Syntax und Formale Grammatiken

HPSG II: Basisstrukturen

Dozent: Gerold Schneider

Übersicht

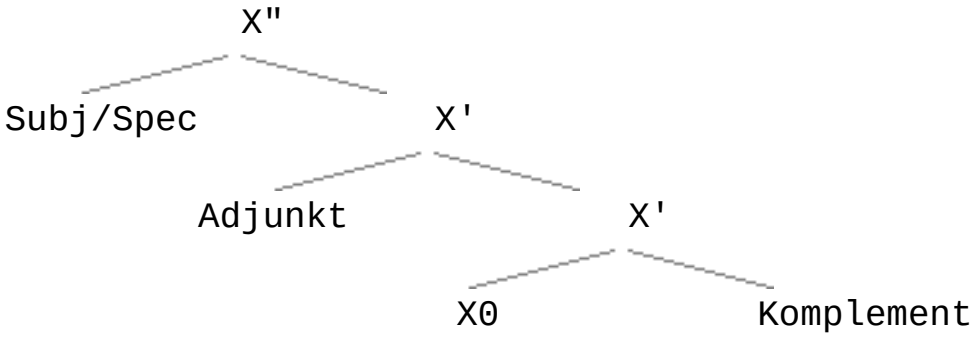
-

Erkenntnisse aus der X-Bar Theorie

In der X-Bar Theorie verbleiben statt vielfältiger Phrasenstrukturregeln (PSG-Regeln) wie

```
S --> NP VP
NP --> Det N'
```

nur noch drei allgemeingültige Regelskelette, die für alle Wortklassen gelten (was durch die Verwendung des Variablenbuchstaben X ausgerückt wird). Diese drei Regelskelette sind die *Specifier-Regel*, die *Adjunct-Regel* und die *Complement-Regel*. Der Lexikoneintrag des Kopfes bestimmt die Art und Anzahl der Specifier und Komplemente, X-bar-Theorie ist nur eine einschränkende (constraining) Regel, die z.B. sicherstellt, dass der Kopf unverändert von X zu X' zu X" projiziert wird.



Adjunkte werden im Deutschen auch freie Angaben, Komplemente auch obligatorische Ergänzungen genannt. Der Kopf eröffnet sog. Valenzen für die obligatorischen Specifier und Komplemente. Sind alle Specifier und Komplemente gefunden worden so spricht man von einer gesättigten Valenz.

-

Die Unterscheidung zwischen verschiedenen Bar-Ebenen ist nicht zwingend nötig. In Chomskys Minimalist Program wie auch in HPSG erhalten X,X' und X" dieselbe Bezeichnung: X. X" entspricht dabei einem X mit vollständig gesättigten Valenzen (leerer Subkategorisierungsrahmen), X' einem X mit teilweise gesättigten Valenzen und X0 einem X mit nur ungesättigten Valenzen (Subkategorisierungsliste aus dem Lexikon), z.B:

```
V<>
% Verb mit gesättigter Valenz -> VP
N<Det>
% Nomen mit teilw. gesättigten Valenzen -> N'
V<NP[kas:nom],NP[kas:akk]>
% Verb mit nur ungesättigten Valenzen -> V0
```

Die allgemeinste valenzbasierte Phrasenstrukturregel

Eine derart revidierte X-bar-Theorie lässt sich vereinfacht so ausdrücken: Ein Kopf fordert und findet Specifier und Komplemente. Das allgemeinste Phrasenstrukturregelskelett sieht also so aus:

```
X<> --> X<Y,Z,...>, Y, Z, ...
% X mit gesättigter Valenz besteht aus X plus
% den Elementen Y, Z etc., für die X eine Valenz
öffnet,
% d.h. für die X subkategorisiert.

           X<>   % leere Subkatliste: gesättigte
Valenz
          / | \
X<Y,Z, ...> Y  Z ...% Knoten X öffnet Valenzen für
Y,Z, ...
```


-

Dieses einfachste Phrasenstrukturregelskelett wird in HPSG durch das Subkategorisierungsprinzip aus der Vorlesung HPSG I, ausgedrückt ([Borsley 97] (Kapitel 6.2, p.83). Seit [Pollard and Sag 94] Kapitel 9 wird es*Valenzprinzip* genannt ([Borsley 97] (Kapitel 6.2, p.84):

"In a headed phrase, for any valence feature F, the F value of the head is the concatenation of the phrase's F value with the list of SYNSEM values of the F daughters".

