constraint erfaßt werden. So wird der Forderung nach einer Sensitivität auf allen Ebenen, die sich aus dem ersten Teil ergibt, nachgekommen. Aus Gründen der Machbarkeit wurden in dieser Arbeit jedoch ausschließlich Modelle evaluiert, die nur direkte syntaktische Abhängigkeiten (von lexikalischen bzw. kategoriellen Einheiten) berücksichtigen. Die durchgeführten Experimente haben gezeigt, daß das Verfahren geeignet ist, sprachliches Wissen aus annotierten Korpora zu extrahieren und bei der Analyse unbekannter natürlichsprachlicher Äußerungen zu nutzen. Einschränkend muß jedoch erwähnt werden, daß die erzielten Ergebnisse weiterhin verbesserungswürdig sind. Dafür wurde eine Reihe von Gründen identifiziert:

- Das verwendete Korpus ist zu klein, um die benötigten Parameter ausreichend sicher schätzen zu können. Inzwischen ist die zugrundeliegende Datenbasis erheblich erweitert worden, so daß sich dieser Punkt in Zukunft relativieren dürfte.

- Teilweise als Folge des ersten Punktes wurde auf ein vereinfachtes stochastisches Modell zurückgegriffen. Versuche mit aufwendigeren Modellen, die z. B. lexikalisch sensitiv sind, führen zu enttäuschenden Ergebnissen, da das Korpus die vergrößerte Anzahl an Parametern nicht zuverlässig schätzen kann.

- Die verwendete Modellklasse beschränkt sich aus Gründen der Praktikabilität und Einfachheit auf relativ simple stochastische Abhängigkeiten. Es ist offen, inwieweit *prinzipiell* komplexere Modelle zur erfolgreichen Analyse natürlichsprachlicher Äußerungen erforderlich sind. Die Untersuchungen im ersten Teil der Arbeit lassen vermuten, daß tatsächlich nur aufwendigere Modelle in Lage sind, den größten Teil der Regularitäten der natürlichen Sprache zu erfassen.

- Das verwendete Korpus ist nicht vollständig konsistent. Inkonsistenzen im Korpus, wenn sie systematisch, d. h. nicht nur vereinzelt, auftreten, bedeuten für alle stochastischen Verfahren, daß sie die Regeln der Sprache nicht mehr lernen können, da sich diese Regeln dann *nicht* in dem Korpus widerspiegeln.

5.2 Ausblick auf zukünftige Arbeiten

In dieser Arbeit können naturgemäß nicht alle aufgeworfenen Fragen beantwortet werden. Einige der sich im Verlauf dieser Arbeit stellenden Fragen bieten einen sinnvollen Ansatzpunkt für weitere ausführliche Untersuchungen. Darunter befinden sich die folgenden:

- Aus der Aufzählung in Abschnitt 5.1 ergibt sich der Wunsch, komplexere bzw. weniger stark vereinfachte Modelle zu evaluieren. Damit ist zum einen eine größere Komplexität des theoretischen Modells gemeint, also eine Ausweitung der Menge der Informationen, die für bestimmte Strukturentscheidungen eine Rolle spielen könnten. In Kapitel 4 wurde z. B. ein Schritt in diese Richtung unternommen, als ein Distanzmaß in das Modell eingeführt wurde. Zum anderen können Überlegungen sinnvoll sein, auf welche Approximationen des Modells bei praktischen Experimenten bei Inkaufnahme von vertretbarem Mehraufwand für die Berechnungen verzichtet werden kann.

- Für weitere Experimente mit Modellen der in dieser Arbeit verwendeten Klasse spricht das erheblich erweiterte Korpus. Zum Beispiel sollte das lexikalisch sensitive Modell aus Kapitel 4 mit der größeren Datenbasis wesentlich bessere Ergebnisse erzielen.

- Die 'ideale' Auswahlfunktion für das eliminative Parsing durch Constraint–Satisfaction (siehe Abschnitt 3.4) bleibt ein Desiderat. Weitere Untersuchungen zu diesem Punkt sind zunächst einmal nötig, um überhaupt festzustellen, ob es eine solche stark verbesserte Auswahlfunktion geben kann

oder ob die zur Verfügung stehenden Informationen für eine in jedem Fall sichere Auswahl prinzipiell nicht ausreichen.

- Alternative Verfahren wie das hierarchische Constraint–Lösen oder die Constraint–Abschwächung bedürfen einer genaueren Untersuchung bzgl. ihrer Eignung für das Parsing mit CDG.

A. Grammatisches System

Im folgenden wird beschrieben, wie das grammatische System für die Arbeit mit PCDG formuliert werden muß. Im einzelnen werden als Komponenten das Lexikon, das mit Constraints formulierte Grammatikwissen und die verwendeten Statistiken vorgestellt.

A.1 Lexikon

Wie in Abschnitt 3.2.1 dargestellt wurde, benötigen (P)CDG ein Lexikon zur Analyse von natürlichsprachlichen Äußerungen. Dieses verwendete Lexikon besteht aus einer Liste von Einträgen für Wortformen. Ein und dieselbe Wortform kann im Falle von lexikalischer Ambiguität mehrfach aufgenommen werden. In den Einträgen wird die Wortform mit zusätzlicher Information angereichert. Tabelle A.1 zeigt die Syntax für das Lexikon.[1]

LEXICON	::=	LEXICONENTRY LEXICON
	|	*empty*
LEXICONENTRY	::=	WORD '=' PATHVALUESEQ ';'
WORD	::=	*identifier*
PATHVALUESEQ	::=	PATHVALUE',' PATHVALUESEQ
	|	PATHVALUE
PATHVALUE	::=	PATH ':' VALUE
PATH	::=	ATTRIBUTE ':' PATH
	|	ATTRIBUTE
ATTRIBUTE	::=	*identifier*
VALUE	::=	*identifier*

Tabelle A.1: Syntax der Lexikoneinträge

In den in Kapitel 4 beschriebenen Experimenten wird ein einfaches Lexikon verwendet, in dem lediglich Informationen zu Kategorie, Kongruenz, Tempus,

[1] Es wird eine erweiterte Backus–Naur–Notation verwendet; dabei sind '::=', '|', '[' und ']' Metazeichen.

Modus und (einfachen) Subkategorisierungsforderungen bereitgestellt werden. Für den verwendeten Ausschnitt der Blaubeurener Dialoge (siehe Anhang B) wurden 524 Lexikoneinträge für 406 verschiedene Wortformen aufgenommen.

A.2 Constraints

Das eigentliche grammatische Wissen wird in Form von Constraints formuliert, nachdem festgelegt wurde, welche Rollen es gibt und welche Beschriftungen für diese Rollen einschlägig sind. Tabelle A.2 zeigt, wie eine bestimmte Rolle eingeführt und eine Menge von Beschriftungen für diese vereinbart wird.

RoleDecl	::=	Role '#' LabelSeq ';'
Role	::=	*identifier*
LabelSeq	::=	Label ',' LabelSeq
	|	Label
Label	::=	*identifier*

Tabelle A.2: Syntax der Rollendeklarationen

Constraints legen fest, zu welchem Grade eine Kombination von Abhängigkeitsbeziehungen grammatisch ist (vgl. Abschnitte 3.2.1 und 3.4.1). Tabelle A.3 zeigt die Syntax eines Constraints.

