Einführung in die Computerlinguistik

Merkmalsstrukturen

WS 2009/2010

Manfred Pinkal

Earley-Algorithmus

- Für einen Eingabesatz der Länge n wird eine Chart mit n+1 Spalten eingerichtet.
- Einträge bestehen aus zwei Informationen, die Zusammen ein (potentiell unvollständiges) Parseresultat kodieren:
  - „Punktierte Regel": Eine Regel A→u, bei der die Symbole auf der rechten Seite durch einen Punkt getrennt sind
  - Paar [i, j], das einen Teilstring der Eingabekette bezeichnet.
- Beispiel: <S → NP • VP, [0, 2]>
  - Wenn die Regel S → NP VP auf den Teil der Eingabe angewandt wird, der an Position 0 beginnt, kann an Position 2 der NP-Teil der Regel vollständig abgearbeitet sein.

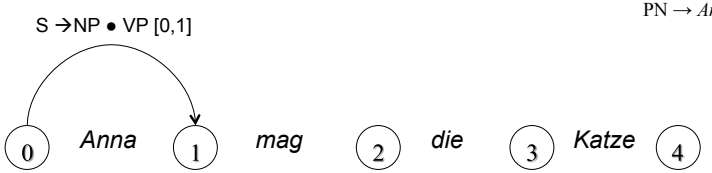
Chart, graphische Darstellung

S → NP VP
NP → Det N
NP → PN
VP → V NP
Det → *die*
N → *Katze*
V → *mag*
PN → *Anna*

Chart, graphische Darstellung

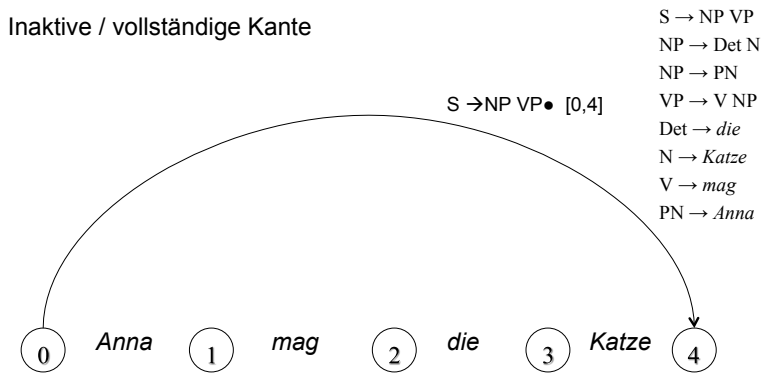
Aktive/ unvollständige Kante:

S → NP VP
NP → Det N
NP → PN
VP → V NP
Det → *die*
N → *Katze*
V → *mag*
PN → *Anna*



