Syntaktische Analyse des Englischen mit JSLIM

Magisterarbeit

im Studiengang

Linguistische Informatik

(Magister Artium)

der

Friedrich-Alexander-Universität

Erlangen-Nürnberg

im

Fachbereich Philosophie und Theologie

Betreuer und Erstkorrektor:

Prof. Dr. Roland Hausser

Zweitkorrektor:

Prof. Dr. Thomas Herbst

vorgelegt von:

Paul Greiner

aus Berkeley (USA)

Erlangen, im September 2011

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

In natürlicher Sprache kommunizierende Maschinen sind in den Werken der Science-Fiction-Literatur allgegenwärtig.[1] In der Realität stellt sich diese Situation allerdings anders dar. Bis zum heutigen Tag ist die Kommunikation mit elektronischen Geräten aller Art relativ umständlich zu praktizieren und erfordert je nach Gerätetyp ein entsprechendes Maß an Training. Ob es sich dabei um die Bedienung eines handelsüblichen PCs per Maus und Tastatur handelt oder ob der Anruf einer Servicetelefonnummer zunächst von einem automatischen Dialogsystem beantwortet wird, von einer der Natur des Menschen entsprechenden Kommunikation ist diese Praxis weit entfernt. Dieser Sachverhalt erweckt das Interesse der modernen Computerlinguistik und lässt sich mit den folgenden Worten zusammenfassen:

> Das Ziel der Computerlinguistik ist es, die natürliche Informationsübertragung nachzubilden, indem die Sprachproduktion des Sprechers und die Sprachinterpretation des Hörers auf geeigneten Computern modelliert werden. [...]
>
> Die Entwicklung sprechender Roboter ist keine Fiktion, sondern eine reale, wissenschaftliche Aufgabe.
>
> Hausser (2000, S. 1)

Auch wenn dieses Ziel zum aktuellen Zeitpunkt noch in weiter Ferne zu liegen scheint, soll die vorliegende Arbeit den Versuch unternehmen, einen weiteren Grundstein zu legen. Dazu wird aufbauend auf einem bereits existierenden, in Kapitel 4 beschriebenen, morphologisch arbeitenden System der englischen Sprache eine grundlegende Syntaxkomponente skizziert, implementiert und evaluiert.

1. Als Beispiel seien hier die Werke des bedeutenden Autoren Isaac Asimov genannt. In seinen Werken beschäftigt er sich in fiktivem Rahmen intensiv mit der Integration synthetischer, kognitiver Agenten in die menschliche Gesellschaft.

1.1 Zielsetzung

Die Syntax natürlicher Sprachen bildet ein extrem komplexes und schwer in allgemeingültige Regeln zu fassendes System. Verdeutlicht wird dieser Umstand durch die Tatsache, dass eine Vielzahl verschiedener, auf unterschiedlichsten Grundsätzen aufbauender, Grammatiktheorien und -formalismen darum konkurrieren, möglichst umfangreiche und endgültige Regelwerke zu dieser Thematik zu erstellen.[2] Ein solch komplexes System kann im Rahmen einer Magisterarbeit nicht in seiner Vollständigkeit abgedeckt werden.

Deshalb setzt sich das in dieser Arbeit präsentierte System das Ziel, die Grundstrukturen des englischen Deklarativsatzes, wie sie von etablierter Literatur identifiziert werden, automatisch zu analysieren.[3] Darüber hinaus werden ebenjene Konstruktionen um eine Reihe von syntaktischen Phänomenen erweitert, die Hausser (2006, Kapitel II) entnommen wurden.

Zum einen wird auf diese Weise verifiziert, dass die in Hausser (2000) beschriebenen Formalismen bzw. Prinzipien der Linksassoziativen Grammatik sowie der SLIM-Sprachtheorie dafür geeignet sind, die Syntax der englischen Sprache automatisch zu verarbeiten. Zum anderen wird ein solides und vor allem ausbaufähiges Grundgerüst dafür gelegt, das Englische in seiner Gesamtheit maschinell zu erfassen und zu analysieren.

1.2 Aufbau der Arbeit

Die Implementierung eines Systems, das die Syntax der englischen Sprache analysieren soll, erfordert Kenntnisse auf zwei Gebieten: Zum einen muss ein fundiertes linguistisches Verständnis des Englischen vorliegen, um den Umfang eines solches Projekts abschätzen zu können und das grundsätzliche Vorgehen zu planen. Zum anderen sind umfassende Kenntnisse in entsprechenden Computerprogrammen erforderlich, in diesem Fall vor allem das an der Friedrich-Alexander-Universität unter Leitung von Prof. Dr. Roland Hausser entwickelte Programm JSLIM. Aus diesem Grund folgt der Aufbau dieser Arbeit dem im Folgenden beschriebenen Schema.

Zunächst soll in den Kapiteln 2, 3 und 4 theoretisches Hintergrundwissen zu den verschiedenen Themen vermittelt werden, die für das vorliegende Projekt von Interesse sind. Das zweite Kapitel beschäftigt sich

2. Vgl. hierfür bspw. die Formalismen der Kategorialgrammatik und der Phrasenstrukturgrammatik, beschrieben in Hausser (2000, S. 107).
3. Für eine genaue Definition dieser Grundstrukturen vergleiche Unterkapitel 2.4.1.

mit der englischen Sprache im Allgemeinen sowie ihrer Syntax im Besonderen. Kapitel drei liefert Hintergrundwissen über das Programm JSLIM und beschreibt die Grundprinzipien, auf denen es basiert. Das vierte Kapitel wird das dieser Arbeit zugrunde liegende Morphologieprojekt erläutern sowie einige Veränderungen an ebenjenem aufzählen und begründen.

Darauf folgend wird der praktische Teil dieser Arbeit behandelt. Kapitel 5 legt hierzu Aufbau und Funktionsweise des Syntaxprojekts der englischen Sprache detailliert dar, woraufhin eine abschließende Evaluierung in Kapitel 6 folgt.

2 Die englische Sprache

Dieses Kapitel liefert grundlegende Informationen über die englische Sprache. Zunächst soll ihre aktuelle Situation dargestellt und darauf folgend ihre historische Entwicklung erläutert werden. Abschließend werden Regeln und Funktionsweise der englischen Syntax dargelegt, soweit sie für die Implementierung des JSLIM-Projekts von Interesse sind.

2.1 Aktuelle Situation

Herbst (2010, S.1) zufolge ist die englische Sprache zum heutigen Zeitpunkt zwar nicht die Sprache mit den meisten Muttersprachlern – diese Stellung nimmt das Chinesische ein – allerdings die verbreitetste und wichtigste Sprache in der internationalen Kommunikation. Als solch moderne *lingua franca* spielt sie in der heutigen globalisierten Welt eine wichtige Rolle in den Bereichen Wirtschaft, Politik und Bildungswesen mit grob geschätzt etwa 1,5 Milliarden Sprechern verteilt auf allen Kontinenten.[1]

Die nachfolgende Auflistung verdeutlicht die aktuelle Verbreitung der englischen Sprache, gegliedert nach der Art des Spracherwerbs, in quantitativer sowie regionaler Hinsicht:[2]

- Englisch als Muttersprache: Etwa 400 Millionen Sprecher; vor allem in Nordamerika, auf den Britischen Inseln, in Australien und Neuseeland.
- Englisch als Zweitsprache: Etwa 400 Millionen Sprecher; beispielsweise in Teilen Kanadas oder auch in Indien.
- Englisch als Fremdsprache: Etwa 600 – 700 Millionen Sprecher; verteilt über die ganze Welt.

1. Vgl. Herbst (2010, S. 1) sowie Quirk u. a. (1990, S. 3).
2. Zusammengefasst aus Herbst (2010, S. 1f.) nach Quirk u. a. (1990) sowie Crystal (1995).

Eine detaillierte Karte über die heutige Verbreitung des Englischen weltweit findet sich in Crystal (1988, S. 8-9). Als bedeutendste Gründe für diese Verbreitung gelten vor allem die Ausdehnung des *British Empire* im 18. und 19. Jahrhundert, die wirtschaftlich dominante Rolle der USA im 20. Jahrhundert, die rasche Entwicklung im Bereich der Informations- und Kommunikationstechnologie, wie auch der aktuell vorherrschende wirtschaftliche Trend der Globalisierung.[3]

2.2 Historische Entwicklung

Die Geschichte der englischen Sprache beginnt im fünften Jahrhundert als germanische Stämme, vor allem Angelsachsen und Juten, von den Küsten des heutigen Norddeutschlands sowie Dänemarks aus an den südöstlichen Küsten Großbritanniens landen. Dort beginnen sie zu siedeln und die Kelten Richtung Westen sowie Norden zu vertreiben.[4]

Die daraus resultierende Entwicklung der englischen Sprache vom sechsten Jahrhundert an wird für gewöhnlich in vier Phasen eingeteilt, nachzulesen in grob stichpunktartiger Fassung in Herbst (2010, S. 6) sowie in ausgiebigerer Abhandlung bei Crystal (1995, Kapitel 3-6). Die Erkenntnisse dieser beiden Autoren sollen in den folgenden vier Abschnitten kurz wiedergegeben werden.

Old English (ca. 600 – 1100)

Diese Form des Englischen orientiert sich stark an der Sprache der germanischen Eroberer und bedient sich dementsprechend eines größtenteils germanisch geprägten Vokabulars. Ob ein Einfluss des lateinischen Lexikons auf diese Phase zurückzuführen ist, oder ob dieser bereits vor dem siebten Jahrhundert stattfand, ist zum heutigen Zeitpunkt unklar.

Im Hinblick auf die Syntax ist festzustellen, dass diese Form des Englischen noch über ein stark ausgeprägtes Flexionssystem und relativ freie Wortstellung verfügte. Bußmann (2008, S. 162) zufolge umfasst dieses Flexionssystem drei Genera, zwei Numeri und vier Kasus.

Middle English (ca. 1100 – 1500)

In dieser Periode nimmt die Verwendung von Flexionsformen stark ab, womit die Durchsetzung einer vergleichsweise streng definierten Satz-

3. Vgl. Herbst (2010, S. 3).
4. Vgl. Crystal (1995, S. 6).

stellung einhergeht.[5] SVO etabliert sich in dieser Zeit als vorherrschende Form der Satzkonstruktion.[6]

Nach der Schlacht bei Hastings im Jahre 1066 nimmt das Französische starken Einfluss auf die englische Sprache, da Wilhelm der Eroberer seine Muttersprache, Französisch, als Sprache der englischen Oberschicht durchsetzt. Dieser Zustand hält bis ins 14. Jahrhundert an und beeinflusst vor allem das englische Vokabular durch französische Lehnwörter. Darüber hinaus erweitern erneut lateinische Lehnwörter, vor allem aus religiösen Kontexten, das Vokabular des Englischen.

Early Modern English (ca. 1500 – 1750)

In dieser Zeit findet der *Great Vowel Shift* statt, in dessen Zuge alle sieben mittelenglischen Langvokale entweder angehoben oder diphtongiert werden. Die Gründe hierfür sind, genau wie die exakte Reihenfolge der Veränderungen, umstritten.[7]

Darüber hinaus ist diese Phase geprägt durch die Einführung der Druckerpresse durch William Caxton im Jahr 1476, die die Vervielfältigung und weite Verbreitung von Texten wie der *King James Bible* oder den Werken Shakespeares ermöglichte. Als Folge dieser Entwicklung werden Rechtschreibung und Interpunktionsregeln im Laufe dieser Epoche standardisiert.

Modern English (ab ca. 1750)

In dieser Phase entstehen die ersten Grammatiken der englischen Sprache, deren Autoren sich vor allem an älteren Werken über das Griechische und Lateinische orientieren. Über lange Zeit bleibt der präskriptive Charakter dieser Regelwerke erhalten und wird erst gegen Ende des 19. Jahrhunderts nach und nach durch ein deskriptives Vorgehen ersetzt.[8]

Dieser Umstand führt zur weiteren Festigung sowohl morphologischer, vor allem aber syntaktischer Strukturen und ist sicher mitverantwortlich dafür, dass die Grundstrukturen der englischen Grammatik, zumindest was weit verbreitete Standardvarietäten betrifft, seit etwa der Mitte dieser Epoche bis zum heutigen Tag relativ unverändert blieben.

5. Verglichen mit diesem Phänomen sei an dieser Stelle beispielsweise die relativ freie Wortstellung des Deutschen, die sich bis zum heutigen Tag erhalten hat.
6. Zur Satzstellung des englischen Deklarativsatzes vgl. Abschnitt 2.4.1.
7. Vgl. Crystal (1995, S. 55).
8. Vgl. Crystal (1995, S. 78f.).

2.3 Herkunft und sprachtypologische Einordnung

Wie bereits festgestellt, basiert das Englische auf west- bzw. nordwestlichen Dialekten des Germanischen. Somit lässt es sich in die Familie der indo-europäischen Sprachen einordnen, wie bei Hogg und David (2006, S. 5) nachzulesen.

Typologisch betrachtet identifiziert Herbst (2010, S. 9f.) die drei im Folgenden beschriebenen Gruppen von Sprachen.[9] Sogenannte **agglutinierende Sprachen** zeichnen sich dadurch aus, dass die meisten linguistischen Phänomene im Bereich der Morphologie behandelt werden. Benötigt dafür wird eine Vielzahl an Affixen (zumeist Suffixe) mit eindeutiger Bedeutung, die nacheinander an eine Wortform angehängt werden und so seine Bedeutung modifizieren.

Die **isolierenden Sprachen** bilden das Gegenstück zu vorhergehender Gruppe und tendieren stark dazu, linguistische Phänomene in den Bereich der Syntax zu verlagern. Der Verzicht auf Flexionsendungen erzwingt eine stark reglementierte Satzstellung.

Die dritte Gruppe bilden die sogenannte **flektierenden Sprachen** wie das Deutsche. Ähnlich den agglutinierenden Sprachen bedienen sich die Mitglieder dieser Gruppe einer Vielzahl von Affixen sowie der Praxis der Wortstammveränderung. Der Unterschied zu Ersteren liegt darin, dass die Bedeutung dieser Affixe mehrfach belegt sein kann, wodurch die Satzstruktur dieser Sprachen einigen Restriktionen unterliegt.

Das Englische seinerseits entwickelt sich laut Herbst (2010, S.10), wie auch bereits in Kapitel 2.2 angeführt, weg vom flektierenden Sprachgebrauch hin zum isolierenden und nimmt damit zum heutigen Tag eine Stellung zwischen den beiden letzten beiden Gruppen ein. In den Worten von Hogg und Davis:

> In its earliest stages English was a heavily inflected language with a relatively free word order [...] rather like modern German today. A host of changes over the centuries has made it into what it is today: a language radically different from that of German.
>
> Hogg und David (2006, S. 109)

2.4 Die Syntax des englischen Deklarativsatzes

In diesem Abschnitt werden die syntaktischen Elemente und Konstruktionen des Englischen erläutert, sofern für die vorliegende Arbeit relevant. Zunächst wird ein Überblick über den Aufbau des Deklarativsatzes

9. Basierend auf den Werken von Wilhelm von Humboldt und August Schlegel, nach Crystal (1988, S. 367), sowie Bußmann (2008, S. 664-666).

gegeben und daraufhin Funktion und Zusammensetzung seiner einzelnen Bestandteile dargelegt.

2.4.1 Grundsätzlicher Aufbau

Sowohl Quirk u. a. (1990, S. 720) als auch Crystal (1995, S. 221) identifizieren für den englischen Deklarativsatz sieben sogenannten *major* bzw. *basic clause types*. Letzterer begründet diese Auswahl wie folgt:

> Clause elements combine into a very small number of patterns. In fact, most sentences can be analysed into one of only seven basic clause types, each minimally consisting of two, three, or four elements.
>
> Crystal (1995, S. 221)

Beide Werke reduzieren diese Standardkonstruktionen auf Kombinationen der folgenden fünf Satzteile:

- S(*ubject*)
- V(*erb*)
- O(*bject*), sowohl direkt als auch indirekt
- C(*omplement*), entweder auf Subjekt oder Objekt bezogen
- A(*dverbial*), in verschiedenster Ausprägung

Die nachfolgende Tabelle 2.1 zählt diese sieben identifizierten Standardfälle auf und illustriert sie an jeweiligen Beispielsätzen:

Tabelle 2.1: Die sieben major clause types *der englischen Sprache*

Satzkonstruktion	Beispielsatz
S V	The dog / sleeps.
S V O	The dog / sees / the ball.
S V C	The dog / is / sleepy.
S V A	The dog / sleeps / in the basket.
S V O O	The man / gives / the dog / the ball.
S V O C	The man / got / his shoes / wet.
S V O A	The girl / eats / the apple / on the table.

Aufbau und Funktion der einzelnen Satzbestandteile werden in den folgenden Kapiteln erläutert.

2.4.2 Subject

Quirk u. a. (1990, S. 724) sprechen dem Subjekt die nach dem Verb bedeutendste Stellung im Satzbau zu, da seine Präsenz im deklarativen Hauptsatz zwingend erforderlich ist. Darüber hinaus kann es in bestimmten Fällen Kongruenzbeziehungen zu beinahe allen anderen Satzteilen

definieren. Für die eingeschränkte Modellwelt des vorliegenden Projekts kann das Subjekt nach den folgenden Kriterien charakterisiert werden:[10]

- **Aufbau:**
  Es wird üblicherweise durch eine Nominalphrase repräsentiert.
- **Position:**
  Es steht direkt vor dem Verb.
- **Semantik:**
  Im Deklarativsatz nimmt es für gewöhnlich die Rolle eines handelnden Agenten ein.
- **Kongruenz:**
  Es steht in Kongruenz mit dem Verb und bestimmt somit dessen Numerus und Person. Darüber hinaus bestimmt es, falls notwendig, Numerus, Person und Genus von Objekten sowie *Complements*.

In den folgenden Beispielsätzen wurden die Subjekte markiert, um obenstehende Charakteristika zu illustrieren:

[He] sleeps.
[The man] sleeps.
[The big man] sleeps.

2.4.3 Verb

Das Verb nimmt insofern eine dominante Stellung im englischen Satzbau ein, als es durch seine Valenz sowie durch andere Merkmale das Vorhandensein sowie die Reihenfolge weiterer Satzglieder definiert. In anderen Worten:

> The fundamental idea of valency theory [...] is that the structure of a sentence is largely determined by the verb. This means that certain properties of the verb have a determining influence on the structure of a clause.
>
> Herbst und Schüller (2008, S. 21)

Verben fallen grundsätzlich in die Klassen der **transitiven** oder **intransitiven Verben**, wobei sich letztere dadurch auszeichnen, dass sie kein Objekt erfordern. Ein Standardbeispiel für ein solches Wort ist das Verb *sleep*, das in einer Konstruktion wie *„The child sleeps.“* einzig und allein durch ein Subjekt begleitet auftreten kann.

Herbst u. a. (1990) zufolge werden transitive Verben entsprechend ihrer Valenz in eine oder mehrere von vier Gruppen eingeordnet. Im

10. Ausgewählt und zusammengefasst aus Quirk u. a. (1990, 724-726).

Folgenden werden diese Gruppen erläutert und ihren jeweiligen Satzkonstruktionen zugeordnet:[11]

- **Monotransitive Verben** erfordern ein direktes Objekt (SVO).
- **Ditransitive Verben** kommen mit direkten und indirekten Objekten vor (SVOO).
- **Komplex transitive Verben** bilden Konstruktionen sowohl mit direktem Objekt als auch zugehörigem *object complement* (SVO oder SVA).
- **Kopulaverben** stehen in Kombination mit einem *subject complement* oder einer obligatorischen Adverbiale (SVOC oder SVOA).

2.4.4 Object

Diese Sektion unterteilt sich in indirekte und direkte Objekte, die eine Reihe an Gemeinsamkeiten aufweisen. Für die vorliegende Arbeit können sie wie folgt charakterisiert werden:[12]

- **Aufbau:**
  Sie werden üblicherweise durch eine Nominalphrase repräsentiert.
- **Position:**
  Objekte stehen für gewöhnlich direkt hinter dem Verb. Ist nur ein Objekt vorhanden, so ist dieses normalerweise das direkte Objekt. In Konstruktionen, in denen beide Arten von Objekten gemeinsam auftreten, folgt üblicherweise das direkte Objekt dem indirekten.
- **Semantik:**
  Das direkte Objekte übernimmt für gewöhnlich die Rolle eines von der beschriebenen Aktion beeinflussten oder auf andere Art und Weise involvierten Gegenstands bzw. einer entsprechenden Person. Das indirekte Objekt hingegen, besetzt üblicherweise die Rolle des Rezipienten einer solchen Aktion.

Im Folgenden werden die oben angeführten Merkmale durch zwei Beispielsätze veranschaulicht. Ersterer enthält nur ein direktes Objekt, letzterer ein indirektes, gefolgt von einem direkten Objekt:

The woman buys [the book].
The woman gives [the boy] [the book].

11. Vgl. Herbst u. a. (1990, S. 146). Für illustrierende Beispielsätze vgl. diese Auflistung mit Tabelle 2.1.
12. Ausgewählt und zusammengefasst aus Quirk u. a. (1990, 726-728).

2.4.5 Complement

Auch diese Kategorie teilt sich in zwei Gruppen: Das *Subject Complement* sowie das *Object Complement*. Wie die Objekte teilen auch diese beiden Gruppen eine Reihe von Merkmalen, die, soweit für das vorliegende Syntaxprojekt interessant, im Folgenden zusammengefasst werden sollen:[13]

- **Aufbau:**
  Complements können durch Nominal- oder auch Adjektivphrasen realisiert werden. Letzteres ist ein wesentliches Unterscheidungsmerkmal zu den vorher beschriebenen Objekten.
- **Position:**
  Das *Subject Complement* steht für gewöhnlich direkt hinter dem Verb, das *Object Complement* folgt üblicherweise auf das direkte Objekt.
- **Semantik:**
  Complements definieren, modifizieren bzw. attributieren das jeweils zugehörige Subjekt bzw. Objekt.

Untenstehende Beispielsätze sollen diese Zusammenfassung verdeutlichen. Der erste illustriert ein *Subject Complement*, realisiert durch ein Adjektiv, der zweite ein *Subject Complement*, das durch eine Nominalphrase gebildet wird.

The book is [old].
The woman is [a teacher].

2.4.6 Adverbial

Auch diese, Quirk u. a. (1990, S. 729) zufolge heterogenste der fünf Gruppen, wird in nachfolgender Auflistung zusammengefasst:[14]

- **Aufbau:**
  Adverbials können sowohl als Adverbialphrasen, Präpositionalphrasen oder Adverbialsätze realisiert werden. Das vorliegende Projekt beschränkt sich auf die ersten beiden Phänomene.
- **Position:**
  Grundsätzlich weisen *Adverbials* eine wesentlich höhere Flexibilät als andere Satzglieder auf, was ihre Position betrifft. In Bezug auf die in Abschnitt 2.4.1 identifizierten *major clause types* jedoch ist diese stark eingeschränkt. *Adverbials* in Kombination mit komplex

13. Ausgewählt und zusammengefasst aus Quirk u. a. (1990, 724-726).
14. Ausgewählt und zusammengefasst aus Quirk u. a. (1990, S. 729f.).

transitiven Verben folgen direkt dem Verb, wohingegen solche in von Kopulaverben beherrschten Konstruktionen auf das direkte Objekt folgen.[15]

- **Semantik:**
  Die Rollen von *Adverbials* können so vielfältig sein wie ihre Erscheinungsformen oder Positionen. Grundsätzlich liefern sie Information über besondere Umstände einer gegebenen Situation oder bilden ein semantisches Bindeglied zwischen verschiedenen, formal in sich geschlossenen Sätzen.

Die folgenden beiden Beispielsätze sollen diese Charakteristika illustrieren. Ersterer zeigt ein *Adverbial*, das durch eine aus einem einzigen Adverb bestehende Nominalphrase realisiert wird und in eine Konstruktion der Form SVA eingebunden ist. Letzterer verdeutlicht ein *Adverbial*, das in Form einer Präpositionalphrase auftritt und in das Satzmuster SVOA integriert ist.

The girl is [sleepy] .
The girl listens [to the music] .

15. Für Informationen bezüglich der Transitivität von Verben vergleiche Abschnitt 2.4.3 sowie Herbst u. a. (1990, S. 145f).

3 JSLIM und die zugrunde liegende Sprachtheorie

Das der Implementierung des Syntaxprojekts dieser Arbeit zugrunde liegende Programm JSLIM wurde an der Abteilung Computerlinguistik der Friedrich-Alexander-Universität Erlangen-Nürnberg unter der Leitung von Prof. Dr. Roland Hausser entwickelt. Die Funktionsweise von JSLIM folgt den von Hausser (2000, 1992) entwickelten Prinzipien der SLIM-Sprachtheorie sowie der Linksassoziativen Grammatik. In diesem Kapitel werden zunächst diese beiden rein theoretischen Themenkomplexe erläutert, bevor JSLIM selbst beschrieben wird.

