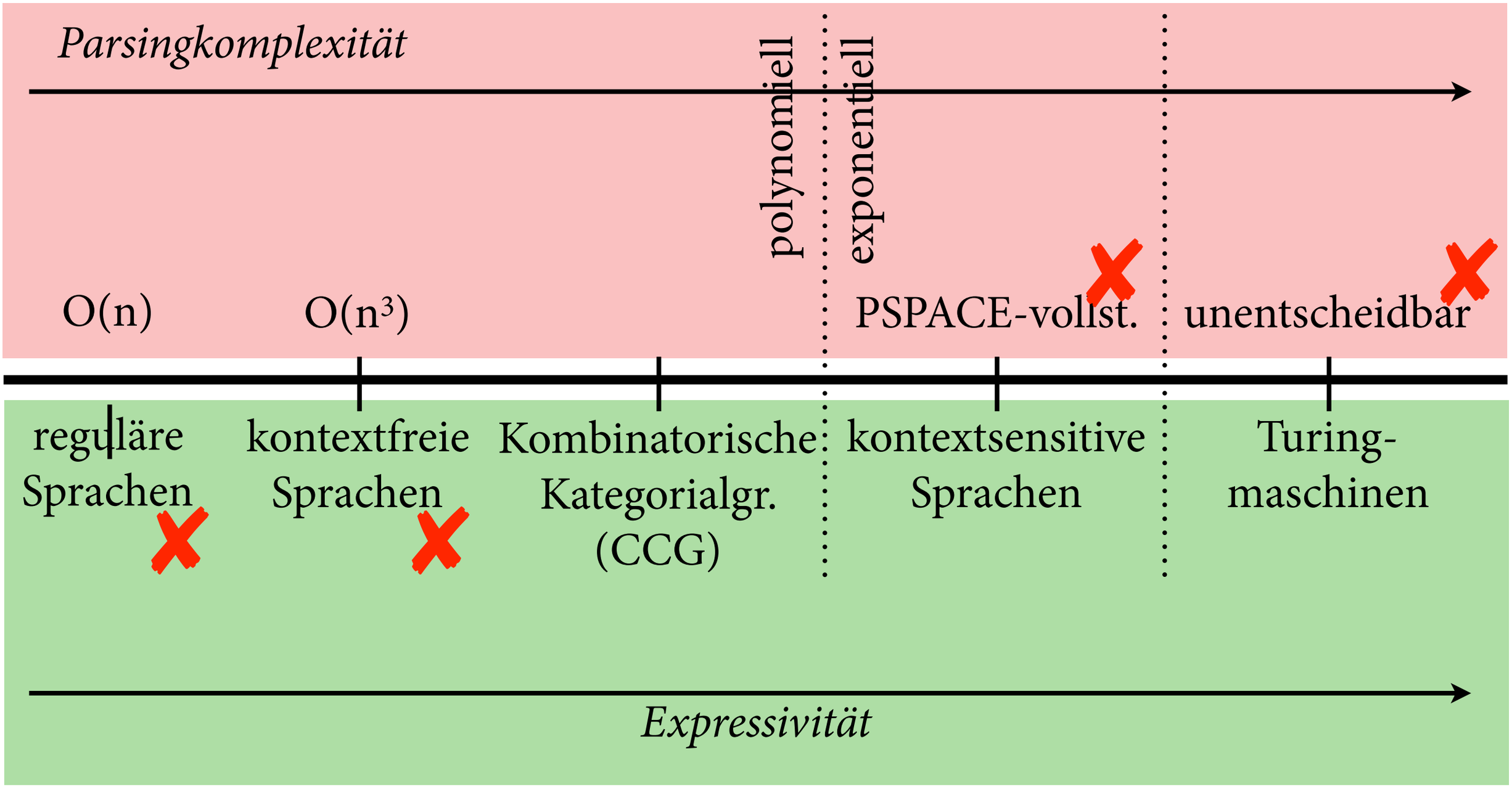
Kombinatorische Kategorialgrammatiken II

Vorlesung “Grammatikformalismen”

Alexander Koller

16. Mai 2017

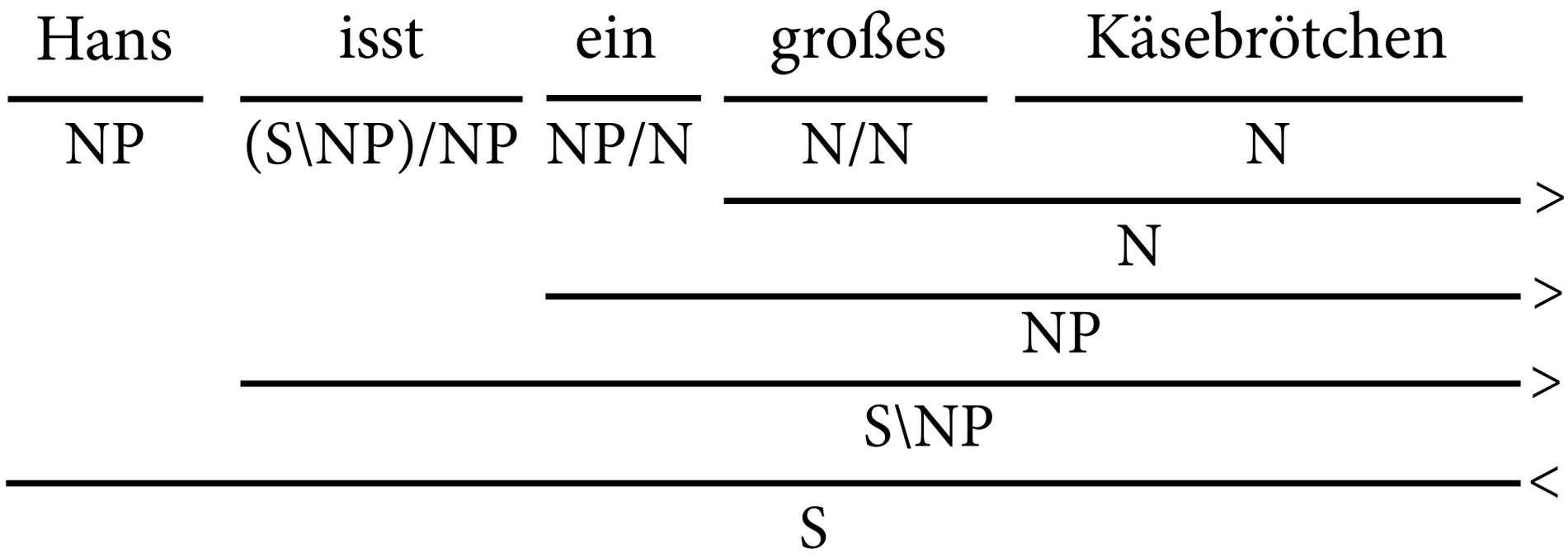
CCG


Parsingkomplexität
polynomiell
exponentiell
O(n)
O(n³)
PSPACE-vollst.
unentscheidbar
reguläre Sprachen
kontextfreie Sprachen
Kombinatorische Kategorialgr. (CCG)
kontextsensitive Sprachen
Turing-maschinen
Expressivität