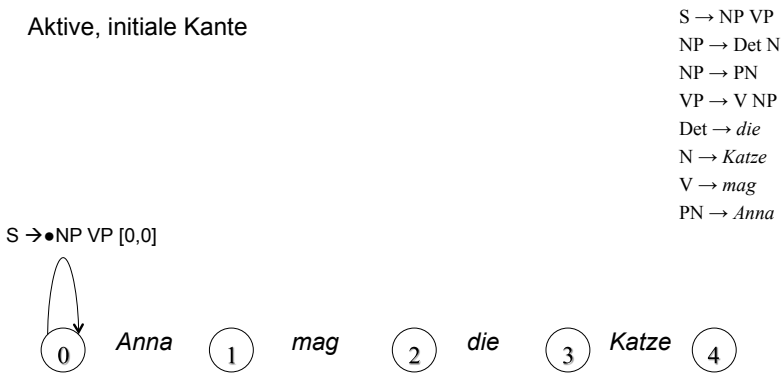
Chart, graphische Darstellung

Inaktive / vollständige Kante



Beispiel-Chart

Aktive, initiale Kante



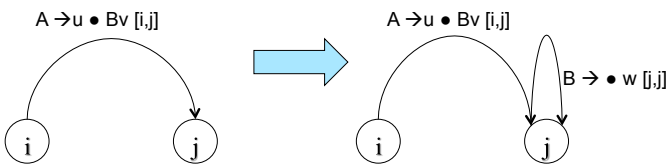
Earley-Algorithmus – Vorgehen

- Es seien u, v ∈ V*: (möglicherweise leere) Ketten von (Terminal- oder Nicht-Terminal-)Symbolen; P punktierte Erzeugungsregeln.
- Initialisiere die erste Spalte der Chart mit <S → • u, [0, 0]> für jede Regel S → u der Grammatik.
- Gehe schrittweise von links nach rechts durch die Chart. In jedem Schritt j:
  - Wende für jeden Eintrag <P, [i, j]> eine der folgenden Operationen an:
  - Wenn P unvollständig und von der Form A → u • av ist (a Terminalsymbol) : Scan
  - Wenn P unvollständig und von der Form A → u • Bv ist (B Nicht-Terminalsymbol) : Predict
  - Wenn P vollständig (von der Form A → u •) ist: Complete.

Earley-Algorithmus – Predictor

- Für Position j, Eintrag <P, [i, j]>:
  - Wenn P unvollständig und von der Form A → u • Bv ist (B Nicht-Terminalsymbol) :
  - Füge für jede Regel B → w der Grammatik <B → • w, [j, j]> zur Position j hinzu.
- Die punktierte Regel A → u • Bv im Eintrag drückt aus, dass als nächstes eine Konstituente der Kategorie B kommen könnte. Der Predictor schreibt alle Regeln in die Chart, die auf der linken Seite ein B haben könnten.

Predictor, Schema

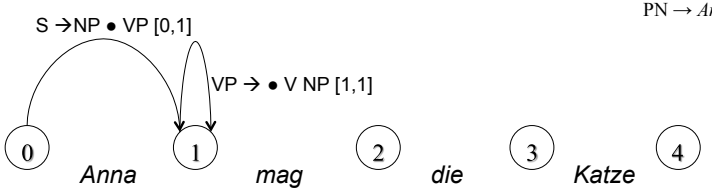


Predict, Beispiel

S → NP VP
NP → Det N
NP → PN
VP → V NP
Det → *die*
N → *Katze*
V → *mag*
PN → *Anna*

Predict, Beispiel

S → NP VP
NP → Det N
NP → PN
VP → V NP
Det → *die*
N → *Katze*
V → *mag*
PN → *Anna*

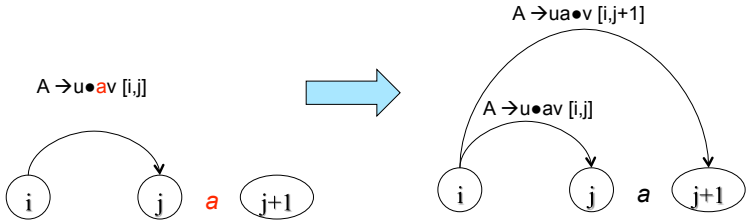


Earley-Algorithmus – Scanner

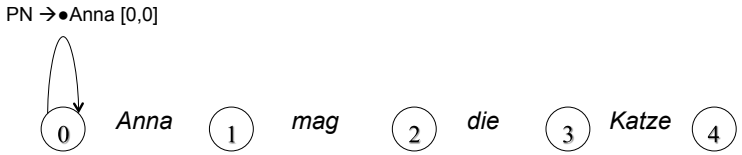
- Für Position j, Eintrag <P, [i, j]>:
  - Wenn P unvollständig und von der Form A → u ● av ist (a Terminalsymbol) , und die Position [j, j+1] der Eingabe mit a gefüllt ist:
  - Füge <A → ua●v , [i, j+1]> zur Spalte j+1 der Chart hinzu.

- Die punktierte Regel A → u ● av drückt aus, dass als nächstes das Eingabesymbol a erwartet wird. Der Scanner liest das Eingabewort und gleicht es mit der Regel ab. Im Erfolgsfall wird der Punkt über a hinweg geschoben und ein neuer Eintrag in der nächsten Spalte der Chart gemacht.

