Frank Richter:
Syntax I

Die Signatur

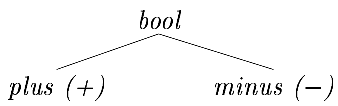
An dieser Stelle fassen wir die Signatur zusammen, die wir unseren Überlegungen im Seminar zugrundegelegt haben. Sie ist nahezu aber nicht immer identisch mit der Signatur des Appendix im Buch von Pollard und Sag.

1 Die Sortenhierarchie

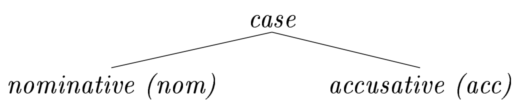
Die Sortenhierarchie als ganzes wird am übersichtlichsten, wenn man sie als einen einzigen Graphen darstellt. Allerdings ist das auf normalgroßem Papier kaum möglich. Daher werden hier die Zweige des Gesamtgraphen gesondert angegeben: Zunächst listen wir die unmittelbaren Untersorten der generellen Obersorte *object* alphabetisch auf und geben dann nacheinander die Teilgraphen wider, die diese Untersorten als Wurzeln haben. Wo dies mit dem gegebenen Platz vereinbar ist, wird sowohl der vollständige Sortenname als auch seine gebräuchlichste Abkürzung genannt.

Unmittelbare Untersorten von *object*:
boolean (bool), case, category (cat), constituent-structure (con-struc), content (cont), context (conx), contextual-indices (c-inds), gender (gend), head, index (ind), list, local (loc), marking, mod-synsem, nonlocal (nonloc), nonlocal1 (nonloc1), number (num), person (per), phoneme-string (phonstring), preposition-form (pform), quantifier-free-parametrized-state-of-affairs (qfpsoa), semantic-determiner (semdet), set, sign, verbform (vform)

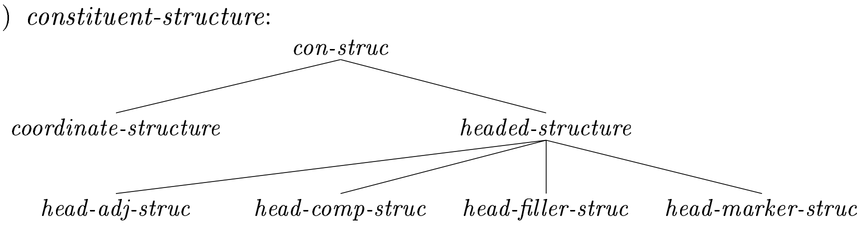
(1) *boolean*:



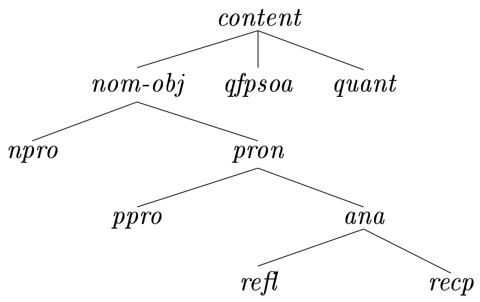
(2) *case*:



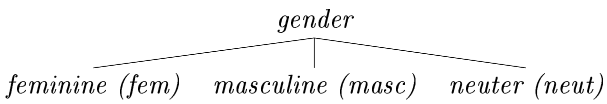
(3) *constituent-structure*:



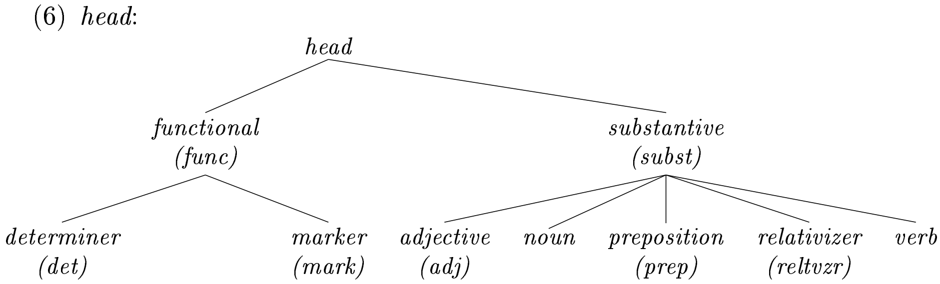
(4) *content*:



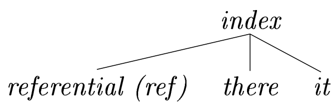
(5) *gender*:



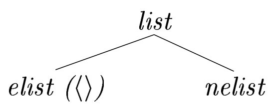
(6) *head*:



