Grammatiken als Logikprogramme

kontextfreie Grammatikregeln:

non-terminal $\rightarrow$ body, z.B.

s $\rightarrow$ np, vp.	det $\rightarrow$ [the].
np $\rightarrow$ det, noun.	noun $\rightarrow$ [cat].
vp $\rightarrow$ trans_verb, np.	noun $\rightarrow$ [dog].
vp $\rightarrow$ intrans_verb.	intrans_verb $\rightarrow$ [sleeps].
	trans_verb $\rightarrow$ [chases].

beschreiben eine *formale Sprache*.

($\rightarrow$ ist hier nicht mit der logischen Implikation zu verwechseln!)

Der Rumpf (die rechte Seite) einer Regel kann aus einem Terminalsymbol (entsprechend den Wörtern einer natürlichen Sprache) oder einer Folge von Nichtterminalsymbolen (entsprechend den Wortklassen/–arten) bestehen.

Eine *Grammatik* besteht aus den endlichen Mengen der Terminalen, der Nichtterminalen und der Regeln, sowie einem der Nichtterminalen, dass als Startsymbol ausgezeichnet wird, hier $\mathtt{s}$.

Ein *Wort* einer von einer Grammatik erzeugten Sprache ist eine Folge von Terminalsymbolen, die durch Anwendung der Regeln aus dem Startsymbol abgeleitet werden können.

Grammatiken als Logikprogramm

kf. Regel in Klauselform

(die S sind Variablen für Wortpositionen):

```
sentence(S0,S) :- np(S0, S1), vp(S1, S).
np(S0,S) :- det(S0, S1), noun(S1, S).
...
```

Das Lexikon:

```
det(S0,S) :- connects(S0, the, S).
noun(S0,S) :- connects(S0, cat, S). ...
```

Der Satz: “the cat chases the dog” wird codiert als:

```
connects(1,the,2).
connects(2,cat,3).
connects(3,chases,4).
connects(4,the,5).
connects(5,dog,6).
```

Test, ob der Satz grammatikalisch ist:

```
?-sentence(1,6).
```

Definite Clause Grammars

Kontextfreie Grammatiken für natürliche Sprachen nicht adäquat (...)

Erweiterung der kf. Grammatikregeln:

- Nicht-Terminale sind komplexe Terme mit Variablen: `np(X, S)` oder `s(S)`

  Zusätzliche Argumente dienen zum Aufbau des Parsebaumes.
- Auf der rechten Regelseite sind Funktionsaufrufe erlaubt:

  noun(N) $\rightarrow$ [W], $\{$rootform(W,N), is_noun(N)$\}$.

Als Logikprogramm:

```
noun(N,S0,S):- connects(S0,W,S), rootform(W,N),
is_noun(N).
```

Diese Transformationen können von den meisten PROLOG-Interpretern automatisch durchgeführt werden.

Erweiterungen

Lexikon

Statt

noun(N) $\rightarrow$ [dog]

für jedes Wort der Kategorie `noun':

noun(N) $\rightarrow$ [N], $\{$is_noun(N)$\}$

und is_noun(...) für alle Wörter.

(Zusätzlichge Argumente und Prozeduren möglich)

Kontextabhängigkeit: **Kongruenz**

Einfügen eines zusätzlichen Arguments für jede Eigenschaft, die kongruieren soll:

is_determiner(every, singular)

is_determiner(all, plural)

– ebenso in allen Grammatikregeln, in denen Artikel vorkommen, Kongruenz wird durch Verwendung von Variablen erzwungen:

noun_phrase(Number, np(Det,Noun)) $\rightarrow$ determiner(Number, Det), noun(Number, Noun).

Termunifikation in der Prädikatenlogik

Um das Resolutionskalkül auf die Prädikatenlogik anzuwenden, müssen nicht nur die Formeln in KNF umgeformt, sondern auch geeignete Substitutionen für die Variablen ausgeführt werden, um die Literale möglichst zu vereinfachen. Dieser Vorgang heisst *Termunifikation*.

Def: *(allgemeinster) Unifikator*

Eine Substitution sub ist ein Unifikator einer endlichen Menge von Literalen **L** $= \{L_1, ..., L_k\}$, wenn $L_1 sub = L_2 sub = ... = L_k sub$.

sub ist ein allgemeinster Unifikator von **L**, wenn für jeden anderen Unifikator sub' gilt: es gibt s mit $sub' = sub\ s$ (d.h. für alle Formeln F gilt $Fsub' = Fsubs$)

Beispiel:

$\{P(x), P(f(y))\}$ wird durch $[x/f(y)]$ unifiziert.