Beispiel: Modifikation

Lexikon:

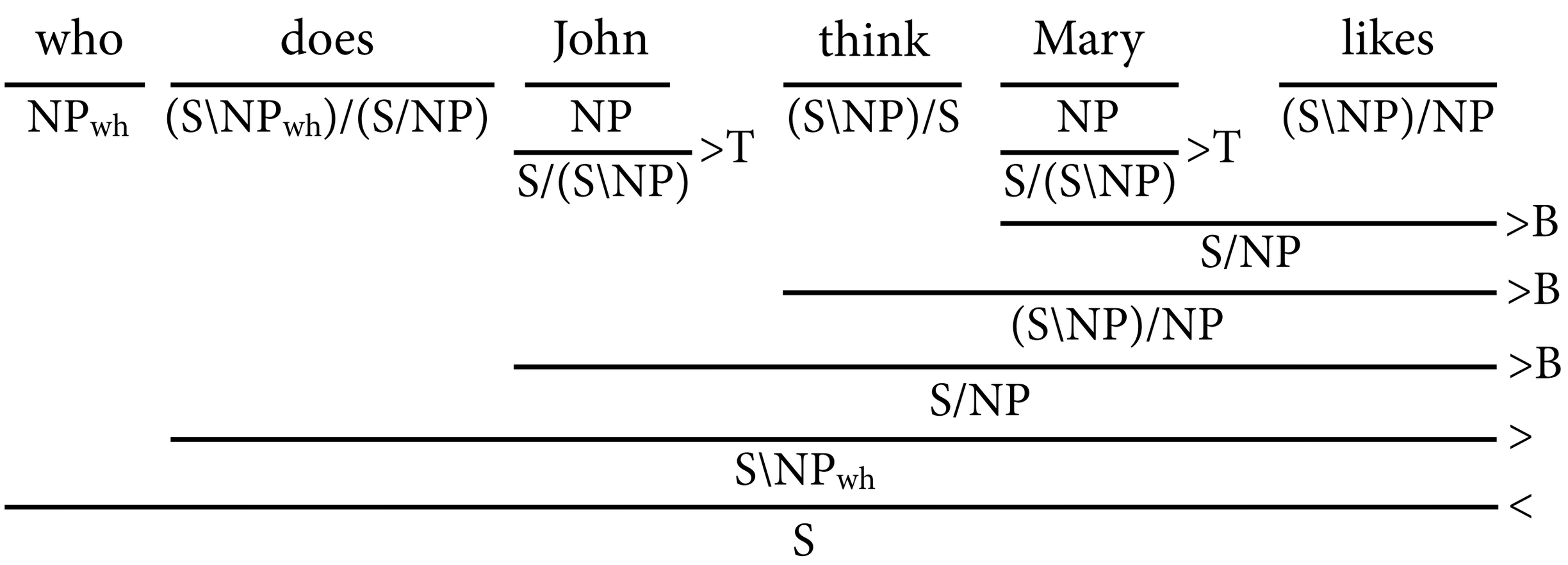
Hans: NP	ein: NP/N
isst: (S\NP)/NP	Käsebrötchen: N
großes: N/N	



Fernabhängigkeiten

Lexikon:

John: NP	think: (S\NP)/S
Mary: NP	likes: (S\NP)/NP
who: NP_{wh}	does: $(S\backslash NP_{wh})/(S/NP)$



Schweizerdeutsch

- Crossed composition erlaubt Modellierung von cross-serial dependencies:

das	mer	em Hans	es huus	hälfed	aastriiche
	NP_{nom}	NP_{dat}	NP_{acc}	$S_{+SUB}\backslash NP_{nom}\backslash NP_{dat}/VP$	$VP\backslash NP_{acc}$
				$S_{+SUB}\backslash NP_{nom}\backslash NP_{dat}\backslash NP_{acc}$	>Bx
			$S_{+SUB}\backslash NP_{nom}\backslash NP_{dat}$		<
		$S_{+SUB}\backslash NP_{nom}$			<
	S_{+SUB}				<

(“VP” = Abkürzung für $S\backslash NP_{nom}$)

Erweiterungen von CCG

- Regeleinschränkungen und Slashtypen
- Featurestrukturen
- Semantikkonstruktion

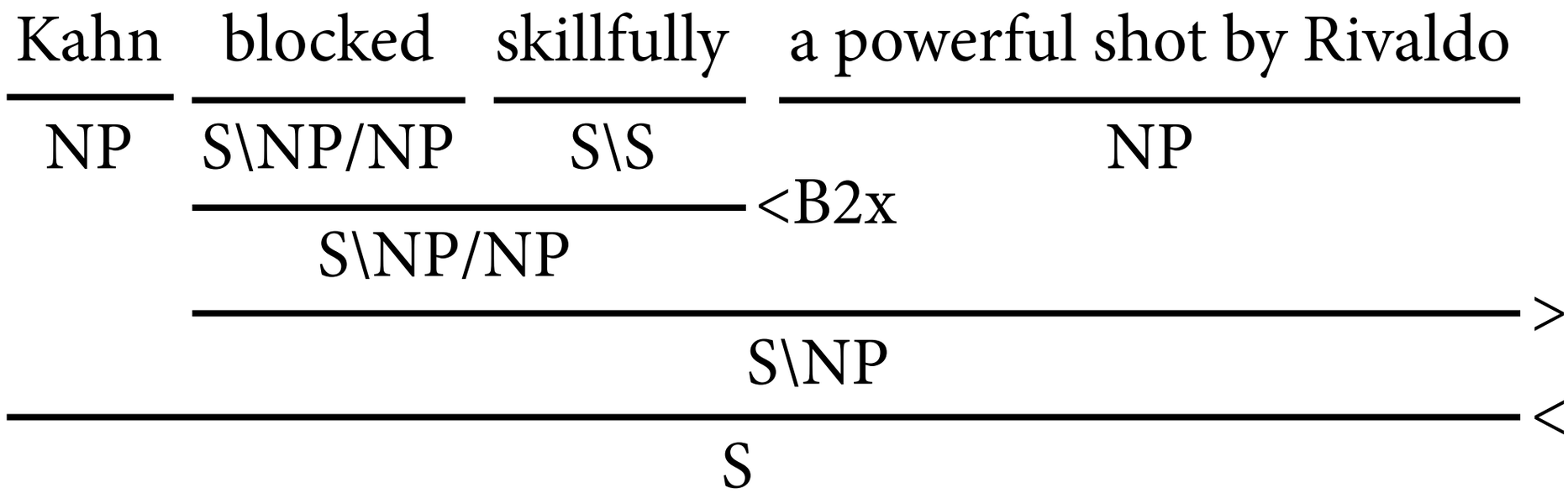
CCG zu permissiv

- Kombinationsregeln von CCG ermöglichen Expressivität bei niedriger lexikalischer Ambiguität.

