Beweisnetze in der Kategorialgrammatik

Armin W. Buch

Seminar f. Sprachwissenschaft, 10.7.2009

Kurz zusammengefasst

- Beweisnetze stammen aus der Linearen Logik, einer ressourcen-sensitiven Einschränkung klassischer Logik
- Dort dienen sie der eindeutigen Darstellung von Beweisen (die sich auf mehreren Wegen herleiten lassen)
- Also kann man damit Parses in der Kategorialgrammatik darstellen
- Und so über Derivationsambiguitäten abstrahieren

Aufbau dieses Vortrags

- Herleitung von Beweisnetzen inkl. Definitionen
- anhand von vielen Beispielen
- Untersuchung ihrer Eigenschaften
- Einzelne interessante Aspekte und Grenzfälle
- Abschlussdiskussion

Grundlegendes über Kategorialgrammatiken

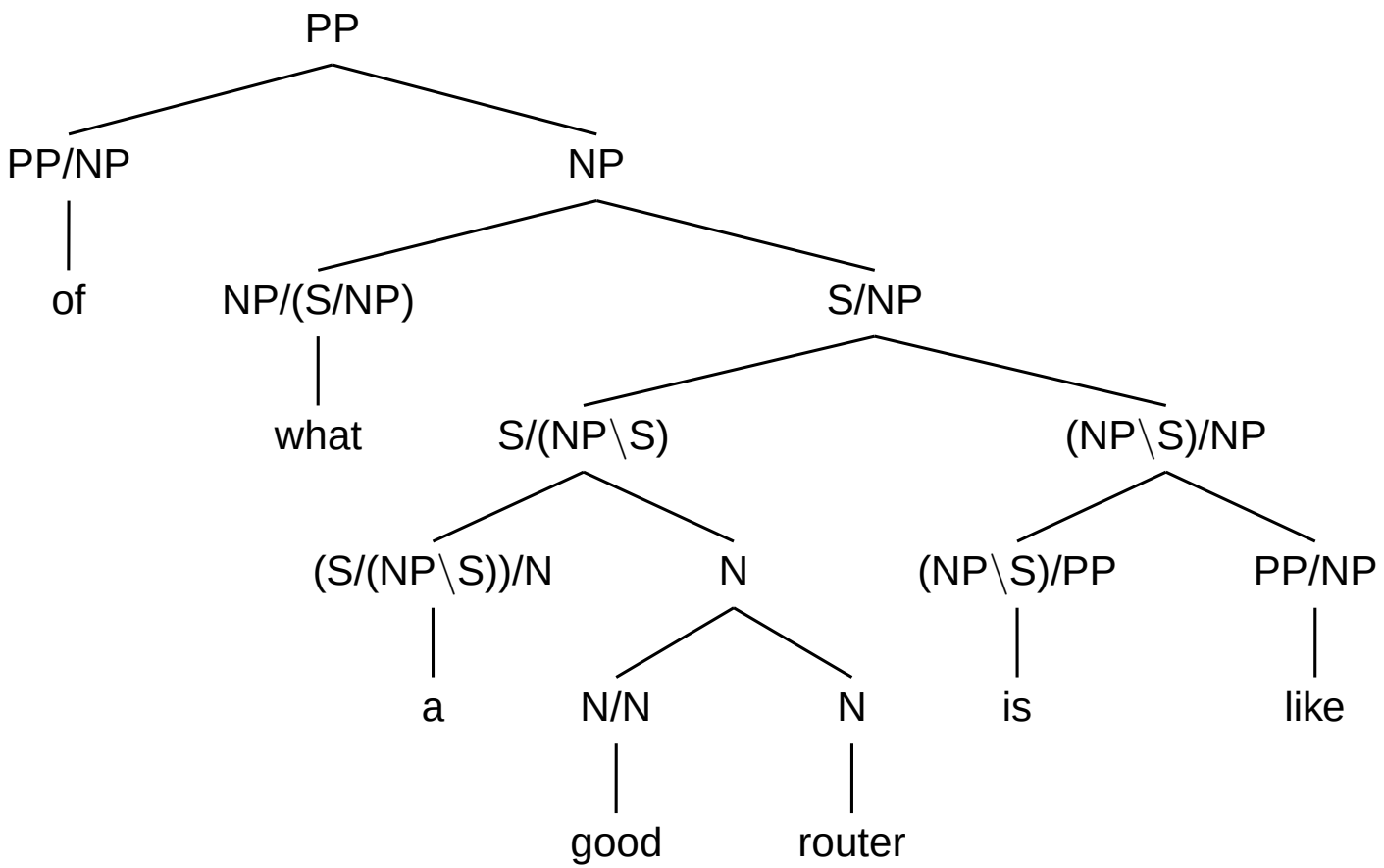
- Primitive und komplexe Kategorien:
  - $\mathrm{CAT}_0 \subset \mathrm{CAT}$
  - $(B\backslash A, A/B \in \mathrm{CAT}) \Leftrightarrow (A \in \mathrm{CAT} \wedge B \in \mathrm{CAT})$
- Kombinatoren:

$(>)$	Applikation	$X/Y : f$	$Y : a$	$\rightarrow$	$X : f(a)$
$(<)$	Applikation	$Y : a$	$Y\backslash X : f$	$\rightarrow$	$X : f(a)$
$(>\mathrm{B})$	Komposition	$X/Y : f$	$Y/Z : g$	$\rightarrow$	$X/Z : \lambda z.f(g(z))$
$(<\mathrm{B})$	Komposition	$Z\backslash Y : g$	$Y\backslash X : f$	$\rightarrow$	$Z\backslash X : \lambda z.f(g(z))$