Unifikationsalgorithmus:

Eingabe: nichtleere Literalmenge **L**

$sub := []$;

L$sub := \{L_1 sub, ..., L_k sub\}$;

solange $|$ **L**sub $| > 1$:

- Durchlaufe **L**sub, bis sich (von links nach rechts gelesen) zwei Literale darin in mindestens einem Zeichen unterscheiden.

  Wenn keines der beiden Zeichen eine Variable ist, stoppe mit “nicht unifizierbar”.

  sonst:

  sei x die Variable, t der andere Term; wenn x in t vorkommt, stoppe mit “nicht unifizierbar”.

  sonst:

  $sub := sub[x/t]$;

  L$sub := \{L_1 sub, ..., L_k sub\}$

Gib sub als allgemeinsten Unifikator aus.

Probleme der prädikatenlogischen **Resolution**: Explosion der Kombinationsmöglichkeiten, Auswahl der Substitutionen, etc.
Abhilfe: Strategien, Restriktionen: Bedingungen, unter denen zwei Klauseln resolviert werden können.

Merkmallogik

Grundmenge: Merkmalstrukturen (*feature structures*), Mengen aus Namen (*labels*) (für Merkmale, Attribute) und deren Werten.

Formal: partielle Funktion von Merkmalen auf Werte.

Notation: $[$ *name: wert* $]$

- Werte selbst können wieder Merkmalstrukturen sein:

$$\begin{bmatrix} \textit{cat:} & np \\ \textit{agreement:} & \begin{bmatrix} \textit{person:} & 3 \\ \textit{number:} & S \end{bmatrix} \end{bmatrix}$$

- dieselben Werte können von mehreren Attributen geteilt werden, dh., es kann auf einen einmal definierten Wert verwiesen werden (*reentrance, structure sharing*, Koreferenz, Koindizierung):

$$\begin{bmatrix} \textit{f:} & \boxed{1} & \begin{bmatrix} \textit{person:} & 3 \\ \textit{number:} & S \end{bmatrix} \\ \textit{g:} & \boxed{1} & \end{bmatrix}$$

Dabei sind zirkuläre Verweise nicht erlaubt!

Grundbegriffe

atomare Merkmalstrukturen:

einfache Symbole wie 3, S, np...

komplexe Ms.: $[$*name: wert*$]$

leere Ms., auch *Variable*: $[\,]$

Extraktion:

Sei D eine Ms., f eines ihrer Attribute. Dann bezeichnet $D(f)$ den Wert von f in D.

Bsp.: $D = \begin{bmatrix} \textit{person:} & 3 \\ \textit{number:} & S \end{bmatrix}$, dann D(*person*) $= 3$.

Definitions- und Wertebereich wie bei Funktionen:

Definitionsbereich, Urbild: dom(D) = $\{$*person*, *number*$\}$

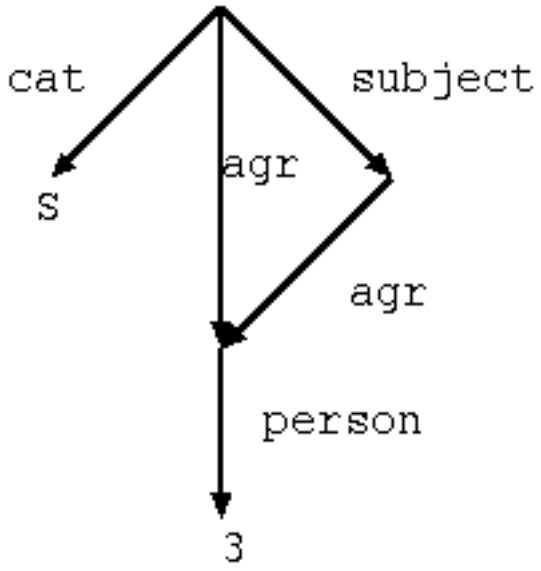
Wertebereich, Bild: range(D) = $\{3, S\}$

Ein *Pfad* innerhalb einer Ms. ist eine Folge von Attributen: $\langle$ *agreement person* $\rangle$

Die Extraktion kann auch auf Pfade angewendet werden...

Merkmalstrukturen als Graphen

$$\begin{bmatrix} \textit{cat:} & S \\ \textit{subject:} & \begin{bmatrix} \textit{agreement:} & \boxed{1} \end{bmatrix} \begin{bmatrix} \textit{person:} & 3 \end{bmatrix} \\ \textit{agreement:} & \boxed{1} \end{bmatrix}$$



(Abb. geborgt von Martin Volk)

Merkmalstrukturen entsprechen den Knoten, Merkmalsnamen den Kanten.