Constraint	::=	Prefix ':' Formula '.'
Prefix	::=	Name [',' PenaltyFactor [',' Class]]
PenaltyFactor	::=	*number*
Class	::=	*identifier*
Formula	::=	Term Operator Term
	|	Formula Junctor Formula
	|	'~' Formula
	|	'(' Formula ')'
Term	::=	Variable'^'Path
	|	Variable'_'Path
	|	Variable'.label'
	|	Variable'.role'
Variable	::=	*identifier*
Path	::=	Attribute ':' Path
	|	Attribute
Attribute	::=	*identifier*
Operator	::=	'=' | '>' | '<' | '>=' | '<=' | '=\='
Junctor	::=	'&' | '—' | '->' | '<->'

Tabelle A.3: Syntax der Constraints

Neben einem Namen und optionaler Bewertung sowie Klasse besteht ein Cons-

traint aus einer aussagenlogischen Formel[2]. Evaluiert die Formel zum Wahrheitswert falsch, gilt das Constraint als verletzt.

A.3 Statistik

Die Statistiken liefern Bewertungen für bestimmte Konfigurationen. Welche Konfigurationen dabei unterschieden werden, legt das stochastische Modell fest.

Um eine einfache, zugleich aber äußerst flexible Realisierung der Statistiken zu ermöglichen, werden die Statistiken mit den exakt gleichen Mitteln beschrieben wie das grammatische Wissen, d. h. mit Hilfe von Constraints. Dies ist deshalb möglich, da Grammatik–Constraints ebenso eine bestimmte strukturelle Konfiguration bewerten. Der wesentliche Unterschied ergibt sich aus der Verarbeitung: Während bei den Grammatik–Constraints die Gesamtheit der Parameter zur Bewertung einer Konfiguration beisteuert, gibt es bei den Statistiken genau einen Parameter, d. h. ein Constraint, das eine bestimmte Konfiguration bewertet. Unter Berücksichtigung dieses Unterschieds lassen sich Statistiken einfach als Menge von bewertenden Constraints formulieren.

[2]Die verwendeten Variablen sind zum Zeitpunkt der Auswertung bereits gebunden, so daß keine Quantifizierung stattfindet.

B. Korpus

Stochastische Modelle von natürlichen Sprachen benötigen immer ein Korpus an natürlichsprachlichem Material zu ihrer Evaluation. Ein solches Korpus wird zum einen für verschiedene Testläufe, d. h. für das kontrollierte Parsing unbekannter Eingaben, und zum anderen für das Training der stochastischen Parameter verwendet. Um die Generalisierbarkeit auf wirklich unbekannte Eingaben garantieren zu können, trennt man das Korpus für diese beiden Zwecke in zwei disjunkte Teile, die sogenannten Test– und Trainingsmengen.[1]

Im folgenden wird das Korpus vorgestellt, mit dem die Experimente in Kapitel 4 durchgeführt wurden.

B.1 Charakteristika, Herkunft und Umfang des verwendeten Korpus

Das in dieser Arbeit erweiterte Verfahren erhebt den Anspruch, auf robuste Weise (sowohl bzgl. Über– als auch bzgl. Untergenerierung) (Teil–)Analysen für gesprochene natürlichsprachliche Äußerungen zu erstellen. Daraus ergeben sich einige der folgenden Forderungen an ein zu verwendendes Korpus:

- Das Korpus sollte Spontansprache im Gegensatz zu geschriebener bzw. kontrolliert gelesener Sprache enthalten. Nur so können tatsächliche Fehler in der menschlichen Sprachproduktion erfaßt werden. Ein Vorliegen des Korpus sowohl in transkribierter Form als auch als Sprachdaten bzw. in Form von Ausgaben eines Spracherkenners ermöglichen Experimente auf beiden Eingabeformaten.

[1] Beste Generalisierbarkeit der Ergebnisse erhält man, wenn man zu Beginn der Versuchsreihen eine zufällig ausgesuchte, im Detail unbekannte Teilmenge zurückbehält, dann die verbleibende Menge in Test– und Trainingsmenge teilt, diese für die verschiedenen (oft in mehreren Iterationen verlaufenden) Versuche verwendet und erst nach allen Feineinstellungen am Modell die (zu veröffentlichenden) Ergebnisse mit der zu Beginn zurückbehaltenen Menge in einem einzigen Testlauf ermittelt. Dadurch verhindert man das indirekte Training der Parameter durch mehrmalige Trainings– und Testläufe.

- Eine möglichst reichhaltige, konsistente Annotation der Äußerungen stellt eine der Voraussetzungen für die Verwendung mit stochastischen Verfahren dar. Die Verfahren aus Kapitel 4 erfordern dabei eine Strukturierung auf der Basis von Abhängigkeitsbeziehungen, d. h., übliche Strukturierungen wie z. B. Phrasenstrukturen können nicht direkt übernommen werden. Die Annotation mit Kategorien legt automatisch das verwendete Kategoriensystem fest. Obwohl in dieser Arbeit nur syntaktische Kategorien und Abhängigkeiten verwendet wurden, ist eine weitere z. B. semantische Strukturierung wünschenswert.

- Das Korpus sollte in deutscher Sprache und in — auch für zukünftige Erweiterungen — genügend großem Umfang vorliegen. Eine Einschränkung auf eine Anwendungsdomäne erhöht die Handhabbarkeit des Korpus und vereinfacht die Durchführung von Versuchen.

- Eine möglichst breite Akzeptanz in der Wissenschaftsgemeinde vergrößert die Chancen, Arbeitsergebnisse direkt vergleichen zu können.

Kein verfügbares Korpus erfüllt die obigen Anforderungen; tatsächlich stand kein Korpus mit Abhängigkeitsstrukturen in deutscher Sprache zur Verfügung. Statt dessen wurde ein Teil des im BMBF–Projekt VERBMOBIL (Wahlster, 1993) erstellten Korpus verwendet, das leider keine Annotierung enthält, sonst aber die wesentlichen Bedingungen erfüllt. Die Domäne dieses Korpus sind Terminabsprachen zwischen zwei Geschäftspartnern.

Das Korpus wurde von Hand an die Bedürfnisse des Verfahrens angepaßt (siehe Anhang B.2) und mit Informationen über Wortkategorien (Anhang B.3) und syntaktische Strukturen (Anhang B.4) versehen.

Basis sind die Dialoge des sogenannten Blaubeurener Ausschnitts der VERBMOBIL–Serie N: n001k, n002k, n003k, n007k, n009k, n011k, n017k und n019k. Diese Dialoge enthalten jeweils zwischen 15 und 43, insgesamt 195 Redebeiträge, von denen 155 verwendet wurden (siehe Anhang B.2). Aus diesen wurden wiederum 235 Sätze extrahiert, die das Korpus für alle Untersuchungen bilden. Die Sätze B.1 bis B.13 zeigen exemplarisch einen (relativ willkürlichen) Ausschnitt aus dem Korpus.

