KOMPOSITIONALE DERIVATION

Die formale Semantik befasst sich mit der Frage, wie natürlichsprache Ausrücke zu ihrer Bedeutung (*Denotation*) kommen; sie bildet ein Teilgebiet der Linguistik, das Aspekte der Logik, Linguistik, Philosophie, Informatik, Kognitionswissenschaft und anderen Disziplinen vereint. Im ersten Teil dieses Skriptums wurde eine Übersicht über unterschiedliche Theorien zur Bedeutung geboten und dargelegt, dass diese über *Wahrheitsbedingungen* definiert werden müssen. Konkret drücken Sätze *Propositionen* aus, wobei eine Proposition formal mit einer Menge von Situationen gleichgesetzt wird.

(1) *Sätze denotieren Mengen von Situationen (aka Propositionen)*

$$\underbrace{[\![\mathbf{\Sigma}]\!]}_{\text{Σ in objektsprache}} \quad =_{\text{Def}} \quad \{s|\ \underbrace{\Sigma}_{\text{Metasprachliches Vorskommen von Σ}} \quad \text{in s}\ \}$$

“die Menge all jener Situationen, in denen Σ wahr ist”

Welche Situationen Teil der Proposition sind, und welche nicht, wird durch die *Wahrheitsbedingungen* des Satzes festgelegt. Zur Erinnerung: die Wahrheitsbedingungen eines Satzes legen fest, wie die Welt aussehen muss, damit dieser Satz wahr ist. Ganz allgemein kann dies schematisch so wie in (2) dargestellt werden. Die Wahrheitsbedingungen sind durch die Box markiert:

(2) *Tarskis T-Schema*
Σ ist wahr, $\boxed{\text{genau dann, wenn p}}$

Wahrheitsbedingungen werden durch die *Form* des Satzes gegeben. Der Satz *Es regnet* hat demnach andere Wahrheitsbedingungen als der Satz *Es regnet nicht* oder der Satz *Maria lacht.* Im Schema (1) werden die Wahrheitsbedingungen durch das metasprachliche Vorkommen von Σ definiert.

Aus diesen Annahmen kann nun der Begriff der Wahrheit abgeleitet werden. Ein Satz ist in einer gegebenen Situation wahr, genau dann, wenn diese Situation ein Element der Proposition ist, die dieser Satz ausdrückt. Etwas formaler:

(3) *Definition von Wahrheit (drei äquivalente Versionen)*

Für jeden Satz Σ und für jede Situation s gilt:
a. Σ ist wahr in s gdw. $s \in \{s|\ \Sigma \text{ ist wahr in s}\}$
b. $[\![\mathbf{\Sigma}]\!] = 1$ in s gdw. $s \in [\![\mathbf{\Sigma}]\!]$
c. $[\![\mathbf{\Sigma}]\!] = 1$ in s gdw. $s \in \{s|\ \Sigma \text{ ist wahr in s}\}$

Dieser zweite Teil des Skriptums erklärt, *wie* man die Bedeutung von Sätzen und anderen komplexen Ausdrücken *kompositional* aus der Bedeutung der Teile ableiten kann. Eine Theorie, die dazu in der Lage ist, besteht aus zwei Teilen:

(4) a. *Lexikalische Bedeutungsregeln* für die atomaren sprachlichen Ausdruck (das sind die Wörter/freie Morpheme wie *Tisch*, *lachen, jeder*, *nicht,..* und eine Teilmenge der gebundenen Morpheme, wie z.B. *un-* in *unklar* oder ξανά- in ξανάδιαβάζω)

b. *Semantischen Kompositionsregeln*, die einfache oder komplexe Bedeutungen rekursiv zu neuen Bedeutungen kombinieren.

Hinweis zum Skriptum. Dieser Teils des Skriptums enthält viel mehr Information, als wir im Seminar behandeln können. Im Laufe des Semesters werden nur die ersten Schritte in Richtung einer kompositionalen Semantik skizziert werden, die Ausführungen werden sich dabei auf eine einfache Semantik von transitiven Verben, so wie sie in Abschnitt 1 vorgestellt wird, beschränken. Wie immer sind natürlich nur jene Teile, die wir auch tatsächlich im Seminar besprechen, für die Prüfung relevant. Abschnitte 2 und 3 führen kurz in jene Techniken ein, die heute allgemein in der semantischen Forschung verwendet werden. Eine letzte Anmerkung: wir beginnen zwar mit einer Semantik, die auch Situationen berücksichtigt, werden die aber im weiteren Verlauf meistens ignorieren, um die Ausführungen etwas einfacher zu halten.

---

1. PRÄDIKATE ALS MENGEN

1.1. LEXIKONEINTRÄGE FÜR EINSTELLIGE PRÄDIKATE

Prädikate sind Ausdrücke, die ein oder mehrere Argumente benötigen, um einen vollständigen Satz zu erzeugen. Man unterscheidet zwischen Prädikaten, die mir einem einzigen Argument einen Satz bilden, sogenannten *einstelligen* Prädikaten (siehe (5)), und Prädikaten, die mehr als ein Argument brauchen, wie *kennen* oder *mögen* (dazu später)[1].

(5) schlafen$_V$, lachen$_V$, grün$_A$, müde$_A$, fröhlich$_A$, Hund$_N$, Tisch$_N$, Linguist$_N$, unten$_P$,...

Semantisch können einstellige Prädikate als Menge aufgefasst werden, genauer gesagt als die Menge der Elemente, auf die das Prädikat zutrifft. Dabei spielt es keine Rolle, ob das Haupt des Prädikates der Kategorie Verb, Adjektiv, Nomen oder Präposition angehört. Relativiert auf Situationen (s. Teil 1 des Skriptums) ergibt dies die Denotationen in (6):

(6) *Lexikalische Bedeutungsregeln*

a. ⟦**schlafen**⟧ = für jede beliebige Situation s: {x|x schläft in s}
b. ⟦**müde**⟧ = für jede beliebige Situation s: {x|x ist müde in s }
c. ⟦**Linguist**⟧ = für jede beliebige Situation s: {x|x ist ein Linguist in s}
d. ⟦**unten**⟧ = für jede beliebige Situation s: {x|x ist unten in s}

Die in (6) bereitgestellte Information ist Teil des mentalen *Lexikoneintrags* der Wörter. Man bezeichnet diese Definition des semantische Beitrags eines jeden Lexikoneintrags auch als die *Bedeutungsregel* für dieses Wort. Wenn ein Sprecher einer Sprache z.B. die Bedeutung des Wortes *schlafen* kennt, dann hat er/sie etwas ähnliches wie die Regel (6)a erlernt. Neben einfachen Prädikatsbedeutungen wurden im ersten Teil des Skriptum auch lexikalische Einträge für Namen eingeführt. Namen *referieren* auf das Individuum, das der Name denotiert:

(7) ⟦**Hans**⟧ = das Individuum mit dem Namen ‘Hans’

1.2. VON WORTBEDEUTUNGEN ZU SATZDENOTATIONEN

Im Folgenden soll die Bedeutung von einfachen intransitiven Sätzen aus den Bedeutungen der Teile abgeleitet werden. Beginnen wir mir dem, was wir bereits wissen: den Bedeutungsregeln für Namen und einstellige Prädikate, sowie die Denotation des ganzen Satzes. Außerdem wurde schon festgestellt, dass Sätze Mengen von Situationen (also Propositionen) denotieren. Welche

---

[1]Noch einfacher sind Verben wie *regnen* oder *schneien*, die ohne Argumente auszukommen scheinen. Semantisch handelt es sich hier um *0-stellige* Prädikate.

Situationen Teil dieser Menge sind, hängt von den Wahrheitsbedingungen des Satzes ab. Genauer gesagt ist eine Situation s genau dann in der Menge der Satzdenotation, wenn der Satz in s wahr ist. Weiters ist ein Satz genau dann in einer Situation s wahr, wenn die Subjektsbedeutung ein Element der VP-Denotation in s ist. Der Satz *Hans schläft* ist z.B. in einer Situation s genau dann wahr, wenn Hans ein Element der Menge ist, welche die schlafenden Individuen in s umfasst. In mengentheoretischer Notation kann dies so ausgedrückt werden.[2]

(8) Für jede Situation s,

$$
\begin{aligned}
[\![\textbf{Hans schläft}]\!] &= 1 \text{ in s gdw } [\![\textbf{Hans}]\!] \in [\![\textbf{schlafen}]\!] \text{ in s} \\
&= 1 \text{ in s gdw Hans in s schläft}
\end{aligned}
$$

An diesem Punkt ist es notwendig, kurz etwas genauer auf die Beziehung zwischen Syntax und Semantik einzugehen. In unserem Modell[3] werden Sätze in der Syntax generiert und anschließend in der Semantik interpretiert. Da die Syntax Baumstrukturen produziert, die aus Knoten mit unterschiedlichen Kategorienamen (*Labels;* NP, V°,...) bestehen, werden auch in der Semantik ganze Bäume verarbeitet, und nicht etwa nur eine einfache Reihe von Wörtern, oder etwa Bäume, deren Knoten keinen Kategorienamen tragen. Für unser Beispiel *Hans schläft* bedeutet dies, dass die Denotationsklammern ([\![.]\!]) einen vollständigen Strukturbaum umschließen, und nicht nur, so wie in (8), zwei einfache Wörter. (9) macht dies explizit:

(9)

$$
\left[\!\!\left[\; [_{\text{IP}} \; [_{\text{NP} (=\beta)} \; [_{\text{N°}} \; \textbf{Hans}]] \; [_{\text{VP} (=\gamma)} \; [_{\text{V°}} \; \textbf{schläft}]]] \; \right]\!\!\right] = \underbrace{1 \text{ in s gdw Hans in s schläft}}_{\text{Satzbedeutung (definiert durch Wahrheitsbedingungen des Satzes)}}
$$

Jetzt kann endlich das Ziel konkret definiert werden, das sich für die Analyse stellt. Es besteht darin, die Wahrheitsbedingungen des Satzes, die auf der rechten Seite des Gleichheitszeichen (‘=’) gegeben sind, aus dem komplexen syntaktischen Ausdruck auf der linken Seite kompositional abzuleiten. Was ‘kompositional ableiten’ bedeutet, ist dabei genau definiert: wenn ein Ausdruck β aus einem Ausdruck α abgeleitet wird, dann muss es möglich sein, α durch die Anwendung einer endlichen Anzahl von Operationen in β zu übersetzen oder umzuformen. Die Operationen müssen dabei die Eigenschaft besitzen, dass sie Ausdrücke mit unterschiedlicher Form, aber immer mit gleicher Bedeutung produzieren. Ein klassisches Beispiel für eine Ableitung in diesem Sinne sind die Regeln der Arithmetik, illustriert durch (10):

(10)

a.		7	+	(5 x 2)	=
b.	=	7	+	10	=
c.	=	17			

[2]Das Symbol ‘∈’ bezeichnet die ‘Element von’ Beziehung; ‘gdw’ bedeutet ‘genau dann wenn’.

[3]Hier wird ein sogenanntes *syntax-zentriertes* Modell vorgestellt, das auf Noam Chomsky zurückgeht, und in dem die syntaktische Komponente Ausdrücke produziert, die im Anschluss daran in der Semantik interpretiert werden (Heim und Kratzer 1998). Es ist wichtig zu betonen, dass es auch alternative Systeme (z.B. *kategoriale Grammatiken*) gibt, in denen Syntax und Semantik gleichzeitig - und nicht hintereinander - ablaufen.

