Universität des Saarlandes
Fachbereich 14
WS 1997/98

Proseminararbeit

Übersetzung von HPSG nach TAG

Jörg Steffen

2. Februar 1998

im Rahmen des Proseminars
"Effizientes Parsing und Generieren
mit Baumadjunktionsgrammatiken"
Wolfgang Wahlster und Tilman Becker

Inhaltsverzeichnis

Abbildungsverzeichnis

1 HPSG – ein Überblick

Bevor ich mit der Besprechung des Papiers über die Übersetzung von HPSG nach TAG beginne, möchte ich zum besseren Verständnis kurz die Hauptmerkmale von Head Driven Phrase Structure Grammars (HPSG) beschreiben. Das Folgende ist ein Crash Kurs in HPSG und weit davon entfernt, sämtliche Feinheiten von HPSG zu erklären.

HPSG ist ein auf Feature–Strukturen basierender Grammatikformalismus. Linguistische Sachverhalte werden dabei durch modulare Generalisierungen dargestellt: traditionelle Grammatikregeln werden generalisiert zu Schemata, die eine abstrakte Definition für grammatische Relationen liefern. Bei HPSG befindet sich die meiste grammatische Information im Lexikon. Die Schemata steuern den Fluß von grammatischer Information aus dem Lexikon und in der globalen Struktur. Im Anschluß werde ich die wichtigsten Schemata sowie einige andere Begriffe kurz erläutern.

1.1 Kopf einer Phrase

Jede Phrase enthält ein Wort, das für die Phrase die meisten syntaktischen Eigenschaften festlegt. Dieses Wort nennt man den *lexikalischen Kopf* der Phrase. So ist z.B. das Verb der lexikalische Kopf einer Verbphrase und die Präposition der lexikalische Kopf einer Präpositionalphrase.

Der *Kopf einer Phrase* ist die Tochter, die entweder selbst der lexikalische Kopf ist, oder die, die eine Phrase mit dem lexikalischen Kopf enthält. Dieser Tochterknoten wird *Head–Daughter* genannt. In dem Satz

Hans trifft Peter.

ist *trifft Peter* der Kopf der Phrase, weil er den lexikalischen Kopf *trifft* enthält.

Die relevanten Informationen (die sog. *Head–Werte*), die der (lexikalische) Kopf einer Phrase enthält, müssen auch beim Mutterknoten verfügbar sind. Deshalb wird diese Information immer nach oben weitergereicht.

1.2 Subkategorisierung

Die Lexikoneinträge und Phrasen, mit denen man bei HPSG arbeitet, werden allgemein *Zeichen* genannt. Dementsprechend unterscheidet man zwischen phrasalen Zeichen und lexikalischen Zeichen.

Die Subkategorisierung eines Zeichens gibt an, mit welchen anderen Zeichen dieses Zeichen kombiniert werden muß, damit es *gesättigt* ist. Bei den Zeichen, die subkategorisiert werden können, unterscheidet man zwischen Subjekt und Komplementen. Ein ditransitives Verb kann z.B. eine Subjekt NP, eine Objekt NP und eine Objekt PP subkategorisieren.

$$\begin{bmatrix} \texttt{SYNSEM} & \begin{bmatrix} \texttt{NON-LOC} & \begin{bmatrix} \texttt{SLASH}\ \langle\ \rangle \end{bmatrix} \\ \texttt{LOC|CAT} & \begin{bmatrix} \texttt{SUBJ} & \left\langle np[\ ] \right\rangle \\ \texttt{COMPS} & \left\langle np[\ ]\ pp[\ ] \right\rangle \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Abbildung 1: Lexikoneintrag für ein ditransitives Verb

1.3 Komplementierung

Die beiden Regelschemata in Abb. 2 und 3 sorgen dafür, daß ein Zeichen mit einem anderen Zeichen kombiniert werden kann, wenn es dieses subkategorisiert.

$$\begin{bmatrix} \texttt{SYNSEM} & \begin{bmatrix} \texttt{LOC|CAT} & \begin{bmatrix} \texttt{HEAD} & \boxed{1} \\ \texttt{SUBJ} & \langle\ \rangle \\ \texttt{COMPS} & \boxed{3} \end{bmatrix} \end{bmatrix} \\ \texttt{DTRS} & \begin{bmatrix} \texttt{HEAD-DTR} & \begin{bmatrix} \texttt{SYNSEM|LOC|CAT} & \begin{bmatrix} \texttt{HEAD} & \boxed{1} \\ \texttt{SUBJ} & \left\langle \boxed{2} \right\rangle \\ \texttt{COMPS} & \boxed{3} \end{bmatrix} \end{bmatrix} \\ \texttt{COMP-DTR} & \begin{bmatrix} \texttt{SYNSEM} & \boxed{2} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Abbildung 2: Head–Subject–Schema

Beim Head–Subject–Schema subkategorisiert die Head–Daughter noch ein Subjekt $\boxed{2}$ und ein Komplement $\boxed{3}$. Die Complement–Daughter ist das subkategorisierte Subjekt. Die beiden Tochterknoten werden zu einem neuen Knoten kombiniert, der nur noch ein Komplement subkategorisiert, aber kein Subjekt mehr.

Beim Head–Complement–Schema subkategorisiert die Head–Daughter noch ein Subjekt $\boxed{2}$ und zwei Komplemente $\boxed{4}$ und $\boxed{3}$. Die Complement–Daughter ist eine der subkategorisierten Komplemente. Die beiden Knoten werden wieder zu einem neuen Knoten kombiniert, der jetzt aber nur noch ein Subjekt und ein Komplement subkategorisiert.

An diesen beiden Schemata ist auch zu sehen, daß die Head–Werte der Head–Daughter zu den Head–Werten der Mutter hochgereicht werden.

$$\begin{bmatrix} \texttt{SYNSEM} & \begin{bmatrix} \texttt{LOC|CAT} & \begin{bmatrix} \texttt{HEAD} & \boxed{1} \\ \texttt{SUBJ} & \boxed{2} \\ \texttt{COMPS} & \boxed{3} \end{bmatrix} \end{bmatrix} \\ \texttt{DTRS} & \begin{bmatrix} \texttt{HEAD-DTR} & \begin{bmatrix} \texttt{SYNSEM|LOC|CAT} & \begin{bmatrix} \texttt{HEAD} & \boxed{1} \\ \texttt{SUBJ} & \boxed{2} \\ \texttt{COMPS} & \text{union}(\boxed{4},\boxed{3}) \end{bmatrix} \end{bmatrix} \\ \texttt{COMP-DTR} & \begin{bmatrix} \texttt{SYNSEM} & \boxed{4} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Abbildung 3: Head–Complement–Schema

1.4 Modifikation

Die Modifikation eines Zeichens mit einem Adjunkt wird durch das Schema in Abb. 4 ermöglicht.

$$\begin{bmatrix} \texttt{SYNSEM} & \begin{bmatrix} \texttt{LOC} & \begin{bmatrix} \texttt{HEAD} & \boxed{1} \\ \texttt{SUBJ} & \boxed{2} \\ \texttt{COMPS} & \boxed{3} \end{bmatrix} \end{bmatrix} \\ \texttt{DTRS} & \begin{bmatrix} \texttt{HEAD-DTR|SYNSEM} \ \boxed{4} & \begin{bmatrix} \texttt{LOC} & \begin{bmatrix} \texttt{HEAD} & \boxed{1} \\ \texttt{SUBJ} & \boxed{2} \\ \texttt{COMPS} & \boxed{3} \end{bmatrix} \end{bmatrix} \\ \texttt{ADJ-DTR|SYNSEM} & \begin{bmatrix} \texttt{LOC|HEAD|MOD} \ \boxed{4} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Abbildung 4: Head–Adjunct–Schema

Die Adjunct–Daughter gibt an, was sie modifiziert, nämlich $\boxed{4}$. Das ist genau die Head–Daughter. Die lokalen Informationen (unter dem Feature LOC) des neu gebildeten Knotens sind die gleichen wie bei der Head–Daughter.