(B.1) dann lassen sie uns doch noch einen termin ausmachen (n001k.000.1)

(B.2) am dienstag den sechsten april haette ich noch einen termin frei allerdings nur nachmittags (n001k.002.1)

(B.3) da habe ich um vierzehn uhr dreissig einen termin beim zahnarzt (n001k.003.2)

(B.4) ich dachte eigentlich an den fuenfzehnten april eine woche spaeter (n001k.006.1)

(B.5) am dreizehnten april bin ich noch im urlaub genauso wie am zwoelften april montag (n002k.002.2.)

(B.6) ich sagte dass ich am donnerstag den fuenfzehnten april von zehn bis vierzehn uhr dreissig ein doktorandentreffen habe (n002k.007.1)

(B.7) auch der fuenfte ist schlecht (n002k.012.2)

(B.8) da habe ich bereits den ganzen tag hindurch termine (n003k.003.2)

(B.9) mein horoskop sagt am mittwoch staenden die sterne gut (n008k.020.1)

(B.10) koennten wir es vielleicht am donnerstag machen (n009k.007.2)

(B.11) wann waere es ihnen denn geschickt (n017k.000.2)

(B.12) donnerstag sechzehnter september sagen sie (n017k.004.1)

(B.13) ich bin mal durch den terminkalender gestolpert (n019k.008.1)

B.2 Manuelle Anpassungen

Die originalen VERBMOBIL–Dialoge liegen als Sprachdaten und als Niederschriften vor. Bei den Niederschriften wurden nicht interpretierbare Bereiche als solche markiert, ansonsten aber auch Interjektionen wie z. B. 'ja', 'prima' und 'wunderbar' oder 'oh', 'ah' und 'nee' direkt in die schriftliche Form aufgenommen. Vor der Anwendung für diese Arbeit werden sowohl die Markierungen für nicht interpretierbare Bereiche als auch alle Interjektionen, sofern sie nicht schlüssig in die syntaktische Struktur aufgenommen werden können, entfernt.

Zusätzlich werden die einzelnen Redebeiträge manuell in zu einem Satz gehörende Abschnitte segmentiert. Als 'zu einem Satz gehörig' wird dabei so verstanden, daß sich die Teile in eine syntaktische Struktur fügen müssen. Dafür essentiell ist das Vorhandensein eines finiten Verbs, welches dementsprechend nicht fehlen darf. Ellipsen anderer Satzteile dagegen werden meist akzeptiert, solange sich eine schlüssige syntaktische Struktur ergibt. Da Koordinationen bei der Analyse zusätzliche Schwierigkeiten machen, wurden diese aus dem Korpus ausgeschlossen.

Zum Beispiel wird der Redebeitrag n002k.000 des Beispiels B.14 in die beiden Sätze der Beispiele B.15 und B.16 aufgeteilt; die fehlenden Bereiche bleiben unberücksichtigt.

(B.14) schön hervorragend dann lassen Sie uns doch noch eine Termin ausmachen wann wäre es Ihnen denn recht ⟨nib⟩

(B.15) dann lassen Sie uns doch noch eine Termin ausmachen

(B.16) wann wäre es Ihnen denn recht

Die durchgeführten manuellen Anpassungen sind für das Funktionieren des verwendeten Verfahrens keine Notwendigkeit; vielmehr dienen diese Maßnahmen dazu, die Komplexität zu verringern. Die Ausarbeitung weiterer Modelle und die Durchführung weiterer Versuche mit Korpora, die nicht manuell verändert wurden, stellen eine Aufgabe für die Zukunft dar.

B.3 Kategorien

Bei der Auswahl des Kategoriensystems wird auf (Thielen, 1995) zurückgegriffen. Das dort definierte hierarchisch strukturierte System wird minimal angepaßt, um eine eindeutige Benennung der Kategorien zu ermöglichen.[2]

Tabelle B.1 zeigt die verwendeten Kategorien.

Symbol	**Erklärung**
N	Nomen
N–N	'normales' Nomen
N–E	Eigenname
V	Verb
V–FIN	finites Verb
V–I	Infinitiv
V–I–NF	'reiner' Infinitiv
V–I–ZU	Infinitiv mit 'zu'
V–PP	Partizip II
ART	Artikel
ADJ	Adjektiv
ADJ–A	attributierendes Adjektiv
ADJ–D	prädikativ oder adverbial gebrauchtes Adjektiv
P	Pronomen
P–PER	Personalpronomen
P–PER–IR	stets irreflexives Personalpronomen *Änderung*
P–PER–RF	reflexives und irreflexives Personalpronomen
P–PER–RF	reines Reflexivpronomen
P–POS	Possessivpronomen
P–POS–S	substituierendes Possessivpronomen
P–POS–AT	attributierendes Possessivpronomen
P–RO	Demonstrativ–/Indefinitpronomen
P–RO–S	substituierendes Demonstrativ–/Indefinitpronomen
P–RO–AT	attributierendes Demonstrativ–/Indefinitpronomen
P–RO–AV	adverbiales Demonstrativ–/Indefinitpronomen
P–W	Interrogativpronomen
P–W–S	substituierendes Interrogativpronomen
P–W–AT	attributierendes Interrogativpronomen
P–W–AV	adverbiales Interrogativpronomen
P–REL	Relativpronomen
P–REL–S	substituierendes Relativpronomen
P–REL–AT	attributierendes Relativpronomen
	fortgeführt auf der nächsten Seite

[2]Genaugenommen werden die Kategoriesymbole so geändert, daß jeder Untertyp den Namen des Obertyps als Präfix enthält. Zum Beispiel beginnen alle verbalen Kategorien mit dem Buchstaben V, die Infinitive mit V–I und Infinitive mit 'zu' erhalten das Symbol V–I–ZU. Auf diese Weise können Constraints leicht für eine bestimmte Klasse formuliert werden.

fortgesetzt von der letzten Seite	
Symbol	**Erklärung**
P–NFL	nicht flektierendes Pronomen
P–ALL	Form von 'all–'
P–BEID	Form von 'beid–'
P–VIEL	nicht flektierte, nicht attributive Form von 'viel', 'wenig', 'zuwenig', 'mehr', 'weniger'
CARD	Kardinalzahl
ADV	Adverb
KO	Konjunktion
KO–U	untergeordnete Konjunktion
KO–U–I	untergeordnete Konjunktion mit Infinitiv
KO–U–S	untergeordnete Konjunktion mit (finitem) Satz
KO–N	nebengeordnete Konjunktion
KO–KOM	vergleichende Konjunktion
AP	Adposition
AP–PR	Präposition
AP–PR–PL	Präposition (plus Artikel) *Änderung*
AP–PR–ART	Präposition mit Artikel
AP–PO	Postposition
AP–ZR	Zirkumposition rechts
ITJ	Interjektion
PTK	Partikeln
PTK–ZU	Infinitiv–'zu'
PTK–NEG	Negationspartikel
PTK–VZS	abgetrennter Verbzusatz von Partikelverb
PTK–ANT	Antwortpartikel
PTK–A	Partikel bei Adjektiv oder Adverb
TRUNC	Kompositionserstglied

Tabelle B.1: Verwendetes Kategoriensystem

B.4 Syntaktisches System

Mangels eines breit akzeptierten Syntaxsystems mit Abhängigkeitsstrukturen für große Datenmengen wird ein eigenes (einfaches) System der syntaktischen Analyse entworfen.

Tabelle B.2 gibt einen Überblick über die verwendeten Beschriftungen mit typischen modifizierenden und regierenden Kategorien (vgl. Abschnitt 3.2.1).