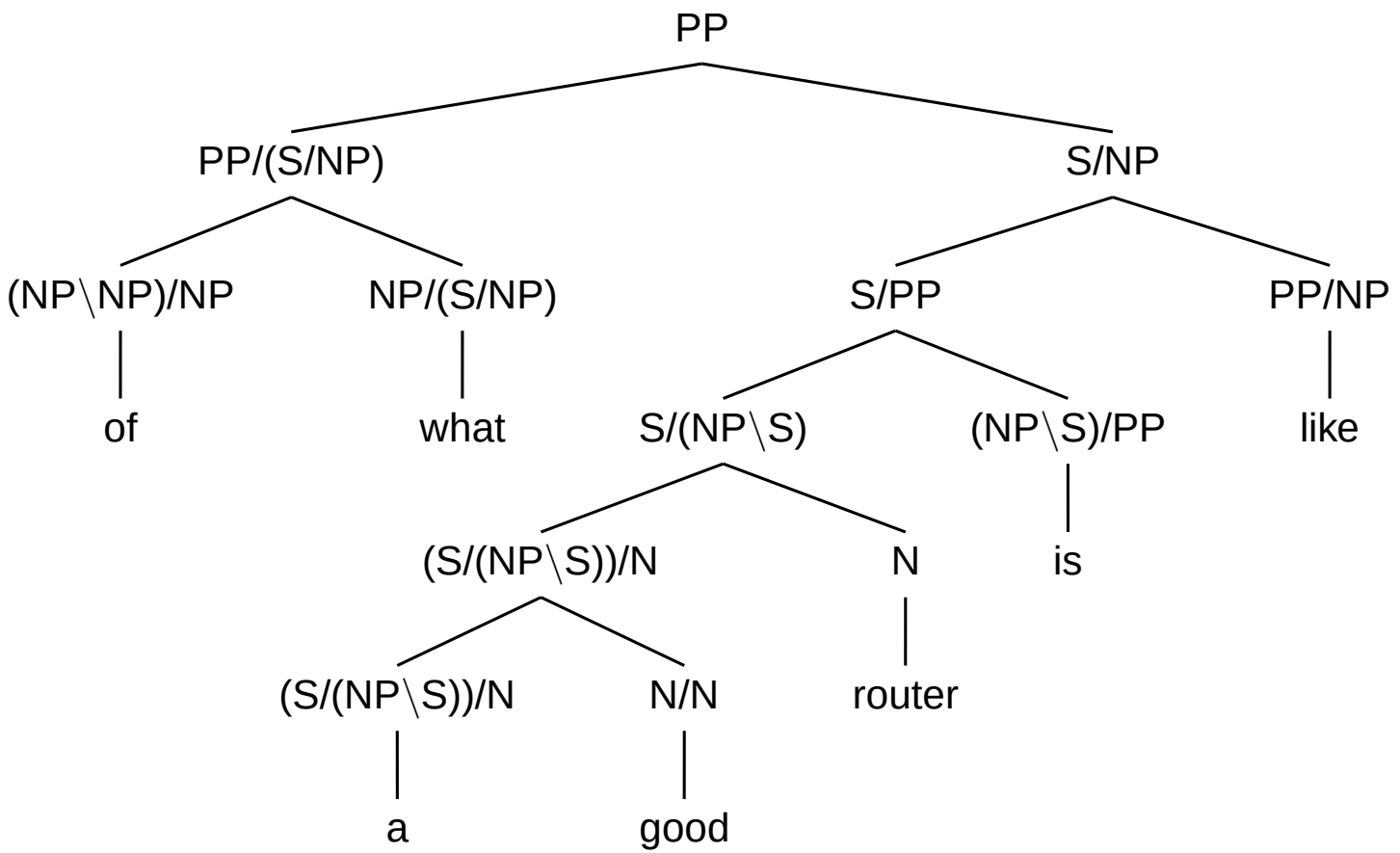
Herleitungsambiguität: Das Problem

- (Fast) Bedeutungslose Herleitungsambiguität in Kategorialgrammatiken
- Mehrere Baumstrukturen
- Selbe Bedeutung
- Nicht zu verwechseln mit echter syntaktischer Ambiguität
- Beispiel kommt sofort

Herleitungsambiguität: Erste Herleitung



Herleitungsambiguität: Zweite Herleitung



Herleitungsambiguität: Fazit

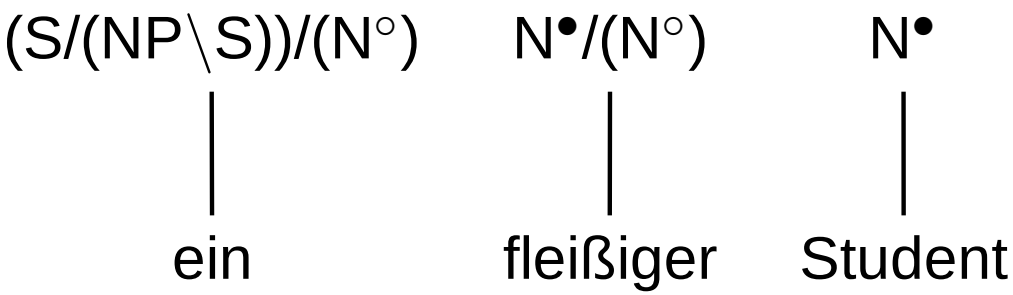
- Eine *normale* und eine *inkrementelle* Herleitung
- Nur zwei unter vielen
- Evtl. ist die zweite für Online-Effekte relevanter (unbewiesen)
- Die Bedeutung aber ist identisch
- Gesucht: Eine Abstraktion über die Herleitung
  - Eine für alle: bedeutet immer noch eine bestimmte
  - Chart: behält die Ambiguität oder hat gar keine Struktur

Grundlegendes über Beweisnetze

- Derivationsunabhängige Darstellung eines Parses in einer Kategorialgrammatik
- unmittelbare Kodierung von syntaktischer und semantischer Struktur
- So nützlich wie Bäume für Phrasenstrukturgrammatiken
- Leider relativ unzugänglich (für Nicht-Logiker) und gewöhnungsbedürftig

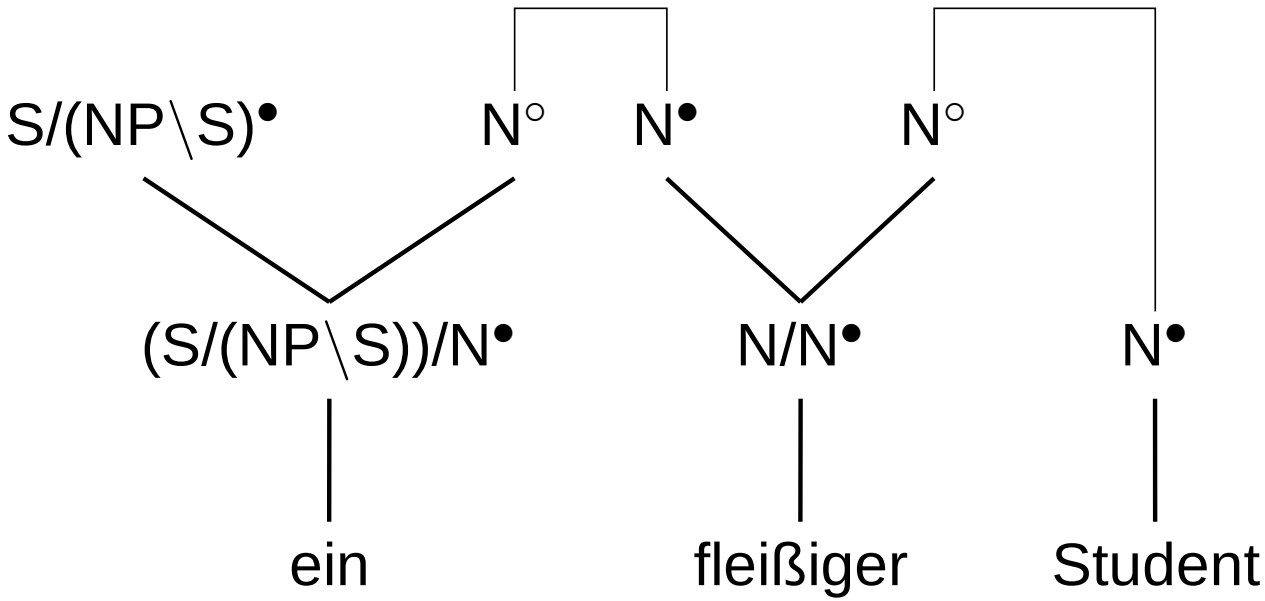
Knoten identifizieren

- Gemeinsamkeit aller äquivalenten Derivationen: Die paarweise Identifikation von Knoten
- Markieren wir diese Knoten mit *Soll* und *Haben* (Polarität)