1.5 Nicht–lokale Abhängigkeiten

Nicht–lokale Abhängigkeiten treten dann auf, wenn subkategorisierte Zeichen nicht in der Domäne des lexikalischen Kopfs sind, sondern außerhalb. Das ist z.B. der Fall bei Topikalisierung von Komplementen wie in dem Satz

To the first test tube, Peter adds water.

Die PP *to the first test tube* wird von *add* subkategorisiert, steht aber außerhalb der Domäne von *add*.

Nicht–lokale Abhängigkeiten werden in HPSG mit Hilfe des *Slash*–Wertes ausgedrückt, der im Gegensatz zu subkategorisierten Subjekten und Komplementen nicht unter LOC steht, sondern unter NON–LOC. Das Schema in Abb. 5 beschreibt die Abarbeitung von Slash–Werten.

$$\begin{bmatrix} \texttt{SYNSEM} & \begin{bmatrix} \texttt{LOC} & \begin{bmatrix} \texttt{CAT} & \begin{bmatrix} \texttt{HEAD} & \boxed{1} \\ \texttt{SUBJ} & \boxed{2}\langle\ \rangle \\ \texttt{COMPS} & \boxed{3}\langle\ \rangle \end{bmatrix} \end{bmatrix} \\ \texttt{NON-LOC} & \begin{bmatrix} \texttt{SLASH} & \langle\ \rangle \end{bmatrix} \end{bmatrix} \\ \texttt{DTRS} & \begin{bmatrix} \texttt{HEAD-DTR} & \begin{bmatrix} \texttt{SYNSEM} & \begin{bmatrix} \texttt{LOC|CAT} & \begin{bmatrix} \texttt{HEAD} & \boxed{1} \\ \texttt{SUBJ} & \boxed{2} \\ \texttt{COMPS} & \boxed{3} \end{bmatrix} \\ \texttt{NON-LOC} & \begin{bmatrix} \texttt{SLASH} & \left\langle \boxed{4} \right\rangle \end{bmatrix} \end{bmatrix} \end{bmatrix} \\ \texttt{FILLER-DTR} & \begin{bmatrix} \texttt{SYNSEM} & \boxed{4} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Abbildung 5: Head–Filler–Schema

Die Filler–Daughter $\boxed{4}$ entspricht dem Wert, der in der Head–Daughter unter Slash steht. Im neu entstandenen Knoten ist deshalb der Slash–Wert leer.

Wenn der Slash–Wert nicht gerade gesättigt wird durch das Head–Filler–Schema, dann wird er, wie die Head–Werte des Kopfes, nach oben zur Mutter weitergereicht.

Die lexikale Regel in Abb. 6 ermöglicht das Besetzen des Slash–Wertes für Lexikoneinträge, deren Slash–Wert defaultmässig leer ist.

Diese Regel entfernt ein Komplement aus der Subkategorisierungsliste, nämlich $\boxed{4}$, und fügt es in die Slash–Liste ein.

1.6 Equi und Raising Verben

Ein Equi Verb, auch Kontrollverb genannt, subkategorisiert ein nicht–finites verbales Komplement mit einem nicht–leeren SUBJ–Wert (Abb.7)

Ein Beispiel für ein Equi Verb ist *versuchen* in

Hans versucht zu fahren.

$$\begin{bmatrix} \texttt{SYNSEM} & \begin{bmatrix} \texttt{LOC} & \begin{bmatrix}\texttt{CAT} & \begin{bmatrix}\texttt{HEAD} & \boxed{1}\\ \texttt{SUBJ} & \boxed{2} \\ \texttt{COMPS} & \boxed{3}\end{bmatrix}\end{bmatrix} \\ \texttt{NON-LOC} & \begin{bmatrix}\texttt{SLASH} & \left\langle \boxed{4} \right\rangle\end{bmatrix}\end{bmatrix} \\ \texttt{DTRS} & \begin{bmatrix}\texttt{LEX-DTR} & \begin{bmatrix}\texttt{SYNSEM} & \begin{bmatrix}\texttt{LOC|CAT} & \begin{bmatrix}\texttt{HEAD} & \boxed{1}\\ \texttt{SUBJ} & \boxed{2}\\ \texttt{COMPS} & \mathit{union}(\boxed{4},\boxed{3})\end{bmatrix}\\ \texttt{NON-LOC} & \begin{bmatrix}\texttt{SLASH} & \langle\ \rangle\end{bmatrix}\end{bmatrix}\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

Abbildung 6: Slash–Termination–Regel

$$\begin{bmatrix} \texttt{SYNSEM|LOC|CAT|COMPS} & \left\langle \ldots, \begin{bmatrix} \texttt{SYNSEM|LOC|CAT} & \begin{bmatrix} \texttt{SUBJ} & \langle [\,] \rangle \\ \texttt{COMPS} & \langle\ \rangle \end{bmatrix} \end{bmatrix}, \ldots \right\rangle \end{bmatrix}$$

Abbildung 7: Equi Verb

Im Gegensatz zu einem Equi Verb subkategorisiert ein Raising Verb ein verbales Komplemente, von dem es subkategorisierte Komplemente übernehmen kann. Die Subkategorisierung wird also vom Komplement aus gesehen nach oben zur Mutter gereicht (*raising*).

$$\begin{bmatrix} \texttt{SYNSEM} & \begin{bmatrix} \texttt{LOC|CAT} & \begin{bmatrix} \texttt{SUBJ} & \boxed{1} \\ \texttt{COMPS} & \left\langle \begin{bmatrix} \texttt{SYNSEM|LOC|CAT} & \begin{bmatrix} \texttt{SUBJ} & \boxed{1} \\ \texttt{COMPS} & \boxed{2} \end{bmatrix} \end{bmatrix} \bullet \boxed{2} \right\rangle \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Abbildung 8: Raising Verb

Beispiele für ein Raising Verb ist *scheinen* in

Hans scheint zu fahren.

Das Subjekt, das von *fahren* subkategorisiert wird, wird auch von *scheinen* subkategorisiert. Raising ist nicht nur für Subjekte möglich, sondern auch für Komplemente.

Raising Verben und die meisten Equi Verben gehören zu den sog. *Bridge Verben.* Bridge Verben übernehmen ihren Slash–Wert von einem ihrer Komplemente.

$$\left[\texttt{SYNSEM}\left[\begin{array}{ll}\texttt{NON-LOC} & \left[\texttt{SLASH}\ \boxed{1}\right]\\ \texttt{LOC|CAT} & \left[\texttt{COMPS}\ \left\langle \ldots, \left[\texttt{SYNSEM|NON-LOC|SLASH}\ \boxed{1}\right], \ldots\right\rangle\right]\end{array}\right]\right]$$

Abbildung 9: Bridge Verb

Die Unterscheidung zwischen Equi Verben und Raising Verben ist ein sehr komplexer und viel untersuchter Bereich. Dabei spielen auch noch semantische Eigenschaften eine Rolle, die hier gar nicht berücksichtigt werden. Es gibt Verben, die sowohl als Equi wie auch als Raising Verben auftreten können. Für das weitere Verständnis dieser Ausarbeitung ist es nur notwendig, den syntaktischen Unterschied zwischen Equi und Raising Verben zu kennen.