Scanner, Schema

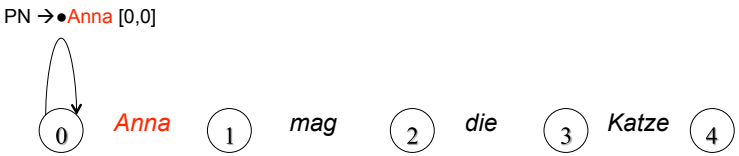


Scan: Beispiel



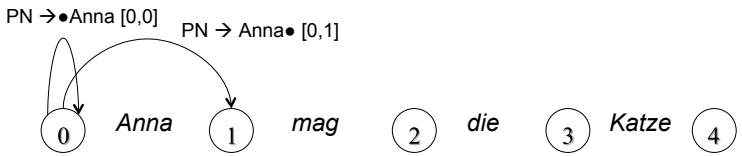
Scan: Beispiel

S → NP VP
NP → Det N
NP → PN
VP → V NP
Det → *die*
N → *Katze*
V → *mag*
PN → *Anna*



Scan: Beispiel

S → NP VP
NP → Det N
NP → PN
VP → V NP
Det → *die*
N → *Katze*
V → *mag*
PN → *Anna*

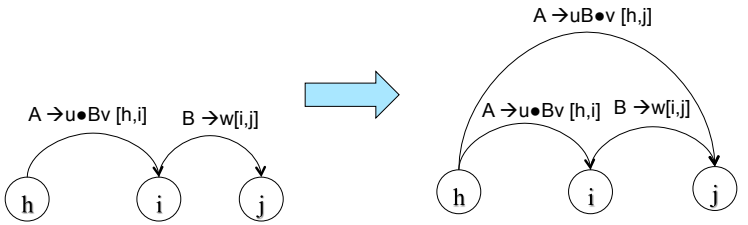


Earley-Algorithmus – Completer

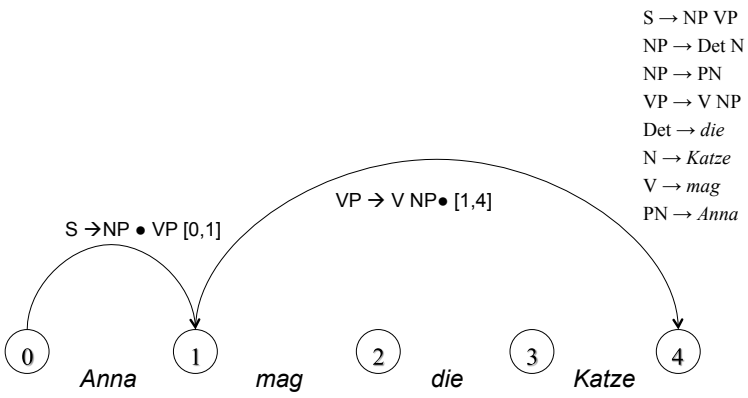
- Für Position j, Eintrag <P, [i, j]>:
  - Wenn P vollständig und von der Form B → u:
  - Füge für jeden bestehenden Eintrag <A → u●Bv , [h, i]> einen neuen Eintrag <A → uB●v , [h, j]> hinzu.

- B → u ● besagt, dass die rechte Seite der Regel vollständig abgearbeitet wurde. Das Resultat kann verwendet werden, um eine bereits bestehende partielle Analyse, die als nächstes einen Ausdruck der Kategorie B erwartet, zu komplettieren.

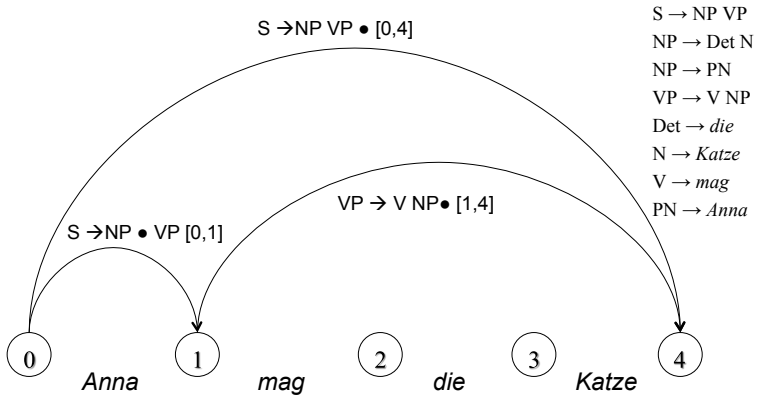
Completer, Schema



Complete, Beispiel



Complete, Beispiel



Earley-Algorithmus: Abschluss

- Die Eingabe ist grammatisch, wenn der Algorithmus einen Eintrag <S → u● , [0, n]> erzeugt.
- Auf den folgenden Folien wird die Chart aufgebaut für die obige Beispielgrammatik und den Beispielsatz