3.1 Linksassoziative Grammatik

Der Formalismus der Linksassoziativen Grammatik (LAG) wurde ursprünglich in Hausser (1985) skizziert. Ausführlichere und aktuellere Informationen zu diesem Thema finden sich in Hausser (2000) und werden im Folgenden kurz zusammengefasst.

Die LAG zeichnet sich vor allem dadurch aus, dass sie sich im Gegensatz zu anderen Grammatikformalismen, wie der Kategorialgrammatik oder der Phrasenstrukturgrammatik, dem *Prinzip der möglichen Fortsetzungen* verpflichtet. Letztere hingegen folgen dem *Prinzip der möglichen Ersetzungen* und verstoßen damit gegen die Zeitlinearität sprachlicher Äußerungen. Hausser beschreibt diese Gegebenheit mit den folgenden Worten:

> Die linksassoziative Ableitungsordnung ist linear in dem Sinn, dass sie ganz regelmäßig immer nur ein Wort nach dem anderen anfügt, und zeitlinear in ihrer Wachstumsrichtung. Somit erfasst ein linksassoziativer Formalismus die zeitlineare Grundstruktur sprachlicher Zeichen.
>
> Hausser (2000, S. 202)

Den Formalismus der LAG beschreibt Hausser (2000, S. 205) als 7-Tupel der Form <W, C, LX, CO, RP, ST_S, ST_F>. Der folgende Auszug erläutert die einzelnen Bestandteile dieses Tupels, wobei zu beachten ist, dass gilt: $n = \{i \mid 0 \leq i \leq n\}$.

> 1. W ist eine endliche Menge von *Wortformoberflächen*.
> 2. C ist eine endliche Menge von *Kategoriesegmenten*.
> 3. $LX \subset (W \times C^+)$ ist eine endliche Menge, die das *Lexikon* umfasst.
> 4. $CO = (co_0 \ldots co_{n-1})$ ist eine endliche Sequenz aus total rekursiven Funktionen aus $(C^* \times C^+)$ in $C^* \cup \{\perp\}$, genannt *kategoriale Operationen*.
> 5. $RP = (rp_0 \ldots rp_n - 1)$ ist eine ebenso lange Sequenz von Untermengen von n, genannt *Regelpakete*.
> 6. $ST_S = \{(cat_s\ rp_s), \ldots\}$ ist eine endliche Menge von *Anfangszuständen*, wobei jedes rp_s, genannt Anfangsregelpaket, eine Untermenge von n ist und jedes $cat_s \in C^+$.
> 7. $ST_F = \{(cat_f\ rp_f), \ldots\}$ ist eine endliche Menge von *Endzuständen*, wobei jedes $cat_f \in C^*$ und jedes $rp_f \in RP$.
>
> Hausser (2000, S. 205f.)

Aus diesen sieben grundlegenden Bestandteile lassen sich vier Gruppen abstrahieren, deren Zusammensetzung und Aufgabengebiet innerhalb eines syntaktisch arbeitenden Systems in der folgenden Auflistung weniger formal erläutert werden sollen:[1]

- Die Mengen W und C vereinigen sich als Lexikon LX. Dieses Lexikon stellt eine Anzahl an dem System bekannten Wortformen zur Verfügung und ordnet ihnen spezielle Kategorien zu, um ihr Verhalten innerhalb einer sprachlichen Äußerung zu repräsentieren.

- Die sechste Menge, ST_S, stellt eine Reihe von Anfangszuständen dar. Diese definieren, mit welchen Wortformen, abgeglichen durch die jeweils zugeordnete Kategorie, eine Ableitung beginnen darf. Darüber hinaus bestimmen diese Anfangszustände, mit welchen Regelanwendungen die Analyse beginnt.

- Aus den kategorialen Operationen und Regelpaketen wird eine Menge von Regeln gebildet. Diese Regeln definieren zum einen genau, welche Wortformen, abgeglichen über die oben angesprochenen Kategorien, miteinander kombiniert werden dürfen und inwiefern diese Kategorien bei erfolgreichem Abgleich angepasst werden. Zum anderen enthält jede dieser Regeln ein Regelpaket mit möglichen Folgeregeln, wodurch die Reihenfolge von Ableitungen restringiert wird.

1. Vgl. Hausser (2000, S. 206).

- Die möglichen Endzustände ST_F definieren, mit welcher Regelanwendung und welcher Kategorie (üblicherweise modifiziert durch die oben beschriebenen Regeln) eine Ableitung erfolgreich abgeschlossen werden kann. Wird kein passender Endzustand erreicht, so schlägt die Ableitung fehl.

In Abbildung 3.1 wird eine linksassoziative Ableitung für die englischsprachliche Äußerung „Mary gives Fido a bone“ illustriert.[2] Im Folgenden werden die einzelnen Bestandteile dieser Grafik beschrieben.

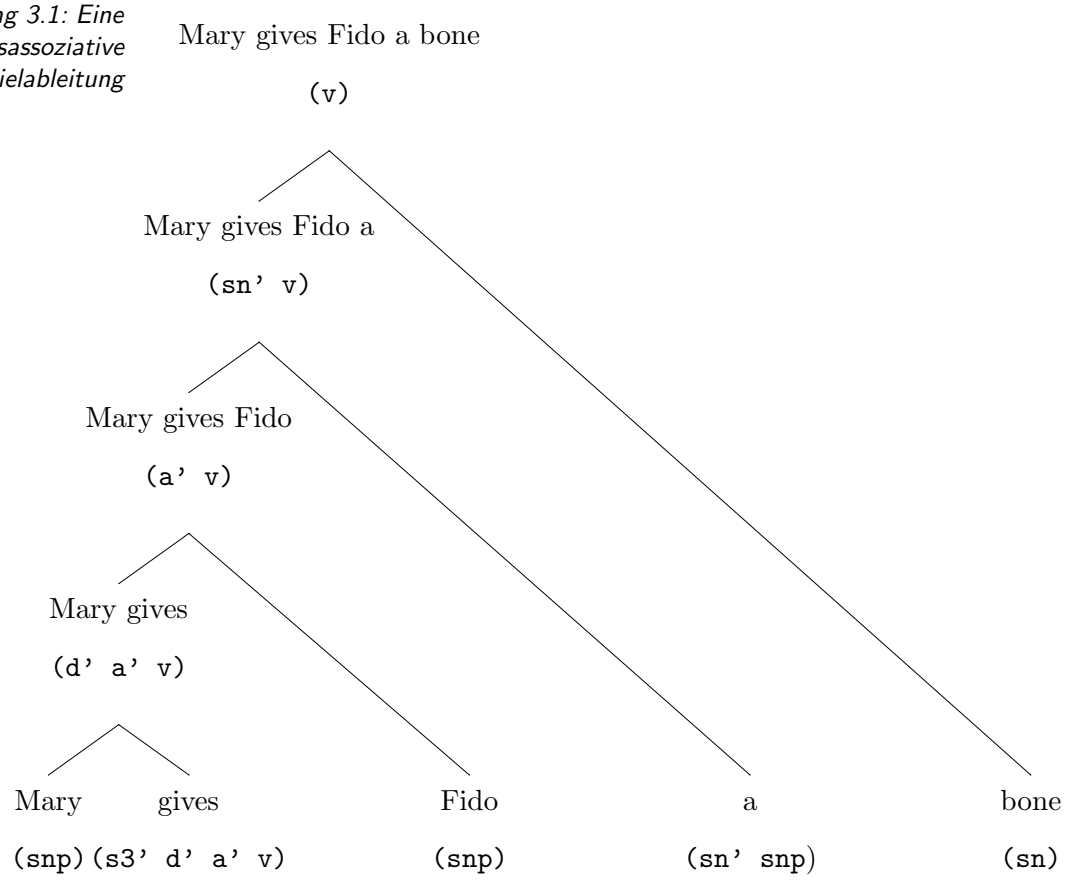
Die untersten beiden Zeilen zeigen die Reihenfolge der Eingabewörter und repräsentieren auch die für diese Analyse benötigten Lexikoneinträge. Unter den jeweiligen Wortformoberflächen *Fido*, *gives*, *Mary*, *a* und *bone* finden sich jeweils in Klammern angegeben die zugeordneten Kategorielisten. `snp` steht dabei für *singular noun phrase*, `v` für *verb* und `sn` für *singular noun*. Die mit Apostroph gekennzeichneten Kategoriesegmente erfordern eine Kürzung. Die Kategorie des Verbs fordert ein Subjekt in der dritten Person Singular (`s3'`) sowie ein indirektes (`d'`) und ein direktes (`a'`) Objekt, die des Artikels ein Nomen im Singular (`sn'`).[3]

Wie bereits angemerkt folgt die eigentliche Analyse streng der Zeitlinearität, was in der Schriftsprache der Leserichtung von links nach rechts entspricht. Zunächst findet ein Abgleich mit den definierten Anfangszuständen statt. Ein passender Anfangszustand muss die Kategorie der Wortform *Mary* als Satzanfang akzeptieren sowie eine Regel enthalten, die es erlaubt, entsprechende Kategorie mit der des folgenden Verbs zu kombinieren. Ist dieser Abgleich erfolgreich, so wird die Regel ausgeführt. Dies hat zur Folge, dass *Mary* und *gives* den neuen Satzanfang bilden, wobei ihre jeweiligen Kategorien von den in der Regel spezifizierten Operationen modifiziert werden. In diesem Fall wird das Segment `sn3'` gekürzt, da *Mary* die vom Verb geforderte Position des Subjekts füllt.

Nach dieser erfolgreichen Kombination gibt das Regelpaket der eben ausgeführten Regel eine Reihe möglicher Folgeregeln an. Akzeptiert eine dieser Regeln die Kategorien des aktuellen Satzanfangs sowie des nächsten Wortes, so werden auch sie miteinander kombiniert und die Kategorien den Operationen der Regel entsprechend angepasst. An

2. Entnommen aus Hausser (2000, S. 216). Für eine ausführlichere Erläuterung siehe ebendort. In der Praxis hat es sich durchgesetzt, dass Großbuchstaben in Kategoriedefinitionen für Variablen verwendet werden, die eine Reihe vordefinierter oder beliebiger Werte annehmen können. Eigentliche Kategoriewerte hingegen werden in der praktischen Implementierung in Kleinbuchstaben definiert. Die Kategoriewerte in Abbildung 3.1 wurden dementsprechend angepasst.
3. Für eine ausführliche Auflistung aller Kategoriewerte sowie deren Bedeutung vgl. Bauer (2011, S. 71ff.).

Abbildung 3.1: Eine linksassoziative Beispielableitung



dieser Stelle ist anzumerken, dass auch mehrere Folgeregeln auf die Kategorien des Satzanfangs und des nächsten Wortes passen könnten. In einem solchen Fall werden alle passenden Ableitungspfade parallel verfolgt, bis ein oder mehrere zu einer erfolgreichen Analyse führen bzw. alle scheitern.

Auf diese Weise werden nun sukzessiv die einzelnen Wortformen aneinandergereiht, bis kein nächstes Wort mehr gefunden wird. In diesem Fall werden die aktuelle Kategorie des Satzanfangs sowie das Regelpaket der zuletzt ausgeführten Regel mit der Menge der möglichen Endzustände verglichen. Findet sich ein Endzustand, der sowohl die Kategorie als auch das Regelpaket akzeptiert, so wird die Ableitung erfolgreich abgeschlossen.

3.2 SLIM-Theorie

Die SLIM-Sprachtheorie wird in Hausser (2000) vorgestellt und modelliert die Verarbeitung natürlicher Sprachen. SLIM bildet hierbei ein Akronym, das auf vier Prinzipien referenziert, die im Folgenden nach Hausser (2000, S. 9) zusammengefasst werden.

Das methodische Prinzip der Oberflächenkompositionalität (***S**urface compositional*) besagt, dass auf die Verwendung von Null-Elementen,

Identitätsabbildungen und Transformationen verzichtet wird. Dem empirischen Prinzip der Zeitlinearität (***L**inear*) zufolge hält sich die Analyse von Sprache an die Reihenfolge der Eingabewörter. Das ontologische und das funktionale Prinzip (***I**nternal* und ***M**atching*) arbeiten in der SLIM-Theorie eng zusammen. Ersterem zufolge wird die Analyse von Sprache als Sprecher-Hörer-interner Prozess analysiert, letztere modelliert sprachliche Referenz als Abgleichung zwischen wörtlicher Sprachbedeutung und Verwendungskontext.

Darüber hinaus ist die Namensvergabe SLIM auch ihrer wörtlichen Bedeutung entsprechend zu verstehen:

> Zusätzlich zur Interpretation seiner einzelnen Buchstaben ist das Akronym SLIM auch als Wort im Sinne von *schlank* motiviert. Denn in ihrer mathematischen und programmiertechnischen Ausarbeitung erweist sich die SLIM-Sprachtheorie in den Bereichen Syntax, der semantischen Interpretation und der Pragmatik als effizient [...].
>
> Hausser (2000, S. 9)

3.3 Das Programm JSLIM

Einen Vorläufer dieses Programms stellt das System JLAG dar, das ähnliche Funktionalität bereits in Grundzügen skizziert und von Kycia (2004) im Rahmen der Magisterarbeit *Implementierung der Datenbanksemantik für die natürlichsprachliche Kommunikation* entwickelt wurde. Eine wesentlich umfangreichere Umsetzung unter dem Namen JSLIM wurde darauf folgend von Handl u. a. (2009) vorgenommen und in Handl (voraussichtlich 2012) detailliert beschrieben.

JSLIM wurde in der Programmiersprache Java implementiert und setzt die in Unterkapitel 3.2 und 3.1 beschriebenen Formalismen und Prinzipien um.[4] Das Programm ist fähig, sowohl auf morphologischer als auch syntaktischer Ebene zu arbeiten. Eine Auswahl bereits implementierter JSLIM-Projekte findet sich in Tabelle 3.1.

3.3.1 JSLIM als Parser

Das System erfüllt den Zweck eines in Hausser (2000, S. 22) beschriebenen Parsers. Seiner Definition zufolge besteht die Aufgabe bzw. Funktionsweise eines solchen Programms aus drei Teilbereichen: Zerlegen, Klassifizieren und Zusammensetzen.

4. Daher rührt der Name des Programms: ***J**ava **S**urface compositional **L**inear **I**nternal **M**atching.*

Tabelle 3.1: Eine Auswahl bereits implementierter JSLIM-Projekte

Thematik	Sprache	Autor
syntaktisch	italienisch	Weber (2011)
morphologisch	englisch	Bauer (2011)
syntaktisch	französisch	Kosche (2011)
morphologisch	schwedisch	Lipp (2010)
morphologisch	rumänisch	Pandea (2010)
morphologisch	polnisch	Niedobijczuk (2009)
morphologisch	französisch	Pepiuk (2009)
morphologisch	arabisch	ben Zineb (2009)
syntaktisch	mehlhaff	Mehlhaff (2007)
morphologisch	italienisch	Weber (2007)

Im ersten der drei Schritte wird die aus einem komplexen Zeichen bestehende Eingabe in ihre elementaren Komponenten aufgespalten. Diese einzelnen Bestandteile werden daraufhin mit einem bestehenden Lexikon abgeglichen und klassifiziert. Der dritte Schritt fügt die Bestandteile nun wieder zusammen und leitet aus den Klassifizierungen der einzelnen Teile sowie deren Beziehungen zueinander eine grammatische Gesamtanalyse des komplexen Zeichens ab.

3.3.2 Syntaktische Analyse mit JSLIM

Bevor in Kapitel 5 das Syntaxprojekt dieser Arbeit ausführlich beschrieben wird, liefert dieser Abschnitt Informationen über die grundsätzliche Funktionsweise der Syntaxkomponente von JSLIM. Um einen Überblick darüber zu erhalten, mit welchen Bausteinen dieses System arbeitet, soll zunächst kurz illustriert werden, wie die vom morphologisch arbeitenden Teil des Systems bereitgestellten Lexikoneinträge aussehen. Danach werden die tatsächlich für die syntaktische Verarbeitung benötigten Bestandteile erläutert.[5]

Die Grundbausteine

Wie in Bauer (2011) ausführlich beschrieben, basiert die Morphologiekomponente von JSLIM auf einem möglichst großen sogenannten Grundformlexikon, das als solches ausschließlich Lemmata ohne Flexions- oder Derivationsformen enthält.[6] Abstrahiert formuliert fasst der Autor einer solchen Morphologiekomponente diese Lemmata darüber hinaus ihrem Verhalten bezüglich Allomorphie, Flexion und Derivation entsprechend in Gruppen zusammen. Mithilfe sogenannter, vom Autoren definierter

5. Für eine ausführlichere Einführung der syntaktischen Analyse vergleiche auch Greiner und Handl (2010).

6. Ein solches Lexikon kann selbstverständlich der Übersicht halber, wie auch von Bauer (2011) praktiziert, auf mehrere Dateien aufgeteilt werden. Darüber hinaus ist es in komplexen Ausnahmefällen auch üblich, Wortformen aufzunehmen, die streng genommen nicht in die Kategorie der Grundformen fallen.

`allo`- und `combi`-Tabellen wird dieses Grundformlexikon dann automatisch aufgefächert, wobei den resultierenden Wortformen eine Reihe von Attributen zugeordnet werden, die ihre linguistischen Charakteristika widerspiegeln. Diese Wortformen mitsamt ihrer Attributierung lassen sich, wie in Abbildung 3.2 gezeigt, graphisch als sogenannte Proplets visualisieren.

Abbildung 3.2: Einige beispielhafte Proplets

sur:	men
noun:	man
cat:	(pn)
mdr:	()
fnc:	()

sur:	sleep
verb:	sleep
cat:	(n-s3' v)
mdr:	()
arg:	()

sur:	green
adj:	green
cat:	(adn)
mdd:	()
arg:	()

An dieser Stelle ist anzumerken, dass die dargestellten Attribute nur eine beispielhafte Auswahl an möglichen Merkmalen bilden. Tatsächlich existieren noch eine Vielzahl mehr, die so verschiedene Sachverhalte wie die Position im Satz oder auch weiterführende semantische Information repräsentieren.

Die Variablendatei der Syntaxkomponente

In dieser Datei werden – wie der Name besagt – Variablen definiert, die im Code der Regeldatei Verwendung finden. Abbildung 3.3 illustriert eine solche Datei in rudimentärer Form.

Abbildung 3.3: Eine rudimentäre Variablendatei

```
1  string N  <- {sn, pn}
2         N' <- {sn', pn'}
3
4  constraint
5         N'  => N:
6         sn' => {sn}
7         pn' => {pn}
```

In den ersten beiden Zeilen werden die zwei Variablen `N` und `N'` definiert. Diesen wird mit dem Schlüsselwort `string` der entsprechende Datentyp zugewiesen. Auf die Vergabe der Variablennamen folgt jeweils auf den Zuweisungsoperator `<-` in geschweiften Klammern eine Liste von durch Kommata getrennten Werten, die die entsprechende Variable annehmen darf. So kann die in der ersten Zeile definierte Variable `N` die Werte `sn` sowie `pn` annehmen und die in der zweiten Zeile definierte Variable `N'` entsprechend die Werte `sn'` und `pn'`.

Die nachfolgenden Zeilen enthalten einen diese beiden Variablen betreffenden *constraint*, der vom Autor dazu benutzt werden kann, zwei bereits definierte Variablen in Beziehung zueinander zu setzen. Folgend auf das Schlüsselwort `constraint` definiert Zeile fünf, durch die Zeichenfolge `=>` getrennt und von einem Doppelpunkt abgeschlossen, die beiden Variablen, die beschränkt werden sollen. Die Art dieser Beziehungen le-

gen die Zeilen sechs und sieben fest, in denen jeweils einem möglichen Wert der Variable `N'` eine Liste an möglichen Werten der Variable `N` zugeordnet wird, wobei diese Listen in dem vorliegenden elementaren Beispiel nur einen einzigen Wert enthalten.

Die Regeldatei

Die für die syntaktische Analyse mit JSLIM unabdingbare Regeldatei setzt die in Kapitel 3.1 angesprochenen LAG-Bestandteile der Anfangszustände, Regeln und Endzustände um und arbeitet dabei eng mit der dazugehörigen Variablendatei zusammen. Abbildung 3.4 liefert das Beispiel einer rudimentären Regeldatei, um die im Folgenden gelieferte Erläuterung zu veranschaulichen.

Abbildung 3.4: Eine rudimentäre Regeldatei

```
1  ST_S = {([cat: (_)] {DET+N})}
2
3  DET+N {}
4
5    [noun: _, cat: (N' _)]
6    [noun: _, cat: (N)]
7
8    setref(nw.noun SS.noun)
9    cancel(N')
10   acopy(nw.sem SS.sem)
11
12 ST_F = {([cat: (_)] rp_DET+N)}
```

Die erste Zeile definiert mit dem Schlüsselwort `ST_S` beginnend die möglichen Anfangszustände. In diesem Fall gibt es nur einen gültigen Anfangszustand. Dieser definiert, dass das erste Wort einer Ableitung mit einem `cat`-Attribut versehen sein muss. Die JSLIM-Standardvariable `_` steht hierbei für einen bis beliebig viele, bezüglich ihrer Werte unbeschränkter, Literale, die diesem Attribut zugeordnet werden können. Darüber hinaus definiert dieser Startzustand eine Regel, mit der die Ableitung begonnen werden darf, falls das erste Wort der Eingabe erfolgreich auf oben geschilderte Restriktion abgeglichen wird. In diesem Fall ist dies die einzige Regel des Projekts, benannt mit `DET+N`.

In den folgenden Zeilen drei bis zehn wird ebenjene Regel genau spezifiziert. Der Name wird als `DET+N` vergeben und auf diesen folgend in geschweiften Klammern das Regelpaket spezifiziert, welches mögliche Folgeregeln enthält. In diesem rudimentären Beispiel wurde das Regelpaket leer gelassen, wodurch nach der Ausführung der Regel `DET+N` direkt der Endzustand abgeglichen werden muss. Zeile fünf gibt den von der Regel geforderten Satzanfang an. In diesem Fall muss dieser ein Proplet enthalten, welches ein `noun`-Attribut enthält, dessen Wert unbeschränkt ist, sowie ein `cat`-Attribut, dessen erstes Segment auf einen Wert abge-

glichen wird, der der Variablen `N'` entspricht. Die folgenden Kategoriesegmente des `cat`-Attributs können beliebige Werte annehmen oder ganz entfallen. Darauf folgend wird in Zeile sechs das von der Regel geforderte nächste Wort spezifiziert. Dessen Proplet muss ebenfalls ein `noun`- sowie ein `cat`-Attribut enthalten. Ersteres ist erneut unbeschränkt, Letzteres muss einem der zulässigen Werte für `N` entsprechen.

Die Zeilen acht bis zehn enthalten die Operationen, die nach erfolgreichem Abgleich der Proplets des Satzanfangs sowie des nächsten Wortes angewendet werden. Im Einzelnen soll an dieser Stelle nicht auf diese Operationen eingegangen werden. Eine Übersicht findet sich im Anhang von Greiner und Handl (2010).

Die Regeldatei schließt mit der Definition von möglichen Endzuständen in Zeile zwölf. In diesem Beispiel wurde nur ein solcher Endzustand definiert, der besagt, dass eine Ableitung unter zwei Bedingungen als erfolgreich angesehen werden kann. Erstens muss der aktuelle Satzanfang ein Proplet enthalten, welches über ein `cat`-Attribut verfügt, dessen Wert unbeschränkt ist. Zweitens muss es sich bei der zuletzt ausgeführten Regel in einem solchen Fall um die Regel `DET+N` handeln.

4 Die dem Syntaxprojekt zugrunde liegende Morphologie

Grundlage der syntaktischen Analyse in JSLIM bildet eine existente und funktionsfähige morphologische Komponente.[1] Aufgabe dieser Komponente ist es, dem für die Syntax zuständigen System ein möglichst umfangreiches sowie fehlerfreies Wortformerkennungssystem zur Verfügung zu stellen. Im Falle der hier vorliegenden Arbeit liefert dieses morphologische System die bereits vorliegende Magisterarbeit von Bauer (2011), betitelt mit *Morphologische Analyse des Englischen mit JSLIM*.

In den folgenden Abschnitten sollen zunächst der Umfang sowie die Abdeckung ebenjener Arbeit dargelegt werden. Darauf folgend wird beschrieben, welche Veränderungen vorgenommen wurden, um das vorliegende System für die syntaktische Analyse zu optimieren, und die jeweilige Begründung für die entsprechenden Änderungen angeführt.