Die syntaktischen Strukturen der Beispielsätze B.17 und B.18 sind in den Abbildungen B.1 und B.2 wiedergegeben und sollen die Verwendung der Beschriftungen veranschaulichen.

Symbol	**untergeordnetes Wort**	**übergeordnetes Wort**	**Funktion**
amod	ADJ, ADV, N	V, N	allgemeine Modifikation
appo	N	N	Beiordnung
atta	ADJ–A	N	adjektivische Modifikation
attc	CARD	N	Modifikation durch ein Kardinal
avz	PTK–VZS	V	lexikalische Abhängigkeit
det	ART	N	Artikelanbindung
kom	KO–KOM	N	Vergleich
neg	NEG	V, N	Satz– oder Phrasennegation
ns	NS	V	Nebensatzunterordnung
objа	N, P	V	Akkusativobjekt
objd	N, P	V	Dativobjekt
pmod	N, V	N, V	Präpositionalgruppenanbindung
pn	AP	N	Präpositionalanbindung
subj	N, P	V	Subjekt
vfin	V–FIN	V Wurzel	(untergeordnetes) Hauptverb
vinf	V–I	V, KO–U	Infinitivanbindung
vpart	V–PP	V	Partizipanbindung

Tabelle B.2: Syntaktische Beschriftungen mit typischen Kategorien

(B.17)

dann	lassen	sie	uns	doch
ADV	V–FIN	P–PER–IR	P–PER-RF	ADV
noch	einen	termin	ausmachen	
ADV	ART	N–N	V–I–NF	

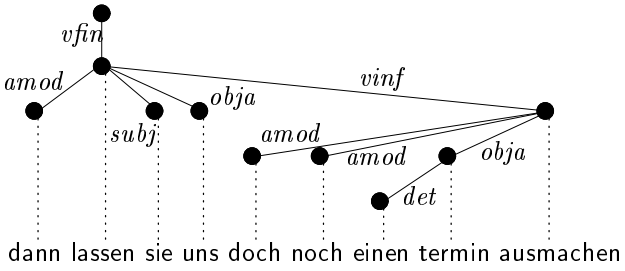


Abbildung B.1: Beispielstruktur 1

Das Beispiel B.17 zeigt deutlich eine Schwierigkeit: Adverben lassen sich oft nicht eindeutig einem bestimmten Wort zuordnen. Die Unterordnung von ‘doch’ und ‘nicht’ unter den Infinitiv geschieht relativ willkürlich; ebensogut wäre eine Unterordnung unter das finite Verb des Satzes möglich. Manchmal bedeutet eine unterschiedliche Zuordnung eine abweichende Semantik; die Zuordnung der Adverben zu den sie regierenden Worten kann aber kaum konsistent durchgehalten werden. Diese Unsicherheit in der Annotation des Korpus bedeutet eine mögliche Fehlerquelle für jedes statistische Verfahren, da es keine konsistente

Zuordnung gibt, die das Verfahren trainieren könnte.

(B.18)

da	habe	ich	um	vierzehn	uhr
ADV	V–FIN	P–PER–IR	AP–PR–PL	CARD	N–N
dreissig	einen	termin	beim	zahnarzt	
CARD	ART	N–N	AP–PR–ART	N–N	

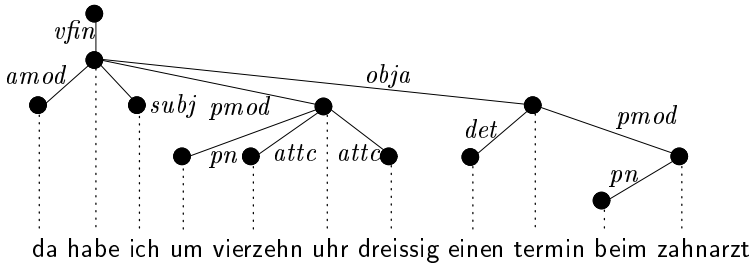


Abbildung B.2: Beispielstruktur 2

Die Abbildung B.2 zeigt die Struktur von Präpositionalphrasen und Zeitausdrücken. Nicht die Präposition, sondern das zugehörige Nomen (oder Pronomen o. ä.) wird als Kopf einer Phrase ausgezeichnet; die Präposition dagegen modifiziert den Kopf. Dieses Vorgehen hat den Vorteil, die u. U. wichtigen lexikalischen Abhängigkeiten zwischen dem Kopf der Präpositionalphrase und dem die Präpositionalphrase regierenden Wort direkt in einer Abhängigkeitsbeziehung nutzen zu können. Bei Zeitausdrücken modifizieren die Zahlworte immer das Zeitwort (hier 'Uhr'). Existiert kein Zeitwort wie z. B. 'um neun' in Beispiel B.19, so übernimmt das Zahlwort dessen Rolle.

(B.19) aber donnerstag vormittag so um neun wäre mir recht (n001k.004.1)

B.5 Weitere Annotationen

In dieser Arbeit wurden nur die kategoriellen und syntaktischen Informationen verwendet. Inzwischen wurde der Umfang des Korpus erheblich erweitert und um semantische und ontologische Annotationen angereichert. Experimente mit statistischen Modellen, die diese Informationen berücksichtigen und das in Abschnitt 3.4.2 beschriebene Systemverhalten nachweisen könnten, stellen eine Aufgabe für die Zukunft dar.

C. Programmsystem

Für die Experimente in Kapitel 3 und 4 ist eine umfangreiche programmtechnische Realisierung der dort beschriebenen Modelle notwendig. Dieser Anhang stellt die verschiedenen Komponenten der Implementation, die sich in die Komponenten Parser, die Statistiken betreffende Teile und allgemeine Hilfsprogramme unterteilt, vor. Eine detaillierte Behandlung bleibt allerdings einem (zu verfassenden) Handbuch vorbehalten; hier geht es im wesentlichen um die Funktion für diese Arbeit und um die Beschreibung des Anteils an dieser Arbeit.

Weitere Informationen zum Programmsystem, ausführliche Dokumentationen zum Korpus und zum Grammatiksystem sowie die detaillierten Ergebnisse der Versuche sind über den Autor verfügbar.

C.1 Allgemeines

Als Programmiersprachen werden JAVA[1] für das komplexere Programmsystem des Parsers und PERL[2] für einfachere Programme verwendet. Folgende Kriterien waren für die Auswahl von JAVA ausschlaggebend.

- Portabilität: JAVA–Quellen werden in einen sogenannten Byte–Kode kompiliert, wodurch die Programme durch einen Interpreter auf einer großen Anzahl von Plattformen ausführbar sind.
- Objektorientiertheit: Das objektorientierte Design von JAVA–Programmen sorgt für eine effektive Daten– und Funktionskapselung.
- Speicherverwaltung: Die automatische Speicherverwaltung entlastet den Programmierer und hilft, schwer zu findende Fehler zu vermeiden.

[1] Die Programmiersprache JAVA wird von der Firma SUN Microsystems entwickelt; nähere Informationen finden sich in (Lemay und Perkins, 1996), (Flanagan, 1996) oder im Internet unter http://www.javasoft.com/.