2 Einführung

Head Driven Phrase Structure Grammar (HPSG) und Tree Adjoining Grammar (TAG) wurden lange Zeit nur getrennt betrachtet. In dieser Ausarbeitung stelle ich einen Algorithmus vor, der eine HPSG in eine lexikalisierte TAG übersetzt.

2.1 Unterschiede zwischen HPSG und TAG

Der größte Unterschied zwischen HPSG und TAG ist die unterschiedliche Verwendung des Begriffs "Lokalität". Lokalität in TAG wird realisiert durch die Domänen der Bäume. Jeder Baum beinhaltete alle Argumente, die von seinem lexikalischen Funktor abhängig sind. In HPSG entspricht der Begriff "Lokalität" eher einem linguistischen Konzept als einer formalen Notation. Dort werden Relationen als lokal betrachtet, die innerhalb der Domäne des lexikalischen Kopfs realisiert sind. Diese Domäne ist normalerweise die maximale Projektion des lexikalischen Kopfs. So ist z.B. die Filler–Gap Relation nicht lokal in HPSG, aber lokal in TAG.

2.2 Motivation für die Übersetzung von HPSG nach TAG

Da sich HPSG und TAG auf den ersten Blick radikal unterscheiden in der Organisation von grammatischen Relationen, fragt man sich, ob es überhaupt sinnvoll ist, eine Übersetzung von einem Formalismus in den anderen anzustreben. Es ist sinnvoll.

Von HPSG aus betrachtet bietet die Übersetzung die Möglichkeit, die Verarbeitungsgeschwindigkeit stark zu verbessern. Um z.B. in HPSG einen Satz mit einem transitiven Verb abzuleiten, müssen zwei Regelschemata angewandt werden, Head–Complement und Head–Subject. Das erste reduziert das Objekt, das zweite reduziert das Subjekt. Bei der Ableitung muß also erst berechnet werden, daß genau diese zwei Regelschemata (und keine anderen) anwendbar sind. Anschließend müssen die Informationen aus den Lexikoneinträgen des Objekts und des Subjekts hochprojeziert werden entsprechend den Beschränkungen durch die Regelschemata.

In einer lexikalisierten TAG wird mit dem Lexikoneintrag für das transitive Verb schon direkt eine Baumstruktur mitgeliefert, die der Kombination der Instantiierungen der beiden Regelschemata entspricht. Die Übersetzung von HPSG nach TAG führt also einen nicht unwesentlichen Anteil der Ableitung bereits während der Übersetzung durch, so daß die spätere, tatsächliche Ableitung wesentlich schneller wird.

Von der TAG Seite aus betrachtet liefert der Übersetzungsalgorithmus die Möglichkeit, eine alternative linguistische Theorie, nämlich HPSG, als Basis für eine TAG zu benutzen. Auf diese Weise wird die Verwendbarkeit von TAG untersucht, wenn es darum geht, andere linguistische Theorien darzustellen.

2.3 Vergleich der Mächtigkeit von HPSG und TAG

TAG gehört zu der Klasse der schwach kontext–sensitiven Grammatikformalismen. Mit HPSG ist es möglich, jede rekursiv aufzählbare Sprache zu beschreiben. Deshalb ist es nicht möglich, jede HPSG in eine TAG zu übersetzen. Allerdings kann man in HPSG Grammatiken Beschränkungen einführen, die ihre Mächtigkeit abschwächen auf das Niveau von schwach kontext–sensitiven Sprachen.

3 Der Algorithmus

3.1 Grundlagen

3.1.1 Selector–Features und Regelschemata

In TAG sind alle Argumente eines Funktors in einer Baumstruktur dargestellt, aber in HPSG ist die Anwendung des Funktors auf seine Argumente verteilt über die Regelschemata. Jedes Regelschema ist also eine partielle Beschreibung eines lokalen Baums. Deshalb muß man für jedes Regelschema feststellen, welche Konstituenten Funktoren sind und welche Argumente.

In TAG sind alle Funktor–Argument Relationen im gleichen Format dargestellt, nämlich als Zweige an dem Pfad vom lexikalen Anker des Baumes zu seiner Wurzel. Leider entspricht der lexikale Anker in TAG Bäumen nicht immer dem Kopf einer HPSG Phrase. So ist z.B. beim Head–Adjunct–Schema nicht die Head–Daughter der Funktor, sondern die Adjunct–Daughter.

Deshalb braucht man eine Verallgemeinerung für die verschiedenen Tochtersorten in HPSG. Um die Funktor–Argument Struktur zu berechnen, führt man nach [Kas92] eine *Selektor* Relation ein. Man hat eine *Selector–Daughter* (SD), die ein *Selector–Feature* (SF) enthält. Dieses Selector–Feature gibt an, wie die Argument–Daughter (auch Nicht–Selector–Daughter genannt) aussehen darf.

Betrachtet man z.B. das Head–Complement–Schema, so erkennt man leicht, daß die Head–Daughter die SD ist und COMPS das SF, denn die Elemente von COMPS geben an, welche Komplemente zulässige Argumente sind.

Die möglichen Selector–Features sind SUBJ, COMP, SLASH und MOD.

3.1.2 Reduktion

Zusammen mit der Selektion findet auch eine *Reduktion* statt. Sei z.B. F ein SF eines Regelschemas, das eine Liste enthält. Ein Element der Liste von F entspricht der Nicht–SD. Dieses Element fehlt in der Liste von F im Mutterknoten. F wird also durch das Schema reduziert.

Jedes der vorgestellten HPSG Schemata reduziert ein SF. Beim Head–Complement–Schema wird COMP reduziert. Beim Head–Subject–Schema ist ebenfalls die Head–Daughter die SD, das reduzierte SF ist SUBJ. Beim Head–Adjunct–Schema ist die Adjunct–Daughter die SD, weil sie mit dem SF MOD die Head–Daughter auswählt. Da MOD ein Head–Feature ist und der Mutterknoten alle Head–Features von der Head–Daughter übernimmt, taucht das MOD Feature der Adjunct–Daughter nicht mehr im Mutterknoten auf, es wird also reduziert. Beim Head–Filler–Schema ist wieder die Head–Daughter die SD und SLASH das SF, das die Filler–Daughter auswählt.

3.1.3 Das Grundschema des Algorithmus

Zur Initialisierung des Algorithmus nimmt man nicht die Lexikoneinträge von konkreten Wörtern, sondern lexikale Typen, die eine ganze Klasse von Wörtern subsumieren. So betrachtet man z.B. nicht einzelne Verben, sondern ganze Klassen, wie intransitive, transitive und ditransitve Verben.

Sei L nun ein lexikaler Typ, der eine solche Klasse repräsentiert. Für L wird ein Knoten des gleichen Typs angelegt. Zu diesem Knoten fügt man einen weiteren Knoten n hinzu, der L dominiert. Der Effekt dieser Dominanzrelation ist, daß n die SF von L bekommt. Im Moment gehen wir davon aus, daß n sämtliche SF von L bekommt, aber keine anderen Features. Diese Dominanzrelationen sind später auch die Punkte, an denen eine TAG Adjunktion möglich ist. Für jedes Regelschema S, das ein SF von n reduziert, baut man den Baum von S an n an, wobei n der SD von S entspricht. Anschließend fügt man noch einen weiteren Knoten m hinzu, der den Wurzelknoten von S dominiert.

Diesen Schritt wiederholt man nun so oft, bis keine weiter Reduzierung von SFs mehr möglich ist.