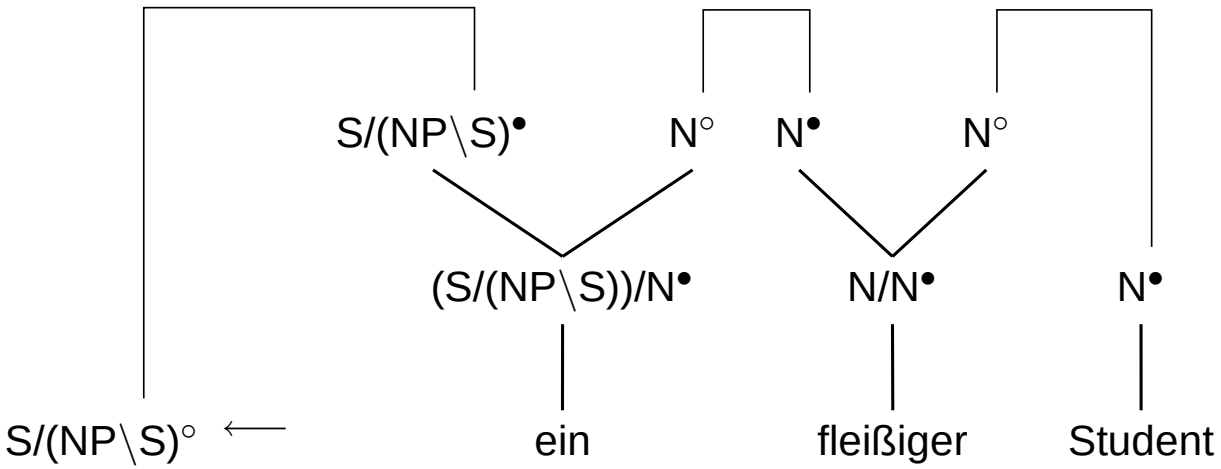
Kategorien ausfalten

- Und verbinden wir die Knoten
- Komplexe Kategorien lassen sich als Bäume deuten

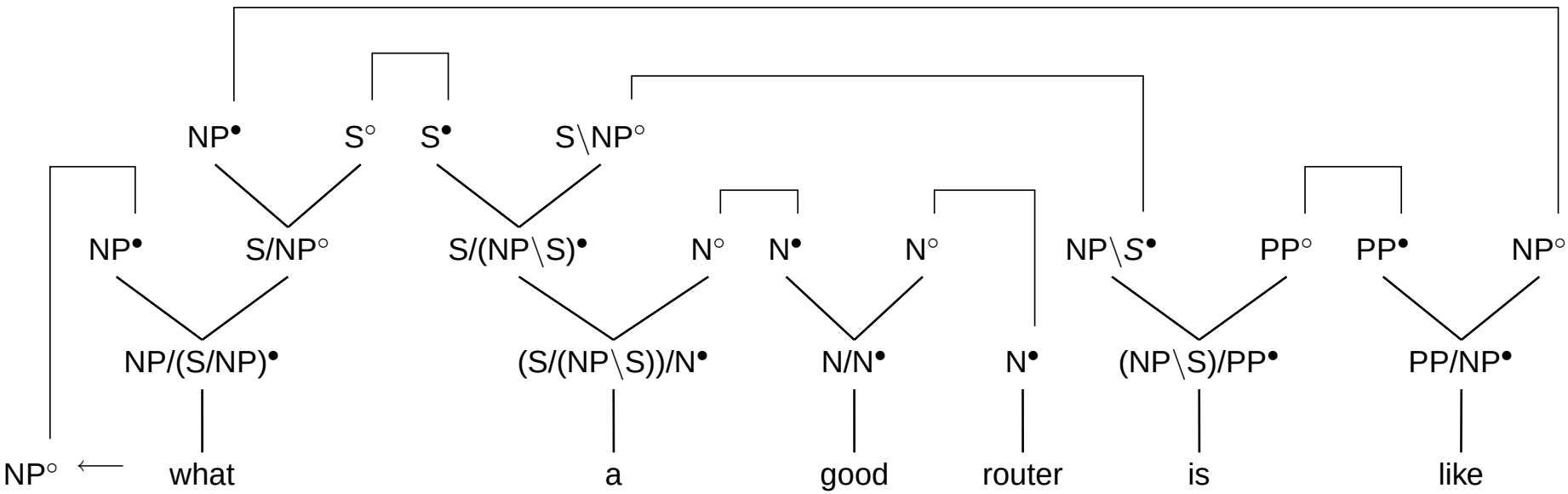


Ergebniskategorie

- Verbinde die offene Kategorie mit dem Ergebnis
- Abkürzungen (komplexen Kategorien verbinden) erlaubt
- Fertig ist das Beweisnetz



Das Beispiel von vorhin



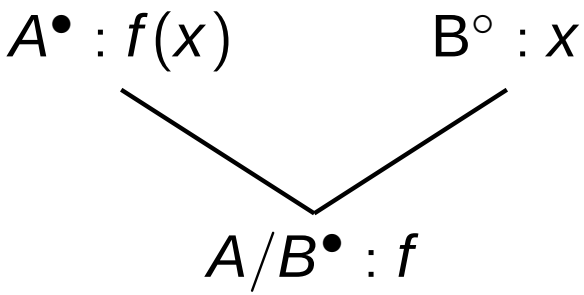
Zwischenfazit

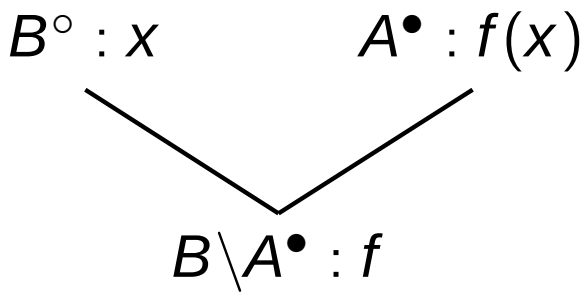
- Das Beweisnetz abstrahiert über alle Derivationen
- Es zeigt die reine syntaktische Struktur
- Zur Semantik kommen wir gleich

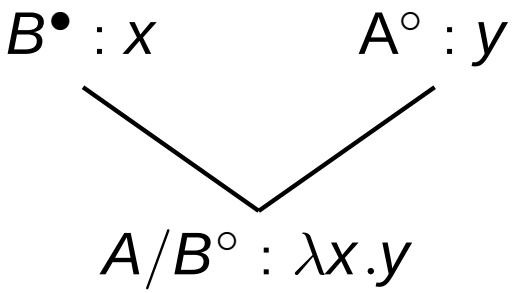
Kurzrezept

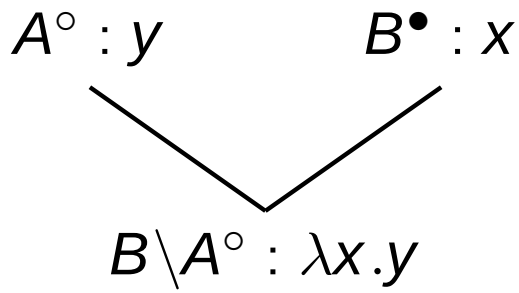
- Markiere alle Kategorien im Satz als Haben
- Füge Ergebniskategorie als Soll hinzu
- Expandiere sie alle (Regeln folgen)
- Verbinde die Kategorienknoten passend
- Prüfe die Gültigkeit

Expansionsregeln (mit Semantik)









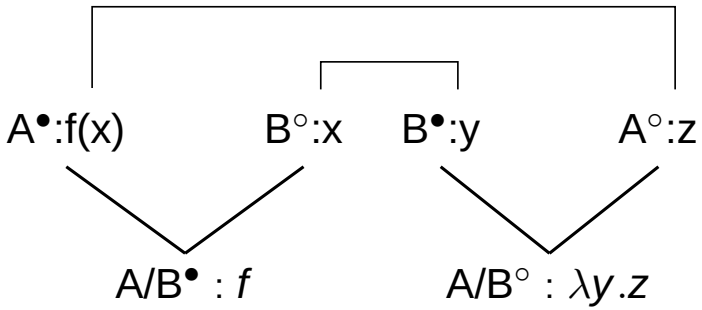
Das *axiom linking*

- Verbinde immer ein $A^\bullet$ mit einem A°
- Gleichsetzung der Bedeutungen an diesen Knoten
- Bis alle Knoten verbunden sind
- Das Ergebnis ist eine Beweisstruktur
- Ein Netz ist es, wenn es zudem wohlgeformt ist