In anderen Worten (dasselbe in verschiedenen Versionen)

- Für ein Valenzmerkmal F (z.B. COMPS) gilt: Der Wert des Merkmals F des *untergeordneten Kopfes X* ist die Liste des Merkmals F des *übergeordneten Knotens X* PLUS die Liste der SYNSEM Werte der Töchter des übergeordneten Knoten X.
- Für ein Valenzmerkmal F (z.B. COMPS) gilt: Der Wert des Merkmals F des *übergeordneten Knotens X* ist der des Merkmals F des *untergeordneten Kopfes X* MINUS die Liste der SYNSEM Werte seiner Geschwister.
- Für dieValenzlisten (z.B. COMPS) gilt: Die Valenzliste des *übergeordneten Knotens X* ist die Valenzliste des *untergeordneten Kopfes X* MINUS die Liste der SYNSEM Werte der Geschwister, die ja die Valenzen sättigen.
- Die Valenz-Anforderungen einer Phrase sind diejenigen der Head-Tochter abzüglich der bereits von den Komplement-Töchtern erfüllten Anforderungen.

Dieses einfachste Phrasenstrukturregelskelett ist vollkommen äquivalent zu einer Dependenzgrammatik (DG). Wie wir in der Diskussion der DG erwähnt haben, entspricht DG aber einer "flachen" X-bar-Theorie, die keine Zwischenkategorien (also X') ausdrückt.

-

Die Übervereinfachung, die obige allgemeinste PSG einführt, ist, dass sie alle Valenzen auf einmal sättigt, so dass keine Reihenfolge mehr erkennbar ist, also ob die Valenz Y oder die Valenz Z zuerst gesättigt werde:

```
Z zuerst:
           X<>   % leere Subkatliste: gesättigte
Valenz
          /   \
        X<Y>   Y   % teilweise gesättigte Valenzen
        /  \
    X<Z,Y>  Z      % nur ungesättigte Valenzen

Y zuerst:
           X<>   % leere Subkatliste: gesättigte
Valenz
          /   \
        X<Z>   Z   % teilweise gesättigte Valenzen
        /  \
    X<Y,Z>  Y      % nur ungesättigte Valenzen
```

Um zwischen X'' und X' unterscheiden zu können, müssen vom lexikalischen Kopf X0 her kommend *zuerst* Komplement-Valenzen gesättigt werden (was X' ergibt) und erst *anschliessend* die Specifier-Valenzen gesättigt werden (was X'' ergibt).

-

Schrittweise Sättigung der Valenzen

Aus Verarbeitungsgründen ist es sinnvoll, die Valenzen schrittweise abzuarbeiten, eine nach der andern. Eine gleichzeitige Abarbeitung würde beispielsweise verschiedene PSG-Regeln erfordern für eine, zwei, oder drei Valenzen:

```
X<A,B,C>        --> X<>, A, B, C.
X<A,B>          --> X<>, A, B.
X<A>            --> X<>, A.
```

Eine rekursive schrittweise Verarbeitung erlaubt die Verwendung einer einzigen Regel plus einer Rekursionsabbruchregel (in 'echtem' Prolog verwendet man dabei [] statt <> für Listen):

```
X<First|Rest> --> X<Rest>, First. % Rekursion
X<>.                             % Rekursionsabbruch
```

E.g. für VPs in PATR (die Nummer hinter vp wird vergeben, damit die Merkmale eindeutig sind. vp1 ist also der übergeordnete, vp2 der untergeordnete vp in der Rekursion):

```
% Complements
vp1 --> vp2, X #
      vp1:head = vp2:head,
      vp2:subcat = [X|Rest],
      vp1:subcat = Rest.
```


-

[Pollard and Sag 94] Kapitel 1-8 verwenden eine einzige Subkategorisierungsliste SUBCAT (das sog.*SUBCAT-framework*). Da es eine geordnete Liste ist, ergibt sich die Reihenfolge der Valenzsättigung aufgrund der Ordung in der Liste. Subjekte beispielsweise, die in GB immer in einer Specifier-Position stehen, sind die zuletzt zu sättigende Valenz.