Die Bäume, die auf diese Weise aufgebaut werden, haben einen sog. *Trunk* vom lexikalen Anker zur Wurzel des Baumes. Alle Knoten, die Geschwister von Knoten sind, die auf dem Trunk liegen, sind Nicht–Selektor Knoten und werden nicht weiter untersucht. Sie dienen später als Fuß– oder Substitutionsknoten.

Folgende Details des Algorithmus werden im Anschluß genauer betrachtet:

- Was sind die Abbruchbedingungen des Algorithmus?
- Wie erkennt man Auxiliarbäume und findet deren Fußknoten?
- Welche Information wird über die Domination Links nach oben gereicht?

3.1.4 Beispiel

Zur Verdeutlichung des Algorithmus wird nun an einem Beispiel gezeigt, wie aus dem lexikalen Typen eines Verbes verschiedene TAG Bäume entstehen.

Der Baum in Abb. 10 ist durch zweifache Anwendung des Head–Complement–Schemas und einmaliger Anwendung des Head–Subject–Schemas entstanden. Man beachte, daß die Knoten über den Domination Links nicht expandiert sind. Sie enthalten aber nur die SF der Knoten, die sie dominieren, d.h. sie sind identisch mit ihnen. Man beachte ebenfalls, daß eine unterschiedliche Reihenfolge der Anwendung der Schemata unterschiedliche Bäume produziert. Wendet man die Schemata in der Reihenfolge Head–Complement, Head–Subject, Head–Complement an, dann erhält man den Baum in Abb. 11.

3.2 Terminierung

Das Ziel des Algorithmus ist es, auf einen gegebenen lexikalen Typ genau die Regelschemata anzuwenden, die alle seine Selektor Informationen reduzieren. Der Algorithmus muß also enden, wenn die Selector–Features irreduzierbare Werte haben. Vorerst nehmen wir an, daß alle Selector–Features über die Domination Links angehoben werden.

Normalerweise erwartet man, daß ein Selector–Feature mit einem spezifizierten Wert durch eine Reihe von Regelschemata zu einer leeren Liste reduziert wird. Es kann aber vorkommen, daß ein Selector–Feature einen unspezifizierten Wert hat. So ergibt z.B. die Anwendung des Head–Complement–Schema auf ein Raising Verb eine Wurzel mit unspezifizierten Werten für SUBJ, SLASH und COMPS (Abb. 12).

Die Anwendung des Head–Complement–Schemas und des Head–Subject–Schemas auf ein Equi Verb ergibt eine Wurzel mit unspezifiziertem Wert für SLASH (Abb. 13).