Wohlgeformtheit

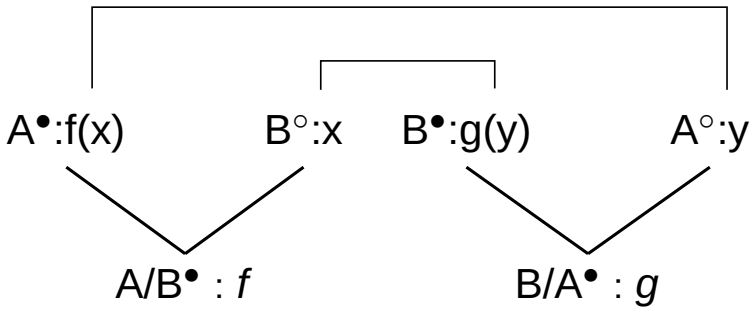
- Es gibt verschiedene, z.T. sehr komplizierte Wohlgeformtheitsbedingungen
- Für unsere Zwecke (nicht die ganze Lineare Logik) reicht eine einfache:
- Zyklen im Netz müssen immer über eine ○-Expansion gehen
- Damit sie nicht semantisch zirkulär werden

Wohlgeformtes Beispiel



- Dieser Zyklus als Teil eines Beweisnetzes
- Per Links ist $y = x$ und $z = f(x)$
- Also $\lambda y.z = \lambda y.f(x) = \lambda y.f(y)$

Nicht wohlgeformtes Beispiel

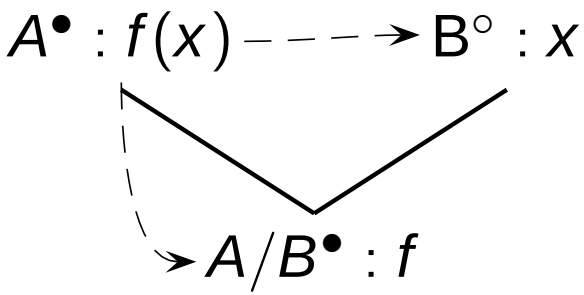


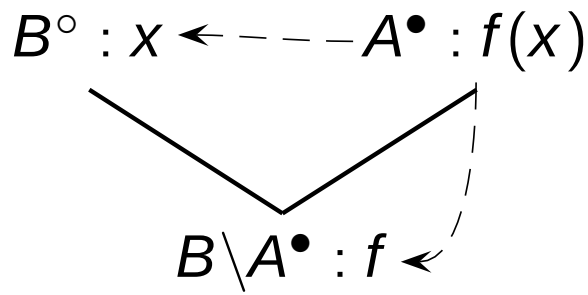
- Per Links ist $y = f(x)$ und $x = g(y)$
- zirkulär $\rightarrow$ nicht wohlgeformt

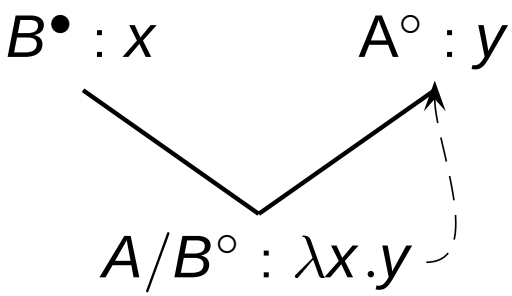
Der *semantic trip*

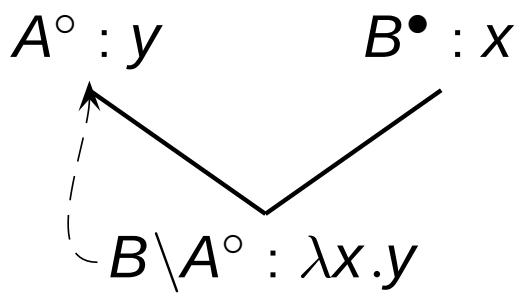
- Reihenfolge, in der der λ-Term vom Netz abgelesen wird
- hat Baumstruktur
- Diese entspricht der Klammerung des unreduzierten λ-Terms

Regeln für den semantic trip (Expansionen)



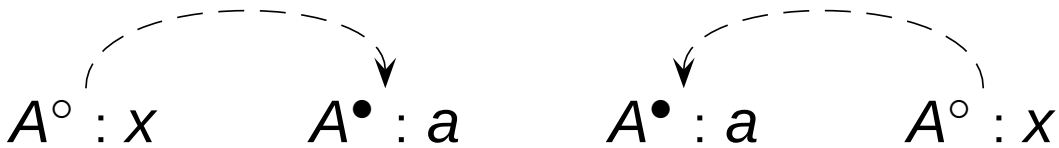




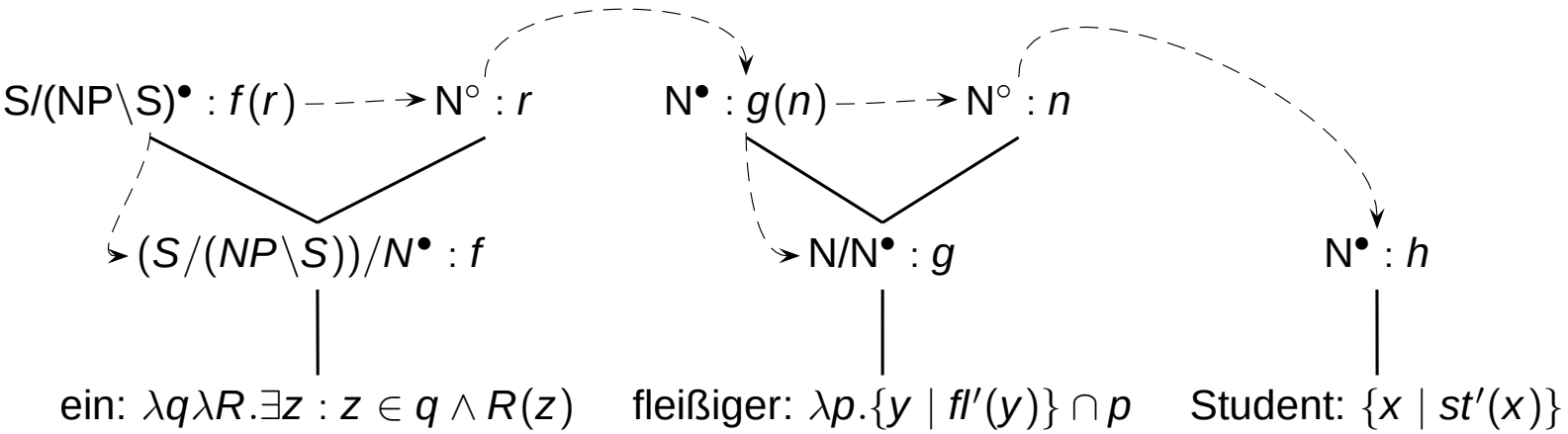


Regeln für den semantic trip (Links)

- Der semantic trip folgt den Links
- Hier denotiert er Substitution der Variablen: $x = a$