[2] Die Skriptsprache PERL wird von Larry Wall (lwall@netlabs.com) entwickelt und steht auf allen verbreiteten Rechnerplattformen zur Verfügung. Nähere Informationen finden sich in (Wall und Schwartz, 1994) oder im Internet unter http://www.perl.org/.

- Ausführungsgeschwindigkeit: Eine effiziente Ausführung der Programme ist möglich, da der erzeugte Byte–Kode schnell interpretiert werden kann. Für die Zukunft ist mit Programmen zu rechnen, die den Byte–Kode kurz vor der Ausführung in Maschinenkode übersetzen (engl. *just–in–time compiler*) und damit eine weitere Geschwindigkeitssteigerung erlauben.

- Typisierung: Die statische Typisierung von Variablen ermöglicht das Aufdecken von Fehlern zur Kompilierzeit und zwingt den Programmierer zu einer klaren Datenmodellierung.

- Klassenbibliothek: Die fest an die Sprache gebundene Klassenbibliothek bietet Zugriff auf eine Reihe von Datenstrukturen und Algorithmen kapselnde Objekte.

Für die Entwicklung der kleineren Programme sind andere Kriterien wichtig. PERL erfüllt die folgenden:

- Entwicklungszeit: Als interpretierte Sprache bietet PERL extrem kurze Entwicklungszyklen (engl. *rapid prototyping*), da ein Kompilieren und Binden vollständig entfällt.

- Reguläre Ausdrücke: Durch die Integration von regulären Ausdrücken in der Sprache erlaubt PERL einen einfachen Umgang mit Zeichenketten bzw. eine einfache Extraktion von Informationen aus textuellen Dateien.

- Typisierung: Die dynamische Typisierung von Variablen erlaubt es dem Programmierer, extrem flexibel und ohne ein vorher ausgearbeites Datenmodell die Problemlösung anzugehen.

Insgesamt umfaßt die gesamte Implementation rund 8.800 Programmzeilen, von denen etwas mehr als die Hälfte in JAVA, der Großteil des Restes in PERL und ein kleiner Rest als Skripte für eine Kommandoschale (engl. *shell script*) geschrieben sind.

C.2 Parser

Der Parser stellt den zentralen Programmteil dar. Er weist einer natürlichsprachlichen Äußerung bei gegebener Grammatik und Statistik eine Struktur zu. Im folgenden werden die wichtigsten Klassen der JAVA–Implementation vorgestellt.

- Die Klassen cdg.RowCol, cdg.Fraction und cdg.Absolute sind die Hauptklassen für den Aufruf des Parsers. Sie realisieren die einzelnen Auswahlfunktionen des Abschnitt 3.5.

- Die Klasse cdg.grammar.Grammar implementiert die verwendeten Grammatiken. Um Grammatiken einlesen zu können, wurde ein LR–Parser cdg.grammar.Parser entworfen. Die Klasse cdg.grammar.Constraint repräsentiert einzelne Constraints.
- Viele Ähnlichkeiten mit den Grammatiken weisen die Statistiken, die durch die Klasse cdg.statistics.Statistics realisiert werden, auf (siehe Anhang A.3). Die Abarbeitung unterscheidet sich jedoch beträchtlich.
- Für die Verarbeitung von natürlichsprachlichen Äußerungen als Wortgraph wird die Klasse cdg.lattice.LexemLattice verwendet.
- Das Lexikon wird durch cdg.lexicon.Lexicon implementiert. Auch hier wird ein LR–Parser cdg.lexicon.Parser für das Einlesen verwendet.
- Die eigentlichen Constraint–Netze werden durch Instanzen der Klasse cdg.net.ConstraintNet modelliert. Sie fassen den Großteil der Objekte zusammen und werden durch den Parser gesteuert.

C.3 Extraktion von Informationen aus Korpora

Die Extraktion von Informationen, z. B. Statistiken, aus Korpora geschieht durch PERL–Skripte. Das Korpus wird zu diesem Zweck in eine Normalform gebracht, die relativ einfach durch Programme verarbeitet werden kann. Die folgenden Programme werden für diese Arbeit verwendet:

- Das Programm nf2statistics erzeugt aus einem Korpus in Normalform eine vom Parser zu nutzende Statistik. Die Art der erzeugten Statistik läßt sich durch Parameter festlegen.
- Für den Vergleich zwischen Parser–Ausgaben und Testmenge wird das Skript info2txt verwendet. Erst mit der Ausgabe dieses Programms kann man bestimmen, wie gut der Parser gearbeitet hat.

C.4 Hilfsprogramme

Im Laufe der Arbeit sind eine Reihe von Hilfsprogrammen entstanden, die verschiedene Aufgaben erfüllen. Der größte Teil dieser Programme dient der Visualisierung von Zwischenergebnissen oder der Dokumentation von Ergebnissen. Einige Programme erzeugen ihre Ausgabe denn auch in HTML (engl. *hypertext markup language*), der Dokumentenauszeichnungssprache des World Wide Web. Im folgenden sind die wichtigsten Skripte aufgeführt:

- Zur Dokumentation des Korpus in Normalform wird das Skript nf2html verwendet, das auch grafische Ausgaben der Abhängigkeitsstrukturen ausgibt.

- Eine Reihe von Programmen (nf2categories, nf2relations u. a. m.) erzeugen Zusammenstellungen über Häufigkeiten von Abhängigkeitskonfigurationen in einem Korpus.
- Die Ergebnisse von Parser–Läufen werden von den Skripten update-txt und update-results zusammengetragen und zusammengefaßt.

D. Basisgrammatik

Wie in Abschnitt 4.3 beschrieben wird bei den Versuchen eine sehr einfache Basisgrammatik verwendet. Diese ist im folgenden vollständig aufgeführt:

$1 : 0.0 : X{\downarrow}id \neq X{\uparrow}id$
'Kein Wort kann sich selber modifizieren.'

$2 : 0.5 : X{\downarrow}cat = PTK\text{–}VZS \rightarrow X{\uparrow}cat1 = V \wedge X{\uparrow}avp = plus$
'Abgetrennte Verbpräfixe modifizieren dafür spezifizierte Verben.'

$3 : 0.5 : X{\downarrow}cat = V\text{–}I\text{–}NF \rightarrow X{\uparrow}aux = inf$
'Infinitive modifizieren dafür spezifizierte Verben.'

$4 : 0.5 : X{\downarrow}cat = V\text{–}PP \rightarrow X{\uparrow}aux = ppart$
'Partizipien modifizieren dafür spezifizierte Verben.'

$5 : 0.4 : X{\downarrow}cat1 = AP \wedge X{\downarrow}cat2 = PR \rightarrow X{\uparrow}case = X{\downarrow}case$
'Präpositionen fordern einen bestimmten Kasus vom Nomen.'

$6 : 0.5 : X{\downarrow}cat = ART \wedge X.label = det \rightarrow X{\downarrow}case = X{\uparrow}case$
'Artikel und Nomen stimmen im Kasus überein.'

$7 : 0.5 : X{\downarrow}cat = ART \wedge X.label = det \rightarrow X{\downarrow}num = X{\uparrow}num$
'Artikel und Nomen stimmen im Numerus überein.'

$8 : 0.5 : X{\downarrow}cat = ART \wedge X.label = det \rightarrow X{\downarrow}gen = X{\uparrow}gen$
'Artikel und Nomen stimmen im Genus überein.'

