Einführung in die Semantik, 12. Sitzung
Quantoren und Variablen, Skopus

Götz Keydana

Göttingen
23. Januar 2007

Variablenbindung bei Quantoren

- Quantifier-raising

Multiple Quantoren und Skopus

- Quantifier-raising
- Quantifying-in
- Cooper-storage

Der Befund

Unsere Analyse der Bindung von Anaphern beruht auf der Tatsache, daß bei Namen oder definiten NPs als Antezedens Sätze des folgenden Typs semantisch äquivalent sind:

(1) Pynchon$_i$ sieht sich$_i$ im Spiegel.

(2) Pynchon$_i$ sieht Pynchon$_i$ im Spiegel.

Ist der Antezedens ein Quantor, so gilt die Äquivalenz nicht:

(3) Jeder$_i$ sieht sich$_i$ gerne im Spiegel.

(4) Jeder$_i$ sieht jeden$_i$ gerne im Spiegel.

Die Lösung: Pränex-Normalformen

Eine Lösung für das Problem kennen wir bereits aus der Prädikatenlogik.

Satz (3) hat in der Prädikatenlogik folgende Repräsentation:

(3′) $\forall(x)S(x,x)$

Quantorenbewegung

Die Pränexnormalform erlaubt es uns, aufgrund von Quantorenbewegung alle Quantoren eines Ausdrucks vor die quantorenfreie Matrix zu ziehen.

Quantorenbewegung in $F_{deutsch}$

Wenn wir uns die Quantorenbewegung der Prädikatenlogik zunutze machen wollen, müssen wir einen Satz wie

(5) $[_S\ [_{NP}\ [_{Det}\ \text{Jeder}]\ [_N\ \text{Schriftsteller}]]_i\ [_{VP}\ [_V\ \text{sieht}]\ [_{NP}\ \text{sich}]_i]]$

in eine semantische Repräsention übersetzen, die wir paraphrasieren können als:

(5′) Jeder Schriftsteller hat die Eigenschaft, ein x zu sein, sodaß gilt: x sieht x.

Wenn wir die syntaktische Struktur von (5) zugrunde legen, verstößt eine Übersetzung im Sinne von (5′) gegen das Kompositionalitätsprinzip.

Ein möglicher Ausweg: LF

Einen Ausweg bietet die Government & Binding Theory:

- GB kennt Bewegung (Move-α) an jedem Punkt in der Derivation.
- GB (wie auch der Minimalismus) geht davon aus, daß die Derivation an einem bestimmten Punkt, der Surface Structure (im Minimalismus bei Spell Out) verzweigt:
  - Ein Weg führt in die Phonologische Form (PF),
  - ein zweiter in die Logische Form (LF).
  - Wenn wir LF als die Repräsentationsebene annehmen, von der die Übersetzung in semantische Repräsentationen ausgeht, steht nichts der Annahme im Wege, daß Quantorenanhebung in der Syntax nach SS (bzw. Spell Out) auf dem Weg zur LF stattfindet.

Syntaktische Quantorenanhebung in GB

Für SS nehmen wir die syntaktische Struktur in (5) an:

(5) $[_S\ [_{NP}\ [_{Det}\ \text{Jeder}]\ [_N\ \text{Schriftsteller}]]_i\ [_{VP}\ [_V\ \text{sieht}]\ [_{NP}\ \text{sich}]_i]]$

In LF wird der Quantor präfigiert. Bei der Bewegung läßt er eine koindizierte Spur (t) zurück:

(6) $[_S\ [_{NP}\ \text{Jeder Schriftsteller}]_i\ [_S\ [_{NP}\ \text{t}]_i\ [_{VP}\ [_V\ \text{sieht}]\ [_{NP}\ \text{sich}]_i]]]$

Ein alternatives Format für (6) ist (7):

(7) $[_S\ [_{NP}\ \text{Jeder Schriftsteller}]\ \lambda i\ [_S\ [_{NP}\ \text{t}]_i\ [_{VP}\ [_V\ \text{sieht}]\ [_{NP}\ \text{sich}]_i]]]$

Das Format in (7) ist für die semantische Interpretation hilfreich, syntaktisch aber nicht unbedenklich. Es geht auf Heim & Kratzer (1998) zurück.