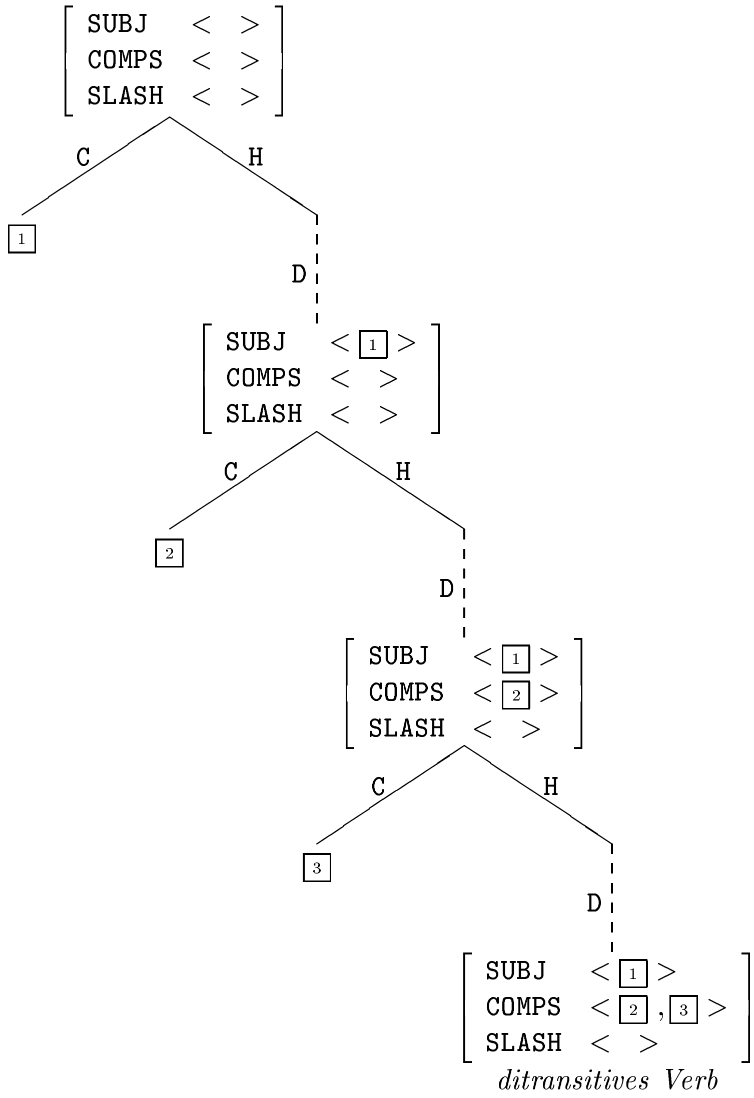

Abbildung 10: TAG Baum für ditransitives Verb

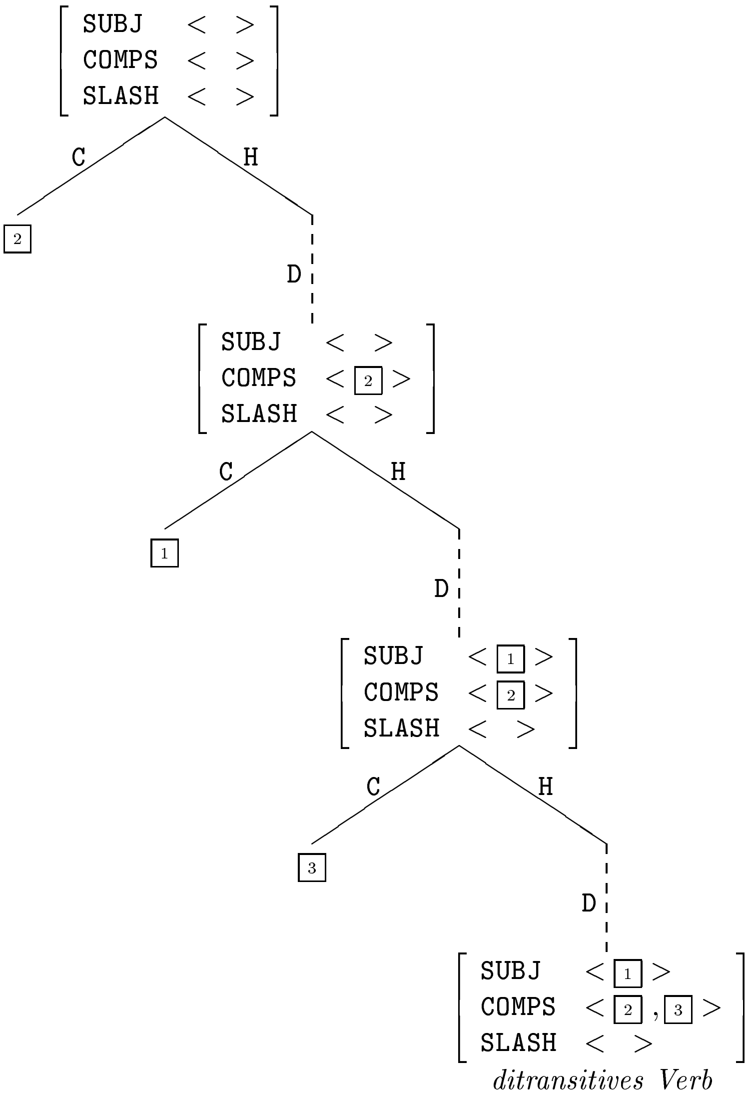

Abbildung 11: alternativer TAG Baum für ditransitives Verb

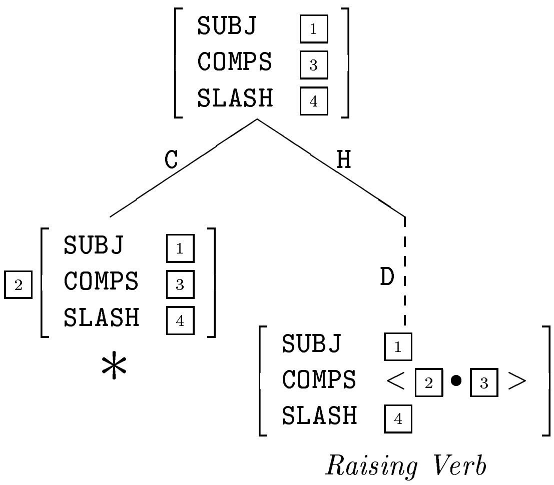

Abbildung 12: Raising Verb

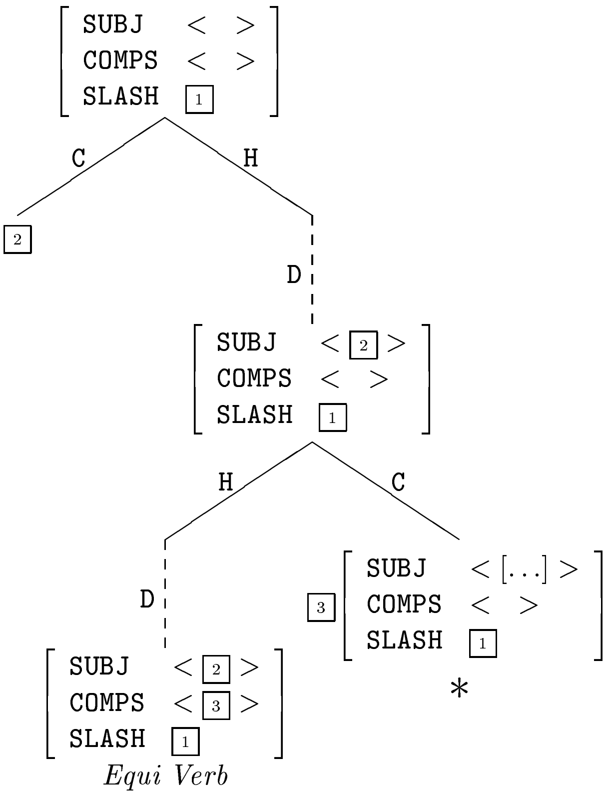

Abbildung 13: Equi Verb

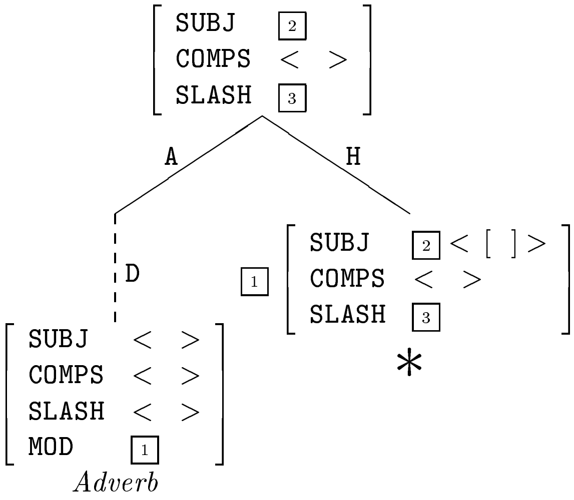

Abbildung 14: Adverb

An den Wurzelknoten all dieser Bäume sind natürlich auch wieder Regelschemata anwendbar. Dies kann aber zu unendlicher Rekursion führen. Außerdem ist es linguistisch nicht sinnvoll, ein unspezifiziertes Selector–Feature weiter zu reduzieren, da das bedeuten würde, daß ein Funktor auf ein Argument angewendet wird, das er nie explizit auswählt.

Eine einfache Lösung wäre es, die Reduktion eines Selector–Features zu verbieten, sobald sein Wert unspezifiziert ist. Leider kann es aber auch vorkommen, daß Werte für Selector–Features von Knoten eingeführt werden, die nicht auf dem Trunk (Pfad vom lexikalen Anker zur Wurzel) liegen. Diese Knoten können auch Constraints einführen, die nichts mit den Argumenten des lexikalen Ankers zu tun haben. Das ist z.B. der Fall nach Anwendung des Head–Adjunct–Schemas auf Adverben (Abb. 14). Die modifizierte Konstituente muß einen nicht-leeren SUB–Wert haben. Diese Selektor Information wird bis zur Wurzel angehoben. Intuitiv wäre es falsch, den SUBJ–Wert weiter zu reduzieren, da er vom lexikalischen Anker nicht lizensiert wird.

Aus obigen Beobachtungen ergibt sich schließlich folgendes Terminierungskriterium:

> Der Wert eines Selector–Features F an einem Wurzelknoten wird nicht weiter reduziert, wenn er eine leere Liste enthält, oder wenn es einen Blattknoten (ausgenommen den lexikalen Anker) gibt, der den gleichen Wert wie F hat.

Dieses Kriterium trifft zu bei dem Adverb Baum in Abb. 14. Sowohl der SLASH– als auch der SUBJ–Wert des Wurzelknotens werden geteilt mit einem Blattknoten, der nicht lexikaler Anker ist. Wenn mit diesem Baum eine Adjunktion durchgeführt wird, dann bleiben diese Werte entlang des Domination Links erhalten. Wenn die Werte reduzierbar sind, dann werden sie in einem anderen

Baum reduziert. Ein Auxiliarbaum sorgt also dafür, daß Argumente, die nicht von ihm realisiert werden, an die richtige Stelle "weitergeleitet" werden.

Aus dem Kriterium folgt auch die Definition von *lokaler Vollständigkeit*. Ein Baum ist lokal vollständig, sobald alle Argumente realisiert sind, die von ihm lizensiert werden und nicht woanders. Initialbäume sind immer lokal vollständig. *Globale Vollständigkeit* ist garantiert, weil "woanders" nur für Auxiliarbäume definiert ist, und Auxiliarbäume müssen mit vollständigen Initialbäumen adjungiert werden.

3.3 Identifikation von Auxiliarbäumen und ihren Fußknoten

In traditionellen TAGs sind Auxiliarbäume minimale rekursive Strukturen, deren Fuß– und Wurzelknoten identisch sind. Die Labels dieser Knoten (z.B. S oder NP) geben an, wo ein Auxiliarbaum adjungiert werden kann. Sie erfüllen quasi die gleiche Funktion wie die Selector–Features von HPSG. In einem Auxiliarbaum wird also ein Pfad von der Wurzel bis zum Fußknoten definiert (genannt *Spine*), an dessen Enden die Knoten die gleichen Selektor Informationen haben.