$9 : 0.5 : X.label = subj \rightarrow X{\downarrow}case = nom$
'Subjekte besizten den Kasus Nominativ.'

$10 : 0.5 : X.label = subj \rightarrow X{\downarrow}per = X{\uparrow}per$
'Subjekt und Verb stimmen in der Person überein.'

$11 : 0.5 : X.label = subj \rightarrow X{\downarrow}num = X{\uparrow}num$
'Subjekt und Verb stimmen im Numerus überein.'

$12 : 0.1 : X.label = obja \rightarrow X{\downarrow}case = acc$
'Ein Akkusativobjekt besizt den Kasus Akkusativ.'

$13 : 0.3 : X.label = obja \rightarrow X{\uparrow}case1 = acc$
'Ein Akkusativobjekt modifiziert ein dafür spezifiziertes Verb.'

14 : 0.1 : X.label = objd $\rightarrow$ X$\downarrow$case =
'Ein Dativobjekt besitzt den Kasus Dativ.'

15 : 0.3 : X.label = objd $\rightarrow$ X$\uparrow$case = dat
'Ein Dativobjekt modifiziert ein dafür spezifiziertes Verb.'

16 : 0.5 : X.label $\neq$ pmod
'Eine Präpositionalgruppenanbindung ist weniger bevorzugt.'

17 : 0.0 : X.label = pn $\wedge$ X$\uparrow$id = Y$\downarrow$id $\rightarrow$ Y.label = pmod
'Wird ein Nomen durch eine Präposition modifiziert, modifiziert es selber als Präpositionalgruppe.'

18 : 0.0 : X$\uparrow$id = Y$\uparrow$id $\wedge$ X.label = subj $\wedge$ Y.label = subj
$\rightarrow$ X$\downarrow$id = Y$\downarrow$id
'Es gibt nur ein Subjekt pro Verb.'

19 : 0.0 : X$\uparrow$id = Y$\uparrow$id $\wedge$ X.label = obja $\wedge$ Y.label = obja
$\rightarrow$ X$\downarrow$id = Y$\downarrow$id
'Es gibt nur ein Akkusativobjekt pro Verb.'

20 : 0.0 : X$\uparrow$id = Y$\uparrow$id $\wedge$ X.label = objd $\wedge$ Y.label = objd
$\rightarrow$ X$\downarrow$id = Y$\downarrow$id
'Es gibt nur ein Dativobjekt pro Verb.'

21 : 0.1 : X$\uparrow$id = Y$\uparrow$id $\wedge$ X.label = pn $\wedge$ Y.label = det
$\rightarrow$ X$\downarrow$pos = Y$\downarrow$pos - 1
'Modifizieren Artikel und Präposition ein Nomen, so steht die Präposition vor dem Artikel.'

E. Experimente

Die Tabelle E.1 zeigt die Endergebnisse der wichtigsten Experimente auf einen Blick. Detailliertere Informationen finden sich insbesondere in Kapitel 4.

No	**Auswahl-funktion**	**Train-ing**	**Fehler**	**Genau-igkeit**	**Para-meter**	**Besonderheiten**
E_1	RowCol 10^{-5};0.1	1	8.6%	0.56	clCd	
E_2	RowCol 10^{-5};0.1	2–19	17.6%	0.62	clCd	
E_3	RowCol 10^{-5};0.1	1	9.4%	0.56	clCd	
E_4	Absolute 10^{-5};0.1	2–19	6.9%	0.13	clCd	
E_5	Absolute 10^{-5};0.01	2–19	9.9%	0.23	clCd	
E_6	Fraction 10^{-5};0.1	2–19	16.7%	0.49	clCd	
E_7	RowCol 10^{-5};0.1	2–19	21.0%	0.37	clC	
E_8	RowCol 10^{-5};0.1	2–19	27.0%	0.30	wlCd	
E_9	RowCol 10^{-5};0.1	2–19	54.4%	0.69	wlWd	
E_{10}	RowCol 10^{-5};0.1	2–19	40.3%	0.32	clW	
E_{11}	RowCol 10^{-5};0.1	2–19	23.2%	0.18	wlC	
E_{14}	RowCol 10^{-5};0.1	1–19	0.4%	0.47	wlWd	
E_{16}	RowCol 10^{-5};0.1	2–19	20.2%	0.39	clCd	keine Basisgram-matik
E_{17}	RowCol 10^{-5};0.1	2–9	17.2%	0.60	clCd	
E_{18}	RowCol 10^{-5};0.1	2	24.9%	0.43	clCd	

fortgeführt auf der nächsten Seite

fortgesetzt von der letzten Seite						
No	**Auswahl-funktion**	**Train-ing**	**Fehler**	**Genau-igkeit**	**Para-meter**	**Besonderheiten**
E_{19}	RowCol 10^{-5};0.1	2–19	17.6%	0.62	clCd	Parameter normali-siert
E_{20}	RowCol 10^{-5};0.1	2–19	16.7%	0.47	wclWCd	Rückfalltechnik
E_{21}	RCQ 10^{-5};0.1	2–19	20.0%	0.72	clCd	Auswahl komb.
E_{22}	RCQ 10^{-5};0.1	2–19	17.6%	0.56	wclWCd	Rückfalltechnik & Auswahl komb.
E_{23}	Quadratic	2–19	15.9%	0.51	clCd	

Tabelle E.1: Ergebnisse der Experimente

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

Black, Ezra, Fred Jelinek, John Lafferty, David M. Magerman, Robert Mercer und Salim Roukos. 1992. Towards history–based grammars: Using richer models for probabilistic parsing. In *Proceedings of the 5th DARPA Workshop on Speech and Natural Language*, New York. cmp-lg/9405007.

Bod, Rens. 1995. *Enriching Linguistics with Statistics: Performance Models of Natural Language*. Ph.D. thesis, University of Amsterdam.

Bod, Rens. 1996. Efficient algorithms for parsing the DOP model? A reply to Joshua Goodman. cmp-lg/9605031.

Brewka, Gerhard, Hans Werner Guesgen und Joachim Hertzberg. 1992. Constraint relaxation and nonmonotonic reasoning. Technical Report TR–92–002, ICSI, University of Berkeley. ftp://ftp.icsi.berkeley.edu/pub/global/techreports/1992/tr-92-002.ps.Z.

Brill, Eric. 1995. Transformation–based error–driven learning and natural language processing: A case study in part–of–speech tagging. *Computational Linguistics*, 21(4).

Bußmann, Hadumod. 1990. *Lexikon der Sprachwissenschaft*. Stuttgart: Alfred Kröner Verlag.

Charniak, Eugene. 1993. *Statistical Language Learning*. Cambridge, MA: The MIT Press.

Chomsky, Noam. 1965. *Aspects of the Theory of Syntax*. Cambridge: The MIT Press.

Collins, Michael John. 1996. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34th Annual Meeting of the ACL*, S. 27–38, Santa Cruz, CA. cmp-lg/9605012.

Collins, Michael John und James Brooks. 1995. Prepositional phrase attachment through a backed–off model. In *Proceedings of the 3rd Workshop on Very Large Corpora*. cmp-lg/9506021.

Dermatas, Evangelos und George Kokkinakis. 1995. Automatic stochastic tagging of natural language texts. *Computational Linguistics*, 21(2):137–163.