Beispiel für den semantic trip

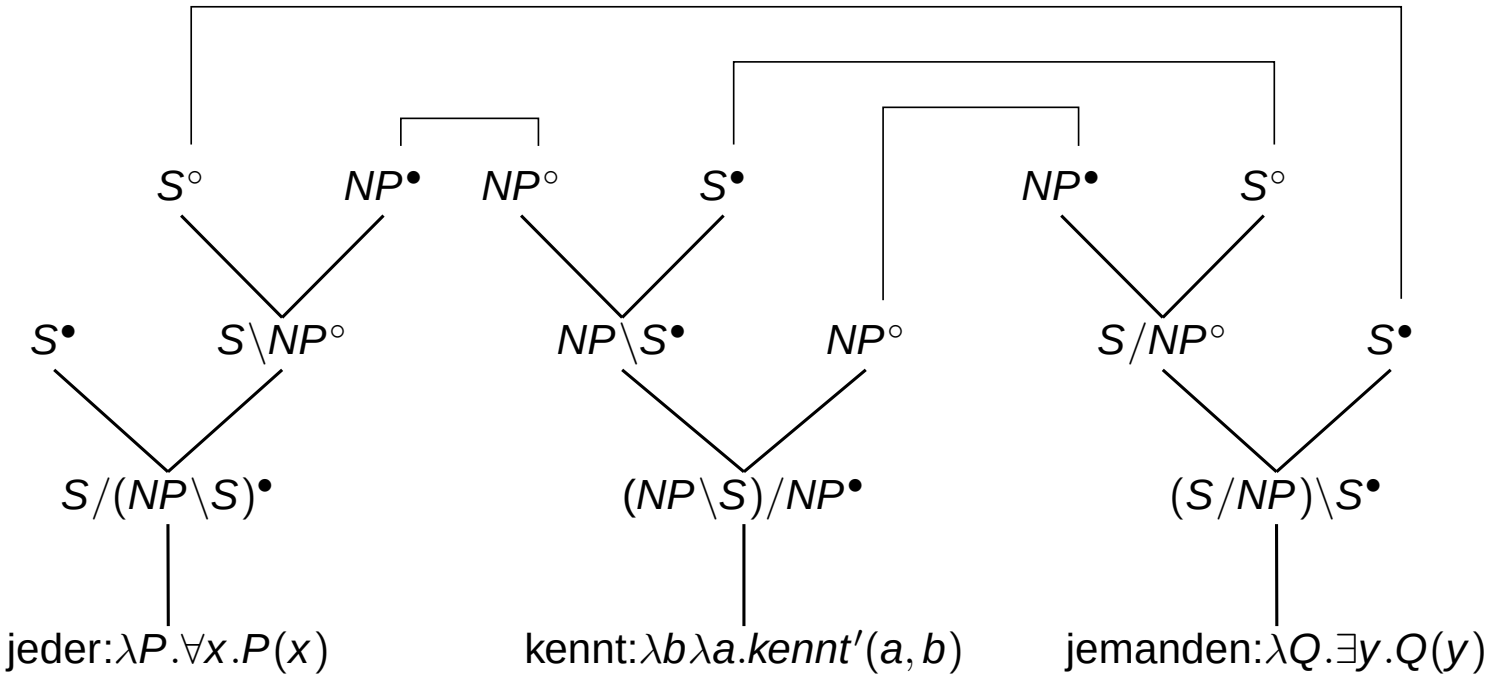


- $f(r) = \lambda R.\exists z : st'(z) \wedge fl'(z) \wedge R(z)$

Echte syntaktische Ambiguität

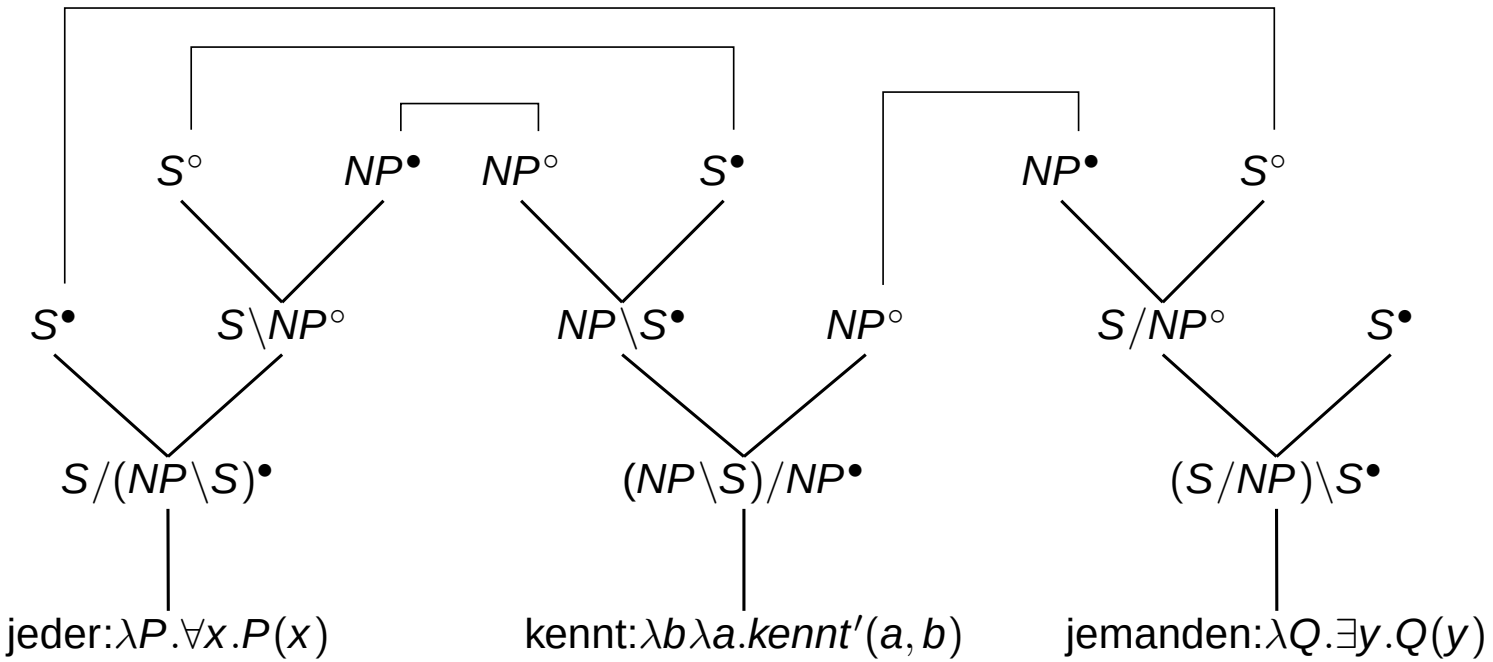
- Echte Ambiguität führt zu verschiedenen Netzen
- Und verschiedene Netze haben verschiedene Bedeutungen
- Am Beispiel Skopusambiguität

Weiter Skopus des Subjekts



- $\forall x \exists y . kennt'(x, y)$
- Hier vereinfacht: Braucht Typenanhebung in CG

Weiter Skopus des Objekts



- $\exists y \forall x . kennt'(x, y)$

Eigenschaften von Beweisnetzen

(a) Wohlgeformtheit

(b) Planarität

(c) Zyklen sind balanciert

(d′) Argumente liegen immer in der geforderten Richtung (primäre Links)