4.1 Umfang

Die Zielsetzung von *Morphologische Analyse des Englischen mit JSLIM* lautet wie folgt:

> Diese Implementierung soll alle Bereiche der Allomorphie und Flexion umfassen und ein möglichst großes Elementarlexikon beinhalten. Es soll dabei Über- als auch Untergenerierung möglichst vermieden werden und bereits Vorbereitung für eine spätere Implementierung einer Syntax getroffen werden.
>
> Bauer (2011, S. 9)

1. Vgl. Hausser (2000, 180).

In Hinblick auf das nachfolgende, die Abdeckung des morphologischen Systems darlegende Kapitel 4.2 kann dieses Ziel durchaus als erreicht angesehen werden. Darüber hinausgehend wurden einige weitere bedeutende linguistische Phänomene zwar theoretisch angerissen, allerdings nicht implementiert.

So wurden Allomorphie und Flexion zwar tatsächlich ausführlich behandelt, Derivation und Komposition aber nur theoretisch angesprochen. Des Weiteren wurden sämtlichen, dem System bekannten Wortformen zwar entsprechende, für die syntaktische Analyse notwendige Kategorien zugewiesen, allerdings verzichtete Bauer (2011) an dieser Stelle darauf, Verben mit Information ihre Valenz betreffend zu versehen. Die Behandlung der beiden erstgenannten Phänomene ist nicht Gegenstand der Themenstellung dieser Arbeit.

Letzteres allerdings – die Valenz von Verben – nimmt eine wichtige Rolle im Satzbau des Englischen ein.[2] Diese Tatsache wird unter anderem dadurch verdeutlicht, dass am Institut für Anglistik und Amerikanistik der Friedrich-Alexander-Universität Erlangen-Nürnberg mit „Introduction to Syntactic Analysis – A Valency Approach“ von Herbst und Schüller (2008) ein Ansatz zur Analyse der englischen Syntax entwickelt wurde, der das Verb als zentrales Element des Satzbaus identifiziert. Einen weiteren Hinweis zur Dominanz des Verbs als zentrales Element der englischen Syntax liefern Quirk u. a. (1990):

> The clause types are determined by the verb class to which the full verbs within the verb constituent belong. Different verb classes require different complementation [...] to complete the meaning of the verb, or [...] no complementation.
>
> Quirk u. a. (1990, S. 720)

Die Erweiterung der Morphologiekomponente um diese, für die Syntax obligatorische Valenzinformation, wird in Abschnitt 4.3.3 behandelt.

4.2 Abdeckung

Das von Bauer (2011) entwickelte System wurde an einer eigens für dieses Projekt erstellten Testliste evaluiert. Diese Testliste setzt sich aus einer Frequenzliste des *British National Corpus* zusammen, aus der sämtliche Wortformen mit einem absoluten Vorkommen kleiner als sechs getilgt wurden, und umfasst somit 190.513 Types, die – aufgerechnet auf ihre jeweiligen Frequenzklassen – insgesamt 95.019.493 Tokens entsprechen.[3]

2. Vgl. Abschnitt 2.4.3.
3. Vgl. Bauer (2011, S. 60).

Beschränkt auf diese Testliste erreicht das vorliegende System eine Abdeckung von insgesamt 97,68% der Tokens sowie 61,61% der Types.[4] Zusammengefasst werden diese Ergebnisse noch einmal in den beiden Abbildungen 4.1 und 4.2.[5]

Abbildung 4.1: Zusammenfassendes Analyseergebnis der Morphologiekomponente für Tokens nach Bauer (2011)

```
* Summary

+--------------+------------+------------+--------------+
| type         | results    | %-results  | %-ambiguity  |
+--------------+------------+------------+--------------+
| confirmed    | 186974     | 0,20       | 106,58       |
| unconfirmed  | 92631196   | 97,49      | 146,88       |
| complete     | 92818170   | 97,68      | 143,40       |
| rejected     | 2201323    | 2,32       | 100,00       |
+--------------+------------+------------+--------------+
```

Abbildung 4.2: Zusammenfassendes Analyseergebnis der Morphologiekomponente für Types nach Bauer (2011)

```
* Summary

+--------------+------------+------------+--------------+
| type         | results    | %-results  | %-ambiguity  |
+--------------+------------+------------+--------------+
| confirmed    | 56         | 0,03       | 119,64       |
| unconfirmed  | 117325     | 61,58      | 121,01       |
| complete     | 117381     | 61,61      | 74,56        |
| rejected     | 73132      | 38,39      | 100,00       |
+--------------+------------+------------+--------------+
```

4.3 Modifikationen und Erweiterungen

Im Zuge der Erstellung des dieser Arbeit beiliegenden Syntaxprojekts war es nötig, einige Veränderungen an dem morphologischen System von Bauer (2011) vorzunehmen. Diese Modifikationen werden im Folgenden erläutert.

4.3.1 Anpassung an die Konventionen des project48

Das *project48* wurde vom Autor dieser Arbeit im Zuge seiner Tätigkeit als studentische Hilfskraft der Computerlinguistik der Friedrich-Alexander-Universität Erlangen-Nürnberg durchgeführt. Ziel dieses Projekts war es, sechs bereits existierende morphologische JSLIM-Projekte

4. Vgl. Bauer (2011, S. 64).
5. Für eine ausführliche Erläuterung dieser Ergebnisse siehe Bauer (2011, S. 63f.).

in eine einheitliche Form zu überführen, um sie für Interessierte besser zugänglich zu machen und dem in JSLIM aktuell vorherrschenden TMTOWTDI-Prinzip entgegenzuwirken.[6] Veränderungen wurden hinsichtlich folgender Punkte vorgenommen:

- Die **Ordnerstrukturen** wurden vereinheitlicht. Dabei wurde vor allem darauf geachtet, dass sämtliche Lexikondateien in einem Ordner `lex` und alle Tabellendateien in einem weiteren Ordner `tbl` zusammengefasst wurden, die nach den Spezifikationen des *project48* einzig und allein für die Aufnahme ebenjener Komponenten bestimmt sind. Ein weiterer Ordner `all` wurde dazu bestimmt, ausschließlich automatisch generierte Allomorphlexika aufzunehmen.[7] Für syntaktisch arbeitende Bestandteile wurde der Ordner `syn` reserviert.

  Darüber hinaus wurden sämtliche Projekte in eigene Verzeichnisse verlegt, deren Benennung der englischen Bezeichnung der jeweiligen Sprache entspricht. Diese Ordner wiederum wurden vereinheitlicht in einem Ordner `grammar` abgelegt, auf dessen Verzeichnisebene auch die ausführende JSLIM-2.1-Komponente zu finden ist. Diese vereinheitlichte, übergeordnete Ordnerstruktur liefert die Basis zur Standardisierung von Makefiles (s.u.).

- Die **Dateinamenvergabe** wurde standardisiert. Ziel war es hierbei, Verwendungszweck und JSLIM-Projekt jeder Datei sofort aus ihrem Namen erschließbar zu machen. Die Projektbezeichnung wurde hierfür als englisches Wort der jeweiligen Sprache (*english-[...]*, *swedish-[...]*, *polish-[...]* usw.) jedem Dateinamen als Präfix hinzugefügt. Der Verwendungszweck wird unter anderem aus der Dateiendung ersichtlich (*[...].rul* für Regeldateien, *[...].var* für Variablendateien, usw.). In Fällen, in denen konventionelle Dateinamen mehrdeutig interpretiert werden können, wurde den Dateienamen noch ein sprechendes Kürzel beigefügt. Für die Allomorphie zuständige Tabellen wurden beispielsweise mit dem Infix `-allo-` versehen und solche, die kombinatorische Phänomene behandeln, mit dem Infix `-combi-`.

- Jedem System wurde ein **Makefile** beigefügt, welches den Programmstart sowie die Ausführung einzelner Komponenten unter Linuxbetriebssystemen vereinheitlicht. Auf diese Weise wurden alternative und inkonsistente Möglichkeiten des Programmstarts wie die Verwendung von Shellskripten obsolet. Die standardisierten Shelleingaben zum Programmstart lauten:

6. TMTOWTDI: „There's more than one way to do it“, Ausgesprochen: [tɪmtəʊdiː]. Vgl. Wall u. a. (2003, S. 19).
7. Diese drei Dateitypen sind elementare Bestandteile jedes morphologischen JSLIM-Systems. Für eine ausführliche Erläuterung siehe Bauer (2011, S. 18ff.).

  - `make allo` führt die Allomorphiekomponente aus, die in der Erstellung automatisch generierter Allomorphlexika resultiert.
  - `make combi` startet das Morphologiesystem und liefert den Prompt zur Eingabe von zu analysierenden Wortformen.
  - `make all` bildet die Kombination aus beiden oben genannten Aufrufen und führt beide sukzessiv aus.
  - `make clean` löscht automatisch generierte Dateien wie oben genannte Allomorphielexika oder unter Linux üblicherweise mit `~` markierte Temporärdateien. Die Verwendung dieser Eingabe trägt maßgeblich dazu bei, die Übersicht zu wahren, sowie inkonsistente Datenbestände zu vermeiden.

- Es wurde eine manuelle **Tilgung überflüssiger bzw. veralteter Dateien** vorgenommen. In die erste Kategorie fallen beispielsweise die durch die Verwendung von Makefiles überflüssig gewordenen Shellskripte. In die zweite Kategorie fallen unter anderem vom jeweiligen Autor zu Debuggingzwecken mit dem Suffix `old` markierte Dateien oder solche, die nicht mehr in die Projektdatei integriert waren.

4.3.2 Einfügen von Interpunktionszeichen

Ein System zur syntaktischen Analyse des Englischen muss auch Interpunktionszeichen erkennen. Da diese von JSLIM grundsätzlich auf die gleiche Weise interpretiert werden wie gewöhnliche Wortformen auch, sind dafür zusätzliche Lexikoneinträge erforderlich. Diese wurden vom Autor dieser Arbeit, wie in Abbildung 4.3 veranschaulicht, in die von Bauer (2011) bereitgestellte und nach den oben erläuterten Spezifikationen des *project48* modifizierte Lexikondatei `english-allo-misc.lex` eingetragen.

Abbildung 4.3: Neuangelegte Lexikoneinträge für Interpunktionszeichen in english-allo-misc.lex

```
[sur: ".", cat: (v' decl), allo: A_null]
[sur: ",", cat: (v' comma), allo: A_null]
[sur: "?", cat: (v' interrog), allo: A_null]
[sur: "!", cat: (v' excl), allo: A_null]
```

Den Oberflächen für Punkt, Fragezeichen und Ausrufezeichen wurden hier mit `decl`, `interrog` und `excl` gültige Resultatkategorien eines korrekt abgeleiteten Satzes zugewiesen. Die Kategorisierung `comma` bildet zwar keine solche gültige Resultatkategorie, wird aber benötigt, um ein entsprechendes Proplet eindeutig zu identifizieren.

4.3.3 Einbinden eines semantischen Lexikons

Das in dieser Arbeit verwendete semantische Lexikon wurde freundlicherweise von Stadlbauer (2011) bereitgestellt. Ihre Arbeit hat es sich

zum Ziel gesetzt, Valenzdaten englischer Verben aus dem *British National Corpus* zu extrahieren. Besonderer Wert wurde dabei auf eine mögliche Verwendung in natürlichsprachlichen Grammatiken gelegt, was die vorliegenden Daten zu einem geeigneten Kandidaten für das Syntaxprojekt dieser Arbeit machen. Das semantische Lexikon wurde neben den regulären Vollformlexika im Ordner `lex` unter der Benennung `english-sem.lex` abgelegt und mit einem entsprechenden Eintrag in der Projektdatei in das Syntaxprojekt eingebunden.

Abbildung 4.4 zeigt zwei beispielhafte Einträge dieses semantischen Lexikons, um seinen Aufbau zu illustrieren. Gezeigt werden die Zuweisung von Valenzstellen für das monotransitive Verb `learn` sowie das ditransitive Verb `give`.

Abbildung 4.4: Drei beispielhafte Einträge des semantischen Lexikons

```
[sur: learn,  cat: <(), (a')>]
[sur: give,   cat: <(), (a'), (a' d')>]
```

4.3.4 Erweiterung um die Syntaxkomponente

Um auf syntaktischer Ebene arbeiten zu können, waren einige Modifikationen an verschiedenen Stellen nötig. Zum einen mussten die Regeldatei `english-syn.rul` sowie die Variablendatei `english-syn.var` angelegt werden. Beide wurden in dem Ordner `syn` abgelegt und mit dem in Abbildung 4.5 Codeausschnitt abgebildeten Eintrag in die Projektdatei `english-syn.pro` in das System integriert.

Abbildung 4.5: Integration der Regel- und Variablendatei innerhalb der Projektdatei

```
lahear
rul:  ../syn/english-syn.rul
var:  ../syn/english-syn.var
```

Darüber hinaus benötigt die syntaktische Analyse andersgeartete Operationen als die morphologische. Diese Operationen werden von speziell für JSLIM geschriebenen Javaklassen gestellt, die der Autor eines JSLIM-Projekts durch einen entsprechenden Eintrag in die Operationsdatei `english.ops` in sein System integriert. D.h. den Javaklassen werden mnemonische Namen zugewiesen, über diese sie dann innerhalb der Regeldatei angesteuert werden. Tabelle 4.1 skizziert diejenigen Operationen, die noch nicht in das morphologische System integriert waren.[8]

8. Für eine Auflistung bereits vorher existierender Einträge siehe Bauer (2011, S. 16).

Tabelle 4.1: Von der Syntaxkomponente zusätzlich benötigte Regeln

Javaklasse	Kürzel	Funktion
StringAppendRefOperation	sappendref	fügt Zeichenketten aneinander und ersetzt sämtliche Vorkommen des Zielwerts im Satzanfang
NAttrOperation	nattr	weist bereits erkannten Proplets neue Attribute zu
PrnIncOperation	inc	inkrementiert den Zähler des `prn`-Attributs
PrnDecOperation	dec	dekrementiert den Zähler des `prn`-Attributs

5 Praktische Umsetzung

Bisher wurden die theoretischen Grundlagen erläutert, auf die sich die vorliegende Arbeit stützt. Der folgende Teil beschäftigt sich nun mit der tatsächlichen Implementierung des englischen Syntaxprojekts.

Die Unterkapitel 5.1 und 5.2 liefern dazu allgemeine Informationen. Ersteres beschreibt den Aufbau und gibt eine kurze Bedienungsanleitung, Letzteres definiert die gültigen Anfangs- und Endzustände einer Ableitung. Im Anschluss daran wird in den Unterkapitel 5.3 bis 5.9 die Funktionsweise der Analyse im Allgemeinen sowie der einzelnen Regeln im Besonderen an den Beispielen der in Abschnitt 2.4.1 definierten *major clause types* dargelegt. Jede dieser Passagen liefert dazu jeweils einen Beispielsatz und beschreibt die einzelnen Satzbestandteile. Darauf folgend wird, soweit relevant, die zugrunde liegende morphologische Analyse kurz zusammengefasst. Darüber hinaus wird die Funktionsweise der jeweils benötigten Regeln erläutert und an einer graphischen Beispielableitung illustriert. Abschließend erweitern die Unterkapitel 5.10 bis 5.11 diese Grundstrukturen um eine Auswahl syntaktischer Phänomene aus Hausser (2006, Teil II). Dabei wird auf graphische Beispielableitungen verzichtet und stattdessen auf die Abschnitte aus Hausser (2006) verwiesen, die entsprechende Illustrationen enthalten.

5.1 Grundsätzliches

An dieser Stelle werden generelle Informationen zum beiliegenden JSLIM-Projekt erörtert. Die folgenden Abschnitte liefern zunächst genaue Informationen zum System und beleuchten den grundsätzlichen Aufbau des Projekts. Danach werden der Programmstart sowie die grundlegende Bedienung beschrieben.

5.1.1 Versionsangabe und Entwicklungsumgebung

Das in dieser Arbeit entwickelte Syntaxprojekt benutzt die aktuell stabile JSLIM-Version 2.1. Die Linuxdistribution openSUSE 11.4 ist das Betriebssystem des verwendeten Rechners.

5.1.2 Aufbau

In Übereinstimmung mit den Spezifikationen des *project48* (vgl. Abschnitt 4.3.1) liegen die Bibliotheken und Startskripte von JSLIM in dem Ordner `jslim2.1`, parallel zu dem Verzeichnis `grammar`, das in einem Unterverzeichnis `english` alle weiteren Komponenten des Systems enthält.[1] In diesem Ordner `english` finden sich die in Tabelle 5.1 illustrierten Verzeichnisse, sowie das im folgenden Abschnitt erläuterte Makefile. Die Ordner `syn` und `test` sowie deren Inhalt wurden für die Syntaxkomponente neu erstellt.

Tabelle 5.1: Tabellarische Darstellung der Ordnerstruktur

Ordner	Inhalt
`allo`	nimmt die automatisch generierten Dateien des Allomorphlexikons auf
`common`	enthält die angepasste Projektdatei `english-syn.pro`, die Attributdatei `englisch.att` und die modifizierte Operationsdatei `english.ops`
`lex`	enthält die Dateien des Elementarlexikons sowie das semantische Lexikon `english-sem.lex`
`morph`	enthält die Regel- und Variablendatei der Kombinatorik `english-combi.rul` und `english-combi.var`
`syn`	enthält die Regel- und Variablendatei der Syntax `english-syn.rul` und `english-syn.var`
`tbl`	enthält die von der Morphologie benötigten Tabellendateien
`test`	enthält ausführliche Testlisten für das Syntaxprojekt sowie ein Makefile zur Pflege ebenjener

5.1.3 Programmstart

Der Start der einzelnen Programmkomponenten erfolgt unter Verwendung eines im Ordner `english` abgelegten Makefiles, welches unter Linuxsystemen von ebenjenem Verzeichnis aus ausgeführt werden kann. Eine kurze Zusammenfassung der einzelnen Aufrufe findet sich in Abschnitt 4.3.1 oder kann durch die Eingabe von `make` oder auch `make help` in einer Linux-Shell auf dem Bildschirm ausgegeben werden.

1. Auf der beiliegenden CD finden sich darüber hinaus parallel zu den Ordnern `grammar` und `jslim2.1` auch diese Arbeit in Druckform (`MA.pdf`) sowie die Datei `readme.txt`, die eine Kurzanleitung zum Programmstart enthält.

Die im Zuge des *project48* implementierte Funktion `make combi` wurde für das Syntaxprojekt durch `make syn` ersetzt, welche die Syntaxkomponente startet und die interaktive Benutzereingabe für zu analysierende Eingaben bereitstellt. Alternativ dazu ist auch der Programmstart mit dem Aufruf `make all` möglich, der vor dem eigentlichen Start der Syntaxkomponente zunächst die Allomorphlexika generiert bzw. aktualisiert.

Das Makefile wurde dieser Arbeit in gedruckter Form beigelegt. Es findet sich in Anhang A.

5.1.4 Bedienung

Den erfolgreichen Start des Programms erkennt man, wie in Abbildung 5.1 illustriert, an einer Bildschirmausgabe, welche unter anderem die Meldung `syntax defined` und die Versionsnummer von JSLIM angibt, sowie an der Veränderung des Eingabeprompts.

Abbildung 5.1: Der JSLIM-Prompt nach erfolgreichem Programmstart

```
disable sem lex hook before morphology run: syntax defined
JSLIM - Java Surface Compositional Linear Internal Matching (v2.1.0.0)
Type '-?' for further information

LAHEAR>
```

Nun kann der Benutzer dem System über die Befehlszeile zu analysierende Eingabesätze übergeben. An dieser Stelle sollte angemerkt werden, dass JSLIM Satzzeichen auf die gleiche Weise wie gewöhnliche Wörter verarbeitet. Da eine syntaktisch zu analysierende Eingabe zunächst an *Whitespaces* in ihre einzelnen Elemente aufgespalten wird, ist es nötig, Interpunktionszeichen mit einem Leerzeichen von der vorhergehenden Wortform abzutrennen.[2]

Eine bequemere Möglichkeit zur Eingabe mehrerer Sätze bieten Testlisten. JSLIM kann Dateien jeglicher Benennung als Testlisten einlesen. Dabei wird jede Zeile der Datei als eine zu analysierende Eingabe interpretiert und danach ein zusammenfassendes Gesamtergebnis ausgegeben. Um Testlisten einzulesen ist folgender Befehl nötig:
`-test` <*Verzeichnispfad*>/<*Dateiname*>

Der Befehl `-?` liefert bei Bedarf weitere Optionen. Das Programm wird mit der Eingabe `-q` beendet.

2. Im Informatikerjargon bezeichnet der Ausdruck *Whitespace* alle nicht druckbaren Zeichen wie Leerzeichen, Tabulatoren oder auch Zeilenumbrüche.

5.2 Anfangs- und Endzustandsdefinition

Wie in Abschnitt 3.3.2 beschrieben, definiert die Regeldatei unter anderem auch die gültigen Anfangs- und Endzustände, mit denen eine Analyse gestartet bzw. beendet werden darf.

Die Anfangszustandsdefinition des vorliegenden Projekts wird in Abbildung 5.2 illustriert und gewährleistet, dass das erste Proplet der Ableitung ein Attribut cat enthalten muss, dessen Wert an dieser Stelle keiner Einschränkung unterliegt. Von diesem ersten Proplet aus kann die syntaktische Analyse mit der Ausführung einer der in geschweiften Klammern angegebenen Regeln beginnen.

Abbildung 5.2: Anfangszustandsdefinition des Syntaxprojekts

```
ST_S = {([cat: _] {DET+N, MAIN+FV, DET+ADJ, ADJ+N, N+CNJ, ADJ+CNJ,
                   N+COMMA, ADJ+COMMA, MAIN+HVBE, TH+N, MAIN+ADV,
                   MAIN+WHICH, WHEN+MAIN})}
```

Die in Abbildung 5.3 gezeigte Endzustandsdefinition des Syntaxprojekts erachtet eine Analyse als korrekt, wenn zwei Bedingungen erfüllt sind: Erstens muss das bisherige Analyseergebnis ein Proplet enthalten, dessen cat-Attribut der Wert decl zugewiesen wurde, und zweitens muss eine erfolgreiche Analyse mit der Anwendung der Regel S+IP abschließen.

Abbildung 5.3: Endzustandsdefintion des Syntaxprojekts

```
ST_F = {([cat: (decl)] rp_S+IP)}
```

5.3 Analyse der Satzkonstruktion SV

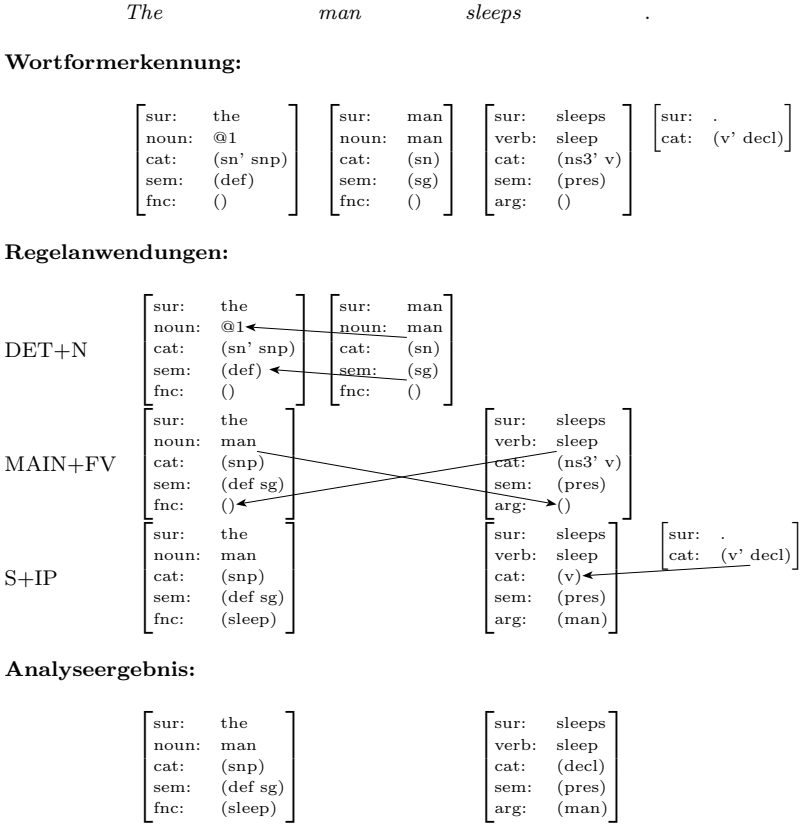
Als Beispiel wird an dieser Stelle der Satz „The dog sleeps.“ analysiert. Der Satzteil *the dog* stellt als Nominalphrase in diesem Fall das Subjekt dar, realisiert durch einen bestimmten Artikel und ein Nomen im Singular. Die Wortform *sleep* übernimmt die Rolle des Verbs.