Anna mag die Katze

Beispiel-Chart: Initialisierung

Position 0	Position 1	Position 2	Position 3	Position 4
S → ●NP VP, [0,0]				

S → NP VP
NP → Det N
NP → PN
VP → V NP
Det → *die*
N → *Katze*
V → *mag*
PN → *Anna*

Beispiel-Chart: Operationen auf Pos. 0

Position 0	Position 1	Position 2	Position 3	Position 4
S → ●NP VP, [0,0]	Scanner:			
Predictor: NP → ●Det N, [0,0]	PN → *Anna*●, [0,1]			
NP → ●PN, [0,0]				
Det → ●*die*, [0,0]				
PN → ●*Anna*, [0,0]				

S → NP VP
NP → Det N
NP → PN
VP → V NP
Det → *die*
N → *Katze*
V → *mag*
PN → *Anna*

Beispiel-Chart: Operationen auf Pos. 1

Position 0	Position 1	Position 2	Position 3	Position 4
S → ●NP VP, [0,0]	Scanner:	Scanner:		
Predictor: NP → ●Det N, [0,0]	PN → *Anna*●, [0,1]	V → *mag*●, [1,2]		
NP → ●PN, [0,0]	Completer:			
Det → ●*die*, [0,0]	NP → PN●, [0,1]			
PN → ●*Anna*, [0,0]	S → NP●VP, [0,1]			
	Predictor:			
	VP → ●V NP, [1,1]			
	V → ●*mag*, [1,1]			

S → NP VP
NP → Det N
NP → PN
VP → V NP
Det → *die*
N → *Katze*
V → *mag*
PN → *Anna*

Beispiel-Chart: Operationen auf Pos. 2

Position 0	Position 1	Position 2	Position 3	Position 4
S → •NP VP, [0,0]	Scanner:	Scanner:	Scanner:	
Predictor: NP → •Det N, [0,0]	PN → *Anna*•, [0,1]	V → *mag*•, [1,2]	Det → *die*•, [2,3]	
NP → •PN, [0,0]	Completer:	Completer:		
Det → •*die*, [0,0]	NP → PN•, [0,1]	VP → V•NP, [1,2]		
PN → •*Anna*, [0,0]	S → NP•VP, [0,1]	Predictor:		
	Predictor:	NP → •Det N, [2,2]		
	VP → •V NP, [1,1]	NP → •PN, [2,2]		
	V → •*mag*, [1,1]	Det → •*die*, [2,2]		
		PN → •*Anna*, [2,2]		

S → NP VP
NP → Det N
NP → PN
VP → V NP
Det → *die*
N → *Katze*
V → *mag*
PN → *Anna*

Beispiel-Chart: Operationen auf Pos. 4

Position 0	Position 1	Position 2	Position 3	Position 4
S → •NP VP, [0,0]	Scanner:	Scanner:	Scanner:	Scanner:
Predictor: NP → •Det N, [0,0]	PN → *Anna*•, [0,1]	V → *mag*•, [1,2]	Det → *die*•, [2,3]	N → Katze•, [3,4]
NP → •PN, [0,0]	Completer:	Completer:	Completer:	Completer:
Det → •*die*, [0,0]	NP → PN•, [0,1]	VP → V•NP, [1,2]	NP → Det•N, [2,3]	NP → Det N•, [2,4]
PN → •*Anna*, [0,0]	S → NP•VP, [0,1]	Predictor:	Predictor:	VP → V NP•, [1,4]
	Predictor:	NP → •Det N, [2,2]	N → •Katze, [3,3]	S → NP VP•, [0,4]
	VP → •V NP, [1,1]	NP → •PN, [2,2]		
	V → •*mag*, [1,1]	Det → •*die*, [2,2]		
		PN → •*Anna*, [2,2]		

S → NP VP
NP → Det N
NP → PN
VP → V NP
Det → *die*
N → *Katze*
V → *mag*
PN → *Anna*