(d″) Eingebettete Argumente (Abstraktionen) gehen entsprechend in die andere, ‘falsche’ Richtung

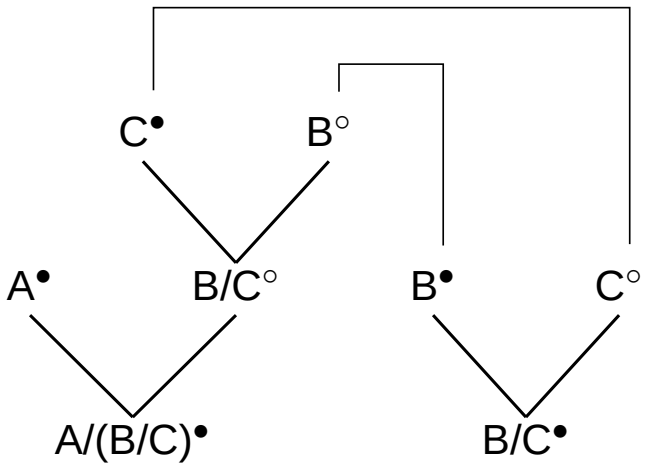
Planare Beweisnetze

- Links in Beweisnetzen für das Lambek-Kalkül überkreuzen sich nie
- Das Netz heißt dann *planar*
- Das Lambek-Kalkül ist äquivalent zu CFGs
- Beweisnetze für CGs mit Appplikation, Komposition und Anhebung
- Folglich ist Planarität der Ausdruck von Kontextfreiheit

Balancierte Zyklen

- Zyklen ergeben sich, wenn komplexe Kategorien identifiziert werden
- Im einfachen Fall völlig symmetrisch
- Die Kategorie kann aber zusammengesetzt sein (Komposition)
- Also allgemeiner: balanciert
- So oft runter wie rauf, in jedem beteiligten Zyklus
- Charakteristisch für Kategorialgrammatik

Richtung von Argumenten



- B wird vom Haupt-/ des ersten Wortes rechts verlangt und steht auch dort
- Bei Steedman: *Principle of Consistency*
- Die Selektionsrichtung für das untergeordnete C verläuft entsprechend

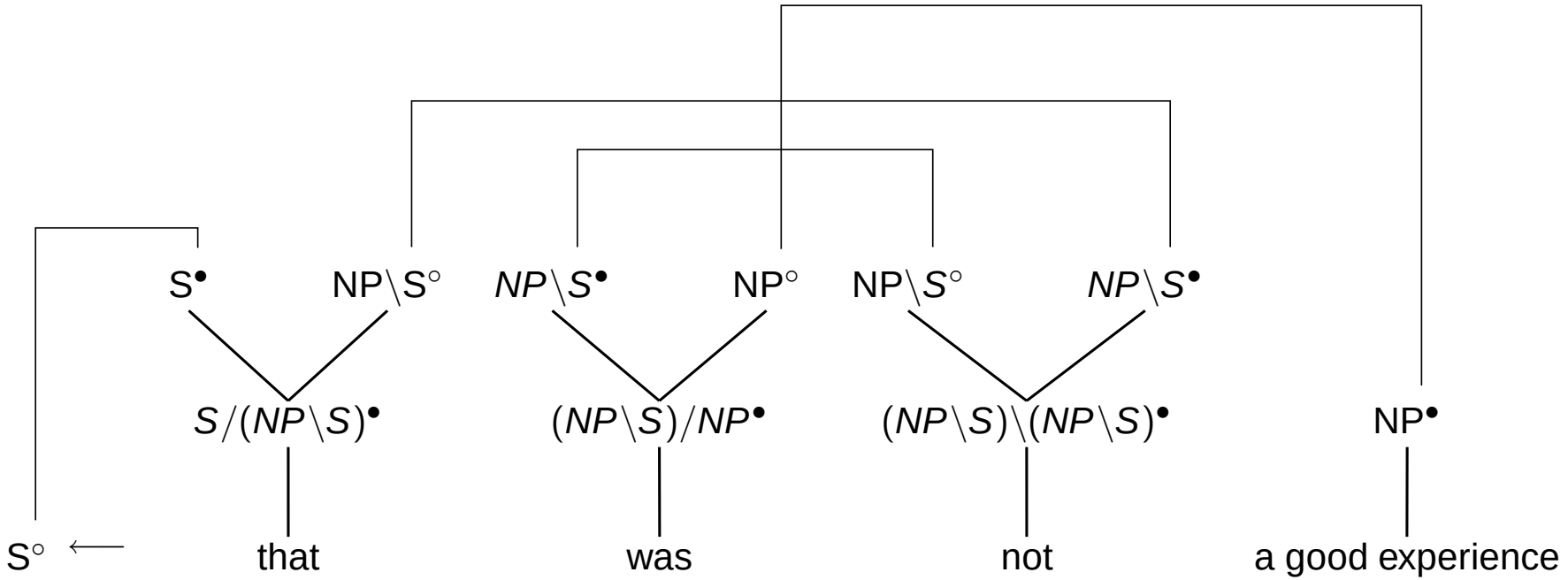
Nicht-planare Beweisnetze

- Kreuzende Komposition für mild kontext-sensitive Grammatiken
- Führt zu kreuzenden Links
- Kein Problem in der linearen Logik
- Die lässt aber viel zu viel zu, z.B. freie Permutation
- Schwierig ist die Eingrenzung, *welche* Überkreuzungen erlaubt werden
- Zu definieren: *milde* Nicht-Planarität

Kreuzende Komposition: Die Regel

$$
\begin{array}{llllll}
>\mathrm{B}_{\times} & X/Y : f & Z\backslash Y : g & \rightarrow & Z\backslash X : \lambda z.f(g(z)) \\
<\mathrm{B}_{\times} & Y/Z : g & Y\backslash X : f & \rightarrow & X/Z : \lambda z.f(g(z))
\end{array}
$$

Kreuzende Komposition: Beweisnetz

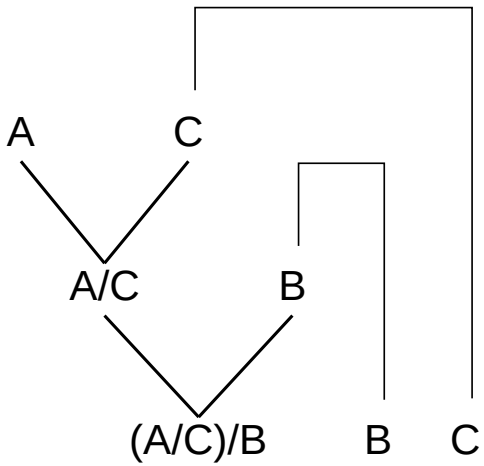


Kreuzende Komposition: Eigenschaften

- Führt zu kreuzenden Links
- (b) gilt nicht mehr
- d′ und d″ werden wichtig
- Für alle Links α über eine Lücke gelten folgende Einbettungsbedingungen:
  - (e) Von dem höchsten Knoten, von dem aus α erreichbar ist, ohne an Abzweigungen vorbeizukommen, sind auch alle Knoten unter α erreichbar (per semantic trip)
    - Das kann auch über α als Zwischenschritt sein, aber:
  - (f) Wenn man dabei aus der Spanne von α hinausgeht, darf man nicht in der selben Richtung wieder herein