Eine Ableitung ist weiters *kompositional*, wenn sie *Freges Kompositionalitätsprinzip* folgt, das untenstehend noch einmal aus dem ersten Teil des Skriptums wiederholt wird:

(11) ***Kompositionalitätsprinzip*** (Gottlob Frege, 1848-1925)
Die Bedeutung eines komplexen Ausdrucks wird vollständig durch
(i) die Bedeutungen seiner unmittelbaren Teile sowie
(ii) die Art, wie diese miteinander verbunden sind (Kompositionsprinzipien), bestimmt.

Eine Konsequenz von Kompositionalität ist, dass jeder Knoten in einem Baum einen semantischen Wert haben muss. Im Laufe einer kompositionalen Ableitung muss also jedem Knoten eine Denotation zugewiesen werden.

Kehren wir nun zu Beispielsatz (9) zurück. Bisher ist nur die Denotation von zwei Knoten bekannt, nämlich von *Hans* und *schlafen*. Diese Bedeutungen werden durch die lexikalischen Bedeutungsregeln, genauer gesagt (6)a sowie (7), zur Verfügung gestellt. Doch was ist mit der Bedeutung aller anderer Knoten? Im vorliegenden Fall fehlen die Denotation für Knoten in (12):

(12) N°, V°, NP, VP, IP

Bei der Suche nach der Antwort ist es sinnvoll, bei der Beobachtung zu beginnen, dass der Baum in (9) zwei unterschiedliche Arten von Knoten enthält: *verzweigende* Knoten (im vorliegenden Fall IP), also Knoten, die zwei Töchter enthalten, und Knoten, die nicht verzweigend sind (N°, V°, NP, VP). Wenden wir uns zuerst den nicht-verzweigenden Knoten zu. Unter der Annahme, dass die syntaktische Kategorie (N, V, ...) in der Semantik keine Rolle spielt, ist die Bedeutung des terminalen Knoten *Hans* ident mit der Bedeutung des Mutterknoten $[_{N^\circ}$ *Hans*$]$. Gleiches trifft auf $[_{N^\circ}$ *Hans*$]$ und $[_{NP}$ *Hans*$]$ zu. Ganz allgemein gilt, dass nicht-verzweigende Knoten keinen semantischen Beitrag zur Bedeutung eines sprachlichen Ausdrucks leisten. Sie werden daher auch als *semantisch leer* bezeichnet. Zu Beginn der Derivation werden all diese semantisch leeren Knoten (NP, N°, VP und V° in (9)) aus dem Baum eliminiert. Dies wird durch eine Regel für nicht-verzweigende Knoten, die in (13) etwas präziser definiert ist, bewerkstelligt:

(13) ***NK. Regel für nicht-verzweigende Knoten***

Für beliebige Knoten α und β gilt:
Wenn α ein nicht-verzweigender Knoten ist und α der Mutterknoten von β ist,
dann $[\![\alpha]\!] = [\![\beta]\!]$.

Wird die Regel NK auf die VP (9), also den Teilbaum (14), angewendet, so erhält man die Ableitung (Derivation) in (15):

(14)
$$\left[\!\!\left[\begin{array}{c} \text{VP} \\ | \\ \text{V}^\circ \\ | \\ \textbf{schläft} \end{array}\right]\!\!\right] = \ ?$$

(15) a. $[\![[_{VP}$ **schläft**$]]\!]$ = $[\![[_{V°}$ **schläft**$]]\!]$
(Anwendung von NK, wobei VP := α und V° := β)

b. $[\![[_{V°}$ **schläft**$]]\!]$ = $[\![$**schläft**$]\!]$
(Anwendung von NK, V° := α und **schläft** := β)

c. $[\![$**schlafen**[4]$]\!]$ = für jede beliebige Situation s: {x|x schläft in s} (Lexikon)

d. $[\![[_{VP}$ **schläft**$]]\!]$ = für jede beliebige Situation s: {x|x schläft in s}
(folgt aus (15)a - (15)c)

(15) besteht, so, wie eine Ableitung in der Arithmetik, aus einer Reihe von Gleichungen. In (15) passiert nichts anderes, als dass die lexikalische Information eines Knoten an den direkt dominierenden Mutterknoten (also von unten nach oben) weitergereicht wird. Die Bedeutung der VP ist in diesem Fall also ident mit der Bedeutung des lexikalischen Kopfes. Die selbe Strategie macht es möglich, die Bedeutung des NP-Knotens zu berechnen. In (16) verläuft die Derivation von unten nach oben (*bottom-up*), wohingegen die Ableitung in (15) von oben nach unten (*top-down*) verläuft. Beide Richtungen ergeben in diesem Fall die selben Resultate.

(16) a. $[\![$**Hans**$]\!]$ = $[\![[_{N°}$ **Hans**$]]\!]$ (Anwendung von NK)

b. $[\![[_{N°}$ **Hans**$]]\!]$ = $[\![[_{NP}$ **Hans**$]]\!]$ (Anwendung von NK)

c. $[\![[_{NP}$ **Hans**$]]\!]$ = $[\![$**Hans**$]\!]$ (folgt aus (16)a und (16)b)

Zusätzlich zu den lexikalischen Bedeutungsregeln für *Hans* und *schlafen* sowie der Regel NK benötigt man eine weitere Regel für verzweigende Knoten. die Diese Regel legt fest, (i) welche Teile in (9) miteinander kombiniert werden und (ii) wie diese Teile miteinander kombiniert werden. Da (9) nur aus zwei Teilen besteht, ist die Wahl, was miteinander kombiniert wird, trivial (später werden wir Fälle kennen lernen, in denen die Entscheidung nicht so einfach ist). Konkret muss die Regel die Subjektsbedeutung mit der VP-Denotation verbinden. Das generelle Bildungsgesetz, das einem erlaubt, aus der Subjekts- und der VP-Denotation die Satzbedeutung abzuleiten, wird *Satzregel* genannt und kann wie folgt definiert werden:

(17) ***S. Satzregel.*** Für beliebige Knoten α, β und γ gilt:
Wenn α die Form [$_{IP}$ β γ] hat,
dann gilt für jede Situation s: $[\![\alpha]\!] = 1$ in s gdw $[\![\beta]\!] \in [\![\gamma]\!]$ in s

Die Ableitung des Beispielsatzes *Hans schläft* wird im Baum (18) skizziert. Ganz oben ((18)a) erlaubt die Satzregel die *Dekomposition* (Zerlegung) des Baumes in zwei Teile, die NP-Bedeutung und die VP-Bedeutung, wobei die Variable β mit dem Subjekt und die Variable γ mit dem VP-Knoten belegt wird. Diese beiden Bedeutungen werden durch die Element-von (∈) Beziehung miteinander verbunden (Details werden weiter unten nachgeliefert werden). Im Anschluss wird zweimal die Regel NK für nichtverzweigende Knoten angewendet ((18)b/c), bis dann die Bedeutung der lexikalischen Ausdrücke aus dem Lexikon eingesetzt werden kann ((18)d).

[4] Die morphologischen Unterschiede zwischen *schlafen* und *schläft* werden hier ignoriert.

(18)

a. Für jede Situation s, $[\![\text{IP}_\alpha]\!] = 1$ in s gdw Hans in s schläft (durch S)

b. $[\![\text{N}°]\!] = [\![\text{NP}_\beta]\!] \in [\![\text{VP}_\gamma]\!] = [\![\text{V}°]\!]$ (NK)

c. $[\![\textbf{Hans}]\!] = [\![\text{N}°]\!] \in [\![\text{V}°]\!] = [\![\textbf{schlafen}]\!]$ (NK)

d. $\text{Hans} = [\![\textbf{Hans}]\!] \in [\![\textbf{schlafen}]\!] = \{x|x \text{ schläft in s}\}$ (Lexikon)

Aus der Ableitung ergibt sich, dass der Satz *Hans schläft* in einer Situation s genau dann wahr ist, wenn Hans ein Element der Individuen ist, die in s schlafen.

Man beachte an diesem Punkt, dass die Derivation des Satzes, so wie gefordert, kompositional verläuft und dass jede Veränderung im Baum - also jeder Schritt von einem Knoten zum nächst höheren (oder tieferen) Knoten - durch eine Regel *legitimiert* (= erlaubt) wird. Außerdem wird, wie aus dem Baum (18) ersichtlich, jedem einzelnen Knoten eine Denotation zugewiesen. Dies ist eine wichtige Konsequenz aus dem Prinzip der Kompositionalität.

(19) In einer kompositionalen Ableitung erhält jeder Knoten eine Bedeutung zugewiesen.

Die Bedeutung des Satzes wird somit vollständig aus der Bedeutung der Teile und der Art, wie diese Teile verbunden werden, abgeleitet.

1.3. MENGENKONVERSION

Die Darstellung in (18) ist nicht besonders ökonomisch, sie barucht viel Platz. Außerdem fehlen in (18) noch zwei wichtige Details. Erstens wird nicht klar ersichtlich, wie die Beziehung zwischen dem Argument (*Hans*) und der Variable x im Lexikoneintrag von *schlafen* (*{x|x schläft in s}*) zu verstehen ist. Warum wird das Prädikat *schlafen* auf der Ebene der Verbbedeutung ((18)d) auf die Variable x angewendet, bei der Satzdenotation (18)d aber auf das Individuum Hans? Zweitens muss beantwortet werden, woher plötzlich die Wahrheitsbedingungen, also der Teil '= 1 gdw Hans schläft' in (18)a kommen. Diese Wahrheitsbedingungen sind, wenn die Derivation so wie in (18) analysiert wird, weder Teil der Verbdenotation noch irgend eines anderen Knoten. (18) ist also nur eine unvollständige Darstellung der semantischen Beziehungen.

<u>Konversion.</u> (20) bietet eine präzisere Version der semantischen Derivation des Satzes *Hans schläft.* ((20) ist gleichzeitig auch ökonomischer, da nun Ausdrücke mit Klammern anstatt von Baumdiagrammes zur Anwendung kommen.)

Wenden wir uns den Details der Derivation zu. Zu Beginn ((20)a) wird die Satzregel angewendet, im Anschluss daran ((20)b) werden aus dem Lexikon die Bedeutungen für *Hans* und *schlafen* eingesetzt.

(20) Für jede Situation s,

a. $[\![[_{\text{IP},\,\alpha}\ \textbf{Hans}_\beta\ \textbf{schläft}_\gamma]]\!] = 1$ in s gdw
$[\![\textbf{Hans}]\!] \in [\![\textbf{schläft}_\gamma]\!]$ in s (Anwendung von S)

b. $[\![[_{\text{IP}}\ \textbf{Hans schläft}]]\!] = 1$ in s gdw
$\text{Hans} \in \{x|x \text{ schläft in s}\}$ in s (Lexikoneinträge von *Hans* und *schlafen*)

c. $[\![[_{IP}$ **Hans schläft**$]]\!] = 1$ in s gdw
Hans schläft in s[5] (durch *Konversion* [s.u.])

Im Übergang von (20)b zu (20)c wird schließlich die Beziehung zwischen dem Argument (*Hans*) und der Variable x, die in (18) noch gefehlt hat, hergestellt. Dieser Schritt ist durch eine Operation der Mengenlehre, der *Mengenkonversion*, legitimiert. Was hier genau passiert, sieht man deutlicher, wenn man (20) mit einem einfachen mathematischen Beispiel vergleicht.