Diesen Ansatz kann man aber nicht so einfach übernehmen für die Selector–Features von HPSG, da es Fuß– und Wurzelknoten gibt, die nur einige Selector–Features gemeinsam haben. Zur Identifizierung von Fußknoten betrachtet man den Wurzelknoten und alle Blätter des Baumes (ohne den lexikalen Anker). Ein Baum ist ein Auxiliarbaum, wenn es einen Blattknoten gibt, der ein oder mehrere nicht leere Selector–Features gemeinsam hat mit dem Wurzelknoten. Dieser Blattknoten wird dann der Fußknoten. Initialbäume sind Bäume, die keinen solchen Knoten haben. Die so identifizierten Fußknoten sind genau die Knoten, die beim Terminierungskriterium dafür sorgen, daß Selector–Features nicht mehr weiter reduziert werden. Fußknoten sind in den Beispielen mit einem $*$ gekennzeichnet.

In Abb. 13 ist die Complement-Daughter der Fußknoten, weil sie das SLASH-Feature mit der Wurzel teilt.

3.4 Anhebung von Selector–Features über Domination Links

Was muß über die Domination Links angehoben werden? Ganz offensichtlich auf jeden Fall die Selector–Features, weil von ihnen abhängt, ob der Algorithmus noch eine weitere Iteration machen kann. Würden die Selector–Features nicht angehoben werden, dann würde der Algorithmus nach der ersten Iteration abbrechen.

Allerdings gibt es auch Probleme, wenn man mehr als nur die Selector–Features anhebt. Z.B. verlangt das Head–Subject–Schema für das Deutsche, daß die Verbphrase finit ist. Würde man zusätzlich zu den Selector–Features auch noch Head–Features anheben, dann wäre das Head–Subject–Schema nicht anwendbar auf infinite Verben. Das hätte zur Folge, daß man nicht die Bäume

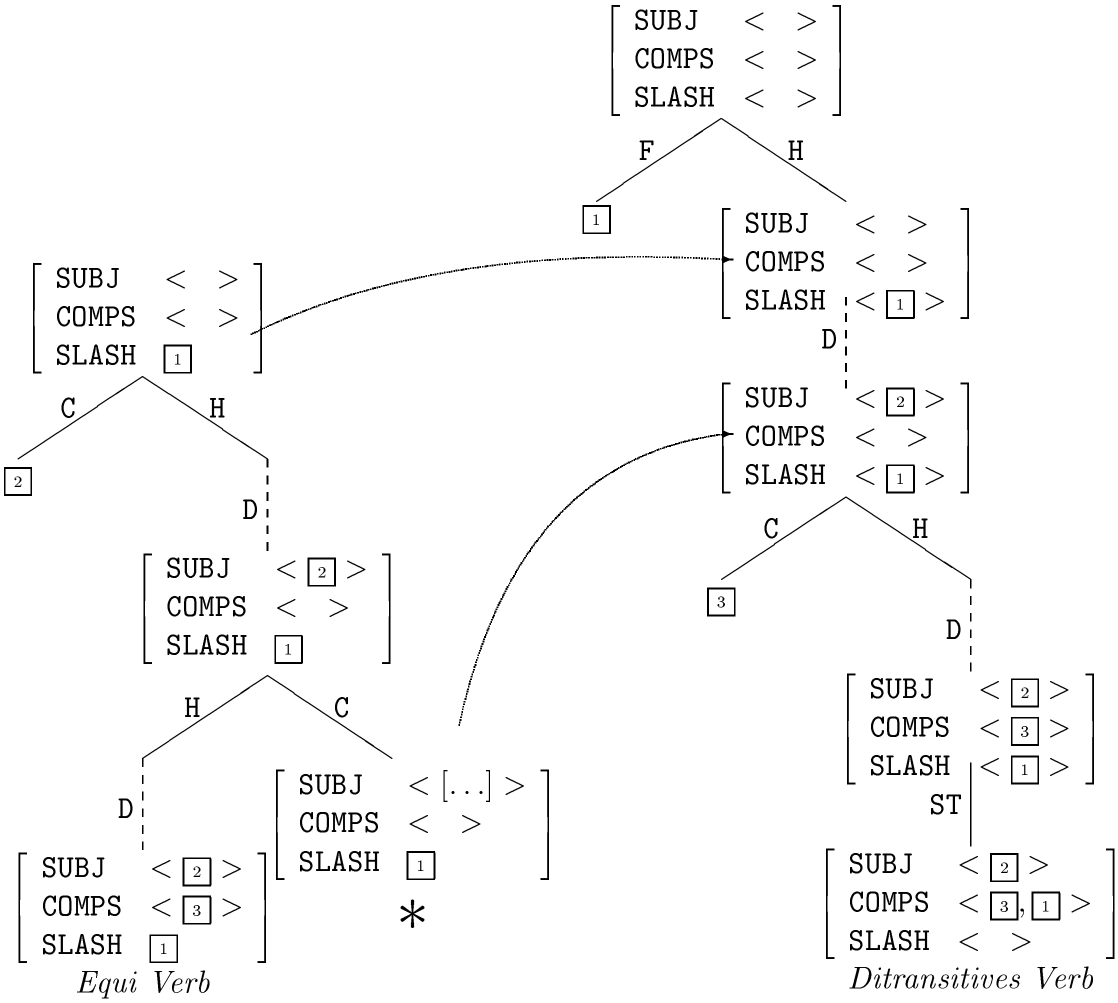

Abbildung 15: Adjunktion eines Equi Verbs an ein ditransitives Verb

produzieren kann, die für die Adjunktion von Raising Verben notwendig sind. Deshalb werden nur Selector–Features über die Domination Links angehoben.

Allerdings gibt es auch Probleme, wenn immer *alle* Selector–Features angehoben werden. Dann müßten die Fuß– und Wurzelknoten in den Auxiliarbäumen genau dieselben Selector–Features haben, damit sie kompatibel sind zu den Selector–Features an den Domination Links in anderen Bäumen, wo eine Adjunktion stattfinden soll. Wie man z.B. an dem Auxiliarbaum in Abb. 13 sehen kann, erfüllen aber nicht alle Auxiliarbäume diese Eigenschaft. Um diesen Auxiliarbaum in den TAG Baum eines infiniten, ditransitiven Verbs zu adjungieren, braucht man dort einen Domination Link, bei dem sich die oberen Selector–Features von den unteren unterscheiden (Abb. 15).

Man könnte alle Selector–Features anheben, wenn nur die Regelschemata für eine Reduzierung von Selector–Features zuständig wären. Das ist aber leider nicht der Fall, da auch lexikale Zeichen eine Rolle bei der Reduzierung spielen können (wie die Equi Verben im letzten Beispiel).

Man kann diese Problem lösen, wenn man einen Baum produziert für jede Untermenge von Selector–Features an jedem Domination Link. Das würde aber zu einer Übergenerierung von Bäumen führen, was nicht wünschenswert ist. Der hier beschriebene Algorithmus löst das Problem durch die Einführung einer zweiten Phase. In der oben beschriebenen ersten Phase werden immer alle Selector–Features angehoben.

3.5 Die zweite Phase

Der Sinn der zweiten Phase ist, Hinweise darauf zu finden, wann nur eine bestimmte Teilmenge von Selector–Features über einen Domination Link angehoben werden soll und wie diese Teilmenge aussieht. Diese Hinweise liefern die Auxiliarbäume, die man in der ersten Phase erhält. Die Unterschiede bei den Selector–Features in Fuß– und Wurzelknoten geben an, in welchen Fällen welche Selector–Features angehoben werden müssen und welche nicht.