Die zuletzt zu sättigende Valenz ist aber nicht immer ein Specifier-Element (e.g. Artikel oder Subjekt).

- In Pro-drop Sprachen (Italienisch, Spanisch, Latein, Finnisch, teilweise gar Schweizerdeutsch) kann das Subjekt weggelassen werden. Die letzte zu sättigende Valenz ist dann ein Komplement ([Borsley 97] (Kapitel 6.3, p.99)).
- Relationale Nomen öffnen Valenzen für Komplemente ([Borsley 97] (Kapitel 6.3, p.98)). Die letzte zu sättigende Valenz ist dann ein Komplement.
- Artikel sind sehr oft fakultativ oder gar ungrammatisch (z.B. bei Eigennamen). Die letzte zu sättigende Valenz ist dann ein Komplement.

Ganz allgemein sind Specifier eher fakultativ als Komplemente, so dass es fraglich erscheint, ob man sie über die gleiche Leiste schlagen soll.

Um Komplemente und Specifier auseinanderzuhalten (und somit auch klar X" undX' ableiten zu können) ist es sinnvoll, eine spezielle Specifier-Subkategorisierungsliste zu führen (das sog. *SUBJ-COMPS framework*, [Pollard and Sag 94] ab Kapitel 9). Wir werden diesen Ansatz weiterverfolgen.

Zusätzlich ist es in HPSG freigestellt, ob die Valenzen schrittweise oder alle auf einmal gesättigt werden sollen ([Borsley 97] (Kapitel 6.2, p.86)). Werden alle auf einmal gesättigt, so kann man mit nur einer Subkategorisierungsliste nur "flache" Strukturen erzeugen, in denen es, wie in Dependenzgrammatik, keine X' gibt.

-

Die Regelschemata der HPSG

Die Regelschemata der HPSG orientieren sich an der X-bar-Theorie. Es gibt in ihr deshalb analog eine *Specifier-Regel*, eine *Complement-Regel* und eine *Adjunct-Regel*. Anders als GPSG ist HPSG bemüht, jeweils nur eine Regel für diese Grundschemata zu benötigen.

X" zu X': Die Specifier-Regel

Die Specifier-Regel hat das allgemeine Format

```
X" --> (Spec) X' % Fakultativer Specifier und
obligatorischer Kopf-single-bar
```

wobei die Reihenfolge von Spec und X' noch nicht festgelegt ist (in GB wird sie als sprachspezifischer Parameter betrachtet).

Beispiele für diese Regel sind

```
N" --> (Det) N'
```

(Artikel stehen immer in Spec-Position)

```
V" --> N" V' % z.B. [The woman] [pushes the bike]
V" --> S' V' % z.B. [That he will come] [pleases us]
```

(In neuerer GB-Tiefenstruktur steht das Subjekt in der VP-Specifier-Position)

Es hängt stark von der Wortklasse, und innerhalb der Wortklasse von der Subkategorisierung ab, ob ein Specifier obligatorisch oder fakultativ ist, und welche Wortklasse er hat. Es ist deshalb sinnvoll, auch den Specifier in die Subkatliste mitaufzunehmen, wie es HPSG konsequent tut, sei es als Element der SUBCAT-Liste ([Pollard and Sag 94] bis Kapitel 9, *SUBCAT-framework*) oder als eigene SPR-Liste und/oder SUBJ-Liste ([Pollard and Sag 94] ab Kapitel 9, *SUBJ-COMPS- framework*).

Falls der Specifier auch in der Subkategorisierungsliste steht, so erkennt man X-single-bar Strukturen daran, dass alle Valenzen ausser der SPECIFIER-Valenz

-

gesättigt sind. Im SUBCAT-framework enhält der SUBCAT-liste nur noch ein Element, im SUBJ-COMPS framework ist die COMPS-Liste ganz leer.

Einige Versionen des SUBJ-COMPS frameworks unterscheiden zwischen Subjekt (in der SUBJ-Liste) und Specifier (in der SPR-Liste). Eine solche Version weicht von der X-bar-Theorie ab, in der Subjekte immer als Specifier auftreten. Wir verfolgen zunächst nur die X-bar-konforme Variante, in der die Subject-Predicate Regel auch durch die Specifier-Head Regel umfasst wird.