Beispiel-Chart: Operationen auf Pos. 4

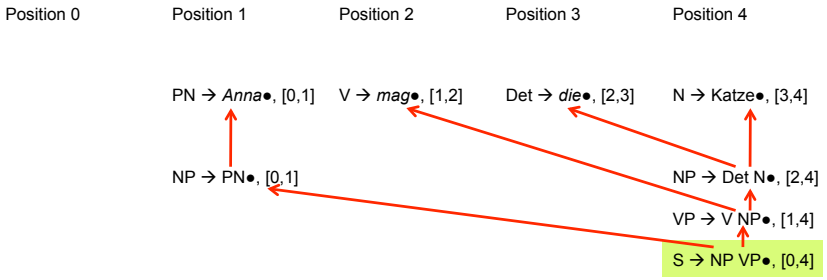
Position 0	Position 1	Position 2	Position 3	Position 4
S → •NP VP, [0,0]	Scanner:	Scanner:	Scanner:	Scanner:
Predictor: NP → •Det N, [0,0]	PN → *Anna*•, [0,1]	V → *mag*•, [1,2]	Det → *die*•, [2,3]	N → Katze•, [3,4]
NP → •PN, [0,0]	Completer:	Completer:	Completer:	Completer:
Det → •*die*, [0,0]	NP → PN•, [0,1]	VP → V•NP, [1,2]	NP → Det•N, [2,3]	NP → Det N•, [2,4]
PN → •*Anna*, [0,0]	S → NP•VP, [0,1]	Predictor:	Predictor:	VP → V NP•, [1,4]
	Predictor:	NP → •Det N, [2,2]	N → •Katze, [3,3]	S → NP VP•, [0,4]
	VP → •V NP, [1,1]	NP → •PN, [2,2]		
	V → •*mag*, [1,1]	Det → •*die*, [2,2]		
		PN → •*Anna*, [2,2]		

Earley-Algorithmus

- Der Eintrag S → NP VP• , [0, 4] zeigt, dass die analysierte Wortkette ein in unserer Beispielgrammatik grammatischer Satz ist.
- Wenn wir die vollständigen Regeleinträge der Chart so miteinander verlinken, dass ein Eintrag auf die Einträge verweist, auf denen er aufbaut, können wir außerdem die syntaktische Struktur / den Parsebaum ablesen.

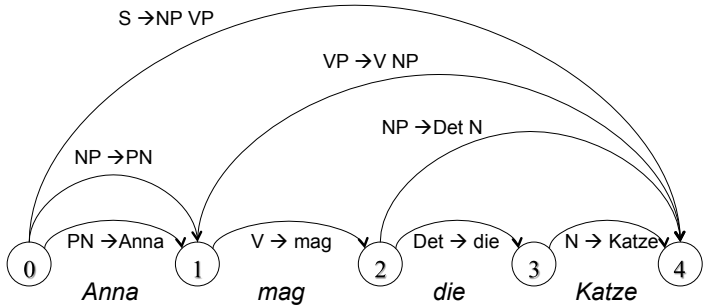
Beispiel-Chart

- Auf vollständige Einträge beschränkt, Abhängigkeiten sind markiert



Beispiel-Chart

- Alternative grafische Repräsentation der vollständigen Einträge



Grammatische Merkmale

- Wie finden Sie die angehängten Bilder? Das sind Fotos, die im Rahmen des TALK-Projektes entstanden sind, uns gehören, und von BMW schon freigegeben waren. Außerdem vermitteln sie besser den Bezug zur Forschung.

Merkmalsabhängigkeiten

- Grammatische Merkmale von Ausdrücken in der syntaktischen Struktur hängen in systematischer Weise voneinander ab.
- Die grundlegenden Typen solcher Beziehungen sind
  - Kongruenz und
  - Rektion oder Subkategorisierung

Kongruenz

- Kongruenz ist die Übereinstimmung von zwei oder mehreren Ausdrücken in Genus, Numerus, Kasus, Person, ...
  - Nominalkongruenz innerhalb der NP zwischen Artikel, Nomen, Adjektiv, Relativpronomen: *die[pl] angehängten[pl] Bilder[pl]*
  - Subjekt-Verb-Kongruenz: sie*[pl] vermitteln[pl]*
  - Pronominalkongruenz zwischen einem „anaphorischen“ Pronomen und der NP, auf die er sich bezieht

    Fotos*[pl] ... sie[pl]*

Subkategorisierung/ Rektion

- Von Rektion oder Subkategorisierung spricht man, wenn ein lexikalischer Kopf Argumente mit bestimmten grammatischen Eigenschaften verlangt.
- Subkategorisierung/ Rektion von Verben
  - *Sie vermitteln den Bezug[NP im Akkusativ]*
  - *Die Bilder gefallen dem Betrachter [NP im Dativ]*
  - *Sie erinnern uns [NP im Akkusativ] an den Urlaub [PP mit Akkusativ]*
- Präpositionen
  - *um das Haus*
  - *bei dem Haus*
  - *wegen des Hauses*
- Adjektive
  - *an computerlinguistischen Fragestellungen interessiert*
  - *einem Baum ähnlich*