Subsumption

Ordnungsrelation $\sqsubseteq$:

$D \sqsubseteq D'$, wenn D weniger (oder dieselbe) Information enthält als D'. D ist allgemeiner, D' spezieller.

Shieber:

$D \sqsubseteq D'$ gdw. für alle $l \in \text{dom}(D) : D(l) \sqsubseteq D'(l)$
und für alle Pfade p, q mit $D(p) = D(q) : D'(p) = D'(q)$

D subsummiert D' (D' wird von D subsummiert), wenn für alle Attribute in D gilt, dass ihre Werte von denen in D' subsummiert werden.

Für atomare Strukturen gilt: $a \sqsubseteq b$ und $b \sqsubseteq a$ gdw. $a = b$

Die leere Struktur subsummiert alle anderen.

Beispiele:

$$[\,] \sqsubseteq \left[\, \textit{cat:}\ np \,\right] \sqsubseteq \left[\begin{array}{ll} \textit{cat:} & np \\ \textit{agreement:} & [\textit{genus:}\ mas] \end{array}\right]$$

Mit der Relation $\sqsubseteq$ bilden Ms. einen Verband!

Unifikation

Unifikation bildet aus zwei *verträglichen* Merkmalstrukturen eine neue, die von beiden subsummiert wird:

$D = D' \sqcup D''$ gdw.
D ist die allgemeinste Struktur mit:
$D' \sqsubseteq D$ und $D'' \sqsubseteq D$

Beispiel:

$$[\textit{number:}S] \sqcup [\textit{person:}3] = \begin{bmatrix} \textit{person:} & 3 \\ \textit{number:} & S \end{bmatrix}$$

Unifikation zwischen nicht verträglichen Strukturen scheitert, d.h. liefert den Wert *fail* oder $\perp$.

Beispiel:

$$[\textit{number:}S] \sqcup [\textit{number:}P] = \textit{fail}$$

Eigenschaften der Unifikation

neutrales Element:

$[\,] \sqcup D = D \sqcup [\,] = D$

Die Verbandseigenschaften:

1. Kommutativität:
   $D \sqcup E = E \sqcup D$
2. Assoziativität:
   $(D \sqcup E) \sqcup F = D \sqcup (E \sqcup F)$
3. Idempotenz:
   $D \sqcup D = D$

Unterschiede zur Termunifikation:

- Mengen vs. N-Tupel, dh. Reihenfolge in der Termunifikation entscheidend.
- Stelligkeit: eine Eigenschaft wie Kongruenz müßte in Prädikatenlogik mit fester Anzahl von Parametern definiert werden.
- Identität und Gleichheit lassen sich in Merkmalstrukturen ausdrücken...

Unifikation mit Identität und Gleichheit

bei Identität (Koreferenzierung)

$$\begin{bmatrix} \textit{agreement:} \boxed{1} & [\textit{number:}\ S] \\ \textit{subject:} & [\textit{agreement:} \boxed{1}] \end{bmatrix}$$

$$\sqcup\ [\textit{subject:}\ [\textit{agreement:}\ [\textit{person:}\ 3]\ ]\ ]$$

$$= \begin{bmatrix} \textit{agreement:} \boxed{1} & \begin{bmatrix} \textit{number:}\ S \\ \textit{person:}\ 3 \end{bmatrix} \\ \textit{subject:} & [\textit{agreement:} \boxed{1}] \end{bmatrix}$$

bei Gleichheit:

$$\begin{bmatrix} \textit{agreement:} & [\textit{number:}\ S] \\ \textit{subject:} & [\textit{agreement:}[\textit{number:}\ S]\ ] \end{bmatrix}$$

$$\sqcup\ [\textit{subject:}\ [\textit{agreement:}\ [\textit{person:}\ 3]\ ]\ ]$$

$$= \begin{bmatrix} \textit{agreement:} & \begin{bmatrix} \textit{number:}\ S \end{bmatrix} \\ \textit{subject:} & \begin{bmatrix} \textit{agreement:} & \begin{bmatrix} \textit{number:}\ S \\ \textit{person:}\ 3 \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Unifikation, Erweiterungen

Disjunktion

aufwendig...

Mengen

Negation

zwei Arten: closed/open World

Typisierte Merkmalstrukturen

Wohlgeformtheit

Typenhierarchien...