Nehmen wir an, wir wollen wissen, ob die Zahl 4 ein Element der Menge $\{x|x > 3\}$ ist. Dazu muss festgestellt werden, ob die Zahl 4 die *Bedingung* auf die Elemente der Menge erfüllt. Diese Bedingung wird durch den Ausdruck nach dem Definitionszeichen '|' ($x > 3$) gegeben. Wie geht man dabei vor? Man folgt dem zweiteiligen Algorithmus in (21). In einem ersten Schritt, den man als *Mengenkonversion* bezeichnet, wird die Variable x in der Bedingung durch die Zahl 4 ersetzt. Im Anschluss daran überprüft man in einem zweiten Schritt, ob die Bedingung eine wahre Aussage ergibt oder nicht. Wenn die Antwort positiv ist, dann ist 4 ein Element der Menge, wenn die Antwort nein ist, dann nicht. Diese logische Beziehung kann folgendermaßen ausgedrückt werden: die Zahl 4 ist ein Element der Menge $\{x|x > 3\}$, genau dann, wenn $4 > 3$ eine wahre Aussage ergibt. Im vorliegenden Beispiel trifft es natürlich zu, dass $4 > 3$, die Zahl 4 ist. Daher ist 4 ein Element der Menge $\{x|x > 3\}$.

(21) *Algorithmus zur Beantwortung der Frage 'Ist $4 \in \{x|x > 3\}$ eine wahre Aussage?'*

*Schritt 1 (**Konversion**)*: Ersetze x in '$x > 3$' durch 4. Resultat: $4 > 3$
Schritt 2 (überprüfe das Resultat): $4 \in \{x|x > 3\}$ gdw $4 > 3$

Antwort: Ja, $4 \in \{x|x > 3\}$

Kehren wir nun zur semantischen Ableitung von *Hans schläft* zurück. Hier sehen wir nichts anderes, als eine weitere Anwendung von Konversion. Konkret findet Konversion im Übergang von (20)b zu (20)c statt, in diesem Schritt wird *Hans* für die Variable x eingesetzt. Aus *Hans*$\in$*x|x schläft in s} in s* erhält man dann *Hans schläft in s*. Der zweite Schritt des Algorithmus ergibt dann die gewünschten Wahrheitsbedingungen. Der Satz *Hans schläft in s* erhält genau dann den Wahrheitswert 1 (ist genau dann wahr), wenn es der Fall ist, dass Hans in s schläft.

Zusammenfassend haben wir bisher gesehen, dass die Wahrheitsbedingungen eines intransitiven Satzes (*Hans schläft*) *kompositional* aus den Bedeutungen der einzelnen Teile (NP und VP) sowie dem Kompositionsprinzip (der Satzregel S) abgeleitet werden können. Bevor wir uns den etwas komplizierteren, transitiven Konstruktionen zuwenden, noch zwei Anmerkungen.

<u>Anmerkung zu den Wahrheitsbedingungen</u>: Die Satzregel (17) hat bei genauerer Betrachtung zwei unterschiedliche Effekte.

(22) a. (17) legt fest, wie Subjekt und VP-Bedeutung miteinander kombiniert werden (über die Element-von-Beziehung '$\in$')
b. (17) führt die Wahrheitsbedingungen ein.

Der zweite Aspekt ist besonders wichtig. Bei genauem Hinsehen wird nämlich offensichtlich,

[5] Die aufmerksame Leserin wird bemerkt haben, dass ein Vorkommen der Variable s plötztlich verschwunden ist, eigentlich sollte man als Resultat *Hans schläft* <u>*in s*</u> <u>*in s*</u> erwarten. Dies folgt aus technischen Details, auf die hier nicht näher eingegangen werden wird. (Konkret kann man annehmen, dass Prädikatsdenotation Argumentsposition für Situationsvariable enthalten, über die abstrahiert werden kann. Das zweite Vorkommen von s wird dann durch λ-Konversion eleminiert.)

dass der Lexikoneintrag des Prädikats (6)c noch gar nicht keine Wahrheitsbedingungen ('ist wahr, genau dann wenn' oder '= 1, genau dann wenn') enthält. (17) ist also dafür verantwortlich, die Bedeutung von Sätzen mit deren Wahrheitsbedingungen in Verbindung zu setzen.

<u>Anmerkung zur Reihenfolge der Regeln:</u> Man beachte zum Abschluss, dass die Reihenfolge, in der die Regeln NK und S angewendet werden, keinen Einfluß auf das Ergebnis hat. Man hätte in der Ableitung der Bedeutung von *Hans schläft* genauso gut zuerst NK anwenden können. Werden die Ergebnisse der Regel NK in den Baum (9) eingesetzt, erhält man den reduzierten Baum in (23), der nur mehr aus einer IP sowie den beiden lexikalischen Knoten *Hans* und *schläft* besteht.

(23) $\left[\!\left[\begin{array}{c} \text{IP} \\ \textbf{Hans} \quad \textbf{schläft} \end{array} \right]\!\right]$

Anwendung von S auf den Baum (23) führt dann zum Ergebnis *⟦(23)⟧ = 1 in s gdw ⟦Hans⟧ ∈ ⟦schläft⟧ in s.* Man erhält also genau das gleiche Resultat wie in (20). Dies ist kein Zufall, sondern folgt aus der Tatsache, dass die Regeln, mit denen Sprache die Bedeutungen berechnet, keine 'Richtung' kennen - es ist gleichgültig, ob die Gesamtbedeutung erst in ihre Teile zerlegt wird, oder ob die Derivation mit lexikalischen Einträgen beginnt, und deren Bedeutung dann verbindet. Die menschliche Sprache verhält sich hier genauso wie formale Sprachen wie etwa die Arithmetik oder die Logik. Auch beim Vereinfachen von komplexen arithmetischen Ausdrücken ist es unwichtig, in welcher Reihenfolge die Ergebnisse miteinander kombiniert werden. Will man z.B. den Wert von (24) berechnen, kann man dies entweder so wie in (24)a oder aber so wie in (24)b tun. Beide Wege führen zum selben Resultat:

(24)		(4 x 3)	+	(5 x 2)	
	a.	12	+	(5 x 2)	= 12 + 10 = 22
	b.	(4 x 3)	+	10	= 12 + 10 = 22

Natürliche Sprache besitzt also eine wichtige Eigenschaft, die sie mit formalen Systemen wie der Logik und der Mathematik teilt.[6]

1.4. EIN PROBLEM: TRANSITIVE PRÄDIKATE

Die bisher vorgestellte Semantik liefert zwar die korrekten Ergebnisse für intransitive Sätze, versagt aber schon bei der kompositionalen Derivation von einfachen transitiven Konstruktionen:

(25) a. Hans [$_{VP}$ kennt Maria]
b. Hans ist [$_{AP}$ Maria fremd/ähnlich]
c. Hans ist [$_{NP}$ ein Feind vieler Worte]
d. Hans ist [$_{PP}$ aus Wien]

Der Grund dafür ist, dass ein Verb wie *kennen* keine Menge von Individuen denotiert, so wie dies bei intransitiven Prädikaten der Fall ist. Daher kann auch nicht die Satzregel zur Anwendung kommen. Vielmehr bezeichnen transitive Verben *Relationen* zwischen Individuen, im Fall von *kennen* z.B. die 'kennen'-Relation, die auf alle Paare von Individuen zutrifft, in denen das erste

---

[6]Man nennt diese Eigenschaft auch die *Church-Rosser* Eigenschaft.

Element des Paares das zweite Element des Paares kennt. Etwas formaler kann dies als eine Menge von geordneten Paaren ausgedrückt werden. Der Lexikoneintrag von *kennen* würde demnach so wie in (26) aussehen:

(26) ⟦**kennen**⟧ = für jede beliebige Situation s: {<x,y>| x kennt y in s}

In (26) wird die Menge (von geordneten Paaren) durch die dem Definitionsstrich ('|') folgende Bedingung angegeben. Genauer gesagt liefert (26) die *Intension* des Prädikats, also die allgemeine, situationsunabhängige Bedeutung, die auch im mentalen Lexikon gespeichert ist. So wie bei einstelligen Prädikaten ist es natürlich auch möglich, die Bedeutung in einer spezifischen Situation, also die *Extension*, anzugeben. Dies geschieht einfach durch Aufzählung der Elemente der Menge. Wenn in einer Situation s_8 z.B. Hans Maria kennt, und Maria Peter kennt, und sonst keine Individuen einander kennen, dann kann die Bedeutung von *kennen* in s_8 wie in (27) angegeben werden:

(27) ⟦**kennen**⟧ in s_8 = {<Hans, Maria>, <Maria, Peter>}

Das Problem, das sich aus der Behandlung transitiver Prädikate für das Kompositionalitätsprinzip ergibt, stellt sich nun wie folgt dar. Keine der oben besprochenen Darstellungen, also weder Intension noch Extension von *kennen*, erlauben eine kompositionale Derivation. Insbesondere setzt jeder der Einträge in (26) - (27) voraus, dass die Verbbedeutung *gleichzeitig* mit der Subjektsdenotation und der Objektsdenotation kombiniert werden muss. Wird (26) zugrunde gelegt, erhält man z.B. die untenstehende Gleichung, in der, um den Satz wahr zu machen, verlangt wird, dass das Paar <Hans, Maria> ein Element der Verbdenotation ist. Auf den ersten Blick scheint dieses Resultat ganz plausibel auszusehen. (Knoten, die keine Bedeutung besitzen und durch NK eliminiert werden, werden ab jetzt einfach weggelassen.)

(28) Für jede Situation s,

$$[\![\textbf{Hans kennt Maria}]\!] = 1 \text{ in s gdw } <\text{Hans,Maria}> \in \underbrace{\{<x,y>|\ x \text{ kennt } y \text{ in s}\}}_{\text{Denotation von } \textit{kennen} \text{ in (26)}}$$

Doch betrachtet man die Details, zeigt sich ein grundlegendes Problem. Konkret verlangt nämlich die Satzregel (17), die untenstehend wiederholt wird, dass die Bedeutung von *kennen* auf ein syntaktisches Objekt appliziert, dessen Bedeutung das Paar <Hans, Maria> darstellt.