5.3.1 Identifizierte Proplets und ihre Attribute

Die Morphologiekomponente erkennt die einzelnen Oberflächen und versieht sie mit einer Reihe von Attributen, wie sie in der Zeile *Wortformerkennung* in Abbildung 5.4 illustriert sind.[3] Die Wortart der Oberflächen von *dog* und *sleeps* lässt sich daran erkennen, dass ihren Proplets das Attribut noun bzw. verb zugewiesen wird. Der bestimmte Artikel *the* kann als solcher durch das Kategoriesegment `sn'` identifziert werden, welches

3. Tatsächlich werden jedem Proplet noch eine ganze Reihe weiterer Attribute zugewiesen. Vgl. dazu Abschnitt 3.3.2. Aus Gründen der Übersicht wird an dieser Stelle sowie in folgenden Abschnitten nur auf die jeweils relevanten Attribute eingegangen.

Abbildung 5.4: Ableitung einer SV-Konstruktion



darauf hinweist, dass ein Nomen im Singular erwartet wird sowie an der Tatsache, dass das Attribut noun des zugehörigen Proplets den als Platzhalter fungierenden Wert `@1` annimmt. Der den Satz abschließende Punkt lässt sich anhand des Kategoriesegments `decl` erkennen.

Darüber hinaus ist die Vergabe der Attribute fnc und arg an dieser Stelle interessant. Am Ende der Ableitung spiegeln sie die Funktor-Argument-Beziehungen zwischen Subjekt und Verb wider.

5.3.2 Die benötigten Regeln

Für die Analyse des Beispielsatzes werden die drei Regeln DET+N, MAIN+FV und S+IP benötigt. Der folgende Abschnitt erläutert ihre Funktionsweise.

Die Regel DET+N übernimmt die Aufgabe, Nomina einzulesen und diese einem entsprechenden, bereits im Satzanfang enthaltenen Artikel zuzuordnen, wobei die Kongruenz des Numerus beider Wortformen sichergestellt wird. Da die Gesamtanalyse eines Satzes nur noch Proplets von Inhaltswörtern enthalten soll, werden die Informationen beider Merkmalstrukturen im Laufe der Ableitung in einem Proplet kombiniert. Insgesamt enthält DET+N sieben Klauseln, die unterschiedliche Ausprägungen der Kombination von Artikel und Nomen behandeln. Die drei wichtigsten dieser Phänomene sind die Kombination des unbe-

stimmten Artikels *a* bzw. *an* mit einem Nomen sowie der Standardfall, der den bestimmten Artikel *the* behandelt und in dem vorliegenden Beispiel Anwendung findet.

Bei erfolgreichem Abgleich von Satzanfang und nächstem Wort modifiziert die Klausel DET+N.STAND die Attribute entsprechender Proplets nach dem folgenden Muster: Zunächst schreibt sie den Wert des noun-Attributs des Proplets des nächsten Wortes in den des noun-Attributs des Proplets des Satzanfangs, woraufhin das Kategoriesegment aus dem Satzanfangsproplet gelöscht wird, das dem erwarteten Nomen entspricht (in diesem Fall `sn'`). Abschließend hängt die Klausel DET+N.STAND den sem-Wert des nächsten Proplets an den des Proplets des Satzanfangs an und verwirft das Proplet des nächsten Wortes, dessen Information sich nun in dem des Satzanfangs wiederfindet.

MAIN+FV identifiziert als nächstes Wort ein Verb sowie das dazugehörige Subjekt aus dem Satzanfang. Dabei wird die Kongruenz zwischen Numerus und Kasus sichergestellt. MAIN+FV enthält acht Klauseln, die eine Reihe an Spezialfällen bearbeiten. In diesem ersten Beispiel tritt wieder die Standardklausel in Kraft.

Wenn der von der Standardklausel MAIN+FV.STAND vorgenommene Abgleich der Proplets des Satzanfangs und des nächsten Wortes gelingt, modifiziert die Klausel deren Attribute. Zunächst wird dazu das Segment der Kategorie des Verbs gekürzt, dass dem erwarteten Nomen entspricht (in diesem Fall `ns3'`). Daraufhin verzeigert die Klausel die beiden Proplets über das jeweilige fnc- bzw. arg-Attribut, um die Funktor-Argument-Struktur der jeweiligen Satzteile widerzuspiegeln. Dies geschieht, indem die Klausel MAIN+FV.STAND den Wert des verb-Attributs des nächsten Wortes in das fnc-Attributs des Satzanfangs kopiert und umgekehrt den Wert des noun-Attributs des Satzanfangs an das arg-Attribut des nächsten Wortes anhängt. Da es sich bei dem nächsten Wort um ein Inhaltswort handelt, das als solches mit einem Proplet in der Gesamtanalyse erscheinen soll, wird das entsprechende, von MAIN+FV modifiziertes Proplet abschließend in den Satzanfang kopiert.

Die Standardklausel der Regel S+IP behandelt die Kombination eines Satzes mit einem abschließenden Interpunktionszeichen. Eine Reihe weiterer Klauseln bearbeitet wiederum Spezialfälle, die, soweit für die ausgewählten Beispiele erforderlich, an anderer Stelle erläutert werden.

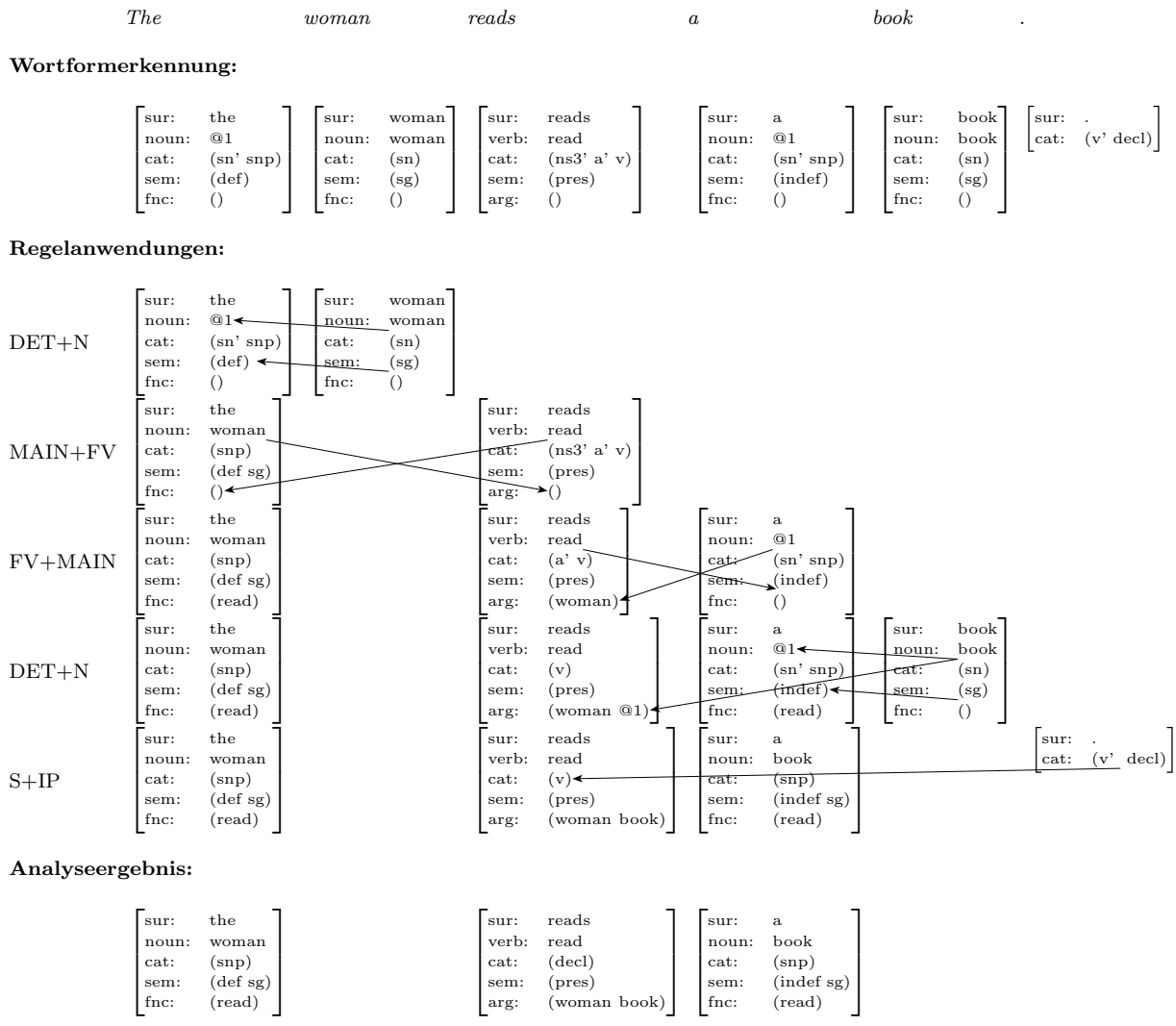
Die in diesem Beispiel angewandte Klausel S+IP.STAND wird ausgeführt, wenn der aktuelle Satzanfang ein Verbproplet enthält, dessen Valenzstellen vollständig gekürzt wurden. Dies ist an dem cat-Wert `v` erkennbar. Ist die Bedingung erfüllt, so wird ebenjenes Attribut mit dem Wert `decl` überschrieben. Dieser Wert des cat-Attributs eines Verbs in

Kombination mit der Tatsache, dass zuletzt die Regel S+IP ausgeführt wurde, führen dazu, dass die Ableitung einer gültigen Endzustandsdefinition entspricht.[4]

5.4 Analyse der Satzkonstruktion SVO

Der Beispielsatz dieser Konstruktion lautet „The woman reads a book.“. Wie im vorherigen Beispiel übernimmt auch diesmal eine Nominalphrase die Rolle des Subjekts, die sich aus einem bestimmten Artikel und einem Nomen im Singular zusammensetzt, nämlich aus *The* und *woman*. Das monotransitive Verb bildet an dieser Stelle die Oberfläche *reads*. Die Nominalphrase *a book*, die von einem diesmal unbestimmten Artikel sowie einem Nomen im Singular gebildet wird, stellt ein direktes Objekt dar.

Abbildung 5.5: Ableitung einer SVO-Konstruktion



4. Zur Funktionsweise der Endzustandsdefinition vgl. Unterkapitel 5.2.

5.4.1 Identifizierte Proplets und ihre Attribute

Auch an dieser Stelle werden die erkannten Wortformen in der zweiten Zeile der illustrierenden Abbildung 5.5 gezeigt. Die ersten zwei Proplets, *The* und *woman*, funktionieren an dieser Stelle genauso wie *The* und *man* im vorigen Beispiel. Das Proplet der Oberfläche *reads* unterscheidet sich vom vorhergehenden Beispiel dahingehend, dass es die Rolle eines monotransitiven Verbs einnimmt, was durch die zusätzliche Valenzstelle `a'` im `cat`-Attribut zum Ausdruck gebracht wird. Darauf folgen die beiden Proplets der Oberflächen *a* und *book*. Letzteres ist bis auf Oberfläche und Grundform identisch mit bisher behandelten Nomina, Ersteres unterscheidet sich als unbestimmter Artikel insofern von den bisher benutzten Artikeln, als dass das Merkmal der Unbestimmtheit durch das Kategoriesegment `indef` ausgedrückt wird. Der den Beispielsatz abschließende Punkt wiederum ist auf die gleiche Weise attributiert wie im vorhergehenden Beispiel.

5.4.2 Regelanwendungen

Die Analyse des Beispielsatzes erfordert insgesamt fünf syntaktische Ableitungsschritte. Zusätzlich zu den bereits beschriebenen Regeln wird an dieser Stelle die Regel FV+MAIN benötigt, um das direkte Objekt zu verarbeiten. Darüber hinaus erfordert der im Beispiel benutzte indirekte Artikel eine zusätzliche Klausel von DET+N.

Nachdem der Satzanfang auch in diesem Beispiel das Proplet von *the* als gültiges erstes Wort identifiziert und die Regel DET+N als mögliche Startregel definiert, wird diese dem vorigen Beispiel entsprechend erfolgreich ausgeführt. Als Resultat dieser Regelanwendung finden sich die Informationen der Oberflächen *the* und *woman* gemeinsam in einem Proplet kodiert wieder.

Auch die Anwendung von MAIN+FV funktioniert ganz ähnlich dem vorigen Beispiel. Die Proplets von Subjekt und Verb werden an dieser Stelle über die Attribute `fnc` bzw. `arg` miteinander verknüpft und das Segment `ns3'`, welches das erwartete Subjekt repräsentiert, gekürzt. Im Unterschied zu dem vorhergehenden Beispiel bleibt allerdings noch eine Valenzstelle `a'` in der Kategorie des Verbs erhalten.

Auf das Verb *reads* folgt in diesem Beispiel der Artikel *a*, wodurch die Ausführung der Regel FV+MAIN ausgelöst wird, welche die Verknüpfung von Verben mit nachfolgenden Nominalphrasen behandelt. FV+MAIN enthält vier Klauseln, von denen im vorliegenden Beispiel wieder die Standardklausel in Kraft tritt.

Diese Klausel FV+MAIN.STAND bearbeitet die Proplets eines bereits eingelesenen Verbs sowie ein nächstes Wort, dessen Proplet ein `cat`-Attribut enthält. Dabei wird sichergestellt, dass die Kategorie des Verbs

noch mindestens eine Valenzstelle enthält. Sind entsprechende Proplets vorhanden, so werden sie, ähnlich der Funktionsweise von MAIN+FV über ihre fnc- bzw. arg-Attribute verzeigert. Dieser Fall unterscheidet sich von den vorhergehenden, von MAIN+FV vorgenommenen Verzeigerungen insofern, als der Artikel selbst noch ein Nomen erwartet, weshalb der Platzhalter @1 an das arg-Attribut des Verbs eingefügt wird. Darüber hinaus wird auch die in der Kategorie des Verbs enthaltene Valenzstelle, in diesem Fall `a'`, gekürzt.

Die nächste zu verarbeitende Oberfäche ist das Nomen *book*, welches ähnlich bereits behandelter Nomina, von der Regel DET+N bearbeitet wird. Allerdings tritt in diesem Fall eine Klausel in Kraft, die speziell die Verarbeitung des unbestimmten Artikels *a* behandelt, benannt mit DET+N.A. Diese Klausel nutzt reguläre Ausdrücke, um sicherzustellen, dass die Oberfläche des nächsten Wortes nicht mit einem Vokal beginnt, und sorgt auf diese Weise dafür, dass inkorrekte Eingaben wie *a eagle* nicht akzeptiert werden. Ansonsten verhält sie sich wie die Klausel DET+STAND und kombiniert die Informationen von Artikel und zugehörigem Nomen in ein Proplet. Anzumerken ist an dieser Stelle, dass die von der Regel DET+N benutzte Operation `setref` nicht nur den im Operationsaufruf adressierten Wert, sondern alle Vorkommen dieses Wertes im Satzanfang modifiziert. Auf diese Weise kann mit der Ausführung einer Operation der Platzhalter @1 sowohl im noun-Attribut des Artikels *a* als auch im arg-Attribut des Verbs ersetzt werden.

Abschließend kommt auch in diesem Beispiel die Standardklausel der Regel S+IP zur Anwendung, da der Punkt das nächste Wort der Eingabe darstellt und alle Valenzstellen des Verbs im Zuge der vorhergenden den Ableitung gekürzt wurden. Die daraus resultierene Kategorie des Verbs `decl` in Kombination mit der zuletzt ausgeführten Regel S+IP entspricht einer gültigen Endzustandsdefinition, so dass die Ableitung korrekt terminiert.

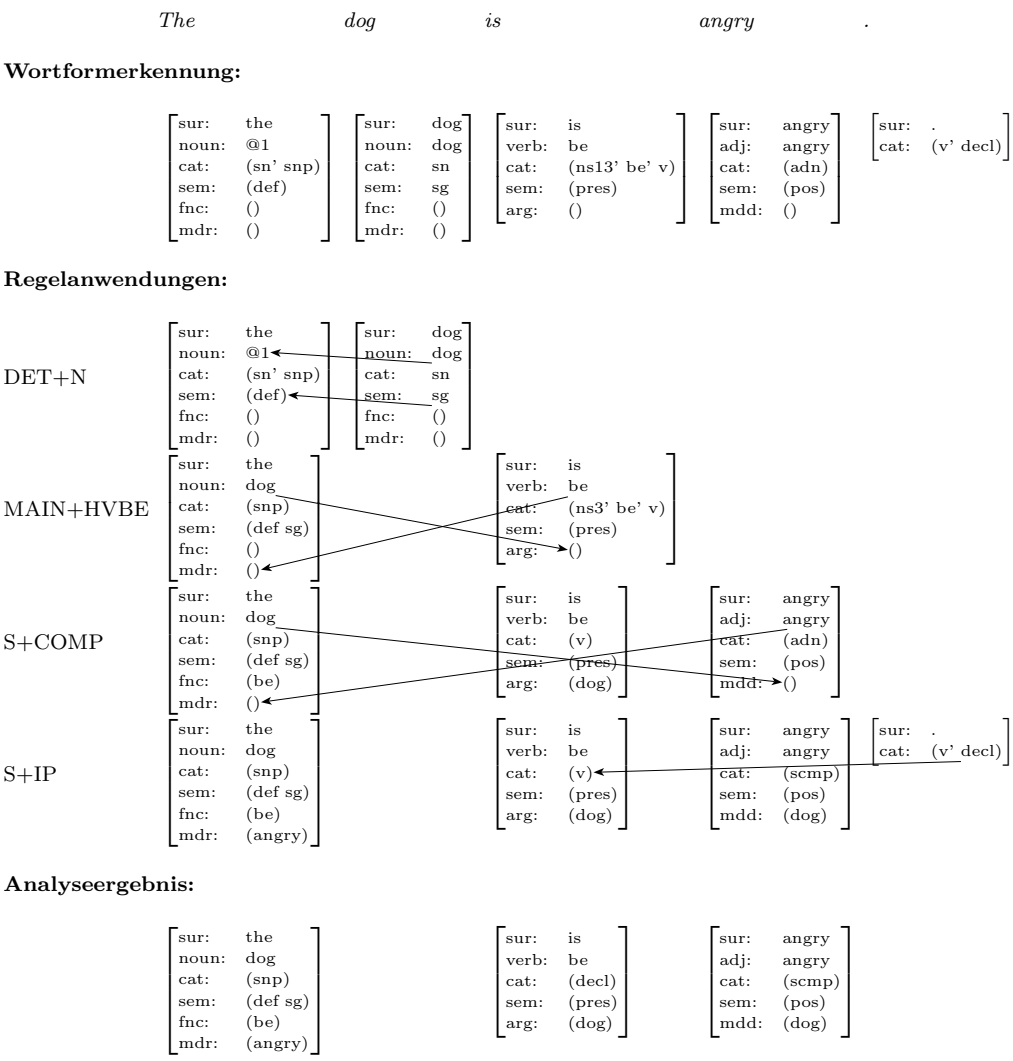
5.5 Analyse der Satzkonstruktion SVC

Als Beispielsatz dieser Konstruktion dient „The dog is angry.“, wobei das Subjekt erneut von einer Nominalphrase aus bestimmtem Artikel und Nomen im Singular gebildet wird. Die Oberfläche *is* stellt in diesem Fall das Verb dar und das *Subject Complement* wird durch das Adjektiv *angry* gebildet.

5.5.1 Identifizierte Proplets und ihre Attribute

Die beiden Proplets von *The* und *dog* weisen die gleichen Merkmale auf wie entsprechende Bestandteile vorheriger Beispiele. Zusätzlich wurde

Abbildung 5.6: Ableitung einer SVC-Konstruktion



hier noch das Attribut mdr in die jeweiligen Merkmalstrukturen aufgenommen, welches für gewöhnlich Verweise zwischen Nomina und dazugehörigen Adjektiven oder Verben und dazugehörigen Adverben realisiert. Da es sich beim *Subject Complement* um eine Phrase handelt, die das entsprechende Subjekt auf ähnliche Weise modifiziert wie ein Adjektiv, wird mdr in diesem Fall auch benutzt, um eine vergleichbare Verzeigerung herzustellen. Darüber hinaus wird das *Subject Complement* im Laufe der Ableitung durch einen speziellen Eintrag in die Kategorie als solches gekennzeichnet. Die Merkmalstruktur des Verbs *is* unterscheidet sich insofern von bisher behandelten Verben, als dass sie unter anderem auch als Hilfsverb bei der Bildung komplexer Nominalphrasen wie dem *present progressive* eine Rolle spielen kann und deshalb über eine entsprechend angepasste Kategorie verfügt. Wie dieser Umstand bearbeitet wird, soll im nachfolgenden Abschnitt aufgezeigt werden. Die Oberfläche *angry* stellt als Adjektiv in diesem Beispiel das *Subject Complement* dar, welches durch das Attribut adj sowie dem Kategoriewert adn als solches

erkennbar ist. Als Gegenstück zu mdr enthält es das Merkmal mdd, welches einen Verweis auf das modifizierte Subjekt aufnimmt.

5.5.2 Regelanwendungen

In dieser Analyse kommen insgesamt vier Regeln zur Anwendung, wobei DET+N und S+IP nach ähnlichem Schema funktionieren wie in vorangegangenen Beispielen. Der durch ein Adjektiv realisierte Satzteil des *Complements* wird allerdings unter Zuhilfenahme der zwei noch nicht erwähnten Regeln MAIN+HVBE sowie S+COMP bearbeitet.

Wie in vorigen Beispielen auch akzeptiert die Definition der Anfangszustände die Oberfläche *The* als erstes Wort und die mögliche Startregel DET+N, die das eben erwähnte Proplet mit dem von *dog* kombiniert, indem die Informationen des Nomens an die entsprechenden Kategorien des Proplets des Artikels geschrieben werden.

Das aktuell nächste Wort ist nun die Oberfläche *is*, deren Proplet aufgrund der speziellen Kategorisierung nicht von der bisher verwendeten Regel MAIN+FV, sondern von MAIN+HVBE berarbeitet wird. Aufgabe dieser Regel ist es, spezielle Verben wie *have* und *be* einzulesen, die nicht nur die Rolle von Vollverben, sondern auch die von Hilfsverben übernehmen können und deshalb spezieller Behandlung bedürfen. In diesem Fall übernimmt die Klausel ähnliche Funktion wie MAIN+FV, nämlich die Verknüpfung von bereits eingelesenem Subjekt und dazugehörigem Verb an der Position des aktuell nächsten Wortes, wobei von *constraints* eingeschränkte Variablen zum Einsatz kommen, um die Kongruenz bezüglich Numerus und Kasus sicherzustellen. Der Unterschied liegt darin, dass MAIN+HVBE die spezielle Kategorisierung solcher Verben beachtet, die auch in komplexen Verbalphrasen oder bei Passivierung Verwendung finden.

Bei erfolgreicher Identifikation eines das Subjekt repräsentierenden Proplets des Satzanfangs sowie eines entsprechenden Verbs als nächstes Wort der Eingabe kommt es zur Modifikation der Attribute beider Proplets. Dazu werden das fnc- und arg-Attribut mit den das entsprechende Lemma repräsentierenden Werten der Attribute verb bzw. noun belegt. Außerdem kürzt MAIN+HVBE bei erfolgreichem Abgleich von Satzanfang und nächstem Wort nicht nur das Kategoriesegment des Verbs, das für das erwartete Subjekt steht, sondern auch den Wert `be'`, der die mögliche Sonderfunktion der Oberfläche *is* widerspiegelt. Anschließend wird das Proplet des Verbs an den Satzanfang kopiert, da es entweder, wie in diesem Fall, als Vollverb im Endergebnis der Analyse vertreten sein soll bzw. als Hilfsverb die Informationen eines zugehörigen Vollverbs aufnehmen könnte.

Nachdem das Verb verarbeitet wurde, bildet das Adjektiv *angry* mitsamt seinem Proplet das nächste zu analysierende Wort und wird von der Regel S+COMP eingelesen. Diese Regel enthält zwei Klauseln, von denen die erste *Subject* und die zweite *Object Complements* behandelt.