- Sie ermöglichen allerdings auch viele Ableitungen, die nicht erlaubt sein sollten.

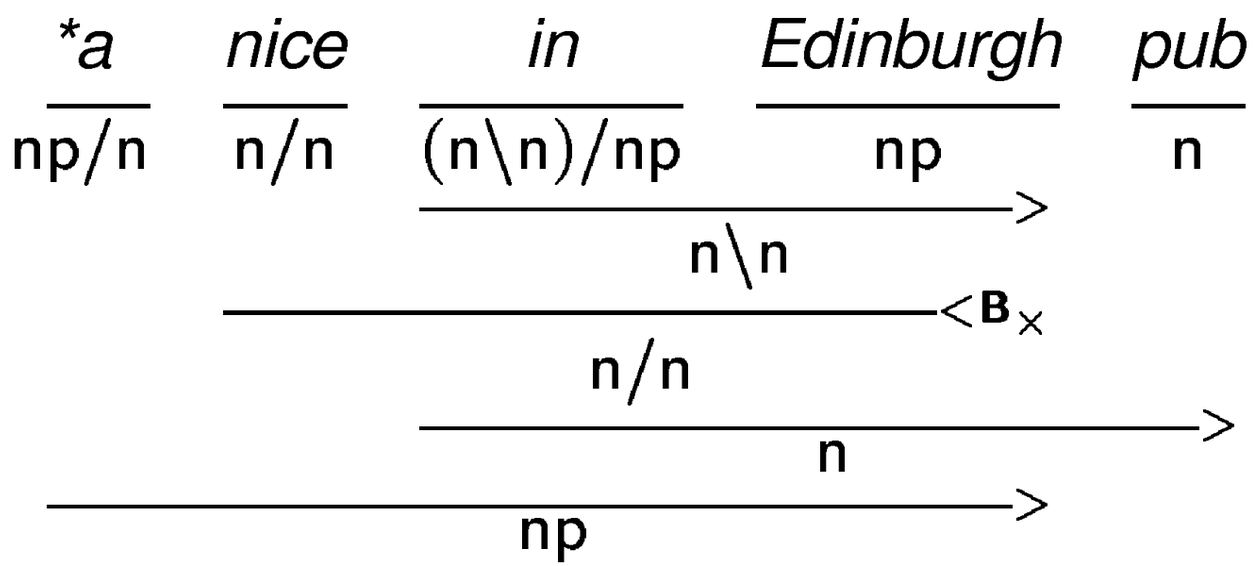
- Insbesondere crossed composition erlaubt unerwartete Worstellungen.