Beim Equi Verb aus Abb. 13 unterscheiden sich Fuß– und Wurzelknoten bzgl. ihres SUBJ–Wertes. Der Fußknoten und die Beziehung zwischen der Selektor Information des Fuß– und des Wurzelknotens sind der gesuchte Hinweis darauf, wann welche Selector–Features angehoben werden: Das Selector–Feature SUBJ wird genau dann nicht angehoben, wenn es eine nicht leere Liste enthält und wenn es ein leeres Selector–Feature COMP am gleichen Knoten gibt.

In der zweiten Phase wird wieder mit den lexikalen Typen gestartet, auf die eine Sequenz von Regelschemata angewandt wird. Jetzt wird allerdings nach jeder Anwendung geprüft, ob die Selector–Features des aktuellen Wurzelknoten mit den Selector–Features des Fußknotens eines Auxiliarbaums unifiziert werden können. Ist das der Fall, dann werden die Selector–Features entsprechend dieses Auxiliarbaumes angehoben. So führt z.B. der Auxiliarbaum aus Abb. 13 dazu, daß in der zweiten Phase ein alternativer Baum für ditransitive Verben erzeugt wird (Abb. 16). Hier wurde zusätzlich zur Verdeutlichung noch ein Komplement mit der Slash–Termination–Regel in die SLASH Liste geschrieben.

Die zweite Phase muß eventuell mehrfach durchgeführt werden, je nachdem ob dort ein neuer Auxilarbaum erzeugt wurde.

4 Beispielableitungen

4.1 (I know) what Kim gives to Sandy

Für die Nominalphrasen *Kim*, *Sandy* und *what*, sowie die Präposition *to* gelten keine besonderen Annahmen. Der Satz läßt sich direkt aus dem Lexikoneintrag für ditransitive Verben ableiten durch Anwendung der Slash–Termination–Regel und der Schemata Head–Complement, Head–Subject sowie Head–Filler. Substitution an den Blattknoten ergibt dann den gewünschten Satz (Abb. 17).