Drosdowski, Günther, Hrsg.. 1989. *Duden „Deutsches Universalwörterbuch“*. Mannheim/Leipzig/Wien/Zürich: Dudenverlag.

Earley, Jay. 1970. An efficient context–free parsing algorithm. *Communications of the ACM*, 13:94–102.

Eisner, Jason M. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics*, S. 340–345, Copenhagen, Denmark.

Eppinger, Bernd und Eberhard Herter. 1993. *Sprachverarbeitung*. München, Wien: Carl Hanser Verlag.

Flanagan, David. 1996. *Java in a Nutshell*. Sebastopol, CA: O'Reilly & Associates.

Franzier, L. und K. Rayner. 1982. Making and correcting errors during sentence comprehension: Eye movements in the analysis of structurally ambiguous sentences. *Cognitive Psychology*, 14:178–210.

Freuder, Eugene C. 1982. A sufficient condition for backtrack–free search. *Journal of the ACM*, 29(1):24–32.

Freuder, Eugene C. 1985. A sufficient condition for backtrack–bounded search. *Journal of the ACM*, 32(4):755–761.

Freuder, Eugene C. und Richard J. Wallace. 1992. Partial constraint satisfaction. *Artificial Intelligence*, 58:21–70.

Frühwirth, Tom, Alexander Herold, Volker Küchenhoff, Thierry Le Provost, Pierre Lim, Eric Monfroy und Mark Wallace. 1993. Constraint logic programming – an informal introduction. Technical report ECRC–93–5, European Computer–Industry Research Center, München.

Gazdar, Gerald und Chris Mellish. 1989. *Natural Language Processing in Prolog*. Wokingham et al.: Addison–Wesley.

Goodman, Joshua. 1996. Efficient algorithms for parsing the DOP model. cmp-lg/9604008.

Görz, Günther, Hrsg.. 1995. *Einführung in die künstliche Intelligenz*. 2. Auflage. Bonn et al.: Addison–Wesley.

Grinberg, Dennis, John Lafferty und Daniel Sleator. 1995. A robust parsing algorithm for link grammars. Technical Report CMU–CS–95–125, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. ftp://ftp.cs.cmu.edu/user/sleator/link-grammar/robust-tech-report.ps.gz.

Gruska, Jozef. 1992. Theoretische Informatik. Vorlesungsskript Universität Hamburg, Fachbereich Informatik.

Harper, Mary P. und Randall A. Helzerman. 1994. Managing multiple knowledge sources in constraint–based parsing of spoken language. Technical Report EE 94–16, School of Electrical Engineering, Purdue University, West Lafayette, IN.

Harper, Mary P., L. H. Jamieson, C. D. Mitchell, G. Ying, S. Potisuk, P. N. Srinivasan, R. Chen, C. B. Zoltowski, L. L. McPheters, B. Pellom und R. A. Helzerman. 1994. Integrating language models with speech recognition. In *Proceedings of the AAAI–94 Workshop on the Integration of Natural Language and Speech Processing*, S. 139–146.

Harper, Mary P., Leah H. Jamieson, Cala B. Zoltowski und Randall A. Helzerman. 1992. Semantics and constraint parsing of word graphs. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, S. 63–66.

Helzerman, Randall A. und Mary P. Harper. 1992. Log time parsing on the MasPar MP–1. In *Proceedings of the 6th International Conference on Parallel Processing*, S. 209–217.

Hertzberg, Joachim. 1995. KI–Lexikon: Anytime–Algorithmen. *Künstliche Intelligenz*, 1:40.

Hess, Michael. 1996. Computerlinguistik und Logikprogrammierung. *Künstliche Intelligenz*, 3:40–45.

Hopcroft, John E. und Jeffrey D. Ullman. 1979. *Introduction to automata theory, languages, and computation.* Reading, MA: Addison-Wesley.

Jelinek, Fred. 1990. Self–organizing language models for speech recognition. In *Readings in Speech Recognition.* Morgan Kaufmann, San Mateo, CA, S. 450–506.

Joshi, Aravind, Leon Levy und M. Takahashi. 1975. Tree adjunct grammars. *Journal of the Computer and System Sciences*, 10.

Karlsson, Fred. 1990. Constraint grammar as a framework for parsing running text. In *Proceedings of the 13th International Conference on Computational Linguistics*, S. 168–173, Helsinki.

Krenn, Brigitte und Christer Samuelsson. 1996. The linguist's guide to statistics. ftp://coli.uni-sb.de/pub/coli/doc/stat/stat_cl.ps.gz.

Kumar, Vipin. 1992. Algorithms for constraint satisfaction problems: A survey. *AI Magazine*, 13(1):32–44.

Lafferty, John, Daniel Sleator und Davy Temperly. 1992. Grammatical trigrams: A probabilistic model of link grammar. In *Proceedings of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language.* Also available as technical report CMU–CS–92–181.

Lahres, Bernhard. 1995. Case frames und CDG: Konzepte und Beispiele. Manuskript Universität Hamburg, Fachbereich Informatik.

Lahres, Bernhard. 1996. A semantic level of description for constraint dependency grammars. Diplomarbeit Universität Hamburg.

Lemay, Laura und Charles L. Perkins. 1996. *Teach yourself Java in 21 days.* Indianapolis, IN: Sams.net.

Li, Hang. 1996. A probabilistic disambiguation method based on psycholinguistic principles. cmp-lg/9606016.

MacDonald, Maryellen C. 1993. The interaction of lexical and syntactic ambiguity. *Journal of Memory and Language*, 32:692–715.

Mackworth, Alan K. 1977. Consistency in networks of relations. *Artificial Intelligence*, 8:99–118.

Mackworth, Alan K. 1992. The logic of constraint satisfaction. *Artificial Intelligence*, 58:3–20.

Mackworth, Alan K. und Eugene C. Freuder. 1985. The complexity of some polynominal network consistency algorithms for constraint satisfaction problems. *Artificial Intelligence*, 25:65–74.

Magerman, David M. 1994. *Natural Language Parsing as Statistical Pattern Recognition.* Ph.D. thesis, Stanford University.

Magerman, David M. 1995a. Book review: Statistical language learning by Eugene Charniak. *Computational Linguistics*, 21(1):103–111.

Magerman, David M. 1995b. Statistical decision–tree models for parsing. cmp-lg/9504030.

Magerman, David M. und Mitchell P. Marcus. 1991. Pearl: A probabilistic chart parser. In *Proceedings of the 5th Conference of the European Chapter of the Association for Computational Linguistics*, S. 15–20.

Marcus, M., B. Santorini und M. Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2). ftp://ftp.cis.upenn.edu/pub/treebank/doc/cl93.ps.gz.

Maruyama, Hiroshi. 1990a. Constraint dependency grammar. Technical Report RT0044, IBM Research, Tokyo Research Laboratory.

Maruyama, Hiroshi. 1990b. Structural disambiguation with constraint propagation. In *Proceedings of the 28th Annual Meeting of the ACL*, S. 31–38, Pittsburgh.

Maruyama, Hiroshi, Hideo Watanabe und Shiho Ogino. 1990. An interactive Japanese parser for machine translation. In *Proceedings of the 13th International Conference on Computational Linguistics*, S. 257–262, Helsinki.