Bx für Englisch

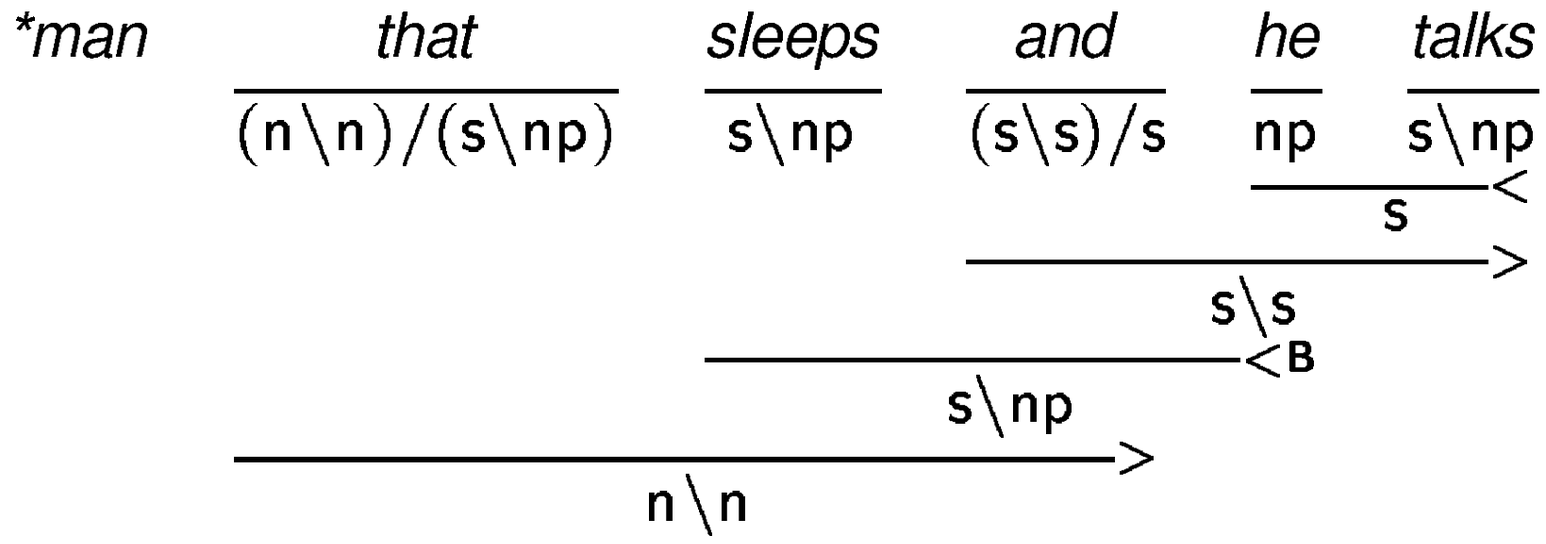


“Heavy NP Shift”; Beispiel aus Baldridge 2001

Übergenerierung



Übergenerierung



Regel-Einschränkungen

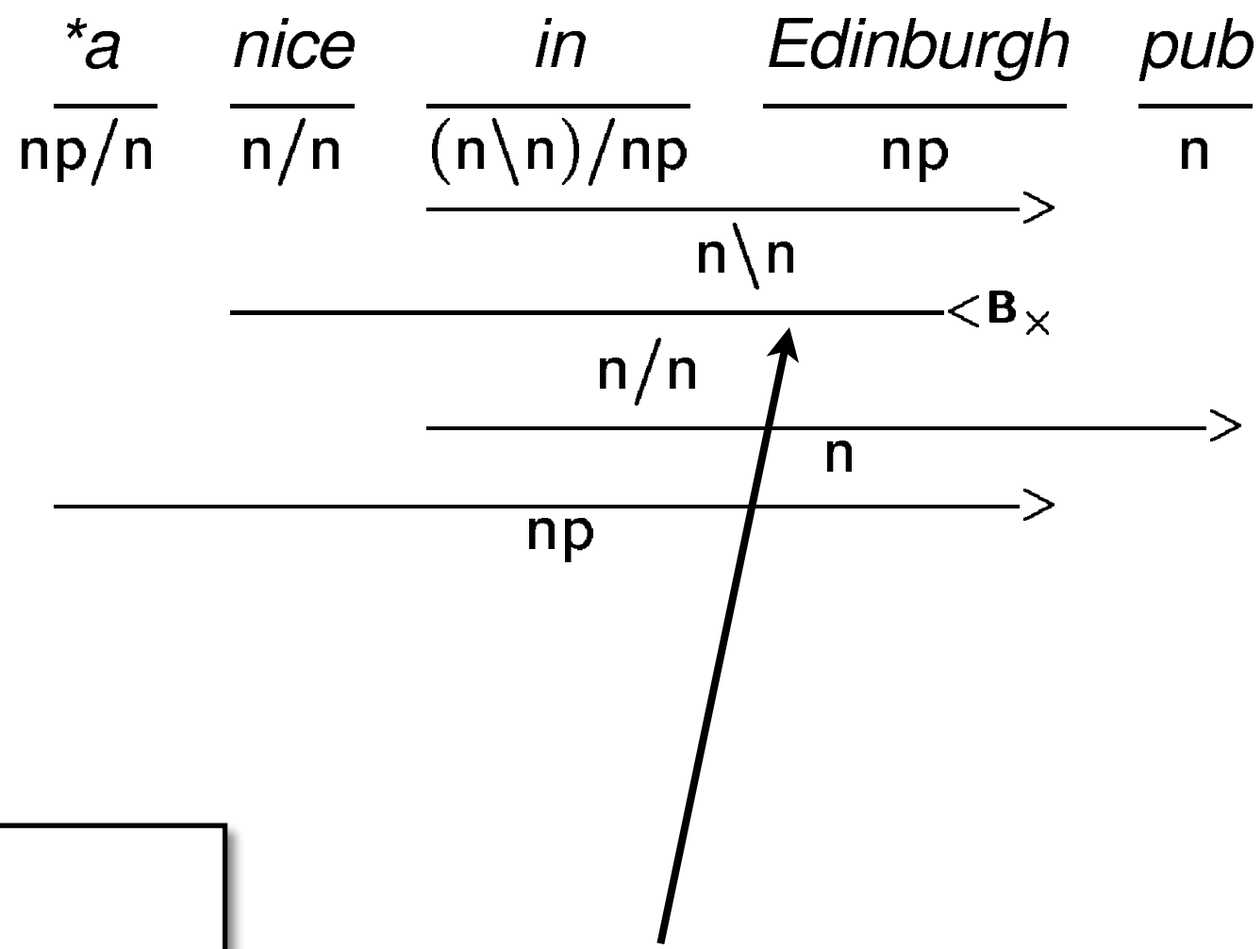
- Klassische CCG erlaubt deshalb, in der Grammatik die zulässigen Instanzen einer Kombinationsregel einzuschränken.

- Beispiel crossed composition in Englisch:

$$\frac{X/Y \qquad Y\backslash Z}{X\backslash Z} >Bx$$

falls X von der Form $S|A_1|...|A_n$ ist

Beispiel



$$\frac{X/Y \qquad Y\backslash Z}{X\backslash Z}\ >Bx$$

falls $X = S|A_1|...|A_n$

Diese Anwendung von <Bx unzulässig.

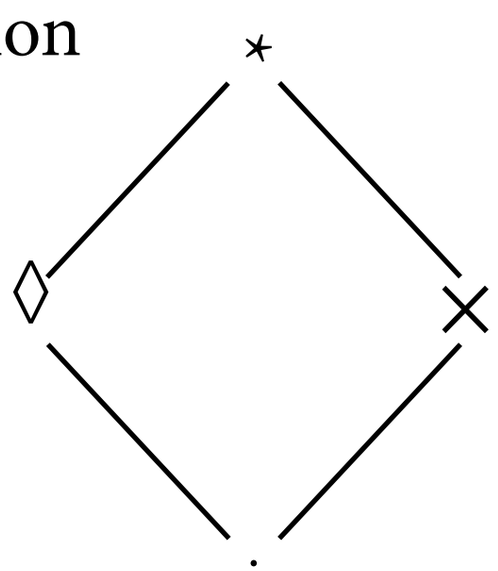
Lexikalisierung

- Ziel: Grammatische Information soll vollständig im Lexikon repräsentiert sein.
- Regel-Einschränkungen brechen dieses Prinzip.
- Können wir CCG so erweitern, dass man Einschränkungen lexikalisieren kann?

Slash-Typen

- *Multimodale* CCG: Jeder Schrägstrich (vorwärts oder rückwärts) bekommt einen *Typ;* verschiedene Typen erlauben verschiedene Kombinationsregeln.

A /⋆ B:
nur Applikation



A /× B: nur
crossed Komposition
(und Applikation)

A /◊ B: nur
harmonische Komposition
(und Applikation)

A /. B:
alle Regeln

Regeln für Slash-Typen

$$\frac{X /_{*} Y \qquad Y}{X} > \qquad \frac{Y \qquad X \backslash_{*} Y}{X} <$$

$$\frac{X /_{\diamond} Y \qquad Y/Z}{X/Z} >B \qquad \frac{Y\backslash Z \qquad X \backslash_{\diamond} Y}{X\backslash Z} <B$$

$$\frac{X /_{\times} Y \qquad Y\backslash Z}{X\backslash Z} >Bx \qquad \frac{Y/Z \qquad X \backslash_{\times} Y}{X/Z} <Bx$$

Beachte: Subtypen sind erlaubt, also z.B. X /. Y Y ⇒ X

Slashes im Ergebnis haben gleichen Typ wie in Prämisse.