SPECIFIER-HEAD RULE ([Borsley 97] (Kapitel 7.2, p.108)) =
SUBJECT-PREDICATE RULE ([Borsley 97] (Kapitel 6.3, p.94))

```
[SPR<>] --> [SPR<[]>,COMPS<>], []
             HEAD               SPR
```

<[]> bedeutet dabei ein Listenelement.

"A sign with the feature specification SPR<> can contain a head daughter with the feature specifications COMPS<> and a specifier daughter."

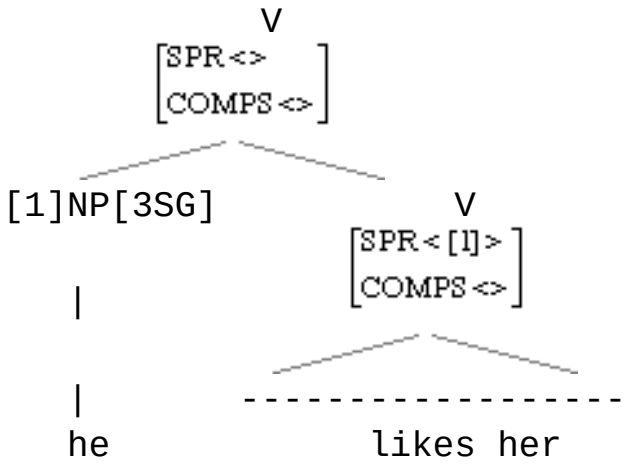
Die leere COMPS-Subkategorisierungsliste stellt sicher, dass alle Komplementsvalenzen gesättigt sind *bevor* die Specifier-Valenzen gesättigt werden, wie es X-bar verlangt.

-

Diese Regel muss noch zusammenarbeiten mit

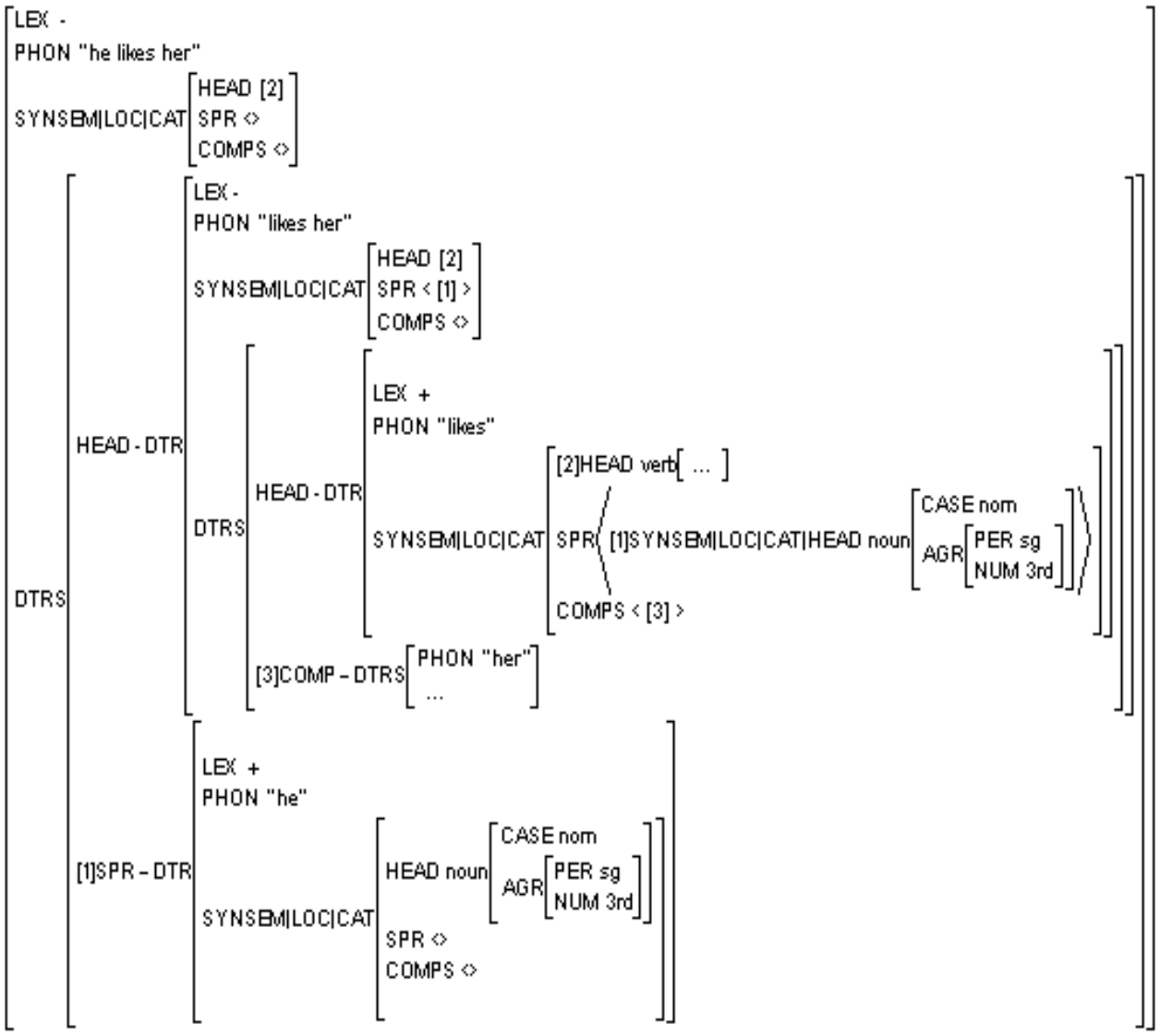
- dem Head-Feature Prinzip aus der Vorlesung HPSG I: Die Bedingung, dass Mutterknoten und Head-Tochter in den Head-Merkmalen übereinstimmen, wird durch das Head-Feature-Prinzip ([Borsley 97] (Kapitel 4.3)) erreicht.
- dem Subkategorisierungsprinzip aus der Vorlesung HPSG I, im SUBJ-COMPS framework Valenzprinzip genannt (([Borsley 97] (Kapitel 6.2, p.84)). Es stellt sicher, dass die geforderten Valenzen (COMPS, SPR) auch tatsächlich von den gefundenen Töchtern erfüllt werden.