Die Interpretation

Wir gewinnen die Bedeutung von (7) durch funktionale Applikation:

(8) $[\![[_S[_{NP}\text{Jeder Schriftsteller}][_S\lambda i[_{NP}\,t]_i[_{VP}[_V\text{sieht}][_{NP}\text{sich}]_i]]]]\!]$

(a.) $\lambda g\lambda w[[\![\text{Jeder Schriftsteller}]\!](g)(w)$
$([\![\lambda i[_S[_{NP}\,t]_i[_{VP}[_V\text{sieht}][_{NP}\text{sich}]_i]]]\!](g)(w))$

Die Bedeutung von *Jeder Schriftsteller* ist ein GQ:

(9) $[\![\text{Jeder Schriftsteller}]\!] =$
$\lambda g\lambda w\lambda P[\lambda x[x\text{ist ein }\mathbf{Schriftsteller}\text{ in }w] \subseteq P]$

Die Gesamtbedeutung muß daher (10) sein:

(10) $\lambda g\lambda w[\lambda x[x\,\text{ist ein }\mathbf{Schriftsteller}\text{ in }w] \subseteq \lambda x[x\,\mathbf{sieht}\,x\text{ in }w]]$

Die Bedeutung des λi-Terms

Wir berechnen die Bedeutung des λi-Terms unter der Annahme, daß die Bedeutung der Spur t die eines gebundenen Pronomens ist:

(11) $[\![\lambda i [_S [_{NP}\, t]_i [_{VP} [_V \text{sieht}] [_{NP} \text{sich}]_i]]]\!]$

(a.) $\lambda g \lambda w [[\![\text{sieht}]\!](g)(w)([\![\text{sich}_i]\!](g)(w))([\![t_i]\!](g)(w))]$

(b.) $\lambda g \lambda w [\lambda g' \lambda w' \lambda y \lambda x [x \textbf{ sieht } y \text{ in } w'](g)(w)$
$(\lambda g'' \langle i \in DOM(g'') \rangle \lambda w'' [g''(i)](g)(w))$
$(\lambda g''' \langle i \in DOM(g''') \rangle \lambda w''' [g'''(i)](g)(w))]$

(c.) $\lambda g \langle i \in DOM(g) \rangle \lambda w [\lambda y \lambda x [x \textbf{ sieht } y \text{ in } w]$
$(g(i))](g(i))]$

(d.) $\lambda g \lambda w [g(i) \textbf{ sieht } g(i) \text{ in } w]$

Rekursion: Varianten von Variablenzuweisungen

(11d.) ist eine Funktion von Zuweisungsfunktionen, die i im Definitionsbereich haben, in Propositionen.

Aus der Prädikatenlogik wissen wir, daß sich eine Funktion g' von g dadurch unterscheidet, daß sie einer Variable (möglicherweise) eine andere Referenz zuweist.

Für die Semantik von $\mathsf{F}_{\text{deutsch}}$ gilt: Die Variante $g' \in G$ unterscheidet sich von $g \in G$ darin, daß sie möglicherweise dem Index i einen anderen Wert zuweist als g.

Wenn wir die Zuweisung $g[i \rightarrow x]$ schreiben, können wir für λi-Terme folgende Bedeutung geben:

(12) $\quad [\![\lambda i[_{\mathsf{S}} \Phi]]\!] = \lambda g \lambda w \lambda x_i [[\![\Phi]\!](g[i \rightarrow x_i])(w)]$

Weiter zu (11.d)

Wir können daher die Bedeutung unseres λi-Terms (11d.) relativ zu der Zuweisung $g[x \rightarrow x_i]$ folgendermaßen angeben:

(13) $\lambda g \lambda w \lambda x_i [\lambda g' \lambda w' [g'(i)\ \textbf{sieht}\ g'(i)\ \text{in}\ w'](g[i \rightarrow x_i])(w)]$

(a.) $\lambda g \lambda w \lambda x_i [g[i \rightarrow x_i]\ \textbf{sieht}\ g[i \rightarrow x_i]\ \text{in}\ w]$