Mellish, Chris S. 1989. Some chart–based techniques for parsing ill–formed input. In *Proceedings of the 27th Annual Meeting of the ACL*, S. 102–109, Vancouver.

Menzel, Wolfgang. 1994. Parsing of spoken language under time constraints. In A. Cohn, Hrsg., *Proceedings of the 11th European Conference on Artificial Intelligence*, S. 560–564, Amsterdam. cmp-lg/9409008.

Menzel, Wolfgang. 1995. Robust parsing of natural language. In *Proceedings of the 19th German Annual Conference on Artificial Intelligence*, S. 19–34, Berlin. cmp-lg/9507003.

Menzel, Wolfgang. 1996. Constraint satisfaction for robust parsing of spoken language. In Walter Hower und Zsófia Ruttkay, Hrsg., *Working Notes of the ECAI–96 workshop W27 Non–Standard Constraint Processing*, S. 49–60, Budapest.

Merialdo, Bernhard. 1994. Tagging English with a probabilistic model. *Computational Linguistics*, 20(2).

Meseguer, Pedro. 1989. Constraint satisfaction problems: An overview. *AI Communications*, 2(1):3–17.

Mohr, Roger und Thomas C. Henderson. 1986. Arc and path consistency revisited. *Artificial Intelligence*, 28:225–233.

Morik, Katharina, 1995. "Maschinelles Lernen". S. 243–297. Kapitel 3 von Görz (Görz, 1995), 2. Auflage.

Nadel, B. 1988. Tree search and arc consistency in constraint–satisfaction problems. In L. Kanal und V. Kumar, Hrsg., *Search in Artificial Intelligence*. Springer–Verlag, New York, S. 287–342.

Narayanan, Srini und Dan Jurafsky. 1996. Exploiting conditional independence in probabilistic models of language understanding. erscheint in Computational Linguistics.

Naumann, Sven und Hagen Langer. 1994. *Parsing*. Stuttgart: B.G. Teubner.

Nudel, Bernhard. 1983. Consistent–labeling problems and their algorithms: Expected complexities and theory–based heuristics. *Artificial Intelligence*, 21:135–178.

O'Shaughnessy, Douglas. 1990. *Speech Communications*. Reading, MA: Addison–Wesley.

Padró, Lluís. 1995. POS tagging using relaxation labelling. cmp-lg/9610001.

Pollard, Carl und Ivan Sag. 1994. *Head–driven Phrase Structure Grammar*. Chicago: The University of Chicago Press.

Resnik, Philip. 1992. Probabilistic tree–adjoining grammar as a framework for statistical natural language processing. In *Proceedings of the 14th International Conference on Computational Linguistics*, S. 418–424, Nantes, France.

Russell, Stuart J. und Shlomo Zilberstein. 1991. Composing real–time systems. In *Proceedings of the IJCAI*, S. 212–217, Sydney.

Samuelsson, Christer. 1996. Relating Turing's formula and Zipf's law. cmp-lg/9606013.

Samuelsson, Christer, Pasi Tapanainen und Atro Voutilainen. 1996. Inducing constraint grammars. cmp-lg/9607002.

Schabes, Yves. 1992. Stochastic lexicalized tree–adjoining grammars. In *Proceedings of the 14th International Conference on Computational Linguistics*, S. 426–432, Nantes, France.

Schabes, Yves und Richard C. Waters. 1995. Tree insertion grammar: A cubic–time, parsable formalism that lexicalizes context–free grammar without changing the trees produced. *Computational Linguistics*, 21(4).

Schömann, Munira. 1995. Menschliche Informationsverarbeitungsprozesse bei der Disambiguierung. Verbmobil–Report 53, Institut für Angewandte Sprachwissenschaft. http://www.dfki.uni-sb.de/verbmobil/htbin/doc-access.cgi.

Schöning, Uwe. 1992. *Theoretische Informatik kurz gefaßt*. Mannheim: BI Wissenschaftsverlag.

Schröder, Ingo. 1995. Analyse natürlicher Sprache durch Beschränkungserfüllung. Studienarbeit Universität Hamburg, http://wsv3.informatik.uni-hamburg.de/~ingo/sa/.

Sleator, Daniel D. und Davy Temperly. 1993. Parsing English with a link grammar. In *Proceedings of the 3rd International Workshop on Parsing Technologies*. ftp://ftp.cs.cmu.edu/user/sleator/link-grammar/LG-IWPT93.ps.gz.

Sleator, Daniel D. K. und Davy Temperley. 1991. Parsing English with a link grammar. Technical Report CMU–CS–91–196, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. ftp://ftp.cs.cmu.edu/user/sleator/link-grammar/LG-tech-report.ps.gz.

Stahel, Werner A. 1995. *Statistische Datenanalyse*. Braunschweig/Wiesbaden: Vieweg Verlag.

Tabossi, Patrizia und Francesco Zardon. 1993. Processing ambiguous words in context. *Journal of Memory and Language*, 32:359–372.

Thielen, Christine. 1995. Ein Tagset fürs Deutsche. Richtlinien für die manuelle Wortarten–Annotierung von Textkorpora. Entwurf vom 8.3.1995, Universität Tübingen.

Trueswell, John C., Micheal K Tanenhaus und Susan M. Garnsey. 1994. Semantic influences on parsing: Use of thematic role information in syntactic ambiguity resolution. *Journal of Memory and Language*, 33:285–318.

Vijay-Shanker, K. 1987. *A Study of Tree Adjoining Grammars*. Ph.D. thesis, Department of Computer and Information Science, Unversity of Pennsylvania.

Wahlster, Wolfgang. 1993. VERBMOBIL: Translation of face–to–face dialogs. In *Proceedings of the 3rd European Conference on Speech Communication and Technology*, S. 29–38, Berlin.

Wahlster, Wolfgang, Anthony Jameson, Alassane Ndiaye, Ralph Schäfer und Thomas Weis. 1995. Ressourcenadaptive Dialogführung: ein interdisziplinärer Forschungsansatz. *Künstliche Intelligenz*, 6:17–21.

Wall, Lary und Randal L. Schwartz. 1994. *Programming Perl*. Sebastopol, CA: O'Reilly & Associates.

Wallace, Richard J. und Eugene C. Freuder. 1995a. Anytime algorithms for constraint satisfaction and SAT problems. In *Proceedings of the IJCAI Workshop on Anytime Algorithm and Deliberation Scheduling*.

Wallace, Richard J. und Eugene C. Freuder. 1995b. Heuristic methods for over-constrained constraint satisfaction problems. In *Proceedings of the CP 1995 Workshop on Over–Constrained Systems*.

Waltz, David. 1975. Understanding line drawings of scenes with shadows. In P. Winston, Hrsg., *The Psychology of Computer Vision*. McGraw–Hill, New York, S. 19–91.

Wilson, Molly und Alan Borning. 1993. Hierarchical constraint logic programming. *The Journal of Logic Programming*, 16:277–318. ftp://cs.washington.edu/tr/1993/01/UW-CSE-93-01-02a.PS.Z.

Erklärung nach § 23 Absatz 9 der Diplomprüfungsordnung

Hiermit erkläre ich, daß ich die vorliegende Arbeit selbständig durchgeführt habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.