um Strukturen zu erlauben wie



-

Dasselbe als (nur leicht) vereinfachte HPSG Merkmal-Wert Struktur:



-

Borsley's Vorschlag ([Borsley 97] (Kapitel 7.2, p.107)), separate Valenzlisten für Specifier (SPR) und Subjekte (SUBJ) zu führen, ist darin begründet, dass gewisse Strukturen sowohl einen Specifier als auch ein Subjekt haben. Dem kann man allerdings entgegenhalten, dass viele Sprachen mehrere Artikel haben können, also ohnehin mehrere Specifier haben können.

il mio fratello (italienisch)
den gamla bil*en min* (norwegisch)

Einige der Beispiele beruhen auch auf der speziellen (der Dependenzgrammtik entlehnten) Behandlung von Kopula: Im Satz

Peter is a candidate

ist nicht das Kopula *is*, sondern das Komplement *candidate* der Kopf des Satzes, von dem sowohl das Kopula als auch das Subjekt abhängen.

-

X' zu X': Die Adjunct-Regel

Die Adjunct-Regel baut eine beliebige Anzahl Adjunkte in die Syntaxstruktur ein, je nachdem wie sie im zu parsenden Text vorliegen. Ihr allgemeines Format ist

```
X' --> Adjunct X'
```

Bei jeder Anwendung dieser Regel wird ein Adjunkt in die Struktur eingefügt. Deshalb wird sie so häufig angewendet wie Adjunkte im zu parsenden Text vorhanden sind, z.B. falls keine Adjunkte vorhanden sind entsprechend null mal.