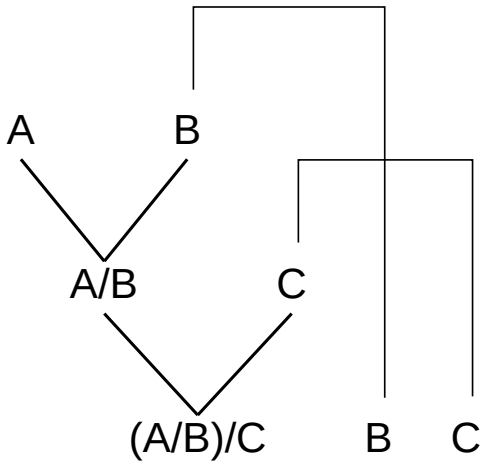
Die erste Einbettungsbedingung

Der C-Link sei α. Die folgende Situation ist erlaubt:



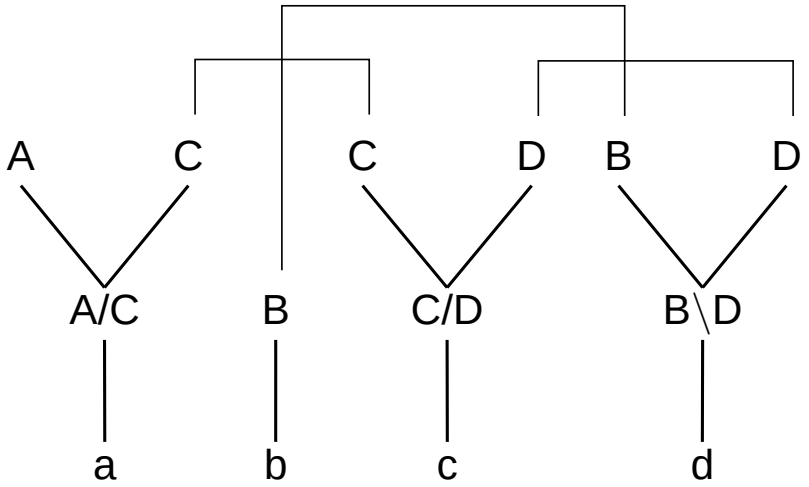
Die erste Einbettungsbedingung 2

Der C-Link sei α. Die folgende Situation wird ausgeschlossen:



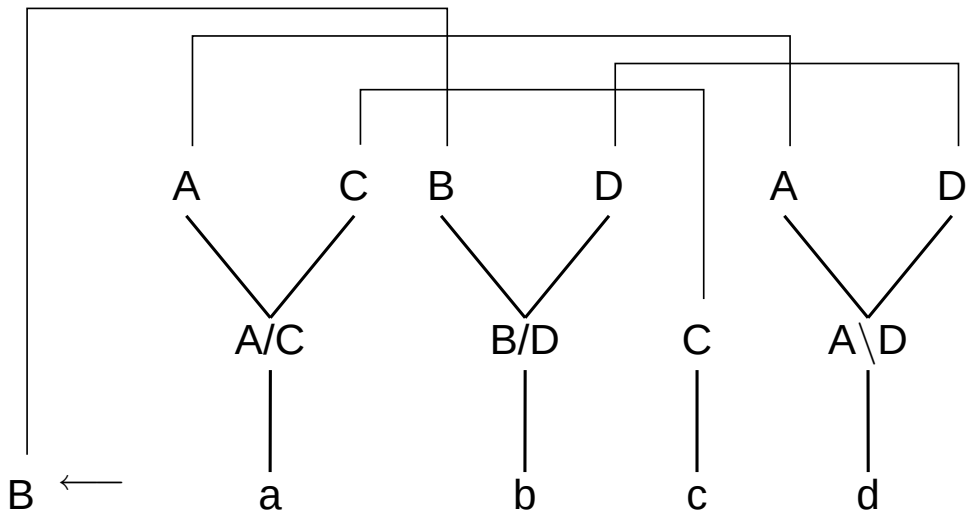
Die zweite Einbettungsbedingung

Es ist erlaubt, wieder in α (der C-Link) hineinzukreuzen:



Die zweite Einbettungsbedingung 2

Aber nicht in der selben Richtung:



CCG-Beweisnetze: Beweisskizzen

- Die Kriterien sind so gemacht, dass sie von CCG-Derivationen nicht verletzt werden können
- In einem Beweisnetz, dass sie erfüllt, kann man immer zwei adjazente und kombinierbare Wörter finden
- Steedmans *Principle of Adjacency*
- Diese zu einem Wort zusammenfassen
- Und so weiter, bis zum Ende
- So findet man immer mindestens eine gültige CCG-Derivation
- Genauer: Alle äquivalenten

Typenanhebung

$$
\begin{array}{llll}
>\mathrm{T} & X : a & \rightarrow & T/(A\backslash T) : \lambda f.f(a) \\
<\mathrm{T} & X : a & \rightarrow & (T/A)\backslash T : \lambda f.f(a)
\end{array}
$$

Typenanhebung: Diskussion

- Bedeutungsneutral
- Notwendig, um im Lambek-Kalkül gültige Herleitungen mit CG nachzubauen
- Führt aber zusammen mit kreuzender Komposition zu freier Wortstellung
- War linguistisch nie zur freien Verwendung vorgesehen
- Syntaktisch notwendig, wenn semantisch von höherem Typ (z.B. Quantoren)
- Kann im Voraus (lexikalisch) erledigt werden