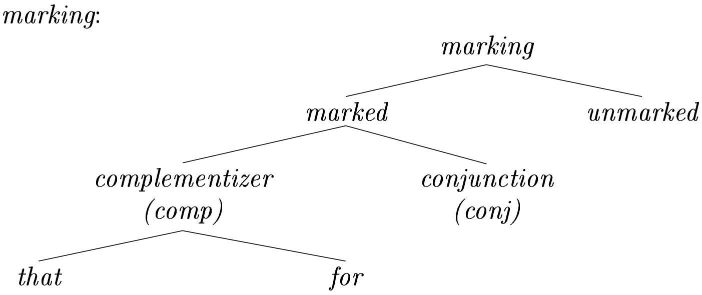
(7) *index*:



(8) *list*:



(9) *marking*:



(10) *mod-synsem*:

mod-synsem

none *syntax-semantics (synsem)*

(11) *number*:

number

singular (sing) *plural (plur)*

(12) *person*:

person

1st *2nd* *3rd*

(13) *phoneme-string*:

phonstring

Mary who gives bagels . . .

(14) *preposition-form*:

pform

to *of* . . .

(15) *semantic-determiner*:

semdet

forall *exists* *the* . . .

(16) *set*:

set

eset ({}) *neset*

(17) *sign*:

sign

word *phrase*

(18) *verbform*:

vform

base	*finite*	*gerund*	*infinitive*	*passive-participle*	*past-participle*	*present-participle*
(bse)	*(fin)*	*(ger)*	*(inf)*	*(pas)*	*(psp)*	*(prp)*

1.1 Anmerkungen zur Sortenhierarchie

Im Unterschied zum Buch haben wir Listen und Mengen nicht hinsichtlich der Sorte der Elemente parametrisiert, die in ihnen vorkommen dürfen. In den Feature Declarations im nächsten Abschnitt werden wir einfach festlegen, daß Listen und Mengen prinzipiell Objekte beliebiger Sorte enthalten können. In Abschnitt 2.1 kommen wir darauf erläuternd zurück.
In (4) wird *qfpsoa* als unmittelbare Untersorte von *content* behandelt im Unterschied zum Buch, wo an der gleichen Stelle die Sorte *psoa* steht. *psoa* können wir für unsere Belange ignorieren, da wir Quantoren nicht näher betrachtet haben, für deren Behandlung im Kapitel 8 des Buches die mit der Sorte *psoa* einhergehende Variante der Sortenhierarchie und Erweiterung der Feature Declarations benötigt und eingeführt wird.
Außerdem erlauben wir es uns, die umfangreiche (und auch im Buch nur fragmentarisch angedeutete) Hierarchie unter *qfpsoa* wegzulassen. So gäbe es etwa für *give* eine maximal spezifische Untersorte *give*, für *want* eine Sorte *want*, undsoweiter. Wenn wir diese (und dazu passende Feature Declarations) für unsere linguistischen Überlegungen benötigen, setzen wir deren Existenz einfach voraus. Entsprechendes gilt für die äußerst vereinfacht gehandhabte Phonologie mit Untersorten von *phoneme-string*, sowie für die Untersorten von *pform* und *semdet*.

2 Feature Declarations

Die Feature Declarations sind im folgenden alphabetisch angeordnet. Es wird jeweils angegeben, welches die höchste Sorte in der Hierarchie ist, für die ein Attribut angemessen ist, und welchen Wert die Angemessenheitsfunktion für diese Sorte und das jeweilige Attribut annimmt. Zum Beispiel ist *functional* die höchste Sorte der *head*-Hierarchie (6), so daß das Attribut SPEC für sie angemessen ist, und der Wert der Angemessenheitsfunktion auf *functional* und SPEC ist *synsem*. Der HPSG-Formalismus stellt sicher, daß die Angemessenheit von SPEC auf die Untersorten von *functional* vererbt wird.

category	HEAD	*head*
	SUBCAT	*list*
	MARKING	*marking*

c-inds	SPEAKER	*ref*
	ADDRESSEE	*ref*
	UTTERANCE-LOCATION	*ref*
	...	

context	BACKGROUND	*set*
	CONTEXTUAL-INDICES	*c-inds*
coord-struc	CONJ-DTRS	*set*
	CONJUNCTION-DTR	*word*
functional	SPEC	*synsem*
head-adjunct-struc	HEAD-DTR	*phrase*
	ADJUNCT-DTR	*sign*
	COMP-DTRS	*elist*
head-filler-struc	HEAD-DTR	*phrase*
	FILLER-DTR	*sign*
	COMP-DTRS	*elist*
head-mark-struc	HEAD-DTR	*phrase*
	MARKER-DTR	*word*
	COMP-DTRS	*elist*
head-struc	HEAD-DTR	*sign*
	COMP-DTRS	*list*
index	PERSON	*person*
	NUMBER	*number*
	GENDER	*gender*
local	CATEGORY	*category*
	CONTENT	*content*
	CONTEXT	*context*
nelist	FIRST	*object*
	REST	*list*
neset	FIRST	*object*
	REST	*set*
nom-obj	INDEX	*index*
	RESTR	*set*
nonlocal	TO-BIND	*nonlocal1*
	INHERITED	*nonlocal1*
nonlocal1	SLASH	*set*
	REL	*set*
	QUE	*set*
noun	CASE	*case*