(17) ***S. Satzregel.*** Für alle beliebigen Knoten α, β und γ gilt:

Wenn α die Form

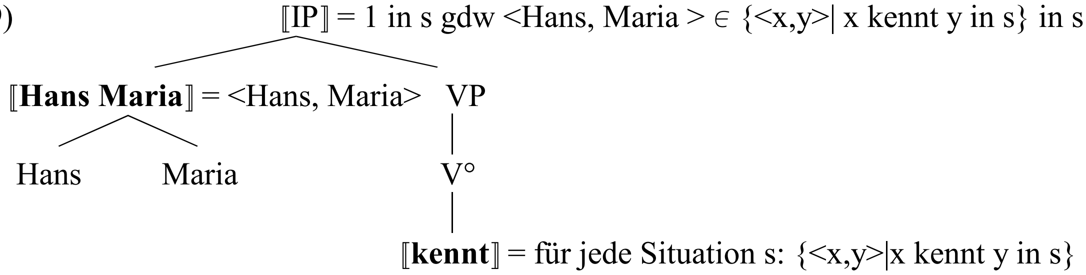
```
  IP
 /  \
β    γ
```

hat,

dann gilt für jede Situation s: ⟦α⟧ = 1 in s gdw ⟦β⟧ ∈ ⟦γ⟧ in s

Um die korrekte Interpretation zu liefern, müßte der Baum demnach so aussehen wie in (29):

(29)



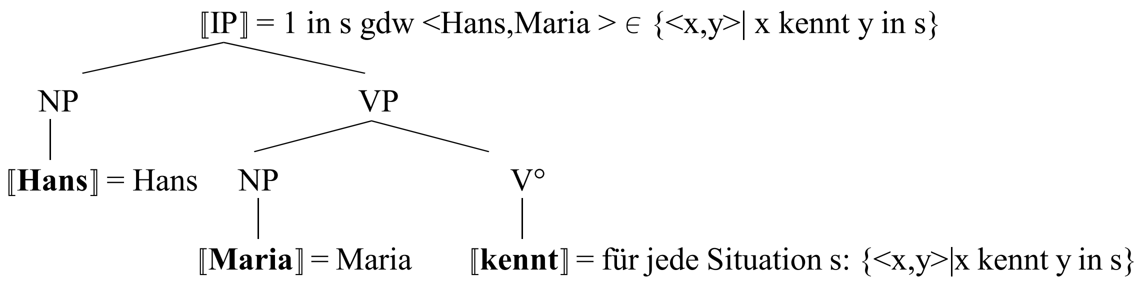
Diese Struktur widerspricht nun aber allen bekannten syntaktischen Gesetzen. Alle Konstituententests legen nämlich nahe, dass das Objekt und das Verb eine Konstituente - die VP - bilden, die das Subjekt exkludiert. So kann zum Beispiel das Objekt zusammen mit dem Verb an die Topikposition (CP) verschoben werden, nicht jedoch das Subjekt mit dem Verb. Und sicherlich bilden auch Subjekt und Objekt keine Einheit, und können daher auch nicht in das Vorfeld verschoben werden:

(30) a. Der Student$_S$ hat [$_{VP}$ einen Artikel$_O$ gelesen].
b. [$_{VP}$ Einen Artikel$_O$ gelesen] hat der Student$_S$.
c. *[$_{VP}$ Der Student$_S$ gelesen] hat einen Artikel$_O$.
d. *[$_{VP}$ Der Student$_S$ einen Artikel$_O$ hat gelesen.

Der Kontrast zwischen (30)b und (30)c folgt z.B. aus den beiden Annahmen (i) dass das Objekt und das Verb eine Konsitutente bildet, Subjekt und Verb jedoch nicht, und (ii) dass nur Konstituenten verschoben werden können. In (29) werden dagegen Subjekt und Objekt als Einheit analysiert, eine Annahme, die mit den Ergebnissen der klassischen Konstituententests nicht kompatibel ist. (Außerdem stellt (29) nicht die korrekte Wortfolge bereit). Es kann daher geschlossen werden, dass eine Analyse, die sich auf (29) stützt, nicht korrekt sein kann.

Wird im Gegensatz dazu die konventionelle, syntaktisch wohlgeformte Struktur (31) interpretiert, führt die semantische Derivation zu keinem Ergebnis. Der Baum (31) kann in unserem System, das bisher nur die Regeln NK und S umfaßt, nicht interpretiert werden.

(31)



Die Bedeutung der Subjekts-NP von (31) ist das Individuum Hans, während die Objekts-NP die Person Maria denotiert. Im syntaktischen Baum findet sich jedoch kein einziger Knoten, dessen Denotation dem Paar <Hans, Maria> entsprechen würde. Derivation der Bedeutung von transitiven Sätzen leidet also an gravierenden Fehlern, sie ist nicht kompositional: der VP in (31) wird keine Bedeutung zugewiesen.

Manche Probleme lassen sich lösen, indem man einfach die Regeln ändert. Vielleicht ist dies auch hier der Fall? Vielleicht werden transitive Sätze nicht mit der Satzregel S ((17)) interpretiert, sondern mit einer eigenen, *konstruktionsspezifischen* Regel? Eine solche allgemeine Methode, die Bedeutung von Sätzen mit transitiven Verben abzuleiten, wird in (32) definiert.

(32) ***TS. Transitive Satzregel*** Für beliebige Knoten α, β, γ und δ gilt:

Wenn α die Form

$$[_{IP}\ \beta\ [_{VP}\ \delta\ \gamma]]$$

hat,

dann gilt für jede Situation s:

$[\![\alpha]\!] = 1$ in s gdw $\langle [\![\beta]\!],[\![\delta]\!]\rangle \in [\![\gamma]\!]$ in s

Die vollständige Derivation des Beispielsatzes *Hans kennt Maria* verläuft nun so wie folgt:

(33) Für jede Situation s gilt:

a. $[\![[_{IP,\,\alpha}\ \textbf{Hans}_{\beta}\ [_{VP}\ \textbf{kennt}_{\delta}\ \textbf{Maria}_{\gamma}]]]\!] = 1$ in s gdw

b. $\langle [\![\textbf{Hans}]\!],[\![\textbf{Maria}]\!]\rangle \in [\![\textbf{kennen}]\!]$ in s gdw (Anwendung von TS)

c. $\langle$Hans,Maria$\rangle \in \{\langle x,y\rangle \mid x$ kennt y in s$\}$ gdw (Lexikon)

d. Hans kennt Maria in s (Konversion)

e. $[\![[_{IP}\ \textbf{Hans}\ [_{VP}\ \textbf{kennt Maria}]]]\!] = 1$ in s gdw
Hans kennt Maria in s (durch Äquivalenz von (33)a und (33)c)

Doch obwohl die Regel TS nun das gewünschte Resultat liefert, weist sie den selben Mangel auf, den wir schon oben beobachtet haben: sie ist *nicht kompositional*. Dies ist so, da der VP, genauso wie im ersten Versuch, keine Denotation zugewiesen wird. Die Regel führt daher zu einer Ableitung der Satzbedeutung, die das Kompositionalitätsprinzip verletzt. Da es ohne Kompositionalitätsprinzip nicht möglich, die grundlegende sprachliche Eigenschaft der Produktivität zu erklären, kann die TS-Analyse also nicht korrekt sein.

Genau das selbe Problem, das oben für das Verb *kennen* diagnostiziert wurde, betrifft natürlich auch alle anderen transitiven Prädikate, ganz unabhängig davon, ob es sich dabei um Verben (*schlagen*, *sehen, bearbeiten, entdecken*,...), Adjektiva, Präpositionen oder Nomen handelt:

(34) a. Hans ist $[_{AP}$ Maria *zugeneigt*]
b. *Hans ist $[_{AP}$ zugeneigt] (s.a. *bewußt, gewiß, überdrüssig, müde, behilflich*,...)

(35) a. Hans ist $[_{PP}$ *aus* Wien]
b. *Hans ist $[_{PP}$ aus] (s.a. *in, unter, über, neben, bei, mit,...*)

(36) a. $[_{NP}$ Die *Schaffung* von Arbeitsplätzen] dauerte Jahre
b. *$[_{NP}$ Die Schaffung] dauerte Jahre (s.a. *Leerung, Erwerb*, *Halbierung*...)

Auch ditransitive Konstruktionen, d.h. Sätze mit dreiwertigen Verben, können nicht mit den bisher bekannten Mitteln analysiert werden:

(37) a. Hans gibt der $\text{Maria}_{\text{IO, DAT}}$ das $\text{Buch}_{\text{DO. AKK}}$
b. Wir muteten $\text{ihm}_{\text{IO, DAT}}$ die schwierigste Aufgabe $_{\text{DO, AKK}}$ zu

(38) a. Sie unterzogen die $\text{Kinder}_{\text{IO, AKK}}$ einer $\text{Prüfung}_{\text{DO, DAT}}$
b. Er setzte die $\text{Freunde}_{\text{IO, AKK}}$ einer $\text{Gefahr}_{\text{DO, DAT}}$ aus (s.a. *zeigen, vorstellen,...*)

Schließlich ist es momentan auch nicht möglich, Kombinationen von Prädikaten mit sententialen Komplementen und viele andere sprachliche Phänomene zu erfassen.

(39) a. Peter glaubt/behauptet, $[_{CP}$ dass Hans Maria kennt]
b. Maria versucht, $[_{CP}$ ein Buch zu lesen]
c. Es ist leicht, $[_{CP}$ dieses Buch zu lesen]

In allen Fällen liegt das Problem am Fehlen einer kompositionalen Analyse für Konstruktionen, die mehr als ein Argument besitzen. Konkret fehlt eine Methode, der VP eine Bedeutung zuzuweisen.

1.5. Semantik transitiver VPs

Eine Möglichkeit, transitiven Prädikaten eine kompositionale Bedeutung zukommen zu lassen, besteht darin, mit komplexeren Mengen zu arbeiten. So kann etwa die Bedeutung von *kennen* so wie in (40) definiert werden.

(40) ⟦**kennen**⟧ = für jede Situation s:
{x | {y | y kennt x in s}}
“die Menge der x, sodass folgendes gilt: y ist die Menge, sodass folgendes gilt: y kennt x in s”

Sehen wir uns nun an, wie diese Änderung des lexikalischen Eintrags hilft, das Kompositinalitätsproblem zu umgehen. Von zentraler Bedeutung ist, dass man mit Hilfe von (40) auch eine Bedeutung für den VP-Knoten erhält. Bevor wir zu den Details übergehen, benötigen wir jedoch noch eine Regel, die es erlaubt, das Verb mit dem Objekt zu kombinieren. Dies wird durch die VP-Regel in (41) bewerkstelligt.

(41) ***VP. VP-Regel.*** Für alle beliebigen Knoten α, β und γ gilt:
Wenn α die Form [VP β γ] hat,
dann gilt für jede Situation s: ⟦α⟧ = ⟦β⟧ ∈ ⟦γ⟧ in s

Nun können wir uns endlich der kompositionalen Derivation der Wahrheitsbedingungen von *Hans kennt Maria* zuwenden, beginnend mit der Berechnung der VP. Relevante Details der Ableitung werden in (42) explizit gemacht.

(42) Für jede Situation s:

a.	⟦$[_{VP}$ **Maria kennt**]⟧ in s	=	
b.	= ⟦**Maria**⟧ ∈ ⟦**kennen**⟧ in	=	(VP-Regel)
c.	= Maria ∈ {x |{y|y kennt x in s}}	=	(Lexikon)
d.	= {y|y kennt Maria in s}		(Konversion)

Der wichtigste Punkt in der Ableitung (42) ist der Übergang von (42)c zu (42)d, hier findet Konversion statt, also *Schritt 1* des Algorithmus, der in (21) eingeführt wurde.[7]

Im zweiten Teil der semantischen Derivation von *Hans kennt Maria* wird die VP-Denotation

[7] Man beachte, dass die Extension der VP in einer Situation keinen Wahrheitswert denotiert, sondern eine Menge von Individuen. Daher finden sich an diesem Punkt auch noch keine Wahrheitsbedingungen (Schritt 2 in (21)).

mit der Subjektsbedeutung kombiniert, und man erhält, wie gewünscht, die Wahrheitsbedingungen in (43)d.