An dieser Stelle kommt die erste dieser beiden Klauseln, benannt mit S+COMP.SC, zur Anwendung. Zum einen setzt sie das Vorhandensein einer das Subjekt darstellenden Nominalphrase voraus. Diese wird identifiziert, indem zwei Proplets aus dem Satzanfang angesteuert und aufeinander abgeglichen werden. Bei dem ersten der beiden Proplets muss es sich um ein Nomen handeln, bei dem zweiten um ein Verb, wobei Variablen benutzt werden, um die Funktor-Argument-Struktur zwischen den beiden Satzteilen sicherzustellen. Findet sich an erster Stelle des arg-Attributs ein Verweis auf das vorangehende Nomen, so ist dieses damit als Subjekt identifiziert. Zum anderen wird anhand der Kategorisierung des nächsten Wortes sichergestellt, dass es sich bei diesem um ein Adjektiv handelt.

Sind die Eingabebedingungen erfüllt, so modifiziert S+COMP.SC beide Proplets. Dazu wird einerseits der Wert des noun-Attributs des Subjekts in den Wert des mdd-Attributs des nächsten Wortes eingefügt und andererseits der Wert des adj-Attributs des Adjektivs an den Wert des mdd-Attributs des Subjekts angehängt. Auf diese Weise werden Subjekt und zugehöriges *Complement* miteinander verzeigert. Letzteres wird darüber hinaus mit dem Eintrag `scmp` in das cat-Atrtibut gekennzeichnet, um seine syntaktische Rolle eindeutig zu identifizieren. Abschließend fügt S+COMP.SC das Proplet des Adjektivs an den bereits analysierten Satzanfang an, da es sich um ein Inhaltswort handelt.

Die Eingabe schließt wie alle Deklarativsätze mit der Oberfläche des Punkts, die wie in vorigen Beispielen von der Regel S+IP behandelt wird und die Kategorie des Verbs auf den Wert `decl` setzt. Damit führt die Anwendung von S+IP zu einem Zustand, der einer gültigen Endzustandsdefinition entspricht.

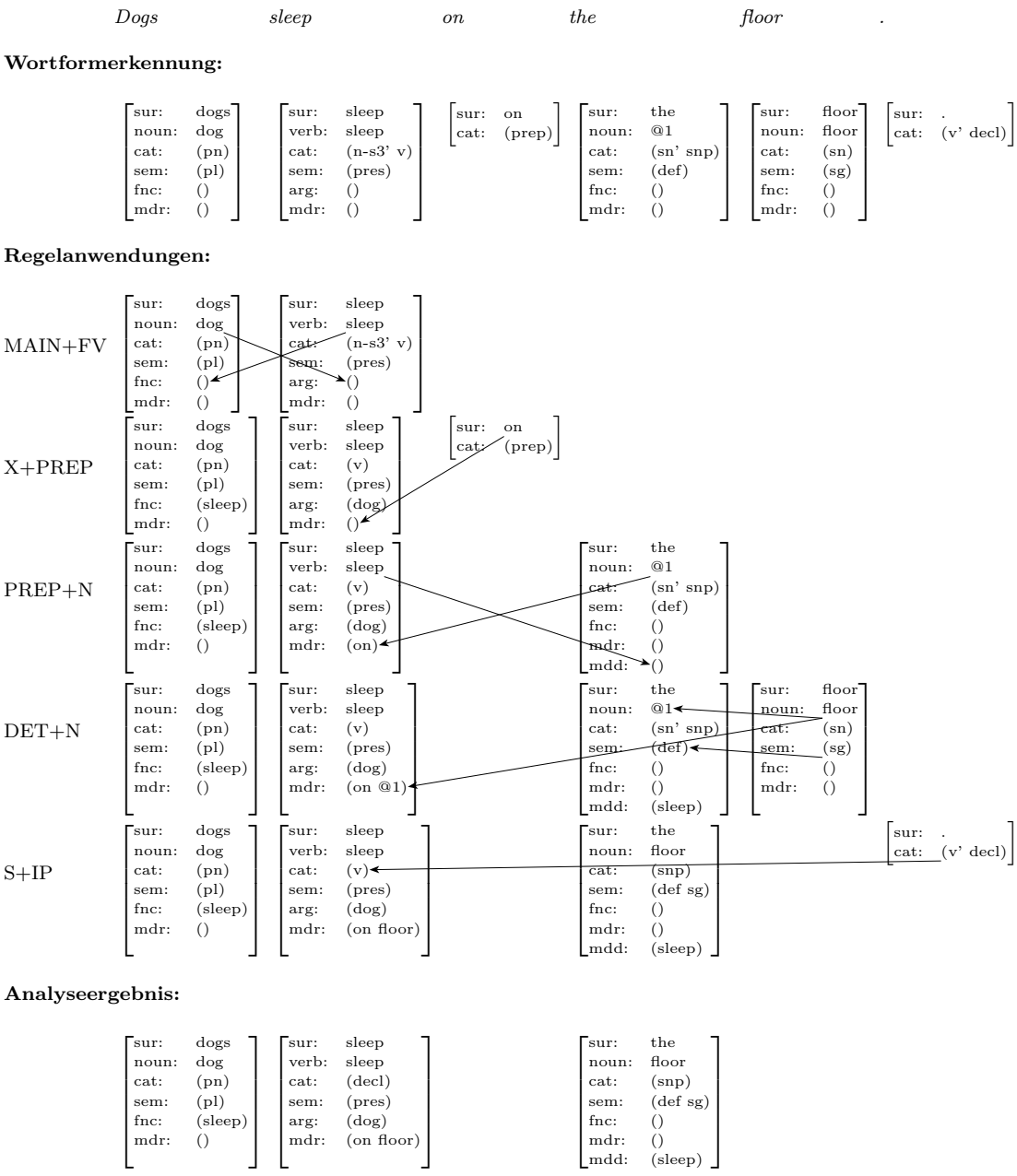
5.6 Analyse der Satzkonstruktion SVA

Der Beispielsatz dieser Konstruktion ist „Dogs sleep on the floor.“. Das Subjekt bildet sich hierbei aus dem alleinstehenden Nomen im Plural *dogs*, und *sleep* übernimmt erneut die Rolle des Verbs. Das *Adverbial* dieses Beispiels wird als Präpositionalphrase *on the floor* realisiert, die sich aus einer Präposition sowie einem Nomen im Singular mit einem dazugehörigen Artikel zusammensetzt.

5.6.1 Identifizierte Proplets und ihre Attribute

Das Merkmal des Plurals ist anhand der Kategorisierung `pn` sowie `pl` des der Oberfläche *Dogs* zugeordneten Proplets erkennbar. Der Kategoriewert `n-s3'` von *sleep* zeigt an, dass diese Oberfläche mit beinahe jedem Subjekt kombiniert werden kann, außer mit solchen, die die dritte Person Singular repräsentieren. Die Präposition *on* ist als solche an ihrer Kategorie zu identifizieren und die Proplets der Oberflächen *the* und *floor* weisen die selben Merkmale auf, wie vergleichbare Proplets aus vorhergehenden Beispielen.

Abbildung 5.7: Ableitung einer SVA-Konstruktion



5.6.2 Regelanwendungen

Die im Folgenden beschriebene Ableitung unterscheidet sich insofern von der in Hausser (2006, S. 101) vorgeschlagenen Verarbeitung von Präpositionalphrasen, als die Information aller Proplets einer solchen Phrase nicht in das Proplet der Präposition, sondern in das des Artikels kodiert wird. Dieses Vorgehen wurde bewusst gewählt, da es erlaubt, eine ganze Reihe von Regeln bzw. Klauseln wieder zu verwenden, die der Kombination von Artikeln mit Nomina oder von Adjektiven mit Nomina dienen, was zu schlankerem und damit übersichtlichem Code führt. Das Proplet einer Präpositionalphrase lässt sich bei dieser Vorgehensweise in der Gesamtanalyse daran erkennen, dass es sowohl über ein mdd- als auch ein mdr-Attribut verfügt. Ersteres dient dazu, einen Verweis auf das von der Präpositionalphrase modifizierte Verb oder gegebenfalls Objekt aufzunehmen.[5] Das mdr-Attribut findet Verwendung, wenn das Nomen einer Präpositionalphrase selbst durch ein Adjektiv modifiziert wird, wie beispielsweise in der Konstruktion „Dogs sleep on the dirty floor.“.[6]

Der Beginn der Ableitung unterscheidet sich in diesem Fall insofern von vorhergehenden Beispielen, als das Nomen des Subjekts nicht von einem Artikel begleitet wird, sondern sich nur aus einem einzelnen Wort zusammensetzt. Da aber auch der Kategoriewert `pn` ein laut Anfangszustandsdefinition gültiges erstes Wort identifiziert, kann die Ableitung mit der Ausführung der Regel MAIN+FV beginnen, die in der Liste der möglichen ersten Regeln der Satzanfangsdefinition enthalten ist.

Auch an dieser Stelle wird die Klausel MAIN+FV.STAND der Regel MAIN+FV ausgeführt. Wie bereits gezeigt, verzeigert sie Proplets für Subjekt und Objekt über die jeweiligen Attribute fnc bzw. arg. Danach wird noch das Kategoriesegment `n-s3'`, welches das erwartete Subjekt repräsentiert, aus dem Proplet des Verbs gekürzt und das Proplet des aktuell nächsten Wortes in den Satzanfang übernommen.

Nun bildet die Präposition *on* das nächste Wort, dessen Analyse von der Regel X+PREP übernommen wird. Diese Regel enthält drei Klauseln, von denen eine die Analyse von Passivkonstruktionen behandelt und eine weitere die von Präpositionalphrasen, welche auf direkte Objekte folgen. Im vorliegenden Fall wird die dritte Klausel X+PREP.STAND ausgeführt, die das Proplet eines Verbs des Satzanfangs sowie das einer Präposition an der Position des nächsten Wortes identifiziert. Sind diese Eingabemuster erfüllt, so fügt X+PREP.STAND die Oberfläche des nächsten Wortes in das mdr-Attribut des Proplets des Satzanfangs ein. Da es sich bei *on* um ein Funktionswort handelt, verzichtet X+PREP.STAND darauf, das entsprechende Proplet in den Satz-

5. Vgl. dazu auch Abschnitt 5.10.5
6. Zur Analyse von Adjektiven vgl. Abschnitt 5.10.2

anfang zu übernehmen. Die einzige mögliche Folgeregel von X+PREP ist PREP+N.

Die Regel PREP+N enthält ebenfalls drei Klauseln, die der Verarbeitung der im vorigen Absatz beschriebenen Phänomene dienen. In diesem Beispiel übernimmt die Standardklausel PREP+N.STAND das Einlesen des nächstes Wortes *the*. Diese Klausel modifiziert die Proplets eines bereits eingelesenen Verbs und eines Nomens in der Position des nächsten Wortes, indem sie dem Nomenproplet zunächst das neue Attribut mdd zuweist und daraufhin dieses Attribut mit dem Wert des verb-Attributs des Verbproplets füllt. Anschließend kopiert sie den Wert des noun-Attributs des nächsten Wortes in das mdd-Attribut des Satzanfangsproplets und stellt auf diese Weise eine Verzeigerung zwischen dem Verb und dem das Verb modifizierende *Adverbial* her.

Danach kommt die bereits bekannte Regel DET+N zum Einsatz, welche die Proplets des Artikels *the* und des nächsten Wortes *floor* miteinander kombiniert. Auch diesmal ist der für ein Nomen stehende Platzhalter `@1` an zwei Stellen im Satzanfang vertreten und wird dementsprechend zweimal durch die Grundform des nächsten Wortes ersetzt.

Abgeschlossen wird die Ableitung wie gewöhnlich durch die Ausführung der Regel S+IP. Die resultierenden Attributwerte entsprechen erneut einer gültigen Endzustandsdefinition, so dass die Ableitung korrekt beendet wird.

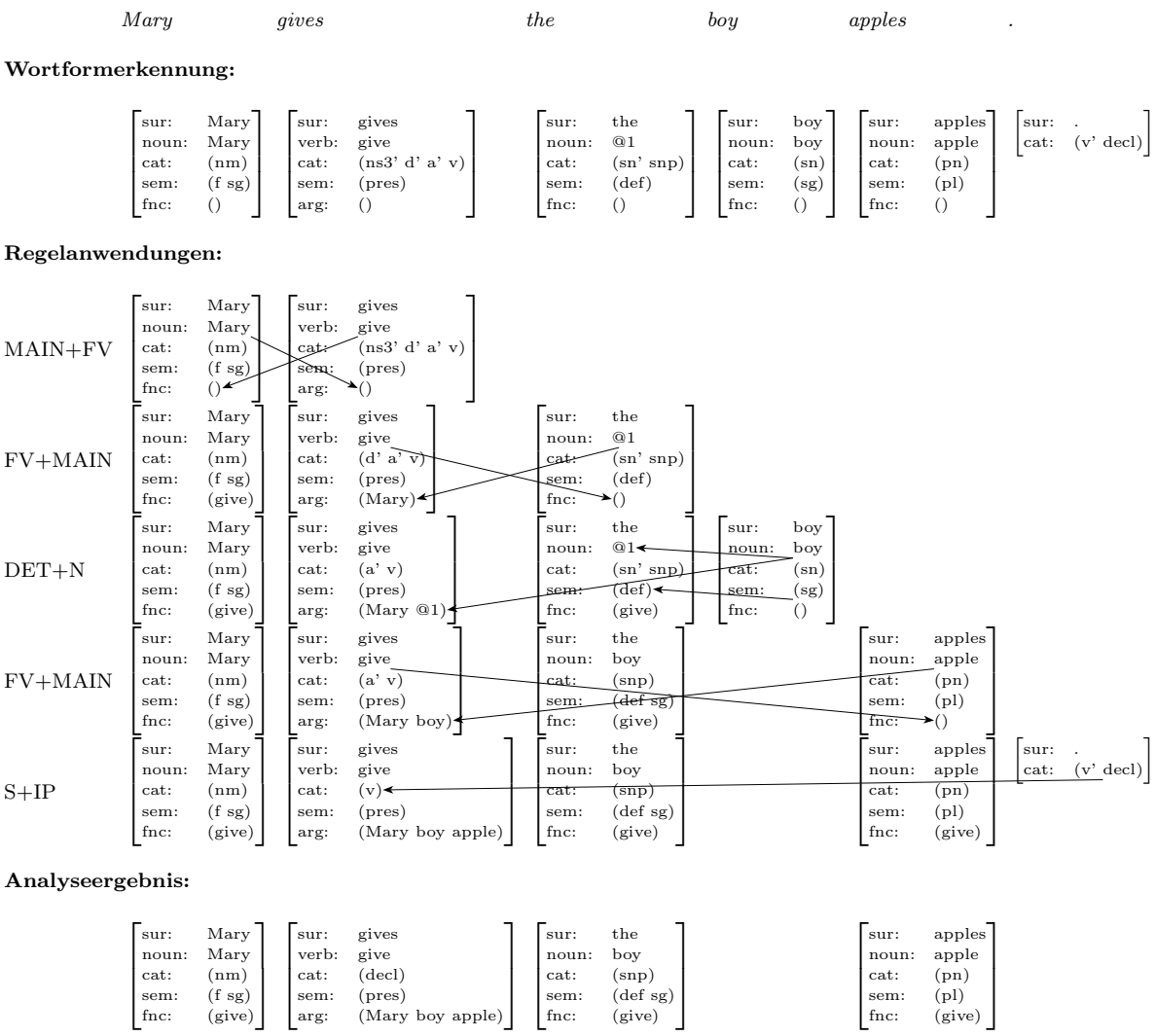
5.7 Analyse der Satzkonstruktion SVOO

Zur Veranschaulichen dieser Konstruktion dient der Beispielsatz „Mary gives the boy apples.“. Die Nominalphrase des Subjekts bildet hierbei der alleinstehende Name *Mary*, das Verb ist *gives*. Die Rolle des indirekten Objekts übernimmt die Nominalphrase *the boy*, die sich aus einem bestimmten Artikel und einem Nomen im Singular bildet. Das direkte Objekt ist in diesem Fall *apples*, ein Nomen im Plural.

5.7.1 Identifizierte Proplets und ihre Attribute

Die von der Morphologiekomponente erkannten Proplets verhalten sich größtenteils wie ähnliche Strukuren aus bereits behandelten Beispielen, wobei zwei Ausnahmen zu beachten sind. Das Subjekt realisiert sich in diesem Fall durch einen Namen, der als solcher durch seine Kategorisierung mit `nm` erkennbar ist. Darüber hinaus nimmt *gives* in diesem Fall die Rolle eines ditransitiven Verbs ein, was an den drei mit Apostroph markierten Kategoriesegmenten des entsprechenden Proplets erkennbar ist. Das erste repräsentiert hierbei das erwartete Subjekt, das zweite das indirekte und das dritte das direkte Objekt.

Abbildung 5.8: Ableitung einer SVOO-Konstruktion



5.7.2 Regelanwendungen

Um den gegebenen Beispielsatz zu analysieren, sind fünf Regelanwendungen nötig. Die verwendeten Regeln sind bereits aus vorhergehenden Abschnitten bekannt. Um das indirekte sowie das direkte Objekt verarbeiten zu können, kommt die Regel FV+MAIN im Zuge der Ableitung zweimal zum Einsatz.

Zunächst wird wieder die Menge der gültigen Satzanfangsdefinitionen überprüft. Da auch die Kategoriesierung des ersten Wortes *Mary* mit `nm` in Kombination mit der Regelanwendung von MAIN+FV akzeptiert wird, beginnt die Ableitung mit dieser Regel.

Die Klausel von MAIN+FV, die hier zum Einsatz kommt, ist erneut die Standardklausel MAIN+FV.STAND. Sie gleicht die Proplets der Oberflächen *Mary* und *gives* aufeinander ab und kopiert die jeweiligen Werte des noun- bzw. verb-Attributs in die Attribute fnc bzw. arg,

wie bereits in vorhergehenden Beispielen erläutert. Darüber hinaus wird das Kategoriesegment `ns3'` aus dem Proplet des Verbs gekürzt, das für das erwartete Subjekt steht. Danach wird das Proplet von *gives* in den aktuellen Satzanfang eingefügt.

Daraufhin wird die Ableitung mit der Regel FV+MAIN fortgesetzt, um das Proplet des aktuell nächsten Wortes *the* zu verarbeiten. Die ausgeführte Klausel ist in diesem Beispiel FV+MAIN.STAND die ganz ähnlich arbeitet, wie die direkt vorher ausgeführte Klausel der Regel MAIN+FV, wobei die Rollen von Satzanfang und nächstem Wort allerdings vertauscht sind. Ihrem Eingabemuster nach muss es sich beim Satzanfangsproplet um das eines Verbs handeln, dessen cat-Attribut noch ein Kategoriesegment enthält, welches ein indirektes oder direktes Objekt repräsentiert. In diesem Fall ist ein solches Segment `d'` vorhanden, das für ein erwartetes indirektes Objekt steht, weshalb das Proplet von *gives* akzeptiert wird. Als nächstes Wort gleicht FV+MAIN.STAND Wörter ab, deren Proplets ein noun-Attribut enthalten und nimmt deshalb das Proplet des Artikels *the* an.

Sind diese Eingabebedingungen erfüllt, so modifiziert die Klausel FV+MAIN.STAND die beiden Proplets, indem die Werte des verb- bzw. noun-Attributs in das arg-Attribut des Verbproplets bzw. das fnc-Attibut des Proplets des Nomens kopiert werden. Außerdem kürzt sie das eine Valenzstelle repräsentierende Kategoriesegment `d'` aus dem cat-Attribut des Verbs und fügt das Proplet des nächsten Wortes in den aktuellen Satzanfang ein.

An dieser Stelle bildet die Oberfläche *boy* mitsamt des dazugehörigen Proplets das nächste Wort und wird von der Regel DET+N vearbeitet. Wie in vorhergehenden Beispielen auch findet die Standardklausel DET+N.STAND Verwendung. Nachdem das Proplet des Artikels aus dem Satzanfang und das des Nomens an der Position des nächsten Wortes gültigen Eingabemustern der Klausel entsprechen, kodiert die Klausel die relevanten Informationen des Letzteren in Ersteres. Dazu werden die Werte des noun- und des sem-Attributs des Nomens an die Stellen der entsprechenden Attribute des Satzanfangsproplets geschrieben. Der Wert des noun-Attributs des nächsten Wortes ersetzt hierbei den Platzhalter @1, der in zwei Proplets des Satzanfangs auftaucht. Abschließend verwirft die Klausel das Proplet von *boy*, da sich seine Informationen im Proplet des Artikels *the* wiederfinden.

Eine der gültigen Folgeregeln von DET+N ist FV+MAIN, deren Klausel FV+MAIN.STAND bereits das vorhergehende indirekte Objekt behandelt hat und nun das Proplet des aktuell nächsten Wortes *apples* verarbeitet. Auch diesmal enthält das Proplet des Verbs im Satzanfang noch ein Kategoriesegment, welches ein erwartetes direktes Objekt repräsentiert, nämlich `a'`. Da das Proplet des nächsten Wortes über

ein noun-Attribut verfügt, entspricht es einem gültigen Eingabemuster der Regel FV+MAIN, welche die Proplets miteinander verzeigert. Dazu schreibt sie erneut den Wert des verb-Attributs des Satzanfangsproplets in das Attribut fnc des Proplets des nächsten Wortes und hängt den Wert des noun-Attributs des nächsten Wortes an den des arg-Attributs des Satzanfangsproplets an. Nachdem MAIN+FV.STAND das Kategoriesegment `a'` gekürzt hat, übernimmt sie das Proplet von *apples* in den Satzanfang.

Wie gewohnt schließt die Analyse daraufhin, indem S+IP das den Satz beendende Interpunktionszeichen einliest und die Kategorie des Verbs entsprechend mit dem Wert `decl` befüllt, was einer gültigen Definition des Endzustands entspricht.

5.8 Analyse der Satzkonstruktion SVOC

Der Beispielsatz „John got the engine running.“ dient der Erläuterung dieser Konstruktion. Das Subjekt wird hierbei erneut durch einen Namen, in diesem Fall *John*, gebildet und die Rolle des Verbs übernimmt die Wortform *got*. Die Nominalphrase *the engine*, die sich aus einem bestimmten Artikel und einem Nomen zusammensetzt, bildet das direkte Objekt. Den Satzteil des *Object Complements* liefert in diesem Beispiel das Adjektiv *running*.

5.8.1 Identifizierte Proplets und ihre Attribute

Die erkannten Proplets verhalten sich genau wie in bisher erläuterten Beispielen. Die Proplets der in Abbildung 5.8 gelieferten Illustration enthalten im Gegensatz zu den meisten vorangegangenen Beispielen allerdings wieder die Attribute mdr und mdd. Sie werden im Laufe der Ableitung mit Verweisen zwischen dem direkten Objekt sowie dem zugehörigen *Complement* versehen.