(b.) $\lambda g \lambda w \lambda x_i [x_i\ \textbf{sieht}\ x_i\ \text{in}\ w]$

Die Gesamtbedeutung von (7) für eine Variablenbelegung $g[i \longrightarrow x_i]$

(11) $\lambda g \lambda w[\lambda g^{prime} \lambda w' \lambda P[\lambda x_i[x_i$ist ein **Schriftsteller** in $w']$
$\subseteq P](g)(w)(\lambda g'' \lambda w'' \lambda x_i[x_i$ **sieht** x_i in $w](g)(w))$

(a.) $\lambda g \lambda w[\lambda P[\lambda x_i[x_i$ist ein **Schriftsteller** in $w]$
$\subseteq P](\lambda g \lambda w \lambda x_i[x_i$ **sieht** x_i in $w])$

(b.) $\lambda g \lambda w[\lambda x_i[x_i$ist ein **Schriftsteller** in $w]$
$\subseteq \lambda x_i[x_i$ **sieht** x_i in $w]])$

Wie in der Prädikatenlogik werden die Varianten von g rekursiv appliziert, um den Wahrheitswert des Satzes relativ zu einer Welt zu errechnen.

Skopusambiguität: Die Daten

Enthält ein Satz mehr als einen Quantor, so können Skopusambiguitäten entstehen.

Beispiele für (vermeindliche) Skopusambiguität

(14) Jemand begleitete jeden.
Someone accompanied everybody.

(15) Ein Mechaniker überprüfte jedes Flugzeug.
A mechanic inspected every plane.

(16) Kein Student hat zwei Bücher von Chomsky gelesen.
No student read two books by Chomsky.

Die Pränex-Normalform

Bewegung der Quantoren in das Präfix des (prädikatenlogischen) Ausdrucks ermöglicht Disambiguierung:

(14′) Jemand begleitete jeden.

(a.) $\exists x \forall y B(x, y)$

(b.) $\forall y \exists x B(x, y)$

Quantifier raising

Wir haben bereits Quantorenhebung als Äquivalent zu Pränexnormalformen angesetzt.

- Wir postulieren LF als Repräsentationsebene nach Spell Out.
- Wir lassen syntaktische Operationen (Move-α) zwischen Spell Out und LF zu.
- Wir bewegen die Quantoren in der oben besprochenen Weise.

Konfigurationaler Ansatz

Dieser Ansatz ist konfigurational, weil er Disambiguierung über unterschiedliche syntaktische Strukturen (Konfigurationen) modelliert. Weil die Interpretation auf der Ebene der LF erfolgt, ist Ambiguität auf anderen Ebenen zulässig.

Parallele Evaluation von Syntax und Semantik

Alternativ können wir postulieren, daß syntaktische Struktur permanent in semantische Struktur übersetzt wird. (Wir haben $F_{deutsch}$ weitgehend in diesem Sinne konzipiert.

- Theorien, in denen semantische Struktur parallel zu syntaktischer Struktur gebaut wird, nennt man derivationale Interpretationstheorien.
- Derivationale Interpretation ist nicht in Grammatiken des GB-Typs implementierbar.
- Montague grammar ist das berühmteste Beispiel für eine derivationale Semantik.

Konsequenzen des derivationalen Ansatzes

Betrachten wir folgendes Beispiel:

(17) Thompson sieht Ellison.

Wir können den Satz grundsätzlich auf zwei Arten repräsentieren:

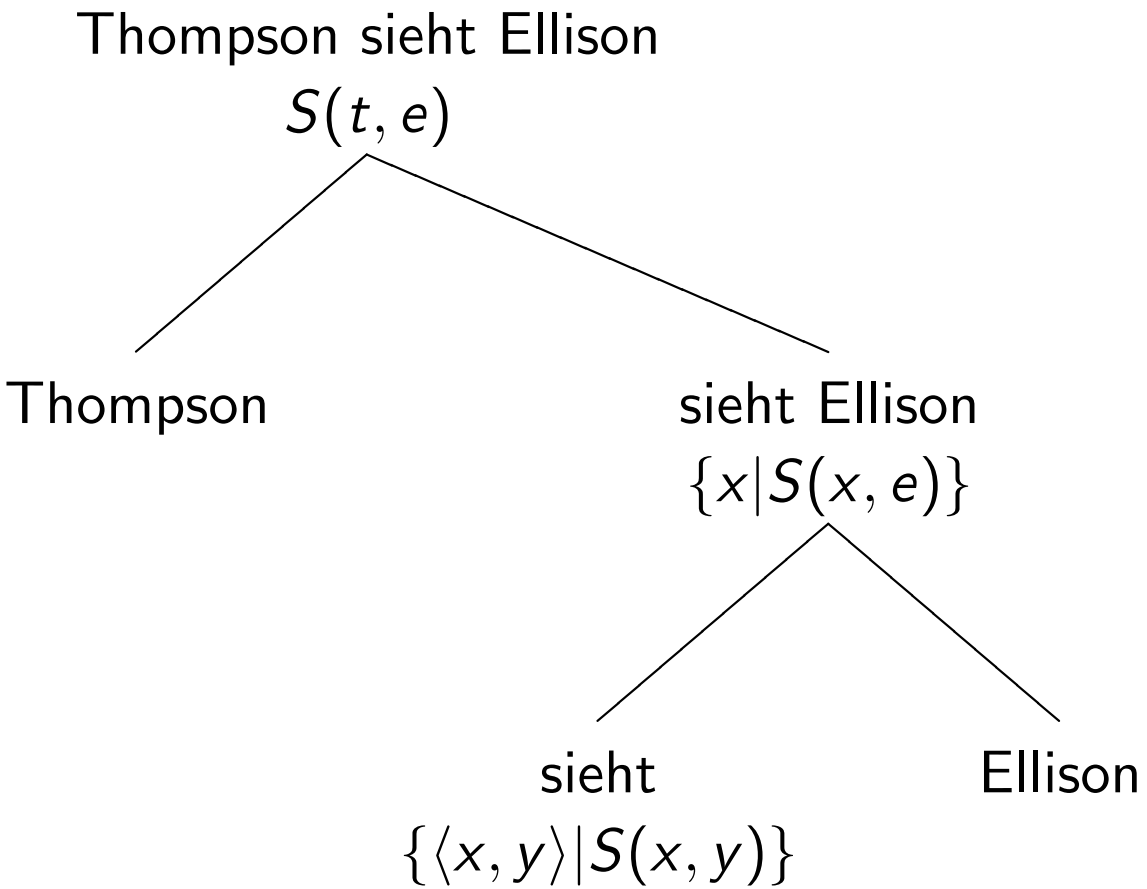
(17′) Thompson + _____ sieht Ellison
Die Eigenschaft, Ellison zu sehen, ist eine Eigenschaft von Thompson.

(17″) Thompson sieht + _____
Die Eignschaft, von Thompson gesehen zu werden, ist eine Eigenschaft von Ellison.
Motiviert kann die zweite Analyse sein z.B. in Sätzen wie

(18) Thompson sieht und Pynchon kennt Ellison

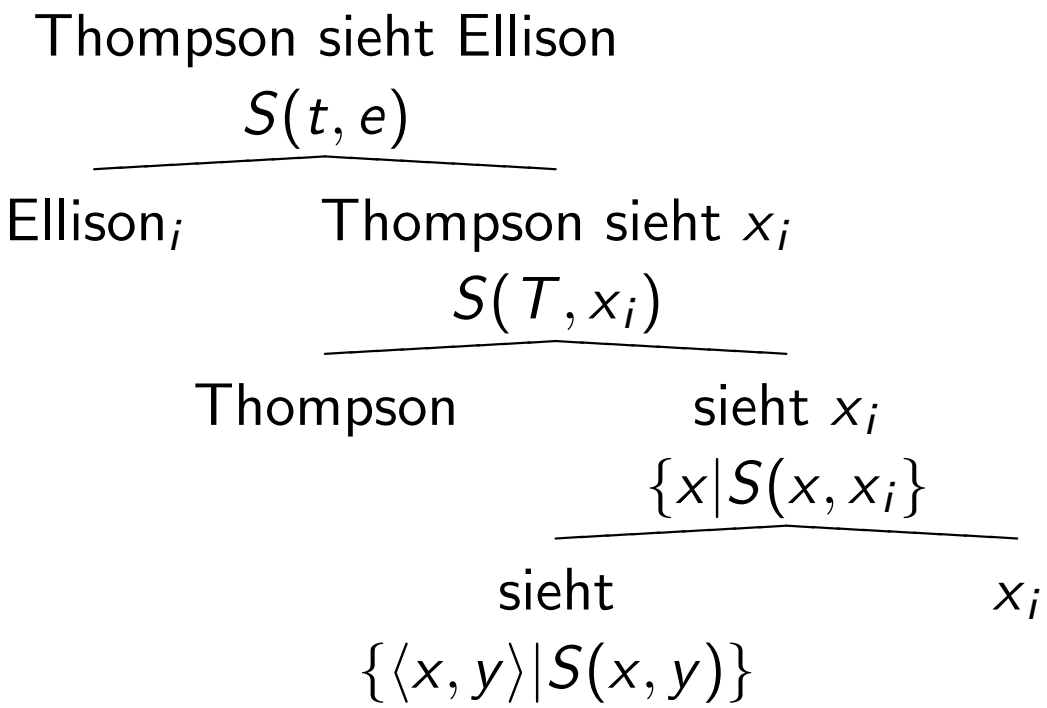
Die Derivation (17′)