Grammatische Merkmale in der CFG

- Beispielgrammatik:

  S → NP VP VP → VT NP
  VP → VI NP → DET N

  VI → *schläft* | *arbeitet*
  VT → *kennt* | *studiert*
  N → *Student* | *Studentin* | *Studenten* | *Studentinnen* | *Fach*
  DET → *der* | *die* | *das* | *den*

- Massive Übergenerierung ohne Berücksichtigung von Kongruenz und Rektion:
  - *die Studenten arbeitet*
  - *der Studentin arbeiten*
  - *der Student kennt der Student*
- Wie können Merkmale in der CFG berücksichtigt werden?

Versuch: Verfeinerung der Kategorien

- Beispielgrammatik:

S → NPSgNom VPSg	S → NPPlNom VPPl
VPSg → VISg	VPPl → VIPl
VPSg → VTSg NPAkk	VPPl → VTPl NPAkk
NPSgNom→ DETSgNomM NSgNomM	NPSgNom→ DETSgNomF NSgNomF
NPPlNom → DETPlNom NPlNom	
DETSgNomM → *der*	DETSgNomF → *die*
NSgNomM → *Student*	NSgNomF → *Studentin* *etc.*

- Nachteile:
  - Information aus dem Lexikon (Kategorie und inhärente Merkmale) und aus der morphologischen Analyse (Variable Merkmale) müssen zu neuen Kategorien fusioniert werden.
  - Regularitäten können nicht ausgedrückt werden
  - Das Regelsystem wird aufgebläht (2 Numeri x 3 Genera x 4 Kasus x 3 Personen x ...)

Explizite Kodierung von Merkmalen

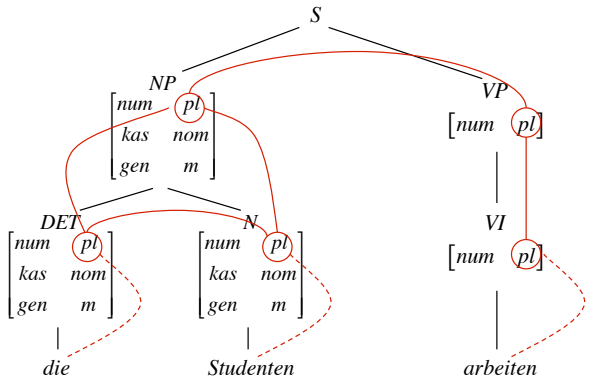
$$S \rightarrow NP\begin{bmatrix} num & sg \\ kas & nom \end{bmatrix} VP[num \quad sg] \qquad S \rightarrow NP\begin{bmatrix} num & pl \\ kas & nom \end{bmatrix} VP[num \quad pl]$$

$$VP[num \quad sg] \rightarrow VI[num \quad sg] \qquad VP[num \quad pl] \rightarrow VI[num \quad pl]$$

$$NP\begin{bmatrix} num & sg \\ kas & nom \\ gen & m \end{bmatrix} \rightarrow Det\begin{bmatrix} num & sg \\ kas & nom \\ gen & m \end{bmatrix} N\begin{bmatrix} num & sg \\ kas & nom \\ gen & m \end{bmatrix}$$

$$NP\begin{bmatrix} num & pl \\ kas & nom \\ gen & m \end{bmatrix} \rightarrow Det\begin{bmatrix} num & pl \\ kas & nom \\ gen & m \end{bmatrix} N\begin{bmatrix} num & pl \\ kas & nom \\ gen & m \end{bmatrix}$$

Explizite Kodierung von Merkmalen



Kontextfreie Grammatik mit Merkmalsstrukturen

- Konstituenten werden mit Paaren aus Kategoriensymbolen und Merkmalsstrukturen ausgezeichnet.
- Eine Merkmalsstruktur ist eine Menge von Merkmal-Wert-Paaren (auch „Attribut-Wert-Paaren“): Die Merkmalsstruktur des NP-Knotens im Beispiel hat drei Merkmale, das erste besteht aus dem Attribut *„num“* und dem atomaren Wert *„sg“*.
- Die explizite Kodierung von Merkmalen erlaubt die Formulierung von Bedingungen / Constraints, z.B. „Numerus von NP und Numerus von VP sind identisch“, oder „Subjekts-NP hat Kasus Nominativ“.
- Regeln der Grammatik sind zweiteilig: Sie bestehen aus einer Ersetzungsregel (wie üblich über Kategorien und lexikalische Ausdrücke formuliert) und einer Menge von Constraints über Merkmalsstrukturen.