Beispiel

$$\dfrac{\dfrac{\dfrac{\text{nice}}{N/{*}N}\quad\dfrac{\text{pub}}{N}\quad\dfrac{\dfrac{\text{in}}{(N\backslash_{\Diamond}N)/NP}\quad\dfrac{\text{Edinburgh}}{NP}}{N\backslash_{\Diamond}N}\,>}{N}\,<}{N}\,>$$

$$*\ \dfrac{\dfrac{\dfrac{\text{nice}}{N/{*}N}\quad\dfrac{\dfrac{\text{in}}{(N\backslash_{\Diamond}N)/NP}\quad\dfrac{\text{Edinburgh}}{NP}}{N\backslash_{\Diamond}N}\,>\quad\dfrac{\text{pub}}{N}}{N\backslash_{\Diamond}N}\,>}{}$$

$$\dfrac{X/{*}Y \qquad Y}{X}\,>$$

$$\dfrac{Y\backslash Z \qquad X\backslash_{\Diamond}Y}{X\backslash Z}\,<B$$

Grenzen von MM-CCG

- Mit MM-CCG kann man viele Regel-Einschränkungen lexikalisieren.

- Man kann zeigen, dass MM-CCG nicht ausreicht, um Regel-Einschränkungen unnötig zu machen (Kuhlmann, K., Satta 2015).

  - Beispiel: COUNT(3) geht
    mit CCG mit Regel-Einschränkungen, aber nicht
    mit MM-CCG ohne Regel-Einschränkungen.

Features in CCG

- Motivation für Features in kfG:
  Kongruenz, Subkategorisierung.

- Subkategorisierung in CCG in lexikalische Kategorien eingebaut: S\NP, (S\NP)/NP, ...

- Bei Kongruenz treten genau die gleichen Probleme auf wie in kfGen.

Features in CCG

- Lösung: CCG mit Features.
  - *Nichtrekursive* Featurestrukturen: Unter jedem Feature steht nur ein atomarer Wert, keine Featurestruktur.
  - Featurestrukturen an *atomare Kategorien* angehängt; funktionale Kategorien haben keine Features.

- Beispiele:
  - N[case=nom, num=pl]
  - S \ NP[case=nom]

Features in Regeln

- Erweiterte Kombinationsregeln: FS der Argumentkategorie und der anderen Prämisse müssen zusammenpassen.

$$\frac{X /\!\!^{\star} Y[f_1] \qquad Y[f_2]}{X}\ > \qquad \text{falls } f_1, f_2 \text{ unifizierbar}$$

- Beispiel:

$$\frac{\text{NP[case=nom, num=pl]} \quad \text{S} \setminus \text{NP[case=nom]}}{\text{S}}\ <$$

Features in OpenCCG

```
feature {
  GEN<2>: masc fem neut;
}

family PN {
  entry: np<2>;
}


family TransV {
       entry: (s<1>\np<2>[nom])\np<3>[acc];
}

word ‘Hans’: PN: nom masc;
```

Semantikkonstruktion

- Ziel von Parsing ist letztlich: Semantische Repräsentationen (SRen) ausrechnen.

- Semantikkonstruktion: Berechnung von SR aus syntaktischer Analyse.

- Kompositionalität: SR von großer Konstituente aus SRen ihrer Teile berechnen.

Semantikkonstruktion in CCG

- CCG erlaubt besonders elegante Beschreibung von Semantikkonstruktion.

- SR: Lambda-Terme.
  - Wörter erhalten ihre SR aus Lexikoneintrag.
  - Jede Kombinationsregel sagt, wie die SRen systematisch kombiniert werden.
  - Schöne Korrespondenz zwischen Kategorien und Typen.

Applikation

Lexikon:

Hans: NP: h^*

isst: (S\NP)/NP: eat'

ein Käsebrot: NP: kb'

$$\frac{\text{X/Y: } f \quad \text{Y: } g}{\text{X: } f(g)} \; >$$

$$\frac{\text{Y: } g \quad \text{X\Y: } f}{\text{X: } f(g)} \; <$$

$$\dfrac{\dfrac{\text{NP: } h^* \quad \dfrac{\text{(S\NP)/NP: } eat' \quad \text{NP: } kb'}{\text{S\NP: } eat'(kb')}\,>}{\text{S: } eat'(kb')(h^*)}\,<}$$

Hans | isst | ein Käsebrot

NP: h^* | (S\NP)/NP: eat' | NP: kb'

S\NP: $eat'(kb')$ >

S: $eat'(kb')(h^*)$ <

Applikation

Lexikon:

Hans: NP: *h**

isst: (S\NP)/NP: *eat'*

ein Käsebrot: NP: *kb'*

$$\frac{\text{X/Y: } f \quad \text{Y: } g}{\text{X: } f(g)} \; >$$

$$\frac{\text{Y: } g \quad \text{X\Y: } f}{\text{X: } f(g)} \; <$$

$$\dfrac{\dfrac{\text{Hans}}{\text{NP: } h^*} \quad \dfrac{\dfrac{\text{isst}}{\text{(S\NP)/NP: } eat'_{\langle e,\langle e,t\rangle\rangle}} \quad \dfrac{\text{ein Käsebrot}}{\text{NP: } kb'}}{\text{S\NP: } eat'(kb')} \; >}{\text{S: } eat'(kb')(h^*)} \; <$$

Applikation

Lexikon:

Hans: NP: h^*

isst: (S\NP)/NP: eat'

ein Käsebrot: NP: kb'

$$\frac{\text{X/Y: } f \qquad \text{Y: } g}{\text{X: } f(g)} \; >$$

$$\frac{\text{Y: } g \qquad \text{X\Y: } f}{\text{X: } f(g)} \; <$$

$$\dfrac{\dfrac{\text{Hans}}{\text{NP: } h^*} \quad \dfrac{\dfrac{\text{isst}}{\text{(S\NP)/NP: } eat'_{\langle e,\langle e,t\rangle\rangle}} \quad \dfrac{\text{ein Käsebrot}}{\text{NP: } kb'_{e}}}{\text{S\NP: } eat'(kb')} \; >}{\text{S: } eat'(kb')(h^*)} \; <$$

Applikation

Lexikon:

Hans: NP: h^*

isst: (S\NP)/NP: eat'

ein Käsebrot: NP: kb'

$$\frac{\text{X/Y: } f \quad \text{Y: } g}{\text{X: } f(g)} \; >$$

$$\frac{\text{Y: } g \quad \text{X\Y: } f}{\text{X: } f(g)} \; <$$

$$\dfrac{\dfrac{\text{Hans}}{\text{NP: } h^*} \quad \dfrac{\dfrac{\text{isst}}{\text{(S\NP)/NP: } eat'_{\langle e,\langle e,t\rangle\rangle}} \quad \dfrac{\text{ein Käsebrot}}{\text{NP: } kb'_{e}}}{\text{S\NP: } eat'(kb')_{\langle e,t\rangle}} \; >}{\text{S: } eat'(kb')(h^*)} \; <$$

Applikation

Lexikon:

Hans: NP: h^*

isst: (S\NP)/NP: eat'

ein Käsebrot: NP: kb'

$$\frac{\text{X/Y: } f \qquad \text{Y: } g}{\text{X: } f(g)} \; >$$

$$\frac{\text{Y: } g \qquad \text{X\Y: } f}{\text{X: } f(g)} \; <$$

$$\dfrac{\dfrac{\text{Hans}}{\text{NP: } h^*_{e}} \quad \dfrac{\dfrac{\text{isst}}{\text{(S\NP)/NP: } eat'_{\langle e,\langle e,t\rangle\rangle}} \quad \dfrac{\text{ein Käsebrot}}{\text{NP: } kb'_{e}}}{\text{S\NP: } eat'(kb')_{\langle e,t\rangle}} \; >}{\text{S: } eat'(kb')(h^*)} \; <$$

Applikation

Lexikon:

Hans: NP: h^*

isst: (S\NP)/NP: eat'

ein Käsebrot: NP: kb'

$$\frac{\text{X/Y: } f \qquad \text{Y: } g}{\text{X: } f(g)} \; >$$

$$\frac{\text{Y: } g \qquad \text{X\Y: } f}{\text{X: } f(g)} \; <$$

$$\dfrac{\dfrac{\text{Hans}}{\text{NP: } h^*_{e}} \quad \dfrac{\dfrac{\text{isst}}{\text{(S\NP)/NP: } eat'_{\langle e,\langle e,t\rangle\rangle}} \quad \dfrac{\text{ein Käsebrot}}{\text{NP: } kb'_{e}}}{\text{S\NP: } eat'(kb')_{\langle e,t\rangle}} \; >}{\text{S: } eat'(kb')(h^*)_{t}} \; <$$

Beobachtungen

- Syntaktisches Kombinieren
  = semantisches Kombinieren.
  - z.B. Applikation = Applikation

- Kategorien ≈ Typen.
  - NP ≈ e, S ≈ t
  - A/B, A\B ≈ ⟨B, A⟩
  - Spezialfall des *Curry-Howard-Isomorphismus*.

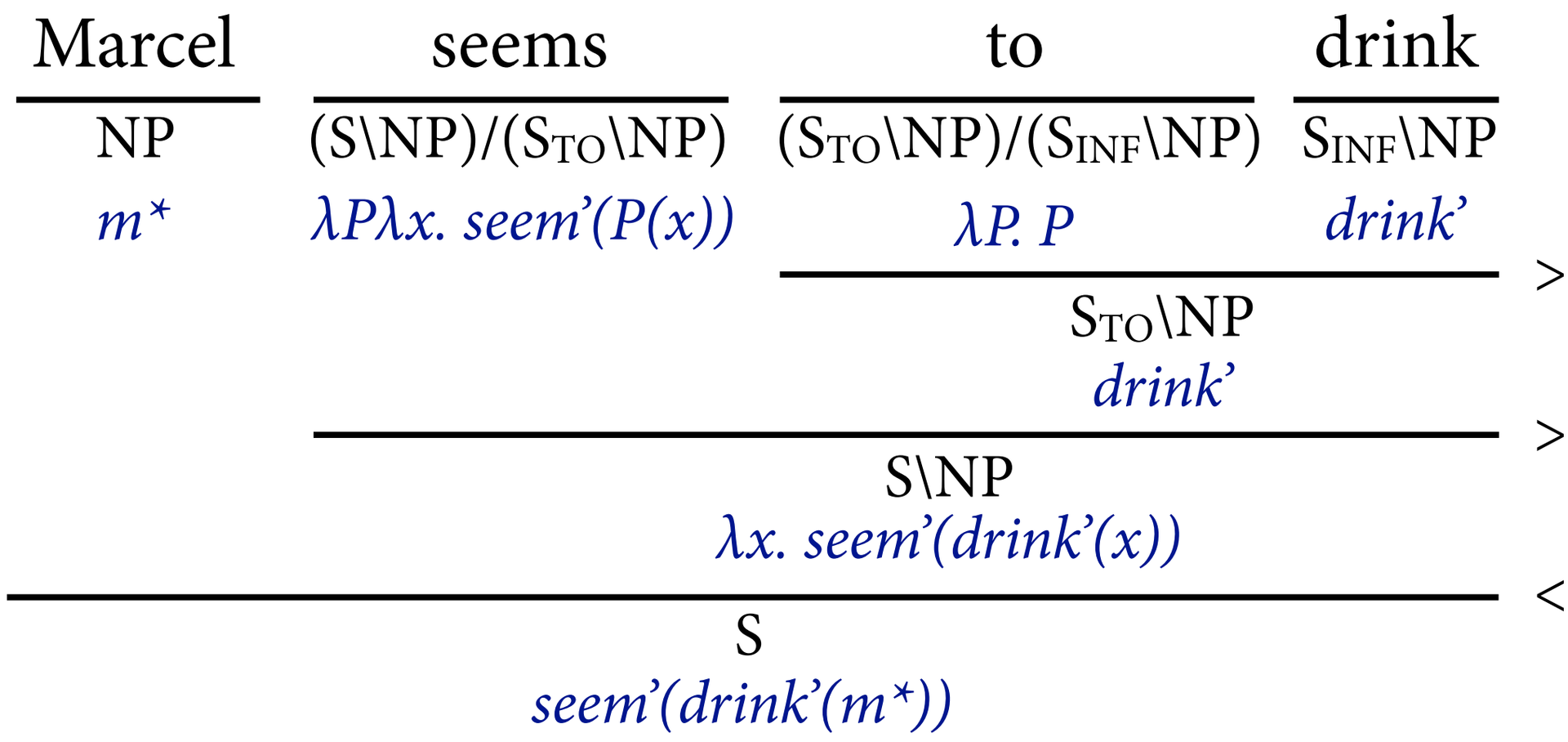
- Diese Prinzipien auf andere Regeln fortsetzen.

Kontrolle

- Subjektkontrolle: ein syntaktisches Argument füllt zwei semantische Argumentpositionen.
  → Nichtlineare Lambdaterme im Lexikon.