(43) Für jede Situation s:

a. ⟦$[_{IP}$ **Hans** $[_{VP}$ **Maria kennt**]]⟧ in s = 1 gdw

b. ⟦**Hans**⟧ ∈ ⟦$[_{VP}$ **Maria kennt**]⟧ in s gdw (Satzregel S)

c. Hans ∈ {y|y kennt Maria in s} in s gdw (aus Lexikon, (42) und (43)c)

d. Hans kennt Maria in s (Konversion)

e. ⟦$[_{IP}$ **Hans** $[_{VP}$ **Maria kennt**]]⟧ in s = 1 gdw
Hans kennt Maria in s (durch Äquivalenz von (43)b und (43)e)

Wie (43)d zeigt, ergibt die Berechnung nun das korrekte Resultat. Die neue lexikalische Bedeutungsregel für *kennen* macht es also möglich, transitive Sätze kompositional zu interpretieren.

Die folgenden beiden Abschnitt gehen näher auf eine alternative (aber im Grunde mit der Mengentheorie äquivalente) Methode ein, Prädikate zu interpretieren. Dabei spielt der Begriff der Funktion und der Verbindung von Funktionen mit deren Argumenten eine zentrale Rolle.

---

2. Prädikate als Funktionen (nicht Prüfungsstoff)

Bisher wurden einstellige Prädikate als Mengen dargestellt, so wie in (44) gezeigt:

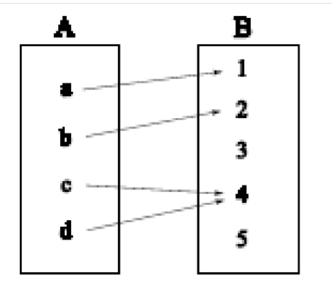
(44) ⟦**schlafen**⟧ = {x|x schläft}

VEREINFACHUNG: Situationsvariable werden ab jetzt weglassen, um die Berechnungen zu vereinfachen. Eine korrekte Theorie der Bedeutung muß diese selbstverständlich inkludieren (für ein System mit Situationen siehe z.B. von Stechow, *Schritte zur Satzsemantik*).

Doch hat sich gezeigt, dass dies zu Problemen bei der Behandlung von mehrstelligen Prädikaten führt, da es sich bei der Denotation eines relationalen Ausdrucks wie *kennen* oder *aussetzen* - im Gegensatz zu *schlafen* - nicht um eine Menge von Individuen handelt. Es gibt einfach keine Menge von Individuen, welche die ‘*kennen*-Eigenschaft’ oder die ‘*aussetzen*- Eigenschaft’ besitzen würden. Als ersten Schritt zu einer Lösung für dieses Problem wird daher eine alternative Analyse vorgestellt werden, in der Prädikate nicht als Mengen von Individuen, sondern als *Funktionen* aufgefasst werden.

Eine Funktion ist nichts anderes als ein Operation zwischen Mengen, die jedem Eingabewert einen Ausgabewert zuordnet. (45) illustriert dies mit der Funktion, die a den Wert 1, b den Wert b, c den Wert 4 und d ebenfalls der Wert 4 zuweist:

(45) *Funktion von A nach B*



(aus Lohnstein 2006: S. 26)

Wird ein Prädikat aus der Liste (44) als Funktion interpretiert, so erhält man einen

Lexikoneintrag, der so wie in (46) aussieht. In (46) wird der objektsprachliche Ausdruck *schlafen* im Lexikon als eine Funktion definiert, deren Eigenschaften dann auf der rechten Seite des Gleichheitszeichens '=' genauer beschrieben werden. Semantisch ist die Funktionsdarstellung (46) vollkommen äquivalent zur Mengeninterpretation (44).

(46) ⟦**schlafen**⟧ = die Funktion f von Individuen zu Wahrheitswerten ({0,1}), sodaß für jedes beliebige Individuum x gilt:
$$\mathrm{f(x)} = \underbrace{\mathrm{1\ gdw\ x\ schläft}}_{\text{Wahrheitsbedingungen für '⟦schlafen⟧(x)'}}$$

Konkret sagt (46) aus, dass der semantische Wert von *schlafen* eine Funktion darstellt, die, wenn sie auf ein beliebiges Individuum x angewendet wird, genau dann den Wert 1 ergibt, wenn dieses Individuum x die Eigenschaft hat, zu schlafen. Andernfalls erhält die Funktion (per Konvention) automatisch den Wert 0.

Transitive Prädikate lassen sich analog behandeln. Der wichtigste Unterschied besteht darin, dass ein zweites Argument eingeführt wird. Der Lexikoneintrag eines Verbs wie *kennen* ist daher auch ein wenig komplexer. Eine erste, noch vorläufige und nicht ganz vollständige Version findet sich in (47).

(47) ⟦**kennen**⟧ = die Funktion f, sodass für alle Individuen x und y gilt:
f(y)(x) = 1 gdw x y kennt

Das Verb *kennen* denotiert demnach eine zweistellige Funktion, die, wenn sie auf zwei beliebige Individuen y und x angewendet wird, genau dann den Wert 1 ergibt, wenn das Individuum x das Individuum y kennt. Andernfalls ist der Wert der Funktion 0. (Was in (47) noch fehlt, ist ein Verfahren, um die Funktion mit jedem der beiden Argumente getrennt zu kombinieren. Diese Methode wird in Abschnitt 2.3. nachgeliefert werden.)

2.1. Funktionale Applikation

Die Funktion in (47) wird als 'zweistellig' bezeichnet, da genau zwei *offene Argumentsstellen* gefüllt werden müssen: y und x. Man nennt die Argumentsstellen, die noch nicht gefüllt worden sind, auch *nicht-saturierte Argumentsstellen*. Sind alle offenen Argumentsstellen saturiert (oder gefüllt), dann erhält man einen Ausdruck, dem ein Wahrheitswert zugewiesen werden kann. '⟦kennen⟧(y)(x)' ist so ein Ausdruck, da man diesen mit '1' oder '0' gleichsetzen kann, je nachdem, ob die Wahrheitsbedingungen erfüllt sind oder nicht. In der Logik und der Semantik nennt man solche Ausdrücke auch *Aussagen* oder *Formeln*. Demnach ist die Bedeutung von (48)a eine dreistellige Funktion, da noch drei Argumente fehlen. (48)b denotiert also eine zweistellige Funktion, (48)a eine einstellige Funktion, und (48)d schließlich eine Formel oder nullstellige Funktion:

(48)	a.	unterziehen	dreistellige Funktion
	b.	einer Prüfung unterziehen	zweistellige Funktion
	c.	die Kinder einer Prüfung unterziehen	einstellige Funktion
	d.	(dass) die Lehrer die Kinder einer Prüfung unterziehen	Formel/Aussage

Das Verfahren, das aus einer Funktion zusammen mit einem ihrer Argumente einen Wert produziert, ist das wichtigste semantische Bauprinzip überhaupt und wird ***funktionale Applikation*** (FA) genannt. FA liegt dann vor, wenn eine Funktion, also ein Ausdruck mit offenen

Argumentstellen, auf ein Argument angewendet wird (man sagt auch ‘appliziert’ wird). Funktionale Applikation kann man immer daran erkennen, dass sie eine n-stellige Funktion in eine (n–1)-stellige Funktion überführt. Im Schritt von (48)a nach (48)b wandelt z.B. FA ein dreistelliges Prädikat - der Wert für n ist hier also 3 - in ein zweistelliges Prädikat um. Durch schrittweise, zweimalige weitere FA erhält man als Resultat in (48)d eine Formel.

Bisher wurde nur formale, also syntaktische Aspekte von FA besprochen. Die Semantik von FA ist durch die Definition in (49) festgelegt:

(49) ***FA. Funktionale Applikation***

Für beliebige Knoten α, β und γ gilt:
wenn α die Form α hat, $[\![\gamma]\!]$ eine Funktion ist und $[\![\beta]\!]$ ein Argument dieser Funktion ist,

$$\begin{array}{c} \alpha \\ \diagup \quad \diagdown \\ \beta \qquad \gamma \end{array}$$

(wobei die Reihenfolge von β und γ irrelevant ist)

dann gilt: $[\![\alpha]\!] = [\![\gamma]\!]([\![\beta]\!])$

(49) besagt, dass zwei Schwesterknoten im Baum so interpretiert werden, dass jener Knoten, dessen Bedeutung eine Funktion darstellt - Knoten α - die Denotation seines Schwesterknoten - Knoten β - als Argument nimmt. Sollten beide Knoten Argumente denotieren, oder beide Funktionen bezeichnen, dann können die Knoten nicht mittels FA interpretiert werden, und das Resultat ist (üblicherweise) semantisch nicht wohlgeformt. Dies erklärt, warum die beiden Ausdrücke in (50) ungrammatisch sind:

(50) a. *[$_{NP=\beta}$ Hans] [$_{NP=\gamma}$ die Bücher]
b. *[$_{VP=\beta}$ schlafen] [$_{AP=\gamma}$ schnarchen]

Die Ungrammatikalität von (50)a und (50)b folgt in der Theorie aus der Annahme, dass den Kombinationen in (50) keine Interpretation zugewiesen werden kann. Im Folgenden wird die Funktionsweise von FA anhand von zwei Beispielderivationen demonstriert werden.

2.2. Derivation Intransitiver Sätze

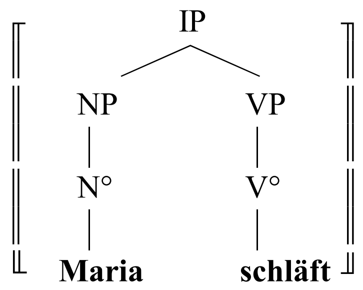
Ziel ist es, die Wahrheitsbedingungen eines einfachen intransitiven Satz wie (51) mit Hilfe der neuen Funktionsschreibweise abzuleiten. Am Ende der semantischen Derivation sollte also die Gleichung (52) stehen.

(51) Maria schläft

(52) $\underbrace{[\![[_{IP}\ \textbf{Maria schläft}]]\!]}_{\text{Objektsprache}} = \underbrace{1 \text{ gdw Maria schläft}}_{\text{metasprachliche Wahrheitsbedingungen}}$

Dabei dürfen, dem Kompositionalitätsprinzip folgend, nur zwei Arten von Information herangezogen werden: (i) die lexikalischen Bedeutungsregeln und (ii) die Prinzipien, welche zwei Denotationen miteinander verbinden (bisher wurde nur die Kompositionsregel der Funktionalen Applikation eingeführt).

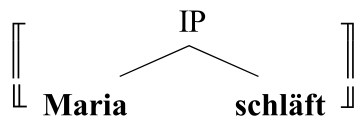
Als Eingabe der Derivation dient, wie schon in Abschnitt 1, ein syntaktischer Baum, der sowohl lexikalische Information (*Maria* und *schlafen*) als auch Information über die Struktur enthält (IP dominiert das Subjekt und die VP). Um anzuzeigen, dass der semantische Wert dieses Baumes abgeleitet werden soll, wird der gesamte Baum in Denotationsklammern gesetzt.

(53)


(Lexikalische Einträge)

Die syntaktische Struktur legt fest, welche Knoten im Baum mittels FA miteinander verbunden werden. Dies wird insbesondere in der Analyse von transitiven Sätzen entscheidend werden, in denen das Verb mit mehr als einem Argument verbunden werden muss, und wo es also wichtig ist zu entscheiden, welches Argument als erstes mit dem Verb kombiniert wird.