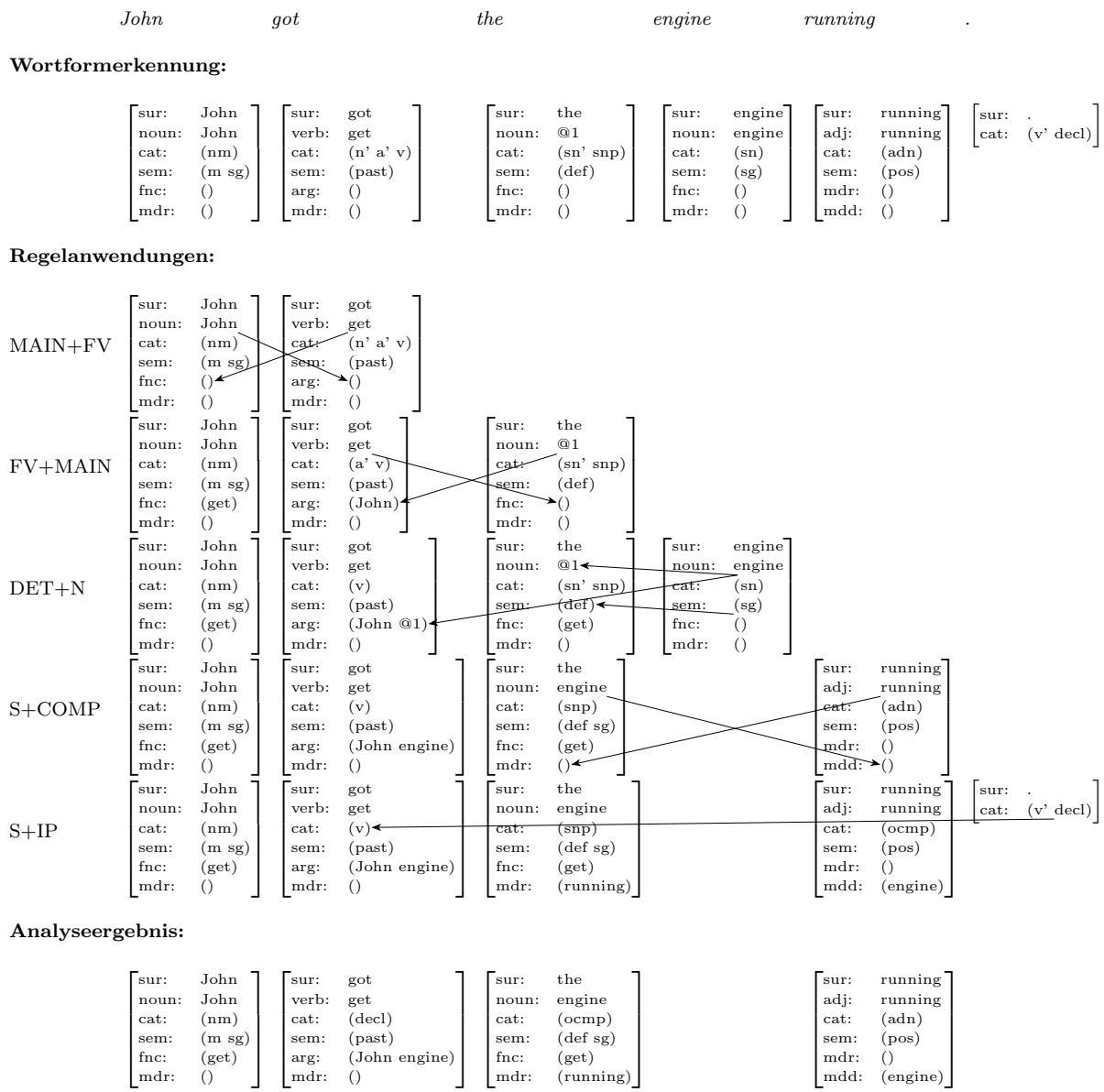
Erneut entsprechen die Kategoriesierung des ersten Wortes sowie der Ableitungsbeginn mit der Regel MAIN+FV einer gültigen Definition der Anfangszustände. Dadurch bildet *John* den aktuellen Satzanfang.

5.8.2 Regelanwendungen

Die Analyse des Beispielsatzes erfordert insgesamt fünf Regelanwendungen. Die benötigten Regeln wurden alle bereits vorgestellt, wobei die Verarbeitung des *Complements* in diesem Fall eine noch nicht erläuterte Klausel der Regel S+COMP voraussetzt, die im Folgenden genauer beleuchtet wird.

Erneut entsprechen die Kategorisierung des ersten Wortes sowie der Ableitungsbeginn mit der Regel MAIN+FV einer gültigen Definition

Abbildung 5.9: Ableitung einer SVOC-Konstruktion



der Anfangszustände. Dadurch bildet *John* den aktuellen Satzanfang und *got* das aktuell nächste Wort.

Die Proplets der beiden Wortformen werden, wie bereits in vorhergehenden Beispielen geschildert, von der Klausel MAIN+FV.STAND der Regel MAIN+FV als Subjekt und Verb akzeptiert und entsprechend modifiziert. Dazu befüllt MAIN+FV.STAND das fnc-Attribut des Satzanfangsproplets mit dem Wert des verb-Attributs des Proplets des nächsten Wortes und das arg-Attribut desselben mit dem Wert des noun-Attributs des Satzanfangsproplets. Anschließend kürzt die Klausel MAIN+FV.STAND das Segment `n'` aus der Kategorie des Verbs und nimmt das Proplet des nächsten Wortes in den Satzanfang auf.

Die Klausel FV+MAIN.STAND der Regel FV.MAIN behandelt die Analyse des Artikels *the* und verknüpft dessen Proplet mit dem des vorhergehenden Verbs. Dazu schreibt sie einerseits den Wert des verb-Attributs des Satzanfangsproplets in das fnc-Attribut des Proplets des nächsten Wortes. Andererseits fügt sie den Wert des noun-Attributs des Proplets des nächsten Wortes an den Wert des arg-Attributs des Proplets des Verbs an. Danach kürzt sie das ein Objekt repräsentierende Segment `a'` aus der Kategorie des Verbs und übernimmt das Proplet des Artikels in den Satzanfang.

Als nächstes Wort folgt das Nomen *engine* mitsamt seinem Proplet, dessen Verarbeitung wie aus vorhergehenden Beispielen bekannt die Klausel DET+N.STAND der Regel DET+N übernimmt. Der im Satzanfang an zwei Stellen vorkommende Platzhalter @1 wird mit dem Wert des noun-Attributs des Proplets des nächsten Wortes überschrieben, der Wert des sem-Attributs desselben Proplets an den des sem-Attributs des Satzanfangsproplets angehängt und das Proplet von *engine* daraufhin verworfen.

Nun bildet die Oberfläche *running* mit seinem Proplet das aktuell nächste Wort und führt zur Anwendung der Regel S+COMP. Dieselbe Regel verarbeitete bereits subjektbezogene *Complements* in Unterkapitel 5.5 mit der Klausel S+COMP.SC. Das in diesem Beispiel vorliegende objektbezogene *Complement* allerdings verlangt nach der Ausführung einer anderen Klausel, benannt mit S+COMP.OC.

Um sicherzustellen, dass sie die richtigen Merkmalstrukturen miteinander verzeigert, identifiziert S+COMP.OC insgesamt drei Proplets des Satzanfangs. Dabei nutzt die Klausel die Reihenfolge der Segmente des arg-Attributs des Verbproplets, um zwischen Subjekt und Objekt zu unterscheiden. Ersteres steht in den von vorliegendem Projekt behandelten Konstruktionen immer an erster Stelle des arg-Attributs, Letzteres an zweiter. Ist dieser Abgleich erfolgreich ausgeführt, so verzeigert die Klausel das nun eindeutig als Objekt identifizierte Proplet mit dem des nächsten Wortes. Dazu setzt sie den Wert des adj-Attributs der Merkmalstruktur des nächsten Wortes an den Wert des mdr-Attributs des Proplets, welches wie oben beschrieben als Objekt identifiziert wurde. Umgekehrt kopiert sie den Wert des noun-Attributs ebendieses Proplets in den Wert des mdd-Attributs der Merkmalstruktur des nächsten Wortes. Darüber hinaus wird das Proplet von *running* mit dem Kategoriewert `ocmp` gekennzeichnet, um im Analyseergebnis seine Rolle als *Object Complement* zu markieren.[7] Da es sich bei *running* als Adjektiv um ein Inhaltswort handelt, übernimmt S+COMP.OC abschließend dessen Proplet in den Satzanfang.

7. Für die Markierung von *Complements* vgl. auch Unterkapitel 5.3.

Auch diesmal bildet der Punkt das letzte Glied der Eingabe und wird von S+IP bearbeitet. Da die Kategorie des Verbs keine zu kürzenden Segmente mehr enthält, überschreibt die Regel sie mit dem Wert `decl` und führt so zu einem Zustand, der der Definition gültiger Endzustände entspricht.

5.9 Analyse der Satzkonstruktion SVOA

Der Satz „Jim eats the cake quickly." soll dazu dienen, diese Konstruktion zu veranschaulichen. Erneut bildet ein einzelner Name das Subjekt, in diesem Fall *Jim*. Die Wortform *eats* übernimmt die Rolle des Verbs und die Nominalphrase *the cake* die des direkten Objekts, welches sich aus Artikel und Nomen zusammensetzt. Das *Adverbial* wird in diesem Fall durch das Adverb *quickly* realisiert.

5.9.1 Identifizierte Proplets und ihre Attribute

Die Kategorisierung der erkannten Wörter entsprechen der vergleichbarer Proplets aus vorhergehenden Beispielen, wobei sich das Adverb *quickly* an dem Kategoriesegment `adv` als solches identifizieren lässt. Erneut wurden die Attribute mdd und mdr in der in Abbildung 5.10 skizzierten Ableitung berücksichtigt. Sie dienen der Verzeigerung von Verb und Adverb.

5.9.2 Regelanwendungen

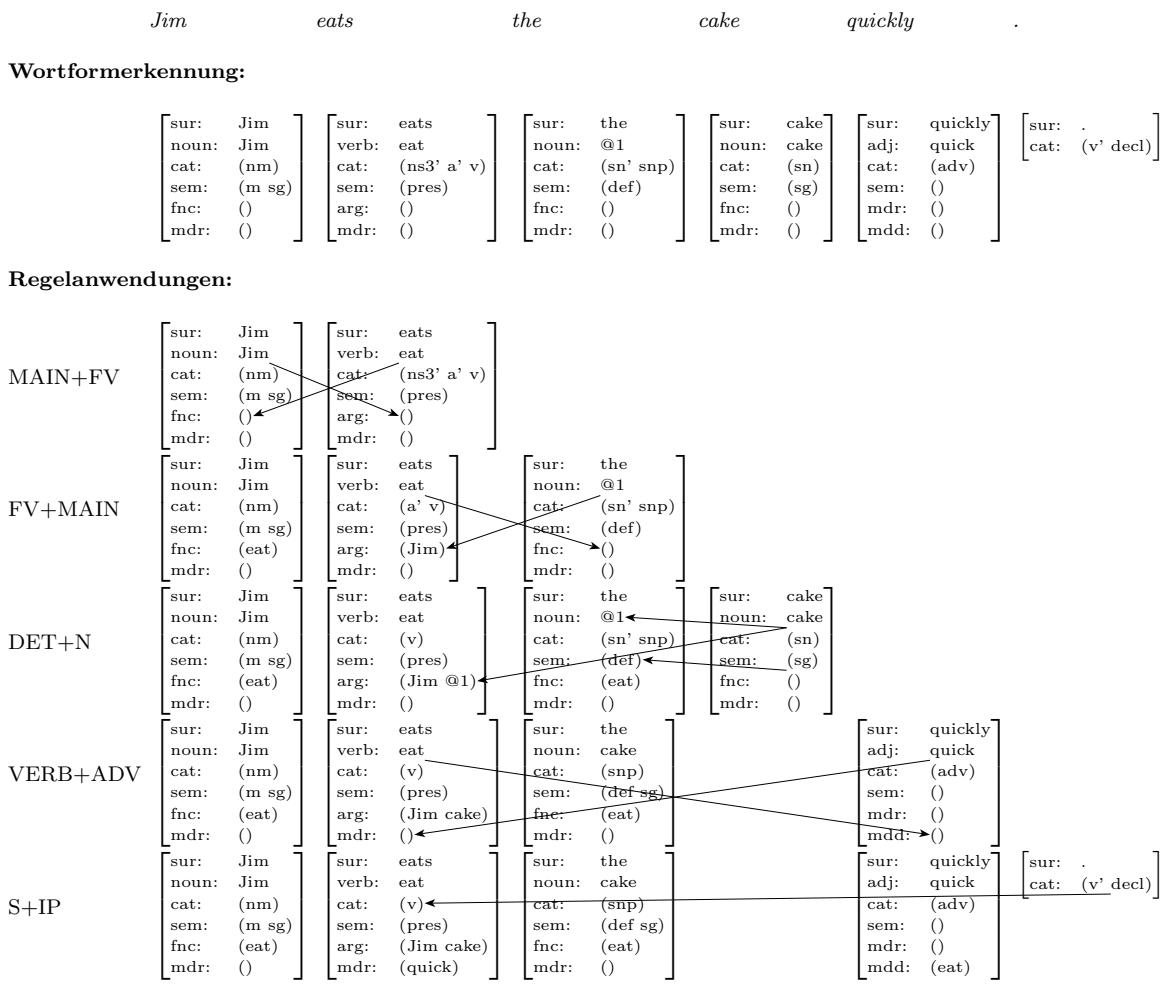
Die fünf benötigten Regeln für dieses Beispiel kamen bereits in vorigen Ableitungen zum Einsatz. Eine Ausnahme hierzu bildet die Regel VERB+ADV, welche die folgendende Erläuterung dementsprechend genauer beleuchtet.

Zunächst wird die Kategorie des ersten Wortes mit der Definition gültiger Startzustände abgeglichen und akzeptiert. Diese Startzustandsdefinition liefert auch die erste ausgeführte Regel MAIN+FV.

Wie bereits erläutert, verzeigert MAIN+FV, oder genauer die Klausel MAIN+FV.STAND, die Proplets des Subjekts und Verbs, indem der Wert des noun-Attributs des Satzanfangsproplets in den des arg-Attributs des Proplets des nächsten Wortes geschrieben wird. Umgekehrt kopiert MAIN+FV den Wert des verb-Attributs aus dem Proplet des Verbs in den des fnc-Attributs des Proplets des Nomens. Abschließend kürzt die Klausel das für das erwartete Subjekt stehende Kategoriesegment `ns3'` aus der Kategorie des Verbproplets und übernimmt dieses in den Satzanfang.

Darauf folgend verarbeitet die Klausel FV+MAIN.STAND der Regel FV+MAIN das aktuell nächste, dem Artikel *the* zugeordnete Pro-

Abbildung 5.10: Ableitung einer SVOA-Konstruktion



plet, indem sie die fnc- bzw. arg-Attribute der Merkmalstrukturen des nächsten Wortes und des Verbs auf die gleiche Weise mit den Werten des verb- bzw. noun-Attributs belegt. Anschließend wird das ein Objekt repräsentierende Kategoriesegment `a'` aus dem Proplet des Verbs gelöscht und das nächste Wort in den Satzanfang übernommen.

Die Ableitung fährt wie üblich mit der Anwendug der Klausel DET+N.STAND der Regel DET+N fort. Sie ersetzt den von vorhergehendem Artikel gelieferten Platzhalter @1 an allen Stellen des Satzanfangs durch die Grundform des nächsten Nomens. Darüber hinaus hängt sie den Wert des sem-Attributs des nächsten Wortes an das sem-Attribut des Artikels an und verwirft abschließend das Proplet des nächsten Wortes.

An dieser Stelle bildet das Adverb *quickly* das nächste zu analysierende Wort, für dessen Verarbeitung die Regel VERB+ADV zuständig ist. Als Eingabemuster identifiziert diese Regel das Proplet des Verbs des Satzanfangs anhand des verb-Attributs und das Proplet eines Adverbs in der Position des nächsten Wortes anhand des Kategoriesegments `adv`. Nachdem diese beiden Muster erfolgreich abgeglichen wurden, verzeigert die Regel die beiden Proplets, indem sie den Wert des adj-Attributs des Proplets des nächsten Wortes in den Wert des mdr-Attributs des Satzanfangsproplets kopiert. Umgekehrt schreibt sie auch den Wert des verb-Attributs des Verbs in das mdd-Attribut des Adverbs. Die Merkmalstruktur von *quickly* wird anschließend in den Satzanfang übernommen.

Das den Satz beendende Interpunktionszeichen wird erneut von der Regel S+IP behandelt, welche die Kategorie des Verbproplets mit dem Wert `decl` überschreibt und die Ableitung damit in eine gültige Definition der Endzustände überführt.

5.10 Analyse intrapropositionaler Funktor-Argument-Struktur

Einige der von Kapitel sechs aus Hausser (2006, S. 81-102) behandelten Phänomene wurden bereits in den vorhergehenden Unterkapiteln 5.3 bis 5.9 dieser Arbeit behandelt. In solchen Fällen verweisen die folgenden Abschnitte auf entsprechende Unterkapitel und werden die Verarbeitung entsprechender Konstruktionen nur kurz umreißen.

5.10.1 Artikel

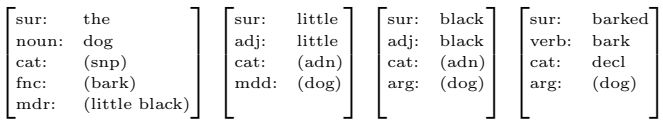
Hausser (2006, S. 89) liefert für die Verarbeitung von Artikeln den Beispielsatz „The man gave the child an apple.“, was einer in Unterkapitel 5.7 erläuterten SVOO-Konstruktion entspricht. Im vorliegenden Projekt verarbeitet die Regel DET+N das Verschmelzen der Information von Artikel- und Nomina-Proplets, wobei der Platzhalter `@1` in der Kategorie der Merkmalstruktur des Artikels in Kombination mit der Operation `setref` verwendet wird, um die korrekte Verzeigerung von Nomina und Verben zu gewährleisten.

5.10.2 Adjektive

Für die Analyse von Adjektiven liefert Hausser (2006, S. 94) den Beispielsatz „The little black dog barked.“, wobei angemerkt werden soll, dass das vorliegende Projekt auch die alternative Verwendung von Kommata berücksichtigt. Abbildung 5.11 veranschaulicht das vom vorliegenden System gelieferte Analyseergebnis.

Das Einlesen und Verarbeiten von Adjektiven übernimmt die Regel DET+ADJ. Sie bestimmt das Proplet eines Artikels aus dem Satzanfang

Abbildung 5.11: Analyseergebnis einer Konstruktion mit Adjektiven

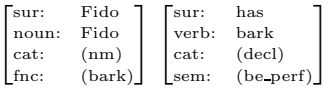


und überprüft anhand der Kategorisierung des nächsten Wortes, ob es sich bei diesem um ein Adjektiv handelt. Ist dieser Abgleich erfolgreich, so verzeigert die Regel die beiden Proplets über die jeweiligen Attribute mdr bzw. mdd, wie aus Abbildung 5.11 ersichtlich. Da es sich bei Adjektiven um Inhaltswörter handelt, übernimmt DET+ADJ abschließend das Proplet des nächsten Wortes in den Satzanfang.

5.10.3 Hilfsverben

Anhand des Beispielsatzes „Fido has been barking.“ erläutert Hausser (2006, S. 97) die Verarbeitung von Hilfsverben, die in diesem Beispiel benutzt werden, um eine komplexe Verbalphrase zu bilden. Abbildung 5.12 illustriert das Analyseergebnis des für diese Arbeit entwickelten Systems.

Abbildung 5.12: Analyseergebnis einer komplexen Verbalphrase



Um die drei Elemente der komplexen Verbalphrase zu analysieren werden drei Regelanwendungen benötigt. Die vorliegende Konstruktion verarbeiten die drei Regeln MAIN+FV, CVP1 und CVP2. Erstere verarbeitet *has*, setzt die entsprechenden Verweise in den Attributen fnc- und arg des Nomen- bzw. Verbproplets, kürzt die entsprechende Valenzstelle aus der Kategorie des Verbs und kopiert das Proplet des nächsten Wortes abschließend in den Satzanfang. Die nächsten beiden Proplets der Oberflächen *has* und *been* werden von den beiden Regeln CVP1 bzw. CVP2 eingelesen, wobei keine der beiden Regeln die Merkmalstruktur des entsprechenden nächsten Wortes in den Satzanfang übernimmt, sondern deren Information stattdessen in das bereits eingelesene Verbproplet kodiert. Im Laufe der Ableitung wird dabei das sem-Segment `be_perf` in den entsprechenden Wert des Verbproplets eingetragen, um die Tempus/Aspektform *present perfect progressive* zu kodieren. Außerdem ersetzt CVP2 den Wert des verb-Attributs des Proplets von *has* durch den des Vollverbs *bark*.

5.10.4 Passivkonstruktionen

Die Bearbeitung des Phänomens des Passivs illustriert Hausser (2006, S. 98f.) mit dem Beispielsatz „The book was read by John.“, dessen Analyseergebnis in Abbildung 5.13 gezeigt wird.

Abbildung 5.13: Analyseergebnis einer Passivkonstruktion

sur:	the		sur:	was		sur:	John				
noun:	book		verb:	read		noun:	John				
cat:	(snp)		cat:	(decl)		cat:	(nm)				
fnc:	(read)		arg:	(John book)		fnc:	(read)				
			sem:	(be_past pass)							

Die Regel PASS wird eingesetzt, um die Information des Proplets des Vollverbs *read* in die Merkmalstruktur des vorhergehenden Hilfsverbs zu übernehmen, woraufhin die Regel das Proplet des nächsten Wortes verwirft. Darüber hinaus markiert die Regel PASS das Verbproplet des Satzanfangs mit dem Segment `pass` im sem-Attribut. Die Oberfläche *by* wird von der spezialisierten Klausel X+PREP.PASS eingelesen, die sicherstellt, dass es sich um eine Präposition handelt, die in Passivkonstruktionen in Erscheinung treten darf, und das Proplet des nächsten Wortes daraufhin verwirft. Darauf folgend verarbeitet die Klausel PREP+N.PASS das Proplet der Oberfläche *John*, indem sie den Wert des noun-Attributs an den Anfang des Werts des arg-Attributs des Verbs kopiert und das Proplet der Oberfläche *John* somit als Subjekt des Satzes ausweist.

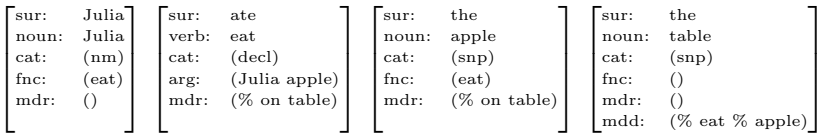
5.10.5 Präpositionalphrasen

Um die Analyse von Präpositionalphrasen zu verdeutlichen, liefert Hausser (2006, S.101) den Beispielsatz „Julia ate the apple on the table.“ einschließlich Ableitung, dessen Präpositionalphrase ganz ähnlich aufgebaut ist wie die des in Unterkapitel 5.6 ausführlich behandelten Beispielsatzes. Hausser (2006, S. 100ff.) merkt an, dass in seinem Beispielsatz eine Ambiguität dahingehend entsteht, dass nicht eindeutig geklärt werden kann, ob die Präpositionalphrase sich auf das Verb *ate* oder das Objekt *the apple* bezieht.

Um dieses Phänomen in der Analyse darzustellen, wird deshalb sowohl das Proplet des Verbs als auch das des Objekts mit dem der Präpositionalphrase über die Attribute mdr bzw. mdd verzeigert und entsprechende Verweise darüber hinaus mit dem auf die Ambiguität hinweisenden Zeichen `%` markiert. Dazu kommen aufeinanderfolgend die beiden Klauseln X+PREP.OBJ und PREP+N.OBJ zum Einsatz. Abbildung 5.14 veranschaulicht das Analyseergebnis des vorliegenden Syntaxgrammtiksystems.[8]

8. Für einige Abweichungen der Kategoriewerte des Präpositionalphrasenproplets im Vergleich zu der von Hausser (2006, S.101) gelieferten Analyse vgl. Abschnitt 5.6.2.

Abbildung 5.14: Analyseergebnis einer Präpositionalphrase



5.11 Analyse extrapropositionaler Funktor-Argument-Struktur

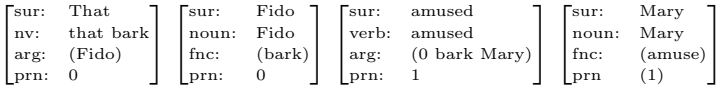
Das Phänomen der extrapropositionalen Funktor-Argument-Struktur bezeichnet in der Terminologie von Hausser (2006, S. 103ff.) solche Konstruktionen, die in der Anglistik ganz allgemein unter den Begriffen *th-* bzw. *wh-clauses* zusammengefasst werden.[9] Die folgenden Abschnitte beleuchten, wie das vorliegende System diese Phänomene verarbeitet.

Die Beispielproplets enthalten dabei das bisher noch nicht erläuterte Attribut prn, welches die Propositionsnummer der entsprechenden Wortform enthält. Dieses Attribut dient dazu, die Verschachtelung von Relativsätzen widerzuspiegeln.

5.11.1 th-clauses

th-Bsp. 1.) Hausser (2006, S. 100) liefert den ersten Beispielsatz einer solchen Konstruktion mit „That Fido barked amused Mary.“. Die Analyse dieses Satzes veranschaulicht Abbildung 5.15.

Abbildung 5.15: Analyseergebnis eines th-clauses *(1)*



Für die Verarbeitung einer solchen Konstruktion sind einige spezialisierte Regeln bzw. Klauseln notwendig. TH+N.SUB kombiniert die Proplets von *That* und *Fido*, wobei sie einen Platzhalter `v1` in beide Proplets einfügt, der im Verlauf der Ableitung durch den Wert des arg-Attributs des den *th-clause* abschließenden Verbs ersetzt wird. Die Klausel TH+V.SUB verarbeitet das Proplet *barked*, übernimmt ebendiese Ersetzung und verwirft das aktuell nächste Wort. Anschließend wird die Klausel MAIN+FV.TH-CL ausgeführt, die die Merkmalstruktur von *amused* bearbeitet. Sie erhöht den Zähler der Propositionsnummer und kopiert die Werte des prn- sowie verb-Attributs des vorhergehenden Verbs in das arg-Attribut des nächsten Wortes, um die Funktor-Argument-Struktur des Satzes darzustellen.

th-Bsp. 2.) Ein weiterer Beispielsatz mitsamt Analyse findet sich in Hausser (2006, S. 101) als „John heard that Fido barked.“. Das ent-

9. Für eine ausführliche Erläuterung dieser Phänomene vgl. Hausser (2006, Kapitel 7).

sprechende Analyseergebnis des in dieser Arbeit entwickelten Projekts zeigt Abbildung 5.16.