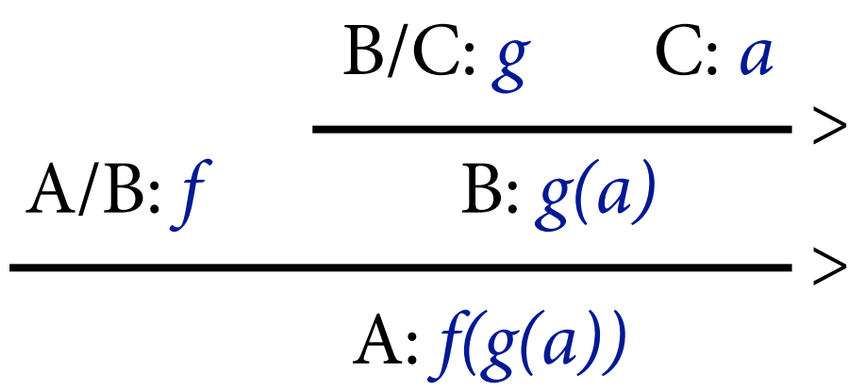
	will	singen
Hans	(S\NP)/(S\NP): $\lambda P \lambda x\ want'(P(x))(x)_{\langle\langle e,t\rangle,\langle e,t\rangle\rangle}$	S\NP: $sing'_{\langle e,t\rangle}$
NP: h^*_e	S\NP: $\lambda x\ want'(sing'(x))(x)_{\langle e,t\rangle}$	
S: $want'(sing'(h^*))(h^*)_t$		

Raising


Marcel
seems
to
drink
NP
(S\NP)/(S_TO\NP)
(S_TO\NP)/(S_INF\NP)
S_INF\NP
m*
λPλx. seem'(P(x))
λP. P
drink'
>
S_TO\NP
drink'
>
S\NP
λx. seem'(drink'(x))
<
S
seem'(drink'(m*))

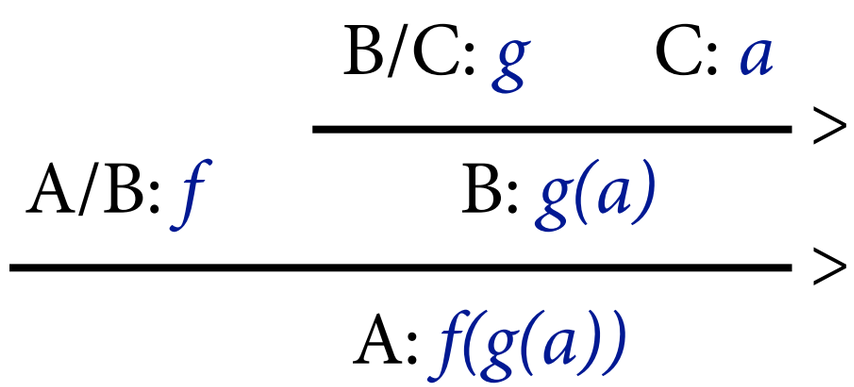
Semantik für Komposition

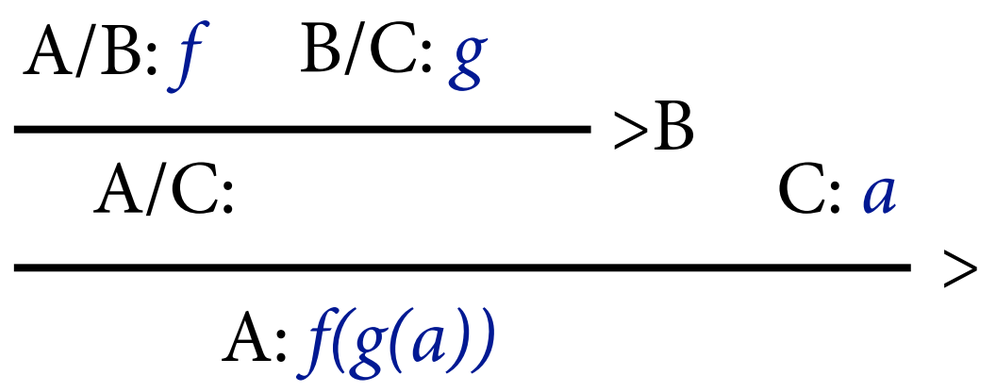
Grundprinzip: “Gleichartige” Ableitungen sollen gleiche SRen berechnen.



Semantik für Komposition

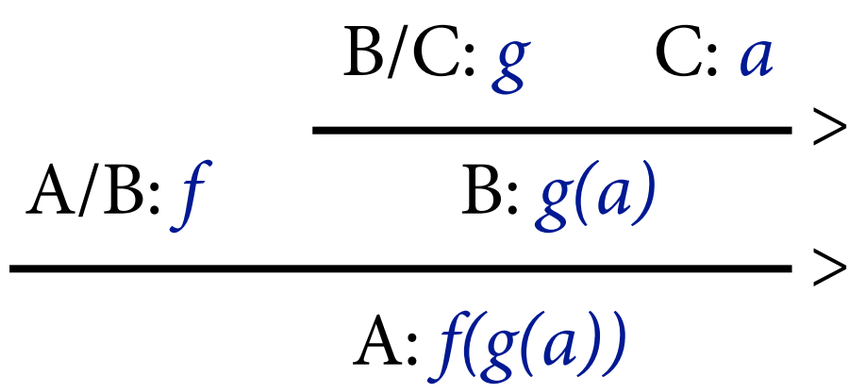
Grundprinzip: “Gleichartige” Ableitungen sollen gleiche SRen berechnen.

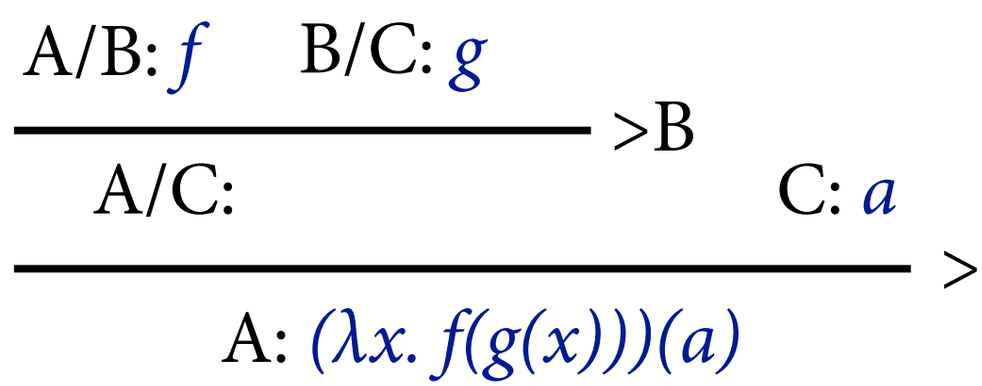




Semantik für Komposition

Grundprinzip: “Gleichartige” Ableitungen sollen gleiche SRen berechnen.