Die semantische Derivation beginnt, ebenfalls parallel zur Derivation mit Mengen, mit der Eliminierung von semantisch leeren, also nicht-verzweigenden Knoten durch Regel NK ((13)). Das Resultat ist, wie schon aus Abschnitt 1, bekannt:

(54)


Nun werden aus dem Lexikon die semantischen Werte (also die Denotationen) von *lesen* und *Peter* extrahiert. Das Ergebnis ist im Detail in ? ausgeführt:

(55) ⟦**Maria**⟧ = (das Individuum) Maria
⟦**schlafen**⟧ = die Funktion f, sodaß für jedes Individuum x: f(x) = 1 gdw x schläft

Das Lexikon stellt die Information zur Verfügung, dass *schlafen* als Funktion, und *Maria* als Argument interpretiert werden muss (und nicht umgekehrt). Damit kann die Kompositionsregel der Funktionalen Applikation (FA) in der Struktur zur Anwendung kommen. Konkret werden die Werte für die beiden Variablen β und γ in der Definition von (49) also wie folgt zugeteilt:

(56) a. β = Maria
b. γ = schläft

Die Anwendung von FA (s. (49)) ergibt dann (57). In (57) fungiert die Bedeutung von *schlafen* als Funktion, und die Denotation von *Maria* als Argument dieser Funktion:

(57) a. $[\![[_{IP}\ \textbf{Maria schläft}]]\!] = [\![\textbf{schlafen}]\!]([\![\textbf{Maria}]\!])$ (Funktionale Applikation; FA)

b. $[\![\textbf{schlafen}]\!]([\![\textbf{Maria}]\!]) =$

$$= \underbrace{[\text{die Funktion f, sodaß für jedes Individuum x:f}(\boxed{x}) = 1 \text{ gdw } \boxed{x} \text{ schläft}]}_{\text{Funktion}}\underbrace{(\text{Maria})}_{\text{Argument}}$$

(57) sagt jedoch noch nicht aus, wie nun genau die Denotation der IP aussieht - die Ableitung der IP-Bedeutung ist aber natürlich das Hauptziel der Analyse. Was ist nun das Ergebnis von FA, welche Veränderungen bewirkt diese Kompositionsregel?

Die Antwort liegt in der Interpretation der FA Regel. In (57)b gibt es eine offene Argumentstelle in ⟦schlafen⟧, nämlich die Variable ‘x’ (durch Box markiert). Anwendung von FA hat nun genau zwei Effekte:

(58) a. alle Vorkommen von x werden im Resultat - d.h. im Wert für ⟦IP⟧ - durch das Argument der Funktion (den Wert von ⟦Maria⟧, also das Individuum Maria) ersetzt;
b. die Formulierung ‘für jedes Individuum x’ wird eliminiert

Das bedeutet also, daß die vollständige Ableitung, mitsamt der IP-Denotation, so wie in (59) aussehen muß. Die IP bezeichnet dann, wie gewünscht, eine Funktion, die genau dann den Wert 1 ergibt, wenn Maria schläft (s. (59)b). Dieses Resultat kann weiter verkürzt werden zu der Darstellung in (59)c.

(59) ⟦[$_{IP}$ **Maria schläft**]⟧ =
a. = [die Funktion f, sodaß für jedes Individuum x: f($\boxed{x}$) = 1 gdw $\boxed{x}$ schläft](Maria) =
b. = [die Funktion f, sodaß f(Maria) = 1 gdw Maria schläft] (Resultat von FA)
c. = 1 gdw Maria schläft (Verkürzung)

Laut (59)c entspricht der Wert der IP dem Wahrheitswert 1, genau dann, wenn Maria schläft. ies sind die korrekten Wahrheitsbedingungen, (60) zeigt den kompletten Baum:

(60) $\left[\!\left[\begin{array}{c} \text{IP} \\ \textbf{Maria} \qquad \textbf{schläft} \end{array} \right]\!\right]$ = 1 gdw Maria schläft

Es wurde also nachgewiesen, daß die Wahrheitsbedingungen des Satzes *Maria schläft* kompositional aus den Bedeutungen der Teile abgeleitbar sind. In (61) werden zusammenfassend noch einmal alle Schritte der Argumentation aufgelistet:

(61) a. ⟦**Maria schläft**⟧ = ⟦**schlafen**⟧(⟦**Maria**⟧) (FA)
b. ⟦**schlafen**⟧ = die Funktion f, sodaß für jedes Individuum x: f(x) = 1 gdw x schläft (Lexikon)
c. ⟦**Maria**⟧ = Maria (Lexikon)
d. ⟦**schlafen**⟧(⟦**Maria**⟧) = (folgt aus (61)a und Lexikon)
= [die Funktion f, sodaß für jedes Individuum x: f(x) = 1 gdw x schläft] (Maria) =
= die Funktion f, sodaß für jedes Individuum x: f(Maria) = 1 gdw Maria schläft =
= 1 gdw Maria schläft (Resultat von FA)
e. ⟦**Maria schläft**⟧ = 1 gdw Maria schläft (folgt aus (61)a und (61)d)

<u>Motivation von (58)a.</u>Wie kann Schritt (58)a motiviert, d.h. mit unabhängigen Prinzipien erklärt werden? Die Lösung ergibt sich aus der Überlegung, daß der Übergang von den beiden Einzelbedeutungen zur Gesamtbedeutung (IP) in ? nichts anderes als die selbe Situation darstellt, die man in einem logischen Schluß wie (62) vorfindet. Man kennt die ersten beiden Prämissen ((62)a und (62)b), weiß, daß sie wahr sind, und folgert daraus die Konklusion ((62)c). Während nun in Prämisse (62)a die QP *alle Individuen* vorkommt, ist diese QP in der Konklusion verschwunden.

(62) a. Alle Individuen schlafen.
b. Maria ist ein Individuum
c. Maria schläft

Wenn also eine Bedingung ‘für alle Individuen x’ gilt, und Maria ein solches Individuum x ist, dann trifft diese Bedingung auch auf Maria zu. Exakt die gleichen Verhältnisse finden sich nun in ?. Genauer gesagt gibt es zwei Prämissen, die in (63)a und (63)b explizit gemacht werden. Die erste Prämisse stammt aus der Bedeutungsregel für das Verb, die zweite aus der Tatsache, daß Maria als Argument der durch *schlafen* bezeichneten Funktion fungiert. Und genauso wie aus (62)a und (62)b die Konklusion (62)c folgt, so folgt nun aus (63)a und (63)b zwingend, daß (63)c wahr ist:

(63) a. die Funktion f, sodaß für jedes Individuum x: f(x) = 1 gdw x schläft
b. Maria ist ein solches Individuum x
c. die Funktion f, sodaß für Maria gilt: f(Maria) = 1 gdw Maria schläft

Motivation von (58)b. Ist dies nun die Bedeutung der IP? Noch nicht ganz, da in (63)c ja noch ‘für Maria gilt:’ aufscheint, und dieser Teil offensichtlich im Endresultat fehlt (siehe IP-Denotation in (59)). Dieser Schritt folgt aus der Beobachtung, daß die Zeile (63)c die gleiche Bedeutung besitzt wie (64):

(64) = 1 gdw Maria schläft

Dies ergibt sich wiederum aus der allgemeinen Beobachtung, daß Sätze der Form *Für* $Name_n$ *gilt, daß ...*$Name_n$ *...,* wobei $Name_n$ für irgendeinen Namen steht, immer die gleiche Bedeutung besitzen wie Varianten ohne den Zusatz *Für* $Name_n$ *gilt.* (65)a belegt dies mit einem einfachen Beispiel. Im Gegensatz dazu darf bei Fällen in denen eine Variable (z.B. ‘x’ in (65)b) vorhanden ist, nicht verkürzt werden, ohne die Bedeutung zu verändern:

(65) a. ⟦**Für Peter gilt: Peter schläft**⟧ = ⟦Peter schläft⟧
b. ⟦**Für jedes Individuum gilt, daß es schläft**⟧ ≠ ⟦Es schläft⟧

Es folgt aus dem oben Gesagten, daß die IP-Denotation in (59) tatsächlich korrekt formuliert wurde. Jeder Schritt, d.h. jede Veränderung, wurde bewiesen.

2.3. Derivation Transitiver Konstruktionen

Das hier vorgestellte System ist nun auch in der Lage, die Bedeutung von transitiven Konstruktionen auf kompositionale Art und Weise zu analysieren. Es wird dies an der semantischen Ableitung der Gesamtbedeutung des Satzes (66) gezeigt werden.

(66) (Sie sagt daß) $[_{IP}$ Hans $[_{VP}$ Maria kennt$]]$

In der Derivation von (66) muss zuerst die Bedeutung von *kennen* mit der Bedeutung von *Maria* kombiniert werden, wie (67) zeigt:

(67) ⟦VP⟧ = ??

⟦**Maria**⟧ ⟦**kennt**⟧ = die Funktion f, sodaß für alle Individuen x und y:
f($\boxed{y}$)(x) = 1 gdw x $\boxed{y}$ kennt

Der erste Schritt, in dem die Bedeutung der VP berechnet werden soll, verläuft exakt nach dem Muster, das in der Derivation intransitiver Sätze zu beobachten war. Zuerst wird festgestellt, welches Element als Funktion und welches als Argument dient. Diese Information kommt aus den Lexikoneinträgen. Die Denotation von *kennen* beinhaltet z.B. die Information, daß in der Verbbedeutung zwei offene Argumentsstellen vorhanden sind, eine für das Objekt und eine für das Subjekt.

(68) *Lexikoneintrag (vorläufige Version)*
⟦**kennen**⟧ = die Funktion f, sodaß für alle Individuen x und y:
f(y)(x) = 1 gdw gilt: x kennt y

Die zwei Variablen x und y fungieren als Platzhalter für diese beiden Argumente. *kennen* ist demnach die Funktion, und Maria das Argument.

Doch stehen wir jetzt vor einem Problem: *kennen* denotiert eine zweistellig Funktion, also eine Funktion mit zwei Argumenten. Da eine solche zweistellige Funktion auf zwei Argumente *gleichzeitig* angewendet werden muss, kann das Verb jedoch nicht mit dem Objekt, so wie in (67), kombiniert werden (s. Diskussion in 1.3). Eine wichtiger Teil der Analyse, und zwar jener, der Kompositionalität sicherstellt, fehlt also noch. Konkret fehlt eine Methode, mit der die zweistellige Funktion in eine Funktion umgewandelt werden kann, die erst auf das Objekt und erst dann auf das Subjekt appliziert. Diese Methode wird durch *Currying* oder *Schönfinkeln* zur Verfügung gestellt.

2.3.1. Currying/Schönfinkeln [nicht Prüfungsstoff]

Unter *Currying* versteht man eine mathematische Operation, mit deren Hilfe eine Funktion mit mehreren Argumenten in eine (zusammengesetzte, komplexe) Funktion mit einem Argument umgewandelt werden kann. Zum ersten mal wurde diese Methode 1924 vom Mathematiker Moses Schönfinkel formuliert, der bewies, dass jede Funktion mit mehr als einem Argumente sich als Reihe von einstelligen Funktionen darstellen läßt. Ein Beispiel illustriert wie Currying zu einstelligen Funktionen führt.