Abbildung 5.16: Analyseergebnis eines th-clauses *(2)*

sur:	John		sur:	heard		nv:	that bark		sur:	Fido
noun:	John		verb:	hear		fnc:	(0 hear)		noun:	Fido
fnc:	(hear)		arg:	(John 1 bark)		arg:	(Fido)		fnc:	(bark)
prn:	0		prn:	0		prn:	1		prn:	1

Nachdem die beiden ersten Wörter nach bekanntem Muster durch die Regel MAIN+FV verarbeitet wurden, liest die Regel FV+TH das Proplet von *that* ein, wobei sie die Propositionsnummer erhöht und einen Platzhalter für eine spätere Verzeigerung des abschließenden Verbs setzt. Außerdem kopiert sie die Werte des verb- und prn-Attributs des bereits verarbeiteten Verbproplets in das fnc-Attribut des Proplets von *that* und kürzt eine Valenzstelle aus der Kategorie des Verbproplets. Anschließend übernimmt die Klausel TH+N.OBJ die Analyse des Proplets von *Fido* und schreibt dessen noun-Wert in das arg-Attribut des Proplets von that. Das Proplet der nachfolgenden Oberfläche *barked* wird von der Klausel TH+N.OBJ eingelesen, welche die vorher eingefügten Platzhalter ersetzt und das Proplet verwirft.

5.11.2 wh-clauses

wh-Bsp. 1.) Hausser (2006, 103) gibt als erstes Beispiel einer *wh*-Konstruktion den Satz „The dog which saw Mary quickly barked.“. Abbildung 5.17 veranschaulicht die dazugehörige Analyse des vorliegenden Systems.

Abbildung 5.17: Analyseergebnis einer wh-*Konstruktion (1)*

sur:	The		sur:	which		sur:	Mary		sur:	quickly		sur:	barked
noun:	dog		av:	(saw)		noun:	Mary		adj:	quick		verb:	bark
fnc:	(bark)		arg	(# Mary)		fnc:	(saw)		mdr:	()		arg:	dog
mdr:	(1 saw)		mdd:	(0 dog)		mdr:	()		mdd:	saw		mdd:	()
prn:	0		mdr:	(quick)		prn:	1		prn:	1		prn:	0
			prn:	0									

Um diese Konstruktion zu verarbeiten sind eine ganze Reihe noch nicht behandelter, spezialisierter Regeln bzw. Klauseln notwendig. Die Regel MAIN+WHICH ist dafür zuständig, die Oberfläche *which* zu verarbeiten. Sie speichert einen Zeiger auf das bereits analysierte Subjekt des Satzes im mdd-Attribut des Proplets des nächsten Wortes, markiert dieses mit dem Zeichen # im arg-Attribut und inkrementiert darüber hinaus den Propositionszähler. Daraufhin bearbeitet WHICH+X.V1 das Proplet des nächsten Wortes *saw*, indem es den Wert des verb-Attributs an die entsprechenden, Verzeigerungen realisierenden Attribute der Proplets des Satzanfangs kopiert. *Mary* wird von der Regel WHICH+MAIN verarbeitet, die den Wert des noun-Attributs des nächsten Wortes an den Wert des arg-Attributs des mit # markierten Proplets schreibt. Daraufhin bildet das Proplet des Adverbs *quickly* das nächste Wort, welches

von der Klausel WHICH+X.ADV bearbeitet wird, die den Wert des adj-Attributs des Proplets des nächsten Wortes in das mdr-Attribut des mit # markierten Proplets kopiert. Die das nächste Wort *barked* verarbeitende Klausel MAIN+FV.WH dekrementiert zunächst den Propositionszähler, da der *wh-clause* als abgeschlossen betrachtet wird. Darüber hinaus verzeigert sie die Proplets des Subjekts mit dem des Verbs an der Position des nächsten Wortes, genau wie die bereits behandelte Standardklausel MAIN+FV.STAND.

wh-Bsp. 2.) Als zweites Beispiel dieser Art liefert Hausser (2006, S. 111) den Satz „The little dog which Mary saw barked.“. Das entsprechende Analyseergebnis des vorliegenden Systems kann in Abbildung 5.18 eingesehen werden.

Abbildung 5.18: Analyseergebnis einer wh-*Konstruktion (2)*

sur:	The	sur:	little	sur:	which	sur:	Mary	sur:	barked
noun:	dog	adj:	little	av:	(saw)	noun:	Mary	verb:	bark
fnc:	(bark)	mdr:	()	arg	(Mary #)	fnc:	(saw)	arg:	dog
mdr:	(little 1 saw)	mdd:	dog	mdd:	(0 dog)	mdr:	()	mdd:	()
prn:	0	prn:	0	mdr:	(quick)	prn:	1	prn:	0
				prn:	1				

Die Ableitung verwendet viele der im vorhergehenden Beispiel genutzten Regeln wieder. Das Proplet der Oberfläche *which* wird erneut von MAIN+WHICH bearbeitet, die den Propositionszähler erhöht und das Proplet des nächsten Wortes an den Satzanfang anfügt. Daraufhin kommt die Regel WHICH+MAIN zum Einsatz, die den Wert des noun-Attributs des nächsten Wortes in das arg-Attribut des Proplets von *which* einfügt und beide Proplets zum Zweck der späteren Verzeigerung des Verbs mit einem Platzhaltersegment v2 versieht. Die Verarbeitung des folgenden Proplets der Wortform *saw* wird von der Klausel MAIN+FV.WH vorgenommen, die den Wert des verb-Attributs des nächsten Wortes in die Werte des av-Attributs des Proplets von *which* und des fnc-Attributs des Proplets von *Mary* einfügt, daraufhin die Merkmalstruktur des nächsten Wortes verwirft und den Propositionszähler dekrementiert. Anschließend wird die Verarbeitung von *barked* von der Klausel WHICH+X.V2 übernommen, die das Proplet des Subjekts des Hauptsatzes, *the dog*, mit dem Proplet des nächsten Wortes über die Attribute fnc bzw. arg verzeigert.

wh-Bsp. 3.) Das dritte Beispiel einer *wh-Konstruktion* findet sich in Hausser (2006, S. 113) mit „When Fido barked Mary smiled.“. Abbildung 5.19 illustriert das Analyseergebnis des Syntaxprojekts.

Abbildung 5.19: Analyseergebnis einer wh-*Konstruktion (3)*

sur:	When	sur:	Fido	sur:	Mary	sur:	smile
av:	(when bark)	noun:	Fido	noun:	Mary	verb:	smile
arg	(Fido)	fnc:	(bark)	fnc:	(smile)	arg:	(Mary)
mdd:	(1 smile)	mdr:	()	mdd:	dog	mdr:	(0 bark)
mdr:	(quick)	prn:	0	prn:	1	prn:	1
prn:	0						

Um diese Eingabe zu verarbeiten wird als erstes WHEN+MAIN angewandt, die den Wert des noun-Attribut des Proplets des nächsten Wortes, in diesem Fall *Fido*, in den des arg-Attributs des Proplets des Satzanfangs kopiert. Abschließend kopiert die Regel das Proplet des nächsten Wortes an den Satzanfang. Die Klausel MAIN+FV.CNJ übernimmt die Verarbeitung des Proplets der nächsten Wortform *barked*. Sie schreibt den Wert des verb-Attributs des nächsten Wortes in den Wert des av-Attributs des *wh*-Proplets und in den des fnc-Attributs des Nomens und verwirft anschließend das Proplet des nächsten Wortes. Das Proplet der Oberfläche *Mary* wird von der Klausel S+IP.CNJ1 eingelesen, die den Propositionszähler erhöht und das Proplet des nächsten Wortes an den Satzanfang anhängt. Anschließend übernimmt die bereits oft eingesetzte Klausel MAIN+FV.STAND die Analyse des Verbs an der Position des nächsten Wortes und kopiert dessen Proplet in den Satzanfang. Um die korrekte Verzeigerung zwischen Haupt- und Nebensatz zu gewährleisten, verarbeitet die Klausel S+IP.CNJ2 den abschließenden Punkt. Sie kopiert das letzte Element des sv-Attributs des *wh*-Proplets und dessen prn-Wert in das mdr-Attribut des Verbs des Hauptsatzes und umgekehrt die Werte des verb- und prn-Attributs des Proplets des Verbs in das mdd-Attribut des *wh*-Proplets.

5.12 Analyse intrapropositionaler Koordination

Um die Beziehungen von in Koordination zueinander stehender Merkmalstrukturen darstellen zu können, enthalten die im Folgenden gelieferten beispielhaften Analyseergebnisse die Attribute nc und ic. Ersteres steht dabei für *next conjunct* und verweist jeweils auf das nächste Glied einer Koordinationskette. Letzteres steht für *initial conjunct* und verweist auf das jeweils erste Glied.[10] Darüber hinaus werden in diesem Unterkapitel Regelanwendungen wie die Kombination von Artikel mit dazugehörigem Nomen, die bereits ausführlich erläutert wurden, außer acht gelassen. Stattdessen konzentrieren sich die folgenden Abschnitte darauf, noch nicht besprochene Regeln und Klauseln zu erörtern.

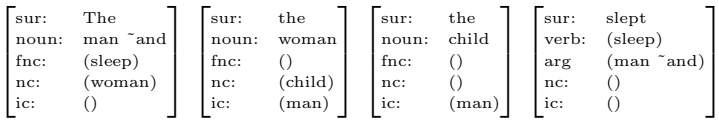
5.12.1 Koordination von Nomina

Hausser (2006, S. 118ff.) liefert für die Koordination von Nomina im Subjekt und im Objekt jeweils einen Beispielsatz. Da beide Konstruktionen mit den gleichen Regeln verarbeitet werden können, soll an dieser Stelle nur das erste Beispiel genauer betrachtet werden. Dieses Beispiel

10. Im vorliegenden Projekt wurde bewusst das Attribut pc (*previous conjunct*) durch das Attribut ic ersetzt. Dies birgt Vorteile bei der eventuellen Weiterverarbeitung von Analyseergebnissen, da in einem einzigen Schritt von einem beliebigen Glied der Koordinationskette aus das erste Glied ebenjener angesteuert werden kann.

lautet „The man, the woman, and the child slept.“ (Hausser, 2006, S. 119).

Abbildung 5.20: Analyseergebnis einer Nominalkoordination



Die Analyse des Beispielsatzes erfordert insgesamt zehn Regelanwendungen. Die Kombination des Proplets von *The man* mit dem nachfolgenden Komma übernimmt die Regel N+COMMA, die das Proplet des nächsten Wortes verwirft und die Ableitung ansonsten nur durch ihre möglichen Folgeregeln beeinflusst. Eine dieser möglichen Folgeregeln ist COMMA+N, deren Standardklausel das Proplet des nächsten Artikels verarbeitet, indem sie den Wert des noun-Attributs des nächsten Wortes in den Wert des nc-Attributs des Satzanfangs kopiert und umgekehrt das noun-Attribut des Satzanfangsproplets in das ic-Attribut des nächsten Wortes schreibt.

N+COMMA verarbeitet auch das nächste Komma, wobei diesmal eine andere Folgeregel zum Einsatz kommt, um die Konjunktion *and* einzulesen. Die Klausel N+CNJ.STAND dieser Folgeregel verwirft das Proplet von *and*, hängt jedoch vorher dessen Oberfläche an die Oberfläche des Proplets des Satzanfangs an, das das erste Glied der Koordinationskette darstellt. N+CNJ.STAND identifiziert dieses Proplet daran, dass seinem ic-Attribut noch kein Wert zugewiesen wurde. Zusätzlich markiert die Regel das Proplet des ersten Glieds einer Koordinationskette mit dem Zeichen ˜, um es im Analyseergebnis sofort als solches erkenntlich zu machen.

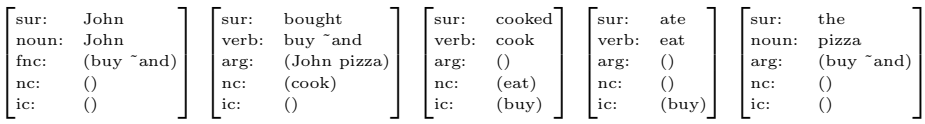
Die Verarbeitung des Proplets des dritten Artikels wird daraufhin von der Klausel CNJ+N.STAND vorgenommen, die den Wert des ic-Attributs des vorhergehenden Koordinationsglieds in den Wert des ic-Attributs des aktuell ersten Wortes übernimmt.

5.12.2 Koordination von Verben bzw. Adjektiven

Für die Verarbeitung dieser beiden Phänomene sind zwei unterschiedliche Sets von Regeln nötig, um den unterschiedlichen Merkmalen der beiden Wortarten gerecht zu werden. Grundsätzlich funktioniert die Koordination in beiden Fällen aber nach sehr ähnlichem Muster, weshalb an dieser Stelle nur das Beispiel für Verben behandelt werden soll.

Als Beispiel für die Koordination von Verben liefert Hausser (2006, S. 121f.) den Satz „John bought, cooked, and ate the pizza.“, dessen Analyseergebnis der erstellten Syntaxgrammatik Abbildung 5.21 zeigt.

Abbildung 5.21: Analyseergebnis einer Verbkoordination



Für die Koordination von Verben werden insgesamt vier noch nicht behandelte Regeln bzw. deren Klauseln benötigt. V+COMMA übernimmt dabei den ersten Schritt der Verarbeitung der Koordination, indem sie auf die Modifikation bereits eingelesener Proplets verzichtet, das Proplet des ersten Kommas an der Position des nächsten Wortes verwirft und den Fortgang der Ableitung allein durch die Spezifikation möglicher Folgeregeln beeinflusst.

Die Klausel COMMA+V.STAND verarbeitet das Proplet der darauf folgenden Wortform *cooked*, indem sie zunächst die Proplets des im Satzanfang enthaltenen Verbs und das des nächsten Wortes miteinander verzeigert. Dazu kopiert sie den Wert des verb-Attributs des nächsten Wortes in den Wert des nc-Attributs des Satzanfangsproplets und umgekehrt den Wert des verb-Attributs des Proplets des Satzanfangs in den Wert des ic-Attributs des Proplets des nächsten Wortes. Daraufhin nimmt COMMA+V.STAND das Proplet des nächsten Wortes in den Satzanfang auf.

Das darauf folgende Komma wird von V+COMMA auf die gleiche Weise bearbeitet, wie das vorhergehende. Allerdings tritt an dieser Stelle eine andere Folgeregel in Kraft.

Das Proplet der Konjunktion *and* an der Stelle des nächsten Wortes wird von der Klausel V+CNJ.VCO eingelesen, die das Proplet des ersten Koordinationsglieds daran erkennt, dass seinem ic-Attribut kein Wert zugewiesen wurde. V+CNJ.VCO markiert daraufhin den Wert des verb-Attributs ebenjenes Proplets mit dem Zeichen ~ und der Oberfläche der Konjunktion. Daraufhin verwirft die Klausel das Proplet des nächsten Wortes.

Die Klausel CNJ+V.STAND ist für die Verarbeitung des letzten Glieds einer Koordinationskette von Verben zuständig, in diesem Fall die Wortform *ate*. Sie kopiert einerseits den Wert des ic-Attributs des Proplets des vorhergehenden Verbs in den Wert des ic-Attributs des Proplets des nächsten Wortes und andererseits den Wert des verb-Attributs des Proplets des nächsten Wortes in den Wert des nc-Attributs des Satzanfangs. Anschließend übernimmt die Klausel das Proplet des nächsten Wortes in den Satzanfang.

5.12.3 Analyse von Subject-, Verb- und Object-Gapping

Zur Illustration dieser drei Phänomene liefert Hausser (2006) die folgenden drei Beispielsätze:

[1] „Bob ate an apple, walked his dog, and read a paper." (S. 127)

[2] „Bob ate an apple, Jim a pear, and Bill a peach." (S. 130)

[3] „Bob bought, Jim peeled, and Bill ate the peach." (S. 133)

Satz [1] bildet hierbei ein Beispiel für *Subject-*, [2] eines für *Verb-* und [3] eines für *Object-Gapping.*[11] Alle drei Phänomene wurden im vorliegenden JSLIM-Projekt umgesetzt, wobei die erstellte Syntaxgrammatik sie alle nach sehr ähnlichen Mustern verarbeitet. Deshalb soll an dieser Stelle die Analyse des *Gappings* exemplarisch am Beispiel des *Subject-Gappings* und damit anhand der Ableitung des Beispielsatzes [1] demonstriert werden. Abbildung 5.22 illustriert das Analyseergebnis des implementierten Systems für diesen Satz.

Abbildung 5.22: Analyseergebnis einer Konstruktion mit Verb-Gapping

sur:	Bob
noun:	Bob
fnc:	(eat)
nc:	()
ic:	()

sur:	ate
verb:	eat ~and
arg:	(Bob apple)
nc:	(walk)
ic:	()

sur:	an
noun:	apple
fnc:	(eat)
nc:	()
ic:	()

sur:	walked
verb:	walk
arg:	(# dog)
nc:	(read)
ic:	(eat)

sur:	his
noun:	dog
fnc:	(walk)
nc:	()
ic:	()

sur:	read
verb:	read
arg:	(# paper)
nc:	()
ic:	(eat)

sur:	paper
noun:	paper
fnc:	(read)
nc:	()
ic:	()

Die Analyse des Beispielsatzes verläuft zunächst nach dem bereits in Unterkapitel 5.4 erläuterten Schema SVO, bis das erste Komma das nächste Wort der Eingabe bildet. Die Verarbeitung seines Proplets übernimmt die Regel N+COMMA, die, ähnlich wie V+COMMA in vorangegangenem Abschnitt, das Proplet des nächsten Wortes unbearbeitet verwirft und den Fortgang der Ableitung nur durch ihre möglichen Folgeregeln beeinflusst.

Die Klausel SUB_GAP.COM+V bearbeitet das darauf folgende Proplet der Wortform *walked* und verzeigert das dazugehörige Proplet mit dem von *ate*, indem sie den Wert des verb-Attributs des Satzanfangsproplets in den Wert des ic-Attributs des Proplets des nächsten Wortes und den Wert des verb-Attributs des Proplets des nächsten Wortes in den Wert des nc-Attributs des Satzanfangsproplets kopiert. Darüber hinaus markiert die Klausel die erste Stelle des arg-Attributs des nächsten Wortes mit dem Zeichen #, um das Phänomen des *Subject-Gappings* widerzuspiegeln. Anschließend übernimmt SUBGAP.COM+V das Proplet des nächsten Wortes in den Satzanfang.

Das nächste Komma des Beispielsatzes wird erneut von der Regel N+COMMA behandelt, woraufhin diesmal allerdings die Klausel

11. Für eine ausführliche Erläuterung dieser Phänomene vgl. Hausser (2006, Unterkapitel 8.4-8.6).

SUB_GAP.N+CNJ zum Einsatz kommt, um die Konjunktion *and* zu verarbeiten. Sie markiert das verb-Attribut des Proplets des ersten Verb der Koordinationskette mit dem Zeichen ~ sowie der Oberfläche der Konjunktion und verwirft daraufhin das Proplet des nächsten Wortes.

Nun verarbeitet die Klausel SUB_GAP.CNJ+V das Proplet des nächsten Verbs, indem sie es mit dem Proplet des vorhergehenden Verbs verzeigert. Dazu kopiert sie den Wert des ic-Attributs des Satzanfangsproplets in den Wert des ic-Attributs des Proplets des nächsten Wortes und den Wert des verb-Attributs des Proplets des nächsten Wortes in den Wert des nc-Attributs des Proplets des Satzanfangs. Außerdem fügt die Klausel das Zeichen # an die erste Stelle des arg-Attributs des Proplets des nächsten Wortes an, um das Phänomen des *Subject-Gappings* im Analyseergebnis zu markieren.

6 Evaluierung des Projekts

Dieses Kapitel wirft einen kritischen Blick auf die Ergebnisse der in JSLIM entwickelten Syntaxgrammatik des Englischen. Zunächst sollen zu diesem Zweck die Zusammenstellung und Gliederung der Daten beschrieben werden, die dieser Evaluierung zugrunde liegen. Im Anschluß daran werden die erzielten Ergebnisse präsentiert und interpretiert.

6.1 Zusammenstellung der Testdaten

Wie bereits in Kapitel 1 erwähnt, bildet die Syntax natürlicher Sprachen ein äußerst komplexes System, welches in seiner Vollständigkeit kaum im Rahmen einer Magisterarbeit abgedeckt werden kann. Zwar wurden die in Abschnitt 2.4.1 beschriebenen Strukturen nicht nur implementiert, sondern auch um eine Reihe der von den Unterkapiteln 5.10 bis 5.12 behandelten syntaktischen Strukturen erweitert, allerdings musste dabei aufgrund des begrenzten Rahmens dieser Arbeit eine große Anzahl an Phänomenen außer Acht gelassen werden. Die Integration von Interrogativ- oder Deklarativkonstruktionen beispielsweise hätte jeweils für sich allein genommen vermutlich den Umfang des vorliegenden Projekts verdoppelt. Darüber hinaus ist die Vollständigkeit der Morphologiekomponente, wie bereits in Unterkapitel 4.1 angemerkt, als fragwürdig einzustufen, da bedeutende morphologische Phänomene, wie beispielsweise die Derivation, sowie eine große Anzahl an Wortformen außer Acht gelassen wurden.

Aus diesen Gründen wird bewusst darauf verzichtet, das vorliegende JSLIM-Projekt anhand realer Daten zu evaluieren. Stattdessen wurden ausgiebige Testlisten angelegt, die speziell die von der implementierten Syntaxgrammatik berücksichtigten Konstruktionen überprüfen. Im Unterordner `test` der beiliegenden CD findet sich diese Zusammenstellung an Testdaten, gegliedert nach positiven und negativen Eingaben, in jeweils 15 Dateien, wobei jede der beiden Kategorien insgesamt mehr als 200 Einträge umfasst. Die genaue Zuordnung der einzelnen Test-

listen zum jeweilig abgefragten syntaktischen Phänomen kann in der Datei `contents` im selben Ordner eingesehen werden.

Positive Testlisten enthalten dabei solche Einträge, die das System als wohlgeformte Eingabe erkennen und darüber hinaus korrekt analysieren soll. Sie dienen dem System somit als eine Art Trainingstext. Negative Testlisten enthalten Einträge, die bewusst fehlerhaft konstruiert wurden, und dienen damit dem Test auf Übergenerierung einer Grammatik.

Der Unterordner `test` enthält darüber hinaus ein Makefile, welches zur Pflege der Testlisten angelegt wurde. Zwei Funktionen dieses Makefiles sind an dieser Stelle von Interesse. Zum Ersten konkateniert der Befehl `make merge-all` alle positiven sowie alle negativen Testlisten und schreibt sämtliche Einträge in jeweils eine Datei pro Kategorie, benannt mit `syn00.pos` bzw. `syn00.neg`. Zum Zweiten kann der Befehl `make clean` dazu genutzt werden, alle von JSLIM beim Einlesen von Testlisten automatisch generierten Dateien zu löschen, und hilft dem Autor einer Grammatik somit, die Übersicht zu wahren.

6.2 Erzielte Resultate

Die Abbildung 6.1 zeigt das vom System gelieferte, zusammenfassende Analyseergebnis, das beim Einlesen der oben erwähnten Testliste `syn00.pos` erzeugt wird.