(17′) können wir in folgendem Strukturbaum abbilden:



Die Derivation von (17″)

Für (17″) nehmen wir folgende Struktur an:



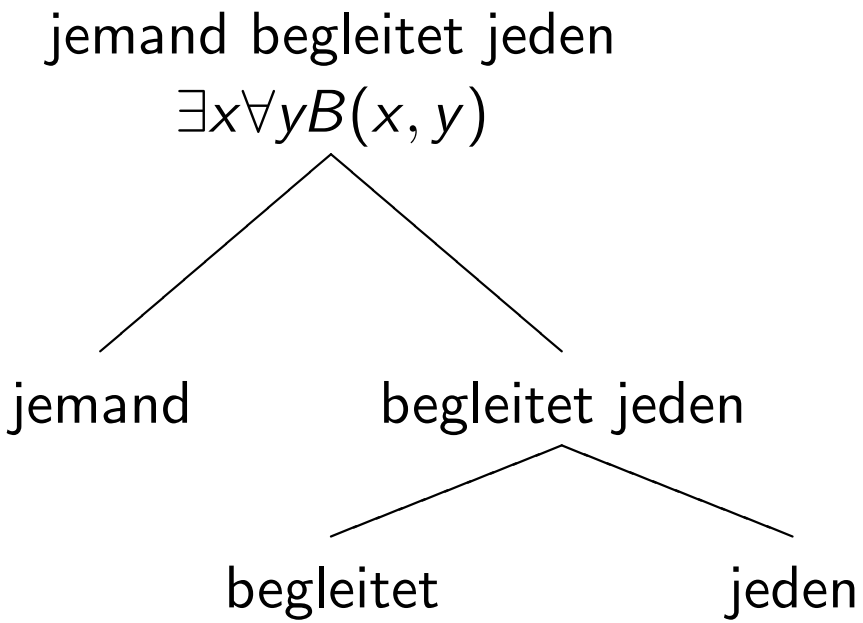
Quantifying-in

Im letzten Schritt der Derivation (17″) schließen wir die ungesättigte propositionale Funktion durch Identifizierung der Variable x_i mit einer Konstante, hier Ellison.