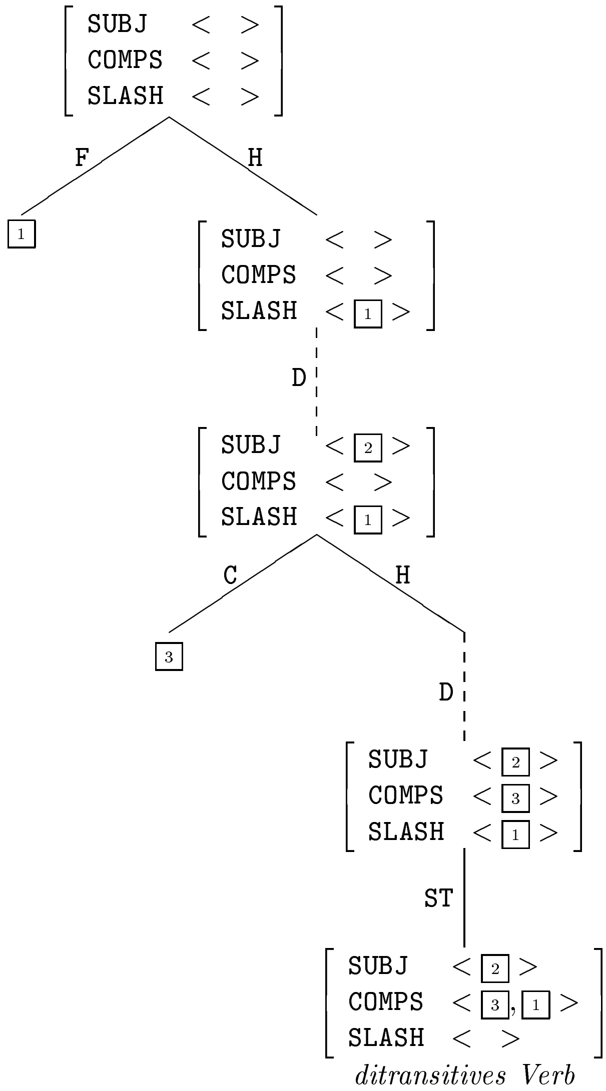

Abbildung 16: TAG Baum für ditransitives Verb in Phase 2

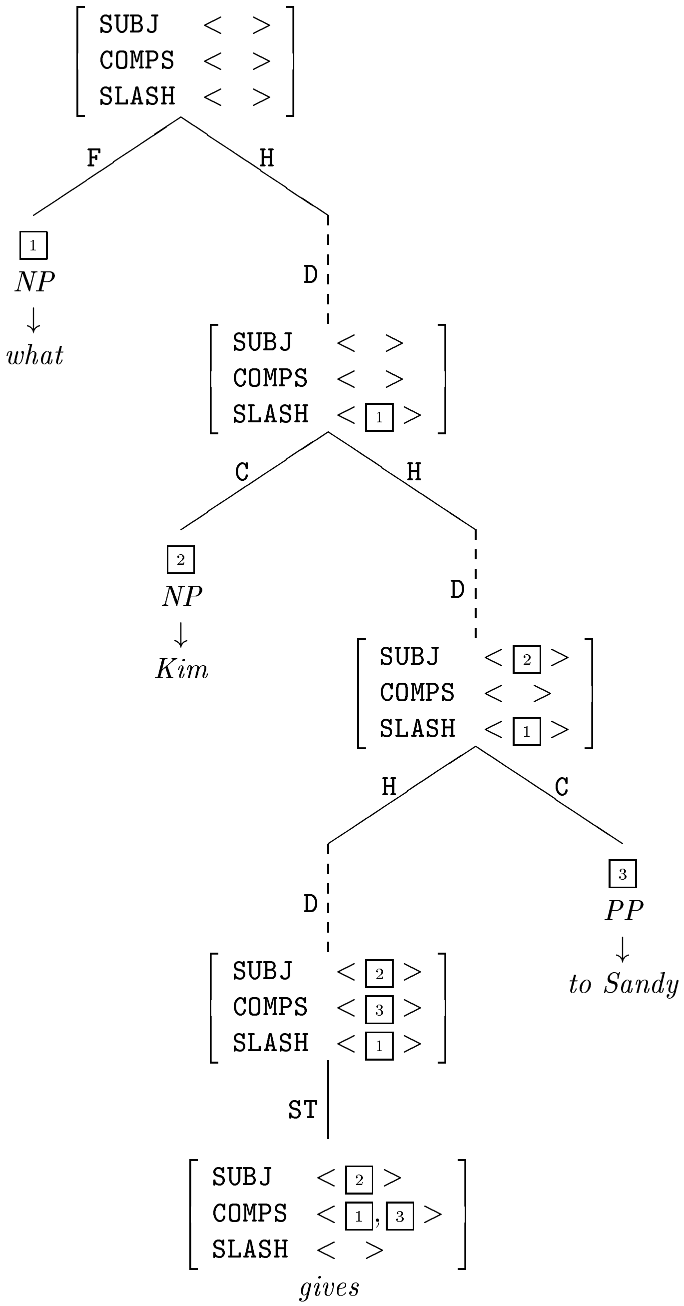

Abbildung 17: what Kim gives to Sandy

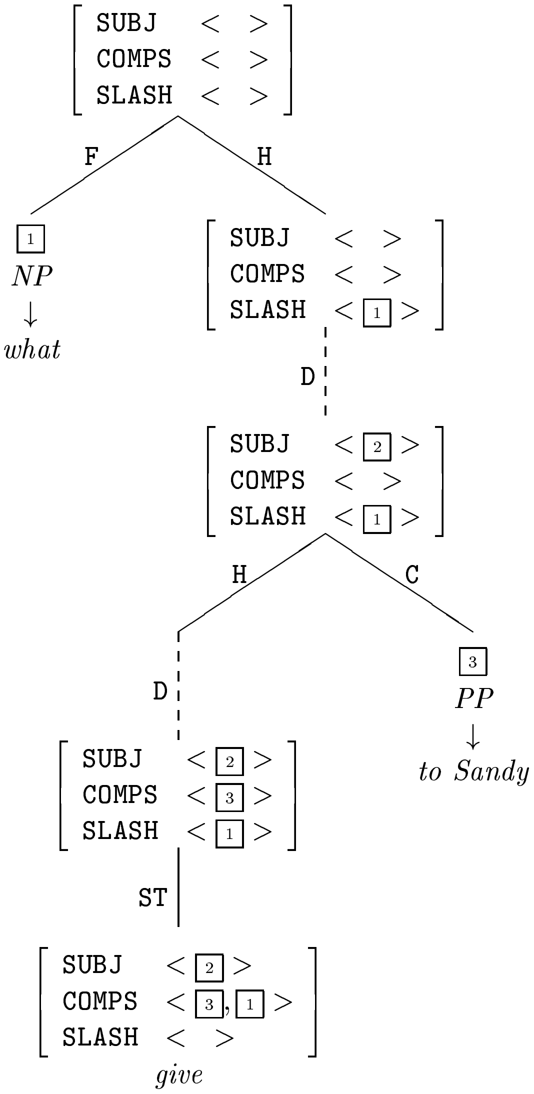

Abbildung 18: TAG Baum für *give* in Phase 2

4.2 (I know) what Kim wants to give to Sandy

Das Verb *want* ist ein Equi Verb und hat deshalb einen Ableitungsbaum wie in Abb. 13. Der Infinitiv Marker *to* verhält sich wie ein Raising Verb und hat deshalb einen Ableitungsbaum wie in Abb. 12, allerdings ist dabei $\boxed{3}$ leer.

In der zweiten Phase leitet man einen alternativen Baum für *give* ab, in den man den Auxiliarbaum für *want* am obersten Domination Link adjungieren kann (Abb. 18). Das gleiche macht man für *to* (Abb. 19).

Durch Adjunktion des Baumes aus Abb. 19 an den obersten Domination Link des Baums aus Abb. 18 erhält man den Baum in Abb. 20, der dem String *what to give to Sandy* entspricht.

Die endgültige Ableitung erhält man durch Adjunktion des Baumes für das

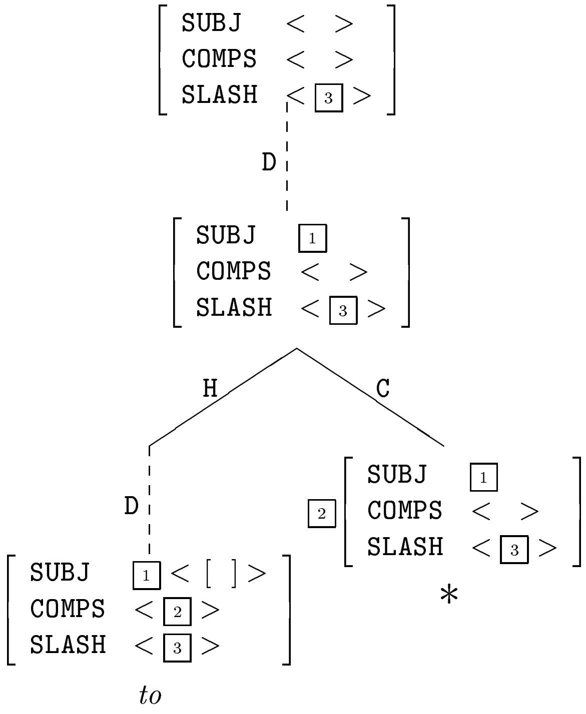

Abbildung 19: TAG Baum für *to* in Phase 2

Equi Verb an den obersten Domination Link des Baums in Abb. 20. Dies führt den String *Kim wants* in den String *what ... to give to Sandy* ein.

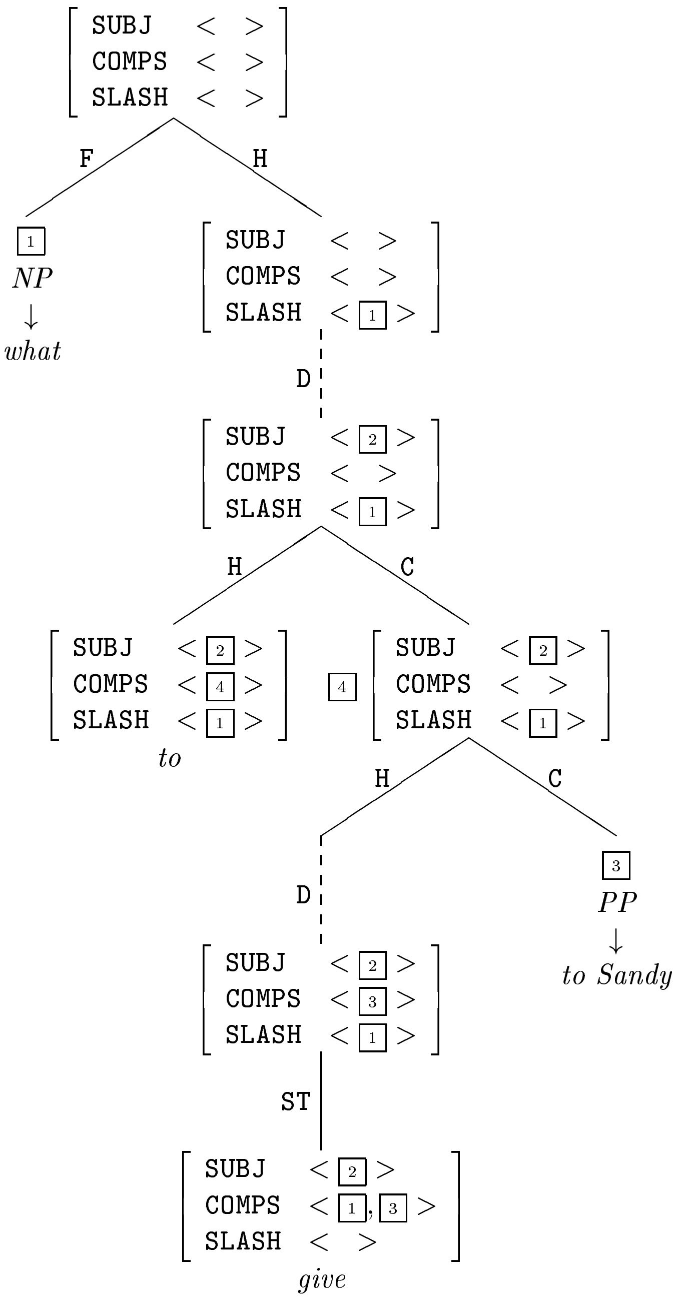

Abbildung 20: what to give to Sandy

Literatur

[Kas95] Kasper, Kiefer, Netter and Vijay-Shanker. Compilation of HPSG to TAG. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pp. 92-99, Cambridge, Massachusetts, 1995.

[Kas92] Robert Kasper. On compiling HPSG in lexikalized TAGs. In *Proceedings of the* 2^{nd} *Workshop on TAGs*, Philadelphia, 1992.

[PS94] Carl Pollard and Ivan Sag. *Head Driven Phrase Structure Grammar*. Studies in Contemproary Linguistics. CSLI,Stanford & University of Chicago Press, 1994.