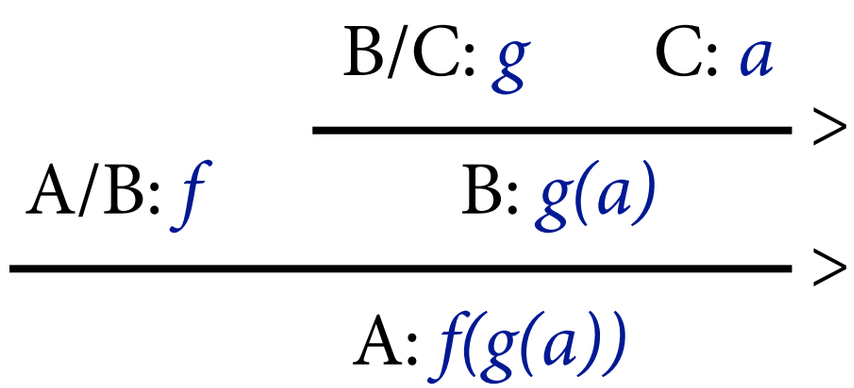


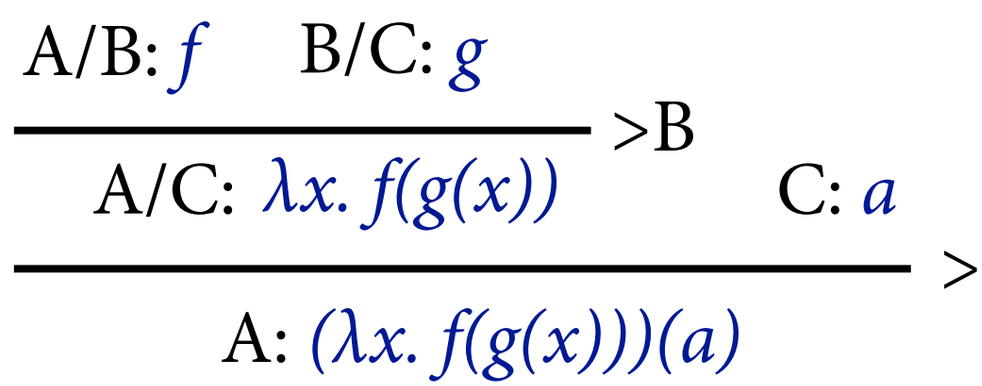


$$(\lambda x.\, f(g(x)))(a) \Rightarrow_\beta f(g(a))$$

Semantik für Komposition

Grundprinzip: “Gleichartige” Ableitungen sollen gleiche SRen berechnen.





$$(\lambda x.\, f(g(x)))(a) \Rightarrow_\beta f(g(a))$$

Semantik für Type-Raising

Grundprinzip: “Gleichartige” Ableitungen sollen gleiche SRen berechnen.

$$\frac{\text{NP: } a \qquad \text{S\NP: } P}{\text{S: } P(a)} \; <$$

Semantik für Type-Raising

Grundprinzip: “Gleichartige” Ableitungen sollen gleiche SRen berechnen.

$$\frac{\text{NP: } a \qquad \text{S\NP: } P}{\text{S: } P(a)}\ <$$

$$\frac{\text{NP: } a^{e}}{\text{S/(S\NP): }}\ >\text{T} \qquad \frac{\text{S/(S\NP): } \qquad \text{S\NP: } P^{\langle e,t\rangle}}{\text{S: } P(a)_{t}}\ >$$

Semantik für Type-Raising

Grundprinzip: “Gleichartige” Ableitungen sollen gleiche SRen berechnen.

$$\frac{\text{NP: } a \qquad \text{S\NP: } P}{\text{S: } P(a)}\ <$$

$$\frac{\text{NP: } a^{e}}{\text{S/(S\NP): } \quad}\ >\text{T} \qquad\qquad \frac{\text{S/(S\NP): } \qquad\qquad \text{S\NP: } P^{\langle e,t\rangle}}{\text{S: } (\lambda Q.\ Q(a))(P)_{t}}\ >$$

$$(\lambda Q.\ Q(a))(P) \Rightarrow_{\beta} P(a)$$

Semantik für Type-Raising

Grundprinzip: “Gleichartige” Ableitungen sollen gleiche SRen berechnen.

$$\frac{\text{NP: } a \qquad \text{S\NP: } P}{\text{S: } P(a)}\ <$$

$$\frac{\dfrac{\text{NP: } a^{\,e}}{\text{S/(S\NP): } \lambda Q.\ Q(a)_{\langle\langle e,t\rangle,t\rangle}}\ >\text{T} \qquad \text{S\NP: } P^{\langle e,t\rangle}}{\text{S: } (\lambda Q.\ Q(a))(P)_{\,t}}\ >$$

$$(\lambda Q.\ Q(a))(P) \Rightarrow_{\beta} P(a)$$

Type-Raising und Komposition

$$\frac{\text{X: } a}{\text{Y/(Y\X): } \lambda P.P(a)}\ \text{>T} \qquad \frac{\text{X/Y: } f \qquad \text{Y/Z: } g}{\text{X/Z: } \lambda x.f(g(x))}\ \text{>B} \qquad \frac{\text{X/Y: } f \qquad \text{Y\Z: } g}{\text{X\Z: } \lambda x.f(g(x))}\ \text{>Bx}$$

$$\frac{\text{X: } a}{\text{Y\(Y/X): } \lambda P.P(a)}\ \text{<T} \qquad \frac{\text{Y\Z: } g \qquad \text{X\Y: } f}{\text{X\Z: } \lambda x.f(g(x))}\ \text{<B} \qquad \frac{\text{Y/Z: } g \qquad \text{X\Y: } f}{\text{X/Z: } \lambda x.f(g(x))}\ \text{<Bx}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\text{Hans}}{\text{NP: } h^*}}{\text{S/(S\NP): } \lambda P.P(h^*)}\ \text{>T} \qquad \dfrac{\text{isst}}{\text{(S\NP)/NP: } eat'}}{\text{S/NP: } \lambda x.(\lambda P.P(h^*))(eat'(x)) \Rightarrow_\beta \lambda x.eat'(x)(h^*)}\ \text{>B} \qquad \dfrac{\text{ein Käsebrot}}{\text{NP: } kb'}}{\text{S: } (\lambda x.eat'(x)(h^*))(kb') \Rightarrow_\beta eat'(kb')(h^*)}\ \text{>}$$

Einschränkungen

- Semantische Repräsentation von komplexen NPs?
  - “ein Käsebrot”: NP/N N ⇒ NP
  - SRen für “ein” und “Käsebrot”?
  - alternative Skopustheorie in CCG (Steedman 2012)
- Umgang mit semantischen Ambiguitäten?
  - für jede syntaktische Analyse nur eine SR
  - muss deshalb gezwungenermaßen semantisch ambigen Sätzen mehrere syntaktische Analysen zuweisen

Zusammenfassung

- Kontrolle des Ableitungsprozesses in CCG:
  - Regel-Einschränkungen
  - Slashtypen in MM-CCG
- Features in CCG
- Semantikkonstruktion