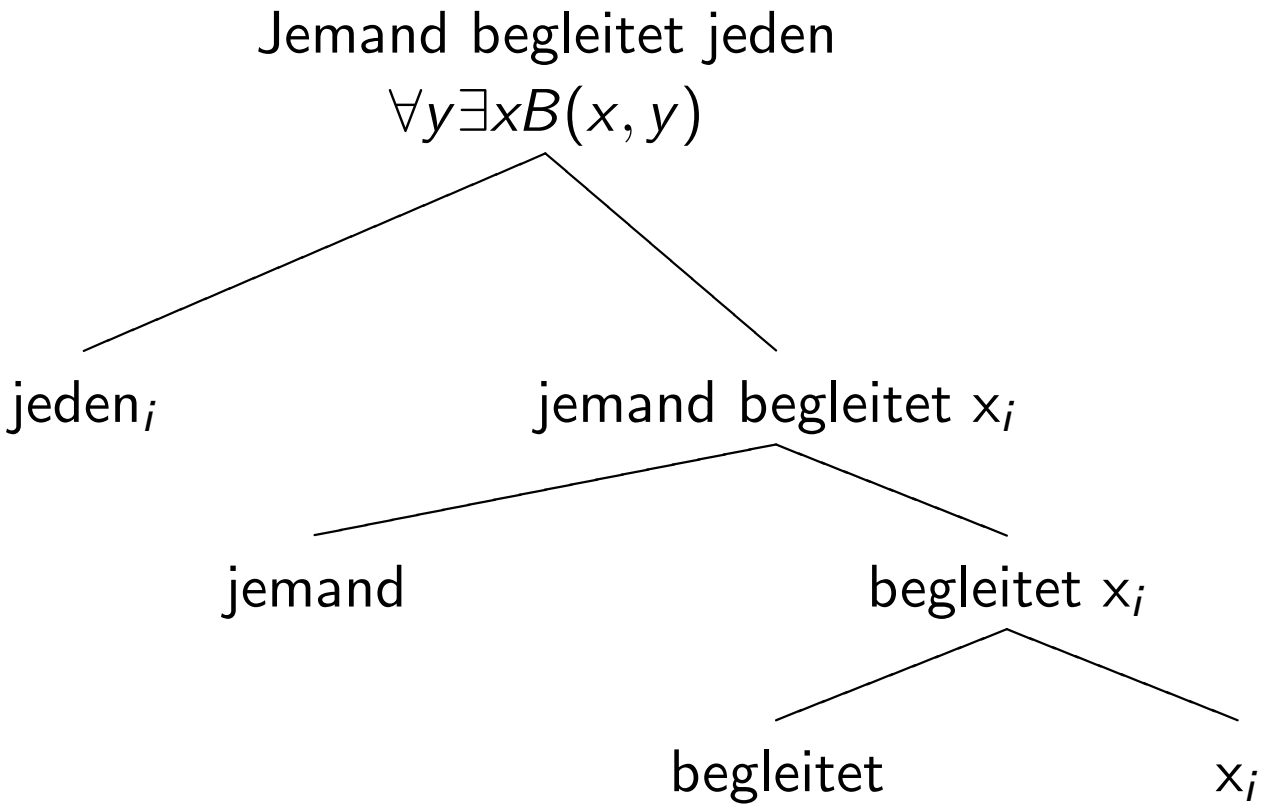
Dieses Verfahren nennt man *Quantifying-in*.

(17′) und (17″) sind semantisch äquivalent. Wir können aber mit diesem Verfahren die verschiedenen Lesarten von Sätzen mit mehr als einem Quantor so modellieren, daß die Struktur zu keinem Zeitpunkt in der Derivation ambig ist.

Satz (14): Weiter Skopus des Existenzquantors



Satz (14): Enger Skopus des Existenzquantors



Vergleich Quantifier-raising vs. Quantifying-in

- Beide Verfahren beruhen auf der Pränex-Normalform.
- Quantifier-raising erlaubt semantisch ambige Strukturen in der Derivation.
- Quantifying-in hat zu keinem Zeitpunkt in der Derivation Ambiguität.
- Quantifier-raising beruht auf Bewegung und LF.
- Quantifying-in hat weder Bewegung noch LF.
- Quantifier-raising (nach Spell Out) hat keine Probleme mit PF.
- Quantifying-in bedarf eines Mechanismus auf PF, der dafür sorgt, daß Quantoren an der Stelle phonologisch realisiert werden, wo die mit ihnen indizierte Variable steht.