phrase	DAUGHTERS	*con-struc*

preposition	PFORM	*pform*

quantifier	DET	*semdet*
	RESTIND	*npro*

sign	PHONOLOGY	*list*
	SYNSEM	*synsem*
	QSTORE	*set*
	RETRIEVED	*set*

substantive	PRD	*boolean*
	MOD	*mod-synsem*

synsem	LOCAL	*local*
	NONLOCAL	*nonlocal*

verb	VFORM	*vform*
	AUX	*boolean*
	INV	*boolean*

2.1 Bemerkungen und Varianten der Feature Declarations

2.1.1 *qfpsoa* und *psoa*

Die Feature Declarations der Untersorten von *qfpsoa* aus dem Buch wurden weggelassen. Wenn nötig setzen wir einfach voraus, daß zum Beispiel für die maximal spezifische Untersorte *give* von *qfpsoa* die Attribute GIVER, GIVEN und GIFT angemessen sind, wobei der Wert der Angemessenheitsfunktion jeweils *ref* ist. Für *demand* (mit Satzkomplement) ergäben sich andererseits die Attribute DEMANDER und (traditioneller Weise) SOA-ARG mit den Werten *ref* und *qfpsoa*. Da wir im Unterschied zum Appendix des Buches die Sortenhiearchie dahingehend vereinfacht haben, daß *qfpsoa* als Untersorte von *content* an die Stelle von *psoa* getreteten ist, können unsere Zeichen keine *psoa*-Objekte enthalten, und die Feature Declarations für *psoa* konnten weggelassen werden. Der Vollständigkeit halber seien sie aber hier erwähnt:

(19) Die Angemessenheitsbedingungen für *psoa* im Buch:

psoa	QUANTS	*list*
	NUCLEUS	*qfpsoa*

Diese Feature Declaration setzt natürlich voraus, daß die Sortenhierarchie im Unterschied zu unserer obigen die Sorte *psoa* überhaupt enthält. Im Appendix des HPSG-Buches ist das auch der Fall.

2.1.2 Parametrisierte Listen und Mengen

Die Buchautoren verwenden nicht einfach Listen und Mengen, wie wir das tun, sondern sogenannte parametrisierte Listen und Mengen. So werden zum Beispiel SUBCAT-Listen in ihrer Signatur nicht einfach als von der Sorte *list* angegeben, sondern als *list(synsem)*, das heißt, sämtliche Elemente dieser Listen müssen qua Signatur *synsem*-Objekte sein. Zum einen sind jedoch parametrisierte Listen ein formal gesehen recht kompliziertes Konzept. Zum anderen stellt sich bei näherem Hinsehen außerdem heraus, daß die Verwendung parametrisierter Listen in der Grammatik des Buches weitgehend eine Redundanz darstellt. So wie die Theorie der Grammatik formuliert ist, ergibt sich bereits unabhängig, daß beispielsweise Elemente von SUBCAT-Listen nur *synsem*-Objekte sein können. Aufgrund dieser Gegebenheiten haben wir darauf verzichtet, parametrisierte Listen und Mengen zu verwenden.
Sofern man wollte, könnte man den im Buch anvisierten Effekt von parametrisierten Listen durch die Verwendung von Relationen erzielen. Wenn wir die hier angegebene Signatur, die außerdem eine einstellige Relation `synsemliste` enthalte, voraussetzen, könnten wir ein Prinzip schreiben, das fordert, daß alle Elemente aller SUBCAT-Listen *synsem*-Objekte sind. Hierzu definieren wir zunächst `synsemliste` so, daß jedes Element ihres listenwertigen Arguments ein *synsem*-Objekt sein muß.