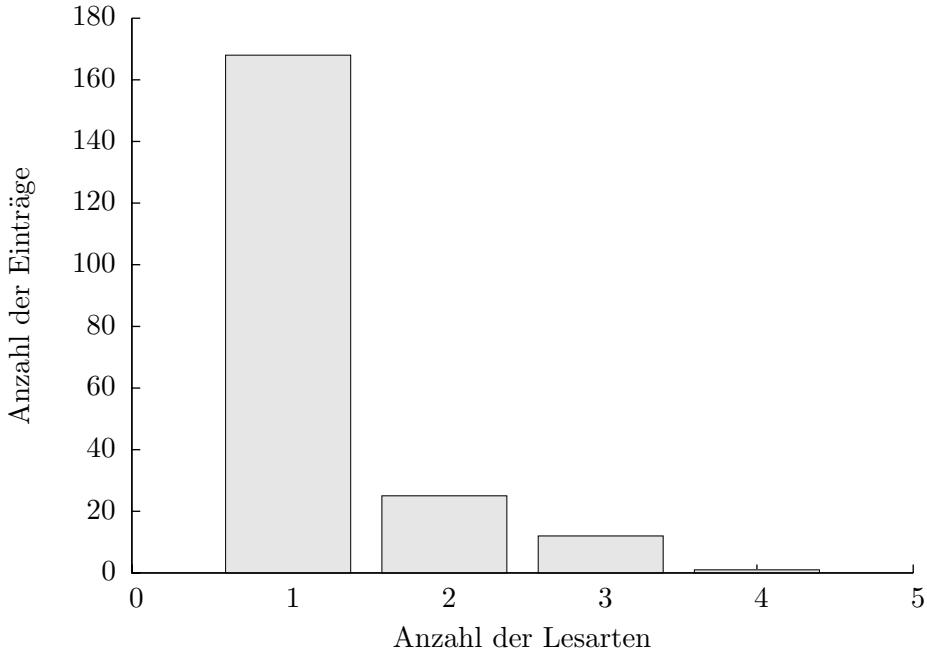
Abbildung 6.1: Zusammenfassendes Analyseergebnis der positiven Testlisten

```
* Wellformed confirmed forms

+------------+------------+------------+
| Readings   | results    | %-results  |
+------------+------------+------------+
| 1          | 168        | 81.55      |
| 2          | 25         | 12.14      |
| 3          | 12         | 5.83       |
| 4          | 1          | 0.49       |
+------------+------------+------------+

* Summary

+--------------+------------+------------+--------------+
| type         | results    | %-results  | %-ambiguity  |
+--------------+------------+------------+--------------+
| confirmed    | 206        | 100.00     | 125.24       |
| complete     | 206        | 100.00     | 125.24       |
+--------------+------------+------------+--------------+
```

In den Spalten `results` bzw. `%-results` des Abschnitts `Summary` gibt die Ausgabe an, dass alle 206 der insgesamt 206 Einträge, und damit 100% der zu analysierenden Eingaben, zu einem gültigen Endzustand

geführt haben. Nachdem das JSLIM-Projekt allerdings mit dem Ziel entwickelt wurde, gerade diese Sätze korrekt zu analysieren, verwundert dieses Ergebnis nicht weiter. Einen differenzierteren Blick auf die Zahlen liefert die Abbildung 6.2, welche die im Abschnitt `Wellformed comfirmed forms` der Abbildung 6.1 gelieferte Information bezüglich der Anzahl der Lesarten visualisiert.

Abbildung 6.2: Ambiguitätsverteilung der Analyseergebnisse der positiven Testlisten



Diese Daten zeigen, dass insgesamt 168 Eingabesätze, und damit 81,55%, nicht nur korrekt analysiert wurden, sondern darüber hinaus keine Ambiguität bezüglich mehrfacher Lesarten aufweisen.

Die mit 25 Eingabesätzen bzw. 12,14% zweitgrößte Gruppe bildet sich aus Testlisteinträgen mit zwei möglichen Lesarten. Bei einem Teil dieser Eingaben sind Doppelanalysen ausdrücklich erwünscht, da sie tatsächliche syntaktische Ambiguitäten aufweisen. Ein Beispiel einer solchen Eingabe ist der Satz „Women read the book.“, bei dem der Oberfläche *read* von der Morphologiekomponente zwei Proplets zugeordnet werden. Eines drückt die Tempusform *present* aus, ein zweites die Tempusform *past*. Da beide korrekt mit einem Subjekt im Plural kombiniert werden können, kommt es zur Ambiguität.

Bei den Testlisteneinträgen mit drei oder vier Lesarten hingegen handelt es sich größtenteils um unerwünschte Mehrfachanalysen, die vermutlich auf zu schwach definierte Variablenbeschränkungen zurückzuführen sind. Diese Übergenerierung ist zwar unwerwünscht, betrifft in seiner Gesamtheit aber nur 6,32% der Eingaben. Speziell die extreme Ausprägung von vier möglichen Lesarten pro Eingabesatz ist mit nur 0,49% in der Gesamtanalyse vertreten und stellt damit eine vernachlässigbare Größe dar.

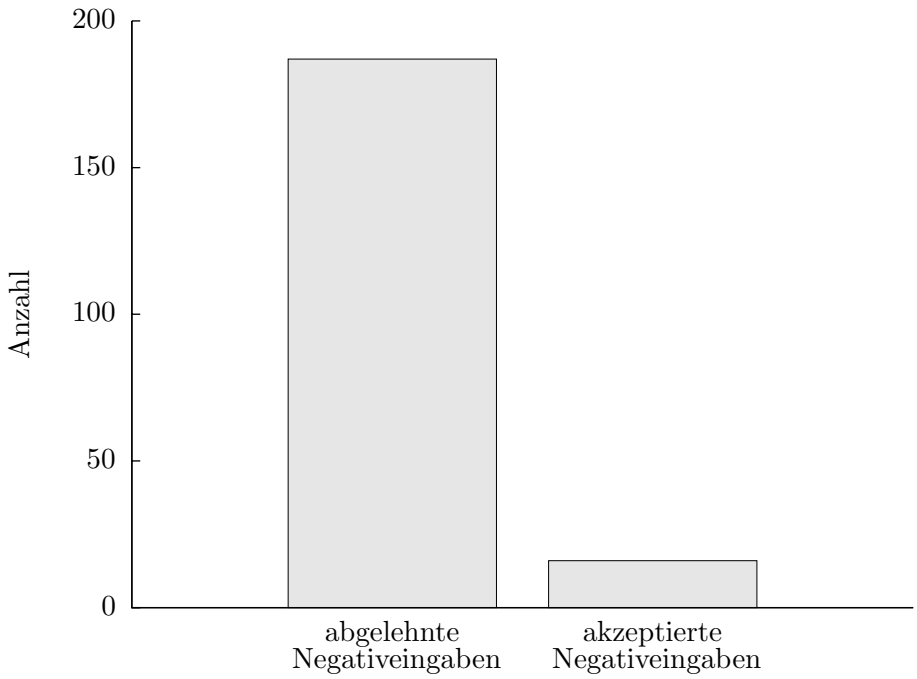
Ein Ausschnitt des zusammenfassenden Analyseergebnisses der negativen Gesamttestliste `syn00.neg` wird in Abbildung 6.3 wiedergegeben und in Abbildung 6.4 veranschaulicht. Da `syn00.neg` solche Eingaben enthält, die vom System nicht als korrekt akzeptiert werden sollen, sind mögliche Mehrfachlesarten bzw. die Ambiguitätsrate an dieser Stelle nicht von Interesse.

Die Daten zeigen, dass das System trotz absichtlich inkorrekter Formulierungen der Testlisteneinträge 7.88% der Eingaben akzeptiert, was vermutlich auf nicht restriktiv genug gestaltete Eingabemuster der Syntaxregeln zurückzuführen ist. Die Tatsache allerdings, dass 92,12% der Testlisteneinträge nicht vom System akzeptiert werden, entspricht dem Sinn dieser Form der Evaluierung und ist als positiv zu bewerten.

Abbildung 6.3: Ausschnitt des zusammenfassenden Analyseergebnisses der negativen Testlisten

```
* Summary

+--------------+------------+------------+--------------+
| type         | results    | %-results  | %-ambiguity  |
+--------------+------------+------------+--------------+
| confirmed    | 16         | 7.88       | 112.50       |
| complete     | 16         | 7.88       | 8.87         |
| rejected     | 187        | 92.12      | 100.00       |
+--------------+------------+------------+--------------+
```

Abbildung 6.4: Anzahl abgelehnter bzw. akzeptierter Einträge der negativen Testlisten



7 Fazit

Im Rahmen der vorliegenden Arbeit wurden die elementaren Strukturen der englischen Syntax mit dem Grammatikentwicklungssystem JSLIM implementiert. Das Projekt umfasst dabei nicht nur die *major clause types*, sondern darüber hinaus noch eine Reihe komplexer syntaktischer Phänomene wie Passiv- oder *Gapping*-Konstruktionen. Auf diese Weise wurde nachgewiesen, dass es sich bei dem Formalismus der Linksassoziativen Grammatik um ein mächtiges Werkzeug zur Analyse natürlicher Sprache handelt, der in der Lage ist, auch komplizierte Strukturen des Englischen zu verarbeiten.

Außerdem versteht sich das vorliegende Projekt als ausbaufähiges Grundgerüst, das dazu geeignet ist, durch aufbauende Arbeiten die englische Sprache in ihrer Gesamtheit zu erfassen. Damit ist das Fernziel, natürliche Sprachen eines Tages vollautomatisch verarbeiten zu können und damit die natürlichsprachliche Kommunikation mit kognitiven Agenten zu ermöglichen, einen Schritt näher gerückt.

Literaturverzeichnis

[Bauer 2011] Bauer, Marc: *Morphologische Analyse des Englischen*, Friedrich-Alexander-Universität Erlangen-Nürnberg, Magisterarbeit, 2011

[Bußmann 2008] Bussmann, Hadumod (Hrsg.): *Lexikon der Sprachwissenschaft.* 4. Auflage. Stuttgart : Alfred Kröner, 2008

[Crystal 1980] Crystal, David: *A First Dictionary of Linguistics and Phonetics.* Cambridge : Cambridge University Press, 1980

[Crystal 1988] Crystal, David: *The English Language.* London : Penguin Books, 1988

[Crystal 1995] Crystal, David: *The Cambridge encyclopedia of the English language.* Cambridge : Cambridge University Press, 1995

[Greiner und Handl 2010] Greiner, Paul ; Handl, Johannes: „A Prototypical Syntax in JSLIM“. 2010. – in: Weber u. a. *CLUE-Arbeitsberichte Nummer 10 – JSLIM 2.1-Dokumentation: Morphologie, Syntax und formale Sprachen / The JSLIM 2.1 Documentation: Morphology, Syntax and Formal Languages*, Hrsg.: Roland Hausser. Abteilung für Computerlinguistik, Friedrich-Alexander-Universität Erlangen-Nürnberg

[Handl voraussichtlich 2012] Handl, Johannes: *Inkrementelle oberflächenkompositionale Analyse und Generierung von natürlicher Sprache.* Erlangen, Friedrich-Alexander-Universität Erlangen-Nürnberg, Dissertation, voraussichtlich 2012

[Handl u. a. 2009] Handl, Johannes ; Kabashi, Besim ; Proisl, Thomas ; Weber, Carsten: „JSLIM – Computational morphology in the frameworks of the SLIM theory of language“. 2009. – in: *The State of the Art in Computational Morphology. Workshop on Systems and Frameworks for Computational Morphology*, Hrsg.: Cerstin Mahlow; Michael Piotrowski. Institute of Computational Linguistics, University of Zürich. Berlin, Heidelberg, New York: Springer.

[Hausser 1985] Hausser, Roland: *NEWCAT: Parsing Natural Language Using Left-Associate Grammar*. Berlin, Heidelberg, New York : Springer, 1985

[Hausser 2000] Hausser, Roland: *Grundlagen der Computerlinguistik*. Berlin, Heidelberg, New York : Springer, 2000

[Hausser 2006] Hausser, Roland: *A Computational Model of Natural Language Communication*. Berlin, Heidelberg, New York : Springer, 2006

[Herbst 2010] Herbst, Thomas: *English Linguistics – A coursebook for students of English*. Berlin, New York : de Gruyter, 2010

[Herbst und Schüller 2008] Herbst, Thomas ; Schüller, Susen: *Introduction to Syntactic Analysis – A Valency Approach*. Tübingen : Narr, 2008

[Herbst u. a. 1990] Herbst, Thomas ; Stoll, Rita ; Westermayr, Rudolf: *Terminologie der Sprachbeschreibung – Ein Lernwörterbuch für das Anglistikstudium*. Ismaning : Hueber, 1990

[Hogg und David 2006] Hogg, Richard ; David, Denison: *A History of the English Language*. New York : Cambridge University Press, 2006

[Kosche 2011] Kosche, Sylvia: *Ein Syntaxparser zur Analyse französischer Gesetzestexte*, Friedrich-Alexander-Universität Erlangen-Nürnberg, Magisterarbeit, 2011

[Kycia 2004] Kycia, Arkadius: *Implementierung der Datenbanksemantik für die natürlichsprachliche Kommunikation*, Friedrich-Alexander-Universität Erlangen-Nürnberg, Magisterarbeit, 2004

[Lipp 2010] Lipp, Susanne: *Automatische Wortformerkennung für das Schwedische*, Friedrich-Alexander-Universität Erlangen-Nürnberg, Magisterarbeit, 2010

[Mehlhaff 2007] Mehlhaff, Judith: *Funktor-Argument-Struktur und Koordination im Deutschen – eine Implementation in JSLIM*, Friedrich-Alexander-Universität Erlangen-Nürnberg, Magisterarbeit, 2007

[Niedobijczuk 2009] Niedobijczuk, Kamila: *Implementierung eines automatischen Wortformerkennungssystems des Polnischen in JSLIM*, Friedrich-Alexander-Universität Erlangen-Nürnberg, Magisterarbeit, 2009

[Pandea 2010] Pandea, Paul: *Implementierung eines automatischen Wortformerkennungssystems im Rumänischen mit JSLIM*, Friedrich-Alexander-Universität Erlangen-Nürnberg, Magisterarbeit, 2010

[Pepiuk 2009] Pepiuk, Stefan: *Implementierung eines automatischen Wortformerkennungssystems für das Französische in JSLIM*, Friedrich-Alexander-Universität Erlangen-Nürnberg, Magisterarbeit, 2009

[Quirk u. a. 1990] Quirk, Randolph ; Greenbaum, Sidney ; Leech, Geoffrey ; Svartvik, Jan: *A comprehensive grammar of the English Language*. 8. Auflage. New York : Longman Inc., 1990

[Stadlbauer 2011] Stadlbauer, Vanessa: *Extraktion von Valenzdaten aus dem BNC für eine Verwendung in natürlichsprachlichen Grammatiken*, Friedrich-Alexander-Universität Erlangen-Nürnberg, Magisterarbeit, 2011

[Wall u. a. 2003] Wall, Larry ; Christiansen, Tom ; Orwant, Jon: *Programmieren mit Perl*. zweite Auflage. Köln : O'Reilly, 2003

[Weber 2007] Weber, Carsten: *Implementierung eines automatischen Wortformerkennungssystems für das Italienische mit dem Programm JSLIM*, Friedrich-Alexander-Universität Erlangen-Nürnberg, Magisterarbeit, 2007

[Weber 2011] Weber, Carsten: *Syntaktisch-semantisches Parsing des Italienischen - Implementierung unter Berücksichtigung von Korpusdaten*. Frankfurt am Main : Peter Lang Verlag - Im Druck, 2011

[ben Zineb 2009] Zineb, Ramzi ben: *Implementierung eines automatischen Wortformerkennungssystems für das Arabische in JSLIM*, Friedrich-Alexander-Universität Erlangen-Nürnberg, Magisterarbeit, 2009

Anhang A
Das Makefile zum Programmstart

```
SYSTEM=../../jslim2.1
ALLO=$(SYSTEM)/bin/allo
JSLIM=$(SYSTEM)/bin/jslim

COMBI_PRO=common/english-syn.pro

ALLO_VERB=english-allo-verbs
ALLO_NOUN=english-allo-nouns
ALLO_MISC=english-allo-misc
ALLO_ADJV=english-allo-adjv

LEX_SUFFIX=lex
ALL_SUFFIX=all

.PHONY: help allo all combi clean syn

help:
        @echo
        @echo "    allo: generate allomorph lexicon"
        @echo "  syntax: start syntax"
        @echo "     all: generate allomorph lexicon and start syntax"
        @echo "   clean: delete files ending in '.all', '.err',"
        @echo "          '.log', '.out', '.lst' or '~'"
        @echo

syn:
        @$(JSLIM) $(COMBI_PRO)

allo: verb noun adjv misc

verb: lex/$(ALLO_VERB).$(LEX_SUFFIX)
        $(ALLO) -G $(COMBI_PRO) -I lex/$(ALLO_VERB).$(LEX_SUFFIX) -O allo/$(ALLO_VERB).$(ALL_SUFFIX)

noun: lex/$(ALLO_NOUN).$(LEX_SUFFIX)
        $(ALLO) -G $(COMBI_PRO) -I lex/$(ALLO_NOUN).$(LEX_SUFFIX) -O allo/$(ALLO_NOUN).$(ALL_SUFFIX)

adjv: lex/$(ALLO_ADJV).$(LEX_SUFFIX)
        $(ALLO) -G $(COMBI_PRO) -I lex/$(ALLO_ADJV).$(LEX_SUFFIX) -O allo/$(ALLO_ADJV).$(ALL_SUFFIX)

misc: lex/$(ALLO_MISC).$(LEX_SUFFIX)
        $(ALLO) -G $(COMBI_PRO) -I lex/$(ALLO_MISC).$(LEX_SUFFIX) -O allo/$(ALLO_MISC).$(ALL_SUFFIX)

all: allo syn

clean:
        @find . -name "*all" | xargs rm -f
        @find . -name "*err" | xargs rm -f
        @find . -name "*log" | xargs rm -f
        @find . -name "*out" | xargs rm -f
        @find . -name "*lst" | xargs rm -f
        @find . -name "*~" | xargs rm -f
```

Anhang B
Implementierte Regeln

Regel	Klauseln	Beispielsatz
WHEN+MAIN		[When + Fido] barked Mary smiled .
		[While + Fido] barked Mary smiled .
MAIN+WHICH		The [dog + which] saw Mary quickly barked .
WHICH+X	ADV	The dog which saw [Mary + quickly] barked .
	V2	The dog which saw Mary [quickly + barked] .
	V1	The dog [which + saw] Mary quickly barked .
WHICH+MAIN	DET2	The dog which saw [the + car] quickly barked .
	NM	The dog which [saw + Mary] quickly barked .
	DET1	The dog which [saw + the] car quickly barked .
FV+TH		[heard + that] Fido barked .
TH+N	OBJ	John heard [that + Fido] barked .
	SUB	[That + Fido] barked amused Mary .
TH+V	OBJ	John heard that [Fido + barked] .
	SUB	That [Fido + barked] amused Mary .
OBJ_GAP	N-TH_GAP	Bob bought , Jim peeled [, + Bill] ate , and ...
	FIRST_GAP	Bob bought [, + Jim] peeled and Bill ate the peach .
SUB_GAP	CNJ+V	Bob ate an apple , + walked his dog [and + read] a paper .
	N+CNJ	Bob ate an apple , walked his [dog + and] read a paper .
	COM+V	Bob ate an apple [, + walked] his dog and read a paper .
X+PREP	PASS	The book was [read + by] the man .
	OBJ	Julia ate the [apple + on] the table .
	STAND	Dogs [sleep + on] the floor .
PREP+N	OBJ	Julia ate the apple [on + the] table .
	OBJ	The dog ate [on + the] floor .
	STAND	Dogs sleep [on + the] floor .
CVP2	PROG	The man [has + been] sleeping .
	FUT	The woman [will + be] sleeping .
CVP1	FUT	The man [will + sleep] .
		The man [would + sleep] .
	PERF	The man [has + heard] .
	PROG	The man [is + giving] .
ADV+COMMA		The dog barks [angrily + ,] loudly and noisily .
COMMA+ADV		The dog barks angrily [, + loudly] and noisily .

Regel	Klauseln	Beispielsatz
ADV+CNJ		The dog barks angrily , [loudly + and] noisily .
CNJ+ADV		The dog barks angrily , loudly [and + noisily] .
N+COMMA		The [man + ,] the woman , and the child slept .
ADJ+COMMA		The [small + ,] ugly and nice man sleeps .
V+COMMA		The dog [sleeps + ,] dreams and snores .
COMMA+V	STAND	John bought [, + cooked] , and ate the pizza .
	NTH	John bought , cooked , ate [, + and] enjoyed the pizza .
COMMA+N	STAND	The man [, + the] woman , and the child slept .
COMMA+ADJ		The dirty [, + little] , black dog barks .
ADJ+CNJ		The dirty , [old + and] big dog sleeps .
CNJ+ADJ		The woman reads the big , red [and + thin] book .
V+CNJ	OGAP	Bob bought , Jim peeled , and Bill ate the peach .
	VCO	The dog sleeps , [dreams + and] snores .
CNJ+V	STAND	The dog sleeps , dreams [and + snores] .
	TWOCO	The dog sleeps and dreams .
N+CNJ	VGAP	Bob ate an apple , Jim a [pear + and] Bill a peach .
	STAND	The man , the [woman + and] the child sleep .
CNJ+N	STAND	The man , the woman [and + the] child sleep .
	TWOCO	The man [and + the] woman sleep
VERB+ADV		The dog [barks + loudly] .
MAIN+ADV		The [dog + quickly] barked .
ADV+FV		The dog [quickly + barked] .
ADJ+N	STAND	The small woman reads [red + books] .
	FIRST	[little + men] sleep .
DET+ADJ		[The + little] dog sleeps .
DET+N	SGAP	Bob ate an apple , walked [the + dog] and read a paper .
	PREP	Julia ate the apple [on + the] table .
	AN	[An + eagle] flies .
		[The + eagle] flies .
	A	[A + man] sleeps .
	STAND	[The + man] sleeps .
		[The + eagle] flies .
MAIN+HVBE		the [man + has] the apple .
MAIN+FV	O_GAP2	Bob bought , Jim peeled , [Bill + ate] , and John ...
	O_GAP1	Bob bought , [Jim + peeled] , and Bill ate the peach .
	V_GAP2	Bob ate an apple , Jim a pear , and [Bill + a] peach .
	V_GAP1	Bob ate an apple , [Jim + a] pear , and Bill a peach .
	CNJ	The dog which saw Mary [quickly + barked] .
	WH	That the [dog + barked] amused Mary .
	STAND	The [man + sleeps] .

Regel	Klauseln	Beispielsatz
MAIN+ADJ		[The + nice] man sleeps .
FV+ADJ		the woman [gives + big] men red book .
FV+MAIN	POSS	John [walks + his] dog .
	PASS	the book was read [by + women] .
	VCO	John bought , cooked , and ate the pizza .
	STAND	the man [likes + books] .
S+COMP	SC	the car [is + green] .
	OC	the man got his shoes wet .
S+IP	SGAP1	Bob ate an [apple + ,] walked his dog and read a paper .
	CNJ2	When Fido barked Mary [smiled + .]
	CNJ1	When Fido [barked + Mary] smiled .
	STAND	The man [sleeps + .]

Lebenslauf

Persönliche Daten

Name	Paul Greiner
Anschrift	Kosbacher Weg 1a
	91056 Erlangen
Telefon	0176 96398973
E-Mail	paulgreiner@gmx.de
Geburtsdatum	31.12.1983
Geburtsort	Berkeley (USA)

Schulbildung

1990 – 1992	Silcherschule (Grundschule) in Tübingen
1992 – 1994	Volksschule (Grundschule) Röttenbach
1994 – 2003	Albert-Schweitzer-Gymnasium Erlangen,
	Abschluss: Allgemeine Hochschulreife

Studium

WS 2004 – SS 2005	Grundschullehramt, Universität Bayreuth
SS 2006 bis heute	Linguistische Informatik und Anglistik: Linguistik,
	Friedrich-Alexander-Universität Erlangen-Nürnberg

Berufserfahrung

2003 – 2004	Zivildienst am Universitätklinikum Erlangen
2005 – 2006	sechsmonatiges Praktikum bei
	Gröpper | Konzept- Grafikdesign
2006 – 2008	Nachtportier im Kreativhotel Luise in Erlangen
2008 bis heute	Studentische Hilfskraft an der Abteilung für
	Computerlinguistik der Friedrich-Alexander-Universität
	Erlangen-Nürnberg

Wahrheitsgemäße Erklärung

Ich erkläre hiermit, dass ich

- die eingereichte Arbeit selbstständig und ohne unerlaubte Hilfe angefertigt habe,
- außer den im Schrifttumsverzeichnis angegebenen Quellen und Hilfsmitteln keine weiteren benutzt und alle Stellen, die aus dem Schrifttum ganz oder annähernd entnommen sind, als solche kenntlich gemacht und einzeln nach ihrer Herkunft unter Bezeichnung der Ausgabe (Auflage und Jahr des Erscheinens), des Bandes und der Seite des benutzten Werkes in der Magisterarbeit nachgewiesen habe,
- alle Stellen und Personen, welche mich bei der Vorbereitung und Anfertigung der Arbeit unterstützten, genannt habe,
- die Magisterarbeit noch keiner anderen Stelle zur Prüfung vorgelegt habe.

Erlangen, den 18. Mai 2017

.....
(Paul Greiner)