Cooper-storage

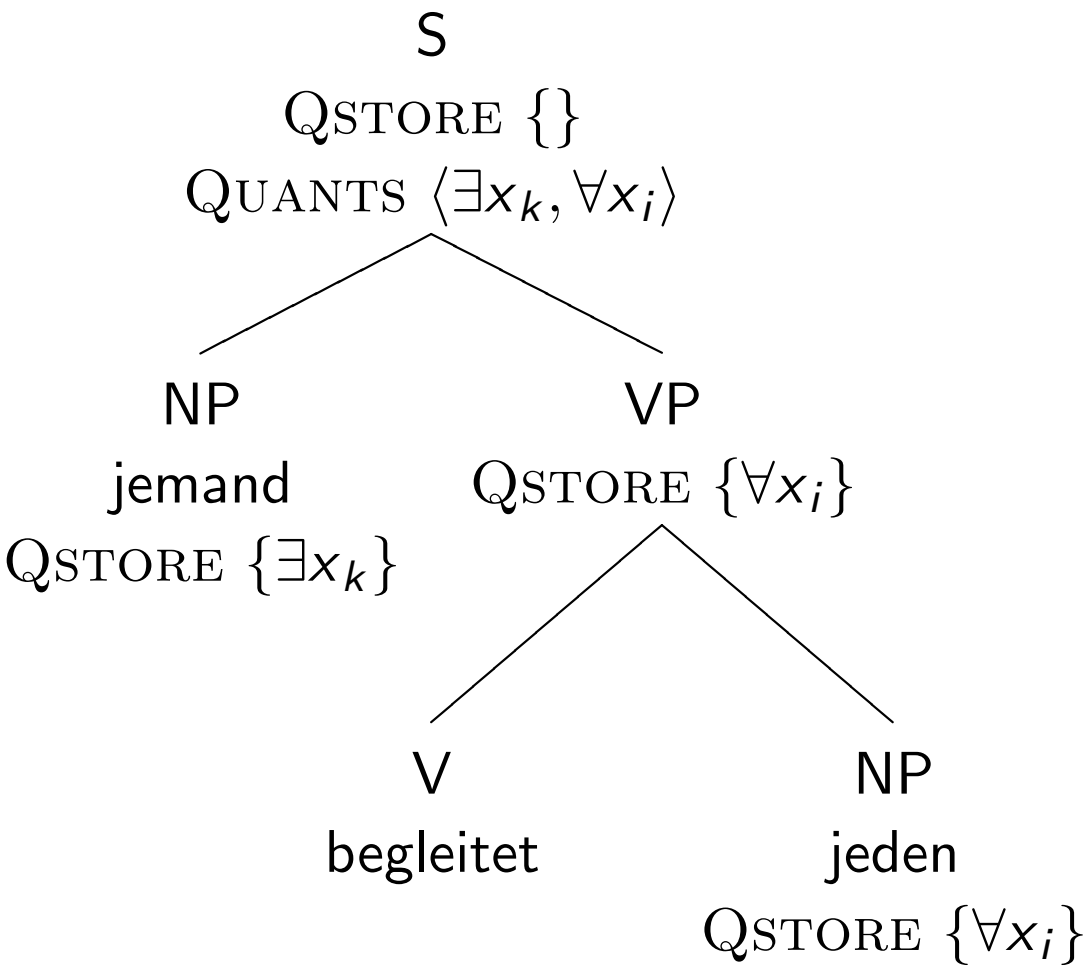
Cooper-storage (benannt nach R.Cooper) ist eine dritte Möglichkeit, mit Skopus-Ambiguität umzugehen.

- Cooper storage basiert auf Quantifying-in.
- Cooper storage verzichtet auf unterschiedliche syntaktische Repräsentationen für verschiedene Lesarten: Skopus-Ambiguität ist ein rein semantisches Phänomen.
- Auf diese Weise verschwindet das PF-Problem von Quantifying-in.
- Cooper storage ist von Cooper (1983) in Montague grammar entwickelt worden. Pollard & Sag (1994) benutzen Cooper storage in HPSG.

Die Grundlagen

- Quantoren werden wie andere NPs in die Derivation eingeführt.
- Sie müssen aber nicht an der Stelle ausgewertet werden, wo sie stehen.
- Die Interpretation eines Quantors kann bis zu einem beliebigen Zeitpunkt in der Derivation zurückgestellt werden.
- Die Interpretation eines Satzes ist nur dann adäquat, wenn sämtliche Quantoren interpretiert sind.

Satz (14): Weiter Skopus des Existenzquantors



Satz (14): Enger Skopus des Existenzquantors