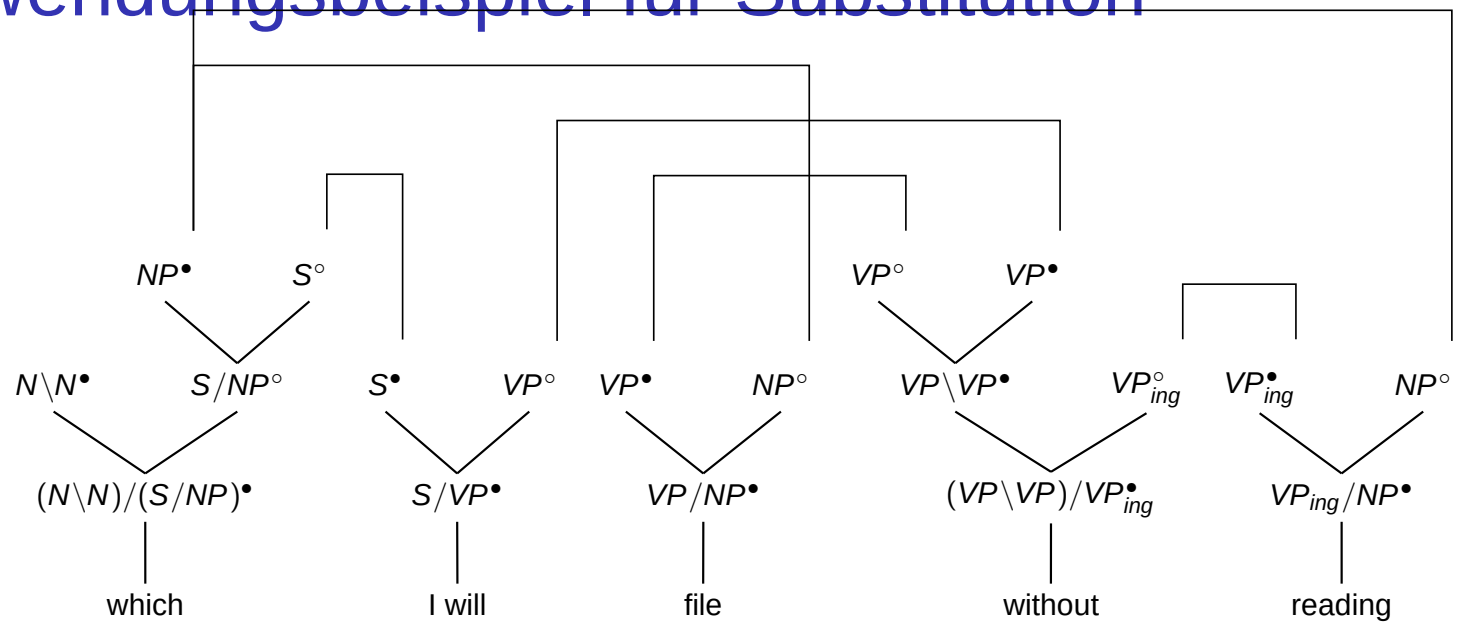
Mehrfachbindung: Das Problem

- Bisher waren alle Operationen einfach bindend
- Je ein λ pro Variable
- In der CCG gibt es für *parasitäre Lücken* eine *Subsitutions*-Regel
- Zwei Lücken, ein Füller
- Beispiele:
  - articles which John filed without reading (them)
  - cities which kids in hate
  - Bücher ohne (sie) zu lesen wegräumen

Die Substitutionsregel

$$
\begin{array}{llll}
>\mathrm{S} & (X/Y)/Z : f & Y/Z : g & \rightarrow \quad X/Z : \lambda z.f(z)(g(z)) \\
<\mathrm{S} & Z\backslash Y : g & Z\backslash(Y\backslash X) : f & \rightarrow \quad Z\backslash X : \lambda z.f(z)(g(z)) \\
>\mathrm{S}_{\times} & Z\backslash(X/Y) : f & Z\backslash Y : g & \rightarrow \quad Z\backslash X : \lambda z.f(z)(g(z)) \\
<\mathrm{S}_{\times} & Y/Z : g & (Y\backslash X)/Z : f & \rightarrow \quad X/Z : \lambda z.f(z)(g(z))
\end{array}
$$

Anwendungsbeispiel für Substitution



- backward crossing substitution mit “file” und “without reading”
- Beachte den doppelten Link
- Bedeutungsskizze: λx.without(file x, read x))

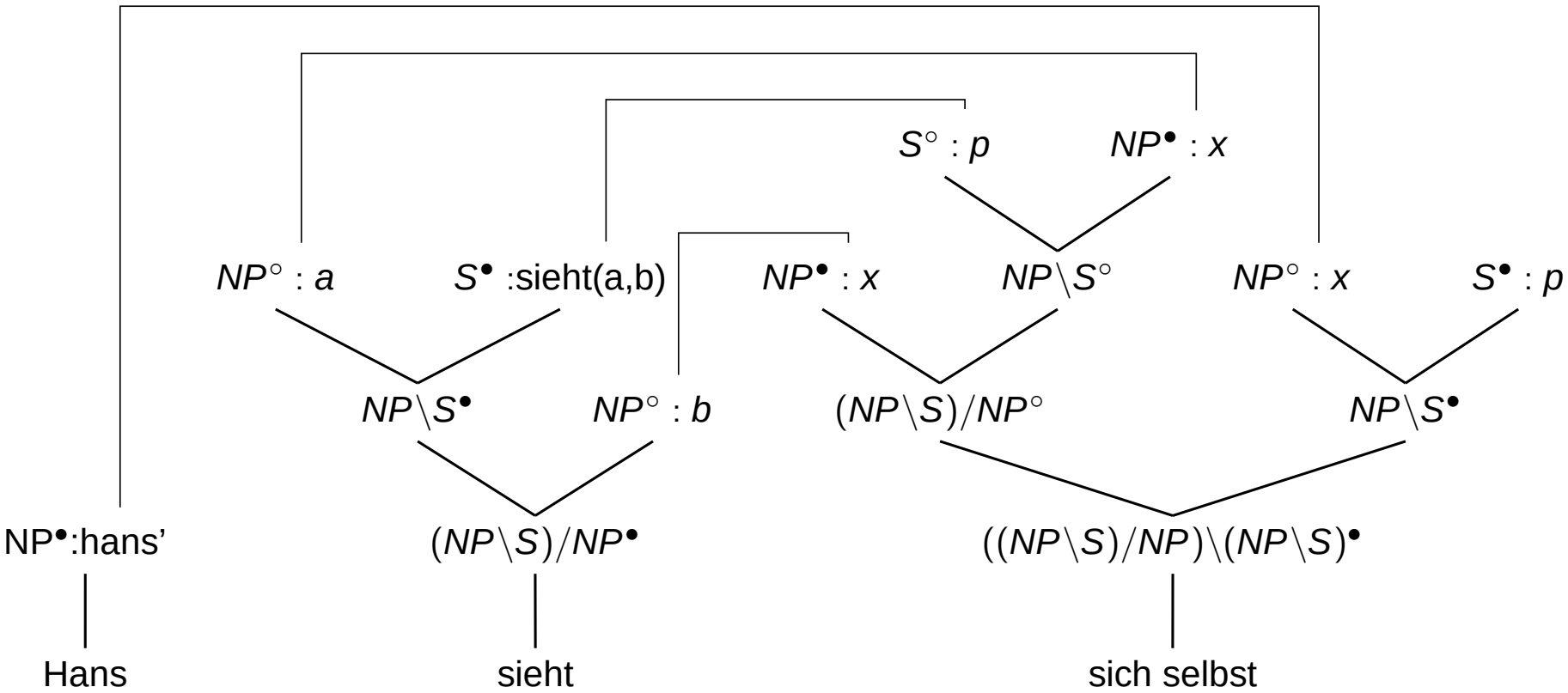
Mehrfachbindung: Diskussion

- Fragwürdiges sprachliches Phänomen
- Darstellbar, wenn man mehr in den Beweisnetzen zulässt
- Schwer abwägbare Konsequenzen
  - Wie beschränkt man in der CCG die Regel richtig?
  - Welche Beschränkungen gelten für die resultierenden Beweisnetze?

Identität von Knoten

- Es gibt semantische Indentitäts-Beziehungen in Beweisnetzen, die nicht als Links auftreten
- Beispiele:
  - Subjekt und Objekt bei Reflexivpronomen (Beweisnetz folgt)
  - Offene Argumente bei Koordination
  - Relativsätze

Reflexivpronomen



Reflexivpronomen: Diskussion

- “sich selbst” ist angehoben
- Daher die Identität der Bedeutung an den S-Knoten
- Einziger weiterer semantischer Beitrag: Identität der NPs
- Direkt an den Variablen notiert
- Bedeutung: sieht′(hans′,hans′)

Zusammenfassung

- Beweisnetze sind für die Kategorialgrammatik, was Bäume für Phrasenstrukturgrammatiken sind
- Eindeutige Struktur
- Parallele Syntax und Semantik

Was ich nicht erzählt habe

- Wie man Kategorien und λ-Terme in eine Schreibweise vereinigt
- Dass die Kategorie N (mit Mengen als Bedeutung) dekomponierbar ist
- Weitere Varianten, wie man mit Typenanhebung umgehen kann
- Multimodale CCG

Was noch nicht erforscht ist

- Weitere (sinnvollere?) Unterklassen von Beweisnetzen
- Effizientes Parsen mit Beweisnetzen (in Arbeit: T. Fowler)
- Lexikalische Ambiguität; idiomatische vs. wörtliche Lesarten (ansatzweise)

Ende

Vielen Dank für die Aufmerksamkeit!