Beispiele für diese Regel sind

```
N' --> Adj" N'
```

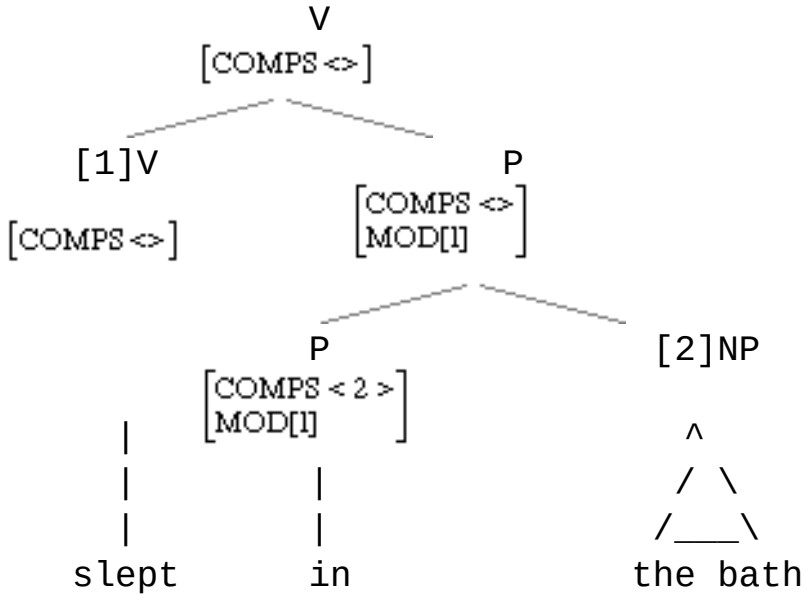
(Einfügung fakultativer Adjektive)

```
Adj' --> Adv Adj'
% z.B. [sehr] [gut], [sehr [sehr]] [gut] (2x Regel)
```

(Adjektivmodifikation durch Adverbien)

-

In HPSG erkennt man X'-Strukturen typischerweise daran, dass die Valenzen teilweise gesättigt sind (siehe SUBJ-COMPS framework oben). Entscheidend ist, dass die Valenz zwischen übergeordnetem und untergeordnetem Kopf unverändert bleibt. Eine Phrase besteht aus derselben Phrase plus dem Adjunkt. Der Unterschied zwischen der übergeordneten und der untergeordneten Phrase ist, dass durch Structure-sharing der Kopf des Adjunkts mit der modifizierten Konstituente verbunden wird.



-

X' zu X: Die Komplement-Regel

Die Komplement-Regel hat die allgemeine Form

```
X' --> Komplement X
```

oder in HPSG-inspiriertem Pseudoprolog:

```
X[COMPS:<First|Rest>] --> X[COMPS:<Rest>], First.
   % Rekursion
X[COMPS:<>].
   % leere Subkatliste -> Rekursionsabbruch
```

Die HPSG-Spezifikation lässt offen, ob alle Komplementsvalenzen auf einmal gesättigt werden, oder ob sie elementsweise (wie oben) abgearbeitet werden ([Borsley 97] (Kapitel 6.2, p.86)). Die üblicherweise verwendete Regel sättigt alle Komplementsvalenzen auf einmal:

```
HEAD-COMPLEMENT RULE ([Borsley 97] (Kapitel 6.2, p.84))
[COMPS<>] --> [COMPS L], []*
               HEAD      COMPS
```

L steht dabei für eine Liste, [] für ein Listenelement, * bedeutet beliebige Wiederholung. Die Regel sagt nur aus, dass eine Struktur mit gesättigter COMPS-Valenz einen Kopf mit einer Vlenz-Liste und beliebig viele Komplemente haben kann.

Auch diese Regel muss noch zusammenarbeiten mit

- dem Head-Feature Prinzip aus der Vorlesung HPSG I: Die Bedingung, dass Mutterknoten und Head-Tochter in den Head-Merkmalen übereinstimmen, wird durch das Head-Feature-Prinzip ([Borsley 97] (Kapitel 4.3)) erreicht.
- dem Subkategorisierungsprinzip aus der Vorlesung HPSG I, im SUBJ-COMPS framework Valenzprinzip genannt (([Borsley 97] (Kapitel 6.2, p.84)). Es stellt sicher, dass die geforderten Valenzen (COMPS, SPR) auch tatsächlich von den gefundenen Töchtern erfüllt werden.

-

Falls ein Kopf keine Komplemente verlangt, so ist es mit dem bisherigen Vorschlag aber nicht möglich, zwischen X0 und X' zu unterscheiden, da sowohl der lexikalische Kopf X0 als auch die Zwischenkategorie X' im SUBJ-COMPS framework COMPS<> haben. HPSG verwendet deshalb ein binäres Merkmal LEX, das den Wert + für lexikalische Köpfe und den Wert - für phrasale Kategorien annimmt (([Borsley 97] (Kapitel 6.2, p.85)).