(20) Definition von `synsemliste` als Grammatikprinzip:

$$\forall x \left(\mathtt{synsemliste}(x) \leftrightarrow \left({}^{x}\left[\mathit{elist} \right] \vee \left({}^{x}\begin{bmatrix} \text{FIRST} & \mathit{synsem} \\ \text{REST} & \boxed{1} \end{bmatrix} \wedge \mathtt{synsemliste}(\boxed{1}) \right) \right) \right)$$

Außerdem schreiben wir noch folgendes Prinzip in die Theorie der Grammatik:

(21) *synsem*-Objekte auf SUBCAT-Listen:

$$\mathit{category} \rightarrow \left[\text{SUBCAT}\ \boxed{1} \right] \wedge \mathtt{synsemliste}(\boxed{1})$$

2.1.3 Mengen

Die Autoren des HPSG-Buches äußern sich nicht präzise dazu, wie sie sich die Behandlung von Mengen in einer Merkmalslogik vorstellen. So führen sie zwar die Mengensorten *set, eset* und *neset* im Appendix ein, sagen aber an anderer Stelle, daß sie diese nicht als Sortensymbole betrachten wollen.
Um mit mengenwertigen Merkmalen einfach und dennoch präzise zu arbeiten, haben wir uns dafür entschieden, *set, eset* und *neset* als normale Sorten zu behandeln, und haben darüber hinaus der *set*-Hierarchie mit den für die Sorte *neset* deklarierten Attributen FIRST (mit Wert *object*) und REST (mit Wert *set*) geeignete Feature Declarations gegeben. Daneben haben wir für die dadurch eingeführten Mengenobjekte Zusatzannahmen verabredet. Im Unterschied zu den Gegebenheiten bei Listen interessieren wir uns nicht für die Reihenfolge, in der die Elemente von Mengen in der durch die Interpretation der REST- und FIRST-Attribute gegebenen Anordnung stehen, und ein Element darf niemals zweimal in einer Menge auftreten. Zyklische Mengen gibt es nicht. Die letzten beiden Verabredungen ließen sich übrigens auch mit Hilfe von Grammatikprinzipien erzwingen.

2.1.4 Zeichen anstelle von Phrasen als Töchter

Wir hatten festgestellt, daß es mit der Forderung von Pollard und Sag, daß Adjunkttöchter, Fillertöchter und Komplementtöchter immer Phrasen sein sollen, Probleme gibt, da aufgrund der gegebenen ID-Schemata viele naheliegende Kandidaten für solche Töchter ausgeschlossen werden: Sie müßten zu Phrasen projizieren, können das aber nicht. Wir haben eine naheliegende Lösung dieses Problems gewählt, indem wir die Feature Declarations so verändert haben, daß die genannten Töchter sowohl Wörter als auch Phrasen sein dürfen. Man beachte, daß sich das für die Komplementtöchter schon alleine daraus ergibt, daß wir nicht mit parametrisierten Listen arbeiten. Daß die Komplementtöchter aber immerhin Zeichen sein müssen, ergibt sich bei der vorliegenden Grammatik des Englischen bereits aus dem im Subcategorization Principle festgelegten Verhältnis von SUBCAT-Listen zu Komplementtöchtern.

2.1.5 Alternativen zur SUBCAT-Liste

In weiten Teilen der HPSG-Literatur werden Varianten der Signatur des HPSG-Buches verwendet. Eine prominente Änderung besteht darin, die SUBCAT-Liste in mehrere Valenzlisten aufzuspalten, was bereits im neunten Kapitel des Buchs von Pollard und Sag diskutiert wird. Zum einen werden die Angemessenheitsbedingungen für *category* so geändert, daß anstelle von SUBCAT das Attribut VALENCE mit dem neuen Wert *valence* definiert wird. *valence* ist eine maximal spezifische unmittelbare Untersorte von *object*. Zum anderen werden Feature Declarations für *valence* eingeführt:

(22) Variante der Subkategorisierung (*valence*):

category	HEAD	*head*
	VALENCE	*valence*
	MARKING	*marking*

valence	SUBJ	*list*
	COMPS	*list*
	SPR	*list*

Es ist klar, daß diese Modifikation unter anderem zu relativ naheliegenden Anpassungen beim Subcategorization Principle und, je nach Ausführung der Idee, bei den ID-Schemata 1 bis 3 und den Untersorten von *headed-structure* führt. Damit setzen wir uns jedoch im Rahmen dieses Seminars nicht weiter auseinander.

2.1.6 Relationen

Eine vollständige Signatur muß nicht nur die Sortenhierarchie, die Attributsnamen und die Feature Declarations spezifizieren, sie muß auch sämtliche Relationsnamen und die Stelligkeit der einzelnen Relationen angeben. Mit dieser Anforderung sind wir wie die Autoren des Buches etwas nachlässig umgegangen und haben bei Bedarf immer einfach vorausgesetzt, daß die benötigten Relationen mit der richtigen Stelligkeit in der Signatur deklariert sind. Beispiele sind etwa `append/3`, `local-o-command/2`, `member/2` und `sign-to-synsem/2` (wobei die den Relationsnamen beigegebene Zahl die Stelligkeit der Relation angibt).