Nehmen wir an, *kennen* hat die Extension, die auf der linken Seite in (69) gegeben ist (= (69)a). *kennen* denotiert also eine zweistellige Funktion von Paaren von Individuen zu Wahrheitswerten. Diese zweistellige Funktion kann nun äquivalent durch eine komplexe einstellige Funktion ersetzt werden - konkret durch eine Funktion von Indivdiuen zu einer Funktion von Individuen zu Wahrheitswerten (s. (69)b). In (69)b appliziert die Funktion zuerst auf das Subjekt. Als Resultat erhält man wiederum eine Funktion. Diese Funktion wird nun auf das Objekt angewendet, und erzeugt einen Wahrheitswert:

(69) *Links-nach-rechts Currying/Schönfinkeln*

a. b. *Subjekt* *Objekt*

$$[\![\textbf{kennen}]\!] = \underbrace{\left[\begin{array}{lcl} \langle \text{Bart, Bart}\rangle & \rightarrow & 0 \\ \langle \text{Bart, Jeff}\rangle & \rightarrow & 0 \\ \langle \text{Jeff, Bart}\rangle & \rightarrow & 1 \\ \langle \text{Jeff, Jeff}\rangle & \rightarrow & 1 \end{array}\right]} = \left[\begin{array}{lcl} \text{Bart} & \rightarrow & \left[\begin{array}{lcl} \text{Bart} & \rightarrow & 0 \\ \text{Jeff} & \rightarrow & 0 \end{array}\right] \\ \text{Jeff} & \rightarrow & \left[\begin{array}{lcl} \text{Bart} & \rightarrow & 1 \\ \text{Jeff} & \rightarrow & 1 \end{array}\right] \end{array}\right]$$

zweistellige Funktion von Paaren von Individuen zu Wahrheitswerten.

Funktion von **Individuen** zu einer Funktion

von **Individuen zu Wahrheitswerten**

Neben (69) existiert noch eine zweite Möglichkeit des Currying, die sich von (69) nur in der Reihenfolge der Argumente unterscheidet. In (70) ist das Objekt das erste Argument, und das Subjekt fungiert als zweites Argument.

(70) *Rechts-nach-links Currying/Schönfinkeln*

Objekt *Subjekt*

$$[\![\textbf{kennen}]\!] = \left[\begin{array}{lcl} \langle \text{Bart, Bart}\rangle & \rightarrow & 0 \\ \langle \text{Bart, Jeff}\rangle & \rightarrow & 0 \\ \langle \text{Jeff, Bart}\rangle & \rightarrow & 1 \\ \langle \text{Jeff, Jeff}\rangle & \rightarrow & 1 \end{array}\right] = \left[\begin{array}{lcl} \text{Bart} & \rightarrow & \left[\begin{array}{lcl} \text{Bart} & \rightarrow & 0 \\ \text{Jeff} & \rightarrow & 1 \end{array}\right] \\ \text{Jeff} & \rightarrow & \left[\begin{array}{lcl} \text{Bart} & \rightarrow & 0 \\ \text{Jeff} & \rightarrow & 1 \end{array}\right] \end{array}\right]$$

Durch rechts-nach-links Currying wird es nun möglich, die Denotation eines transitiven Prädikat zuerst mit dem Objekt und danach mit dem Subjekt zu kombinieren. Diese Repräsentation ist nun besonders gut geeignet, um die Bedeutung von transitiven Prädikaten darzustellen, da Kompositionalität verlangt, dass auch in diesem Fall die Verbdenotation zuerst mit der Objektsbedeutung und dann mit dem Subjekt verbunden wird. Der entsprechende Lexikoneintrag sieht so wie in (71) aus:

(71) *Lexikoneintrag (endgültige Version)*

⟦**kennen**⟧ = die Funktion f, sodaß für alle Individuen x gilt:
f(x) = die Funktion g, sodaß für alle Individuen y gilt:
g(y) = 1 gdw gilt: y kennt x

Rechts-nach-links Currying/Schönfinkeln bildet also die Grundlage für die kompositionale Interpretation von transitiven Sätze.

2.3.2. Anwendung: Funktionale Applikation [nicht Prüfungsstoff]

Kehren wir nun zur Derivation (67) zurück, um zu sehen, welchen Beitrag der Lexikoneintrag (71) im Detail leistet. Die wichtigste Eigenschaft der Funktion (71) ist, dass sie auf jedes ihrer Argumente getrennt angewendet werden kann. Daher ist es nun möglich, die Verbdenotation direkt mittels FA mit der Objektsdenotation zu verbinden. Die Kombination der beiden Bedeutungen führt zu dem in (72) gezeigten Resultat. (72)a drückt aus, dass die VP-denotation ident ist der Kombination der Verbdenotation und der Objektsdenotation. Konkret ist die

Verbdenotation (⟦kennen⟧) die Funktion, und das Objekt (⟦Maria⟧) das Argument der Funktion:

(72) a. ⟦[$_{VP}$ **Maria kennt**]⟧ = ⟦**kennt**⟧(⟦**Maria**⟧)

b. ⟦**kennt**⟧(⟦**Maria**⟧) = (FA)

= [die Funktion f, sodaß für alle Individuen x gilt:
f($\boxed{x}$)) = die Funktion g, sodaß für alle Individuen y gilt:
g(y) = 1 gdw gilt: y kennt $\boxed{x}$] (Maria) =

c. = [die Funktion g, sodaß für alle Individuen y gilt:
g(y) = 1 gdw gilt: y kennt Maria] (Resultat von FA)

Der Schritt (72)a, in dem die VP in eine Funktion und deren Argument zerlegt wird, ist eine gültige Umformung, die direkt aus FA und dem Prinzip der Kompositionalität folgt: die Bedeutung der VP kann vollständig aus der Bedeutung der Teile abgeleitet werden. Noch ein weiterer Aspekt in (72)a ist wichtig: die Bedeutung von *kennen* und *Maria* ‘erzwingen’ sozusagen, dass *kennen* die Funktion darstellt, und *Maria* das Argument, und nicht umgekehrt. Die Semantik leitet somit die Art und Weise, wie die Bedeutungen miteinander kombiniert werden.

Im nächsten Schritt, in (72)b, werden die natürlichsprachlichen Ausdrücke (*kennen* und *Maria*) interpretiert, sie werden also durch ihre jeweiligen Lexikoneinträge ersetzt. In (72)c kommt schließlich FA zur Anwendung: die Variable ‘x’, die der Objektsstelle entspricht, wird durch *Maria* ersetzt, und aus der Formel werden überflüssige Teile (genauer gesagt der Teil: ‘die Funktion f, sodaß für alle Individuen x gilt: f(x)) = ’) gelöscht. Das Ergebnis ist die einstellige Funktion (72)c.

Nach Anwendung von FA auf die zweistellige Funktion ⟦**kennen**⟧ und ihr Argument (die Objektsbedeutung ⟦**Maria**⟧), erhält man also eine einstellige Funktion. Diese Funktion stellt die Bedeutung der VP dar, wie im untenstehenden Baum genauer spezifiziert:

(73) ⟦VP⟧ = die Funktion g, sodaß für alle Individuen y:
g(Maria)(y) = 1 gdw gilt: y kennt Maria

Maria kennt

Die VP-Denotation ist eine Funktion, die noch eine offene Argumentsstelle enthält. Es fehlt der VP-Bedeutung also nur noch das Subjekt, um zu einer vollständigen Satzbedeutung zu gelangen. Da es nun in jedem Satz genau ein Argument außerhalb der VP gibt - das Subjekt - stellen VP-Bedeutungen immer den gleichen Typ von Funktion dar, gleichgültig ob das Verb transitiv oder intransitiv sind - alle VPs denotieren einstellige Funktionen:

(74) *Beobachtung*: Alle VP-Denotationen sind einstellige Funktionen

Im letzten Teil der semantischen Derivation wird die Subjektsbedeutung zur VP-Denotation hinzugefügt. Es wiederholt sich dann exakt die gleiche Vorgehensweise, die in (72) Objekt und Verb verbunden hat, noch einmal mit Subjekt und VP-Denotation. Funktionale Applikation in (75)a legt fest, dass die gesamte IP-Bedeutung den gleichen semantischen Wert besitzt wie das Resultat der Kombination von VP-Bedeutung und Subjektsdenotation. In (75)b wird das Ergebnis ausgerechnet. Wird die VP-Denotation auf das Argument Hans appliziert, dann muss die Variable y in der VP-Denotation (durch Box markiert) durch Hans ersetzt werden. Als Resultat von Funktionaler Applikation erhält man also die Formel (75)c. (Die IP-Denotation bezeichnet natürlich, wenn Situationen berücksichtigt werden, die Proposition in (75)d):

(75) a. ⟦$[_{IP}$ **Hans Maria kennt**]⟧ = ⟦$[_{VP}$ **Maria kennt**]⟧(⟦**Hans**⟧)

b. ⟦$[_{VP}$ **Maria kennt**]⟧(⟦**Hans**⟧) =

= [die Funktion g, sodaß für alle Individuen $\boxed{y}$ gilt:

g(y) = 1 gdw gilt: $\boxed{y}$ kennt Maria] (Hans) = (FA)

c. = 1 gdw gilt: Hans kennt Maria (Resultat von FA)

d. ⟦$[_{IP}$ **Hans Maria kennt**]⟧ = {s|Hans kennt Maria in s}

Das Baumdiagramm (76) zeigt noch einmal die jeweiligen Denotationen für IP, VP und das Verb:

(76) ⟦IP⟧ = 1 gdw Hans Maria kennt

⟦**Hans**⟧ ⟦VP⟧ = die Funktion g, sodaß für alle y: g(y) = 1 gdw y Maria kennt

⟦**Maria**⟧ ⟦**kennt**⟧ = die Funktion f, sodaß für alle Individuen x gilt:
f(x) = die Funktion g, sodaß für alle Individuen y gilt:
g(y) = 1 gdw gilt: y kennt x

Wie aus der Tatsache ersichtlich wird, daß jeder Knoten in (76) eine Interpretation erhält, erlaubt das System demnach nun auch eine kompositionale Derivation von transitiven Strukturen.

2.3.3. Ein weiteres Beispiel

(77) illustriert anhand eines weiteren Beispiels und in einem etwas anderen Format, die Rolle von Funktionaler Applikation, und wie durch FA und den Lexikoneintrag die Wahrheitsbedingungen eines Satze abgeleitet werden. Der Vollständigkeit halber werden in dieser Ableitung alle Knoten, also auch alle nicht-verzweigenden Knoten, wieder angegeben:

(77) a. ⟦$[_{IP}[_{NP}[_{N°}$ **Bart**]] $[_{VP}[_{V°}$ **raucht**]]]]⟧ = ⟦**raucht**⟧(⟦**Bart**⟧) (5 x NK und Satzregel)

b. ⟦**raucht**⟧(⟦**Bart**⟧) = $\boxed{\left[\begin{array}{l}\text{f: D} \rightarrow \{0,1\} \text{ sodass für jedes a} \\ \text{f(a) = } \boxed{\text{1 gdw a raucht}}\end{array}\right]\text{(Bart)}}$

⇓ ⇓
Function Application
⇓ ⇓

c. ⟦**raucht**⟧(⟦**Bart**⟧) = $\boxed{\text{1 gdw Bart raucht}}$

d. ⟦$[_{IP}[_{NP}[_{N°}$ **Bart**]] $[_{VP}[_{V°}$ **raucht**]]]]⟧ = $\boxed{\text{1 gdw Bart raucht}}$

Die große Box in (77)b bezeichnet den Teil, in dem FA appliziert. Konkret wird der Lexikoneintrag von *rauchen* auf die Bedeutung von *Bart* angewendet, und man erhält also Resultat die kleiner Box in (77)c. Da die gesamte Satzdenotation mit der Box in (77)b äquivalent ist, und da diese große Box mit (77)c ident ist, folgt, daß die Bedeutung des Satzes so wie in (77)d dargestellt werden kann. Man erhält also, wie gewünscht, die Wahrheitsbedingungen des Satzes.