Markers

Die Kopf-Komplement-Regel ist die vielleicht wichtigste syntaktische Grundregel. Es gilt aber zu beachten, dass einige der in GB als Köpfe erachteten Kategorien keine Köpfe in HPSG sind, so Complementizer, Artikel und (teilweise) Präpositionen (([Borsley 97] (Kapitel 6.2, p.87)). Artikel werden als Specifier behandelt, wie in früheren Varianten der GB.

`Mary saw (the) complex bracketted structures.`

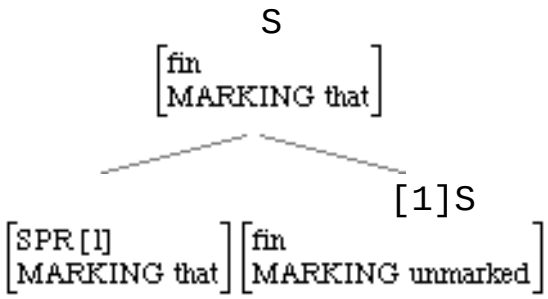
Complementizer und Präpositionen werden als Marker erachtet, eine Morphologie-ähnliche Art der Dependenz, die anders ist als Specifier- oder Komplementsdependenz. Diese Abweichung von der X-bar Theorie löst das syntaktische Problem, dass Specifier und Complementizer häufig fakultativ sind, man also leere Köpfe annehmen müsste ([Borsley 97] (Kapitel 6.2, p.87)). Besonders [Pollard and Sag 94] argumentieren gegen die Verwendung leerer Köpfe.

`Peter thinks (that) Syntax is boring.`

Wenn man Präpositionen als kasusähnliche Marker betrachtet, so kann man den sog. Dative shift einfacher behandeln:

`Mary gives a book` *`to Peter`*`.`
`Mary gives` *`Peter`* `a book.`

Markers funktionieren ähnlich wie Adjunkte. Eine Phrase besteht aus derselben Phrase plus dem Marker. Der Unterschied zwischen der übergeordneten und der untergeordneten Phrase ist der Wert des Flags MARKING, das anzeigt, ob ein Marker vorhanden ist.



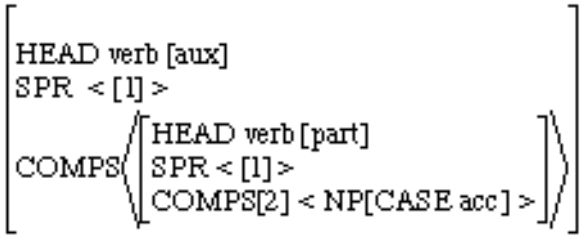
Argumentskomposition

Auch die sog. Argumentskomposition ist eine Abweichung von der X-bar-Theorie. Syntaktisch getrennte Elemente bilden in ihr ein einziges semantisches Element (vgl. Aufbau der f-Struktur in LFG). Argumentskomposition kann verwendet werden, um Hilfs- und Hauptverben zu einer Einheit zusammenzufassen. Dies erlaubt es, Sätzen mit synthetischen (z.B. deutsches oder englisches Perfekt) und analytischen (z.B. deutsches oder englisches Imperfekt) ähnliche Strukturen zu geben.

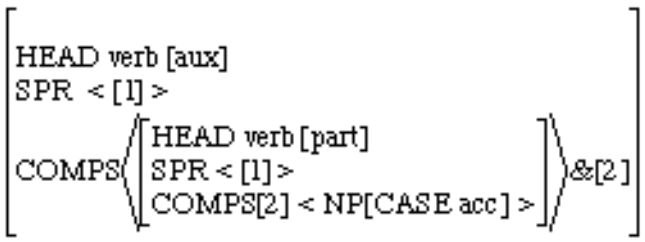
-

In einer Hilfsverb-Partizip-Konstruktion subkategorisiert das finite Hilfsverb für das Subjekt (SPR<[1]>) und das Partizip, die anderen Valenzen (Objekte etc.) stammen aber vom Partizip (COMPS(COMPS<[2]>)).

Im Beispiel eines transitiven Verbs sieht das so aus:



Um die Komplementsvalenzen des Partizips (COMPS(COMPS<[2]>)) an das Hilfsverb hochzugeben, werden diese ganz einfach an die Komplementsliste (COMPS<HEAD verb[part] ... >) angehängt (konkateniert). Der Konkatenationsoperator in HPSG ist '&'.



-