CFG mit Merkmalsconstraints, Beispiel

S → NP VP
Numerus der NP = Numerus der VP
Kasus der NP = nom

VP → VI
Numerus der VP = Numerus von VI

VP → VT NP
Numerus der VP = Numerus von VT
Kasus der NP = akk

NP → DET N
Numerus von DET = Numerus von N
Genus von DET = Genus von N
Kasus von DET = Kasus von N
Numerus der NP = Numerus von N
Genus der NP = Genus von N
Kasus der NP = Kasus von N

VI → *arbeitet*
Numerus von VI = sg

VI → *arbeiten*
Numerus von VI = pl

N → Student
Numerus von N = sg
Genus von N = m
Kasus von N = nom

DET → *der*
Numerus von DET = sg
Genus von DET = m
Kasus von DET = nom

Merkmalstrukturen: Erste Erweiterung

Die Notation von Constraints lässt sich stark vereinfachen, wenn gleichzeitig auf Mengen von Merkmalen Bezug nehmen kann. Wir erlauben komplexe Merkmalsstrukturen, in denen Attribute nicht nur atomar Werte, sondern auch Merkmalsstrukturen als Werte haben können. Beipiel:

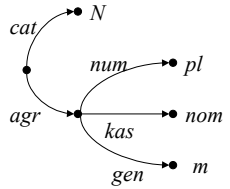
$$\begin{bmatrix} agr & \begin{bmatrix} num & pl \\ kas & nom \\ gen & m \end{bmatrix} \end{bmatrix}$$

„agr“ für englisch „agreement“ (Kongruenz) nimmt als Wert eine Merkmalsstruktur, die die Kongruenzmerkmale spezifiziert. Wir können

Statt der Aufzählung einzelner Kongruenzmerkmale in der NP-Regel können wir formulieren
Kongruenzmerkmale von DET = Kongruenzmerkmale von N
Kongruenzmerkmale von NP = Kongruenzmerkmale von N

Was sind Merkmalsstrukturen eigentlich?

$$\begin{bmatrix} cat & N \\ agr & \begin{bmatrix} num & pl \\ kas & nom \\ gen & m \end{bmatrix} \end{bmatrix}$$



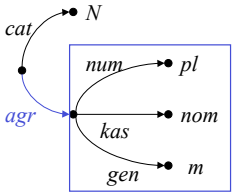
Merkmalstrukturen sind formal als gerichtete azyklische Graphen mit Kanteninschriften darstellbar.

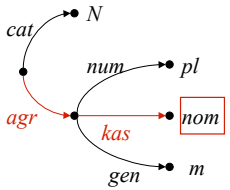
Merkmalsstrukturen, die wir bisher betrachten, sind (ungeordnete) Bäume mit einer Wurzel. Die Kanteninschriften sind Attribut-Label. Die Blätter sind mit atomaren Werten beschriftet.

Merkmalspfade sind Folgen von Kantenlabeln. Wir schreiben sie in der Form <agr> bzw. <agr kas> und können

Merkmalspfade, Pfadgleichungen

$$\begin{bmatrix} agr & \begin{bmatrix} num & pl \\ kas & nom \\ gen & m \end{bmatrix} \end{bmatrix}$$





CFG mit Merkmalsconstraints, Beispiel

S → NP VP
<NP agr num> = <VP agr num>
<NP agr kas> = nom

VP → VI
<VP agr num> = <VI agr num>

VP → VT NP
<VP agr num> = <VT agr num>
<NP agr kas> = akk

NP → DET N
<DET agr> = <N agr>
<NP agr> = <N agr>

VI → arbeitet
<VI agr num> = sg

VI → arbeiten
<VI agr num> = pl

N → Student
<N agr num> = sg
<N agr gen> = m
<N agr kas> = nom

DET → der
<DET agr num> = sg
<DET agr gen> = m
<DET agr kas> = nom