3. MEHR ZU FUNKTIONEN

3.1. FUNKTIONEN VS. MENGEN

Mengen- und Funktionsdarstellung sind systematisch miteinander verwandt.[8] Dennoch erweist sich in der Semantik für unterschiedliche Zwecke die eine oder die andere Darstellung als besser geeignet. So gibt es z.B. zwei wichtige Eigenschaften der Funktionsdarstellung von Prädikaten, die vorteilhaft in der kompositionlen Ableitung von Sätzen genutzt werden.

> **Vorteil A.** Die kompositionale Derivation der Satzbedeutung aus den Bedeutungen der Teile läßt sich leichter durch Funktionen darstellen. Erst die Bedeutungsregel (71) macht es mit einfachen Mitteln möglich, in der semantischen Derivation eines Satzes die Argumente *schrittweise* mit der Prädikatsdenotation zu kombinieren. In der semantischen Ableitung müssen daher nicht mehr alle Argumente *gleichzeitig* zum Prädikat hinzugefügt werden, wie dies in der früheren, nun veralteten Relationsdarstellung (78), noch der Fall war.

(71) ⟦**kennen**⟧ = die Funktion f, sodaß für alle Individuen x gilt:
f(x) = die Funktion g, sodaß für alle Individuen y gilt:
g(y) = 1 gdw gilt: y kennt x

(78) ⟦**kennen**⟧ = {<x,y>|x y kennt}

> **Vorteil B.** Im Lexikoneintrag lassen sich die 'Bedingungen für das Wahrsein' (etwa im Teil *'= 1 gdw x schläft'* in (46)) explizit wiederfinden. Daher können aus der Wortbedeutung direkt die Wahrheitsbedingungen abgeleitet werden. Es ist nicht länger nötig, diese über den Umweg einer Kompositionsregel einzuführen (s. Diskussion der Eigenschaft B. der Satzregel).

Die Funktionsschreibweise (71) ist noch relativ unübersichtlich. Es gibt jedoch eine alternative Art und Weise, Funktionen darzustellen, die hier Abhilfe hilft. Diese alternative Schreibweise - man sagt auch: *Notation* - nennt man die λ-Notation. (Eine detailliertere Einführung in die λ-Notation und den sogenannten λ-Kalkül findet sich z.B. hier: http://users.uoa.gr/~wlechner/IEL%20Lambda2014.htm.) Die λ-Notation wurde zum ersten man in den 1970er Jahren von Richard Montague in der Analyse von natürlicher Sprache verwendet, und stellt sozusagen die internationale Arbeitssprache der formalen Semantiker dar.

3.2. DIE LAMBDA-NOTATION FÜR FUNKTIONEN

Funktionen (WH). Eine Funktion ist eine Relation von einer Menge A zu einer Menge B, für die gilt, daß jedem Element von A genau ein Element der Menge B zugewiesen wird.[9]

[8]Genauer gesagt können Mengen und Funktionen über den Begriff der *charakteristischen Funktion* ineinander übersetzt werden (Grewendorf, Hamm & Sternefeld 1987: 341; Heim & Kratzer 1998).

[9]Daraus folgt, dass nicht alle Relationen auch Funktionen sind. Beispiele für natürlichsprachliche Relationen, die keine Funktionen sind: *Kind*, *Tochter*, *Bruder*, *größer_als*, *Nachbar_von*, *kennen*, ...

(79) *Beispiele für Funktionen*

a. ⟦**Vater**⟧ = jene Funktion, die jedem Individuum x den Vater von x zuweist.

b. ⟦**Geburtsjahr**⟧ = jene Funktion, die jedem Individuum x das Geburtsjahr von x zuweist

c. ⟦**Nachfolger einer natürlichen Zahl**⟧ = jene Funktion, die jeder natürlichen Zahl ihren Nachfolger zuweist

Traditionelle Notation für Funktionen. Es existieren mehrere Möglichkeiten, Funktionen in der klassischen Funktionsschreibweise zu notieren. Die beiden unten angeführten Notationen unterscheiden sich nur minimal, und werden hier als äquivalent behandelt.

(80) a. ⟦**Vater**⟧ = jene Funktion f (von Individuen zu Individuen), sodaß für jedes Individuum x gilt:
f(x) = der Vater von x

b. ⟦**Vater**⟧ = x ↦ der Vater von x

(81) a. ⟦**Geburtsjahr**⟧ = jene f (von Individuen zu natürlichen Zahlen), sodaß für jedes Individuum x gilt:
f(x) = das Geburtsjahr von x

b. ⟦**Geburtsjahr**⟧ = x ↦ das Geburtsjahr von x

(82) a. ⟦**Nachfolger**⟧ = jene f (von Individuen zu Zahlen zu Zahlen), sodaß für jede Zahl n gilt:
f(n) = n+1

b. ⟦**Nachfolger**⟧ = n ↦ n+1

Lamda Kalkulus. Eine alternative Schreibweise für Funktionen ist die sogenannte *Lambdanotation*, kurz *λ-Notation*. In dieser Notation sehen die obigen Funktionen wie folgt aus:

(83) *Funktionen in λ-Notation*

a. ⟦**Vater**⟧ = λx.der Vater von x
b. ⟦**Geburtsjahr**⟧ = λz.das Geburtsjahr von z
c. ⟦**Nachfolger**⟧ = λn.n+1

Eine Funktion in der λ-Notation besteht immer aus drei Teilen:

(84) *Bestandteile eines λ-Ausdrucks*
a. der Lambdaoperator (λ)
b. eine *Variable* (x, y, P,...)
c. der *Skopus* des λ-Ausdrucks, der die Funktion definiert (der Teil nach dem Punkt).

(85) $\underbrace{\lambda x.}_{\text{λ-Operator und Variable}}\ \underbrace{\text{der Vater von x}}_{\text{Skopus des λ-Operators}}$

<u>Interpretation von λ-Ausdrücken</u>. λ-Ausdrücke werden so interpretiert, dass der Teil hinter der Variable und dem Punkt '.' den Wert der Variable bestimmt. Dieser Teil ist im untenstehenden Beispiel in ein $\boxed{\text{Box}}$ gesetzt:

(86) a. $\underbrace{\lambda x.}_{\text{Individuum}}\ \underbrace{\boxed{\text{der Vater von x}}}_{\text{Individuum}}$

b. Interpretation: "die Funktion von Individuen zu Individuen, die für jedes Individuum x den Vater von x ergibt"

Die Funktion kann auch einen Wahrheitswert als Wert haben. In diesem Fall bezeichnet der λ-Ausdruck (im einfachsten Fall) ein Prädikat:

(87) a. $\underbrace{\lambda z.}_{\text{Individuum}}\ \underbrace{\text{z schläft}}_{\text{Wahrheitswert}}$

b. Interpretation: "die Funktion von Individuen zu Wahrheitswert, die, wenn sie auf ein beliebiges Individuum z angewendet wird, den Wert 1 ergibt, wenn z schläft, und den Wert 0 ergibt, wenn z nicht schläft"

λ-Ausdrücke können auch rekursiv erweitert werden, sodaß ein einziger Ausruck mehr als nur einen λ-Operator enthält (die allgemeinen Regeln werden hier nicht näher erläutert werden).

(88) a. ⟦**schlafen**⟧ = λx.x schläft (lexikalischer Eintrag, ohne Situation)
b. ⟦**kennen**⟧ = λx.λy.y kennt x (Lexikon, ohne Situation)
c. ⟦**zeigen**⟧ = λx.λy.λz.z zeigt y x (Lexikon, ohne Situation)
d. ⟦**schlafen**⟧ = λx.λs.x schläft in s (lexikalischer Eintrag, diesmal mit Situation)

Diese Eigenschaft ist nun besonders für die Analyse von transitiven Prädikaten interessant, da λ-Notation eine einfache Methode zur Verfügung stellt, mit der transitive Sätze interpretier werden können. Anstatt die Denotation eines Verbs wie in (71) anzugeben, können wir nun einfach schreiben: 'λx.λy.y kennt x'.

<u>λ-Konversion:</u> Wird ein λ-Ausdruck der Form λα mit einem (passenden) Argument verbunden, dann wird jedes Vorkommen der Variable α durch das Argument ersetzt.

(89) a. $\underbrace{\lambda x.\text{x schläft}}_{\text{Funktion}}\ \underbrace{\text{(Maria)}}_{\text{Argument}} = \underbrace{\text{schläft(Maria)}}_{\text{Resultat von }\lambda\text{-Konversion}}$

b. Interpretation: "die Funktion von Individuen zu Wahrheitswert, die den Wert 1 ergibt, wenn Maria schläft, und 0, wenn Maria nicht schläft."

λ-Konversion drückt also ein bekanntes semantisches Prinzip aus: funktionale Applikation.

<u>λ-Notation und kompositionale Derivation.</u> In der neuen, einfacheren λ-Schreibweise für Funktionen können nun die Wahrheitsbedingungen des transitive Satzes *Maria kennt Peter* so wie in (90) abgeleitet werden.

(90) a. ⟦[$_{IP}$ **Maria kennt Peter**]⟧ =
= ⟦**kennt Peter**⟧ (⟦**Maria**⟧) =
= ⟦**kennt**⟧ (⟦**Peter**⟧) (⟦**Maria**⟧)

b. ⟦**kennt**⟧ (⟦**Peter**⟧)(⟦**Maria**⟧) =
= [λx.λy.y kennt x](Peter)(Maria)

c. [λx.λy.y kennt x](Peter)(Maria) =
= [λy.y kennt Peter](Maria)
"jene Funktion von Individuen zu Wahrheitswerten, die jedem Individuum y den Wert 1 zuweist, wenn y Peter kennt, angwewendet auf Maria"

d. Maria kennt Peter
"jene Funktion von Individuen zu Wahrheitswerten, die Maria den Wert 1 zuweist, wenn Maria Peter kennt"

Auf Details, wie z.B. die Frage, warum die Funktion in genau dieser Reihenfolge auf die Argumente angewendet wird, oder woher genau die Wahrheitsbedingungen in der Ableitung kommen, kann hier nicht näher eingegangen werden. Aus den obigen Ausführungen läßt sich jedoch ein erster Eindruck darüber gewinnen, wie λ-Ausdrücke in der Analyse von natürlichsprachlichen Ausdrücken zur Anwendung kommen. Weitere, einführende Information zur λ-Notation findet sich z.B. in Heim und Kratzer (1998), Lechner (2012) und Lohnstein (1996).

Literatur

Grewendorf, Günther, Fritz Hamm, und Wolfgang Sternefeld. 1987. *Sprachliches Wissen.* Frankfurt a. M.: Suhrkamp.
Heim, Irene und Angelika Kratzer. 1998. *Semantics in Generative Grammar*. Oxford: Blackwell.
Lechner, Winfried. 2012. Compositional Interpretation and the Lambda Calculus. http://users.uoa.gr/~wlechner/IEL%20Lambda2014.htm
Lohnstein, Horst. 1996. *Formale Semantik und natürliche Sprache. Einführendes Lehrbuch.* Opladen: Westdeutscher Verlag.
Zimmermann, Thomas Ede. 2014. *Einführung in die Semantik.* WGB.