Formale Semantik
07. Einfach getypte höherstufige L-Sprachen

Roland Schäfer

Institut für Germanistische Sprachwissenschaft
Friedrich-Schiller-Universität Jena

Folien in Überarbeitung. Englische Teile (ab Woche 7) sind noch von 2007!

Stets aktuelle Fassungen: `https://github.com/rsling/VL-Semantik`

Inhalt

Kernfragen in dieser Woche

Wie unterscheidet sich Montagues System von GB-Semantik?

Welche Rolle spielen Typen?

Was sind λ-Sprachen?
Und woher kennen Sie den λ-Operator eigentlich schon?

Text für heute: Dowty u. a. (1981: Kapitel 4) | Chierchia & McConnell-Ginet 2000: Kapitel 7

Einfachere Semantik

Montague vs. Generativismus

Es geht wie immer auch ohne Bewegung.

- Chierchia | auf Grundlage von GB-Syntax
  - Syntax und Semantik in Phrasenstrukturen
  - Sprache wird zu Logik durch unsichtbare Bewegung.
  - Semantik als eigene Repräsentationsebene
- Montague | Sprache ist Logik!
  - Direkte Interpretation von Zeichen als logische Symbole
  - Logische Form (lf) als Sichtbarmachen logischer Eigenschaften
  - Keine Überseztung

Vorbemerkung | Charakteristische Funktionen

Mengen, über Funktionen definiert

- Große Bedeutung von Mengen in formaler Semantik
- Charakteristische Funktion von Mengen
  $\mathcal{S}(a) = 1$ *iff* $a \in S$, *else* 0
- CF als Einsortierung in ihre Menge
- CF in Mengendefinitionen
  $S = \{x : x \text{ mod } 2 = 0\}$
  $\mathcal{S} = f(x)[x \text{ mod } 2 = 0]$
- Äquivalenz von Mengendenotation und CF-Dontation

Vorbemerkung | T-Sätze und Funktionsapplikation

Funktionsapplikation als allgemeiner Kompositionsmechanismus

- Etwas umständliche Interpretation mit T-Sätzen
  $[\![[_S \textit{ NP VP}]]\!]^{\mathcal{M},g} = 1$ *iff* $[\![\textit{NP}]\!]^{\mathcal{M},g} \in [\![\textit{VP}]\!]^{\mathcal{M},g}$
- CF statt Mengen | Funktion appliziert direkt!
  - $[\![\textit{Mary}]\!]^{\mathcal{M},g} = \textit{Mary in } \mathcal{M}$
  - $[\![\textit{sleeps}]\!]^{\mathcal{M},g}$ *be the CF of the set of sleepers in* $\mathcal{M}$
  - $[\![S]\!]^{\mathcal{M},g} = [\![\textit{sleeps}]\!]^{\mathcal{M},g}([\![\textit{Mary}]\!]^{\mathcal{M},g})$
  - Kein Bedarf an T-Sätzen

Vorbemerkung | Funktionen von Mengen zu Mengen

Funktionen von Mengen von (Tupeln von) Individuen zu Ausdrücken usw.

- Funktionen Definitionsbereich $S_1 \rightarrow$ Wertebereich S_2 | ${S_2}^{S_1}$
  $S_2^{S_1}$ | Die Menge aller Funktionen von S_1 zu S_2
- Beispiel | Einstellige und Zweistellige Prädikate
  - $T = \{0, 1\}$ | Wahrheitswerte
  - D | Diskursuniversum (Menge aller Individuen)
  - $D \times D$ | Menge aller 2-Tupel von Individuen
  - T^D | Menge aller Funktionen von Individuen zu Wahrheitswerten
    Menge der CFs aller einstelligen Prädikate
  - $T^{D \times D}$ | Menge aller Funktionen von 2-Tupeln von Individuen zu Wahrheitswerten
    Menge der CFs aller zweistelligen Prädikate

Einfach getypte Sprachen

Kein Bedarf an Phrasenkategorien

Logik hat bereits Typensysteme, um Syntax zu strukturieren!

- L_{Type} | Prädikatenlogik L_1 plus Typen
- Typen | Semantisch fundierte Klassen von Ausdrücken
- Einfache Typen
  - Terme | $\langle e \rangle$
  - Wffs/Formeln | $\langle t \rangle$ | Ersetzt Startsymbol S der PSG!
- Komplexe/funktionale Typen
  - Einstellige Prädikate | $\langle e, t \rangle$
  - Zweistellige Prädikate | $\langle e, \langle e, t \rangle \rangle$
- Allgemein | $\langle \sigma, \tau \rangle$-Ausdrücke denotieren Funktionen von Denotaten von $\langle \sigma \rangle$-Ausdrücken zu Denotaten von $\langle \tau \rangle$-Ausdrücken.

Modell | Denotate getypter Ausdrücke

Homogenes Diskursuniversum D (auch U und bei Dowty u. a. 1981 A)

- Allgemein D_α | Menge von Denotaten von Ausdrücken des Typs α
- Einfache Typen
  - $D_{\langle e \rangle} = U$
  - $D_{\langle t \rangle} = \{0, 1\}$
- Komplexe Typen | Rekursiv definierte Denotate
  -
  - Allgemein | $D_{\langle \alpha, \beta \rangle} = D_{\langle \beta \rangle}^{D_{\langle \alpha \rangle}}$
  - Einstelliuge Prädikate | $D_{\langle e,t \rangle} = D_{\langle t \rangle}^{D_{\langle e \rangle}}$
  - Zweistellige Prädikate | $D_{\langle e,\langle e,t \rangle \rangle} = (D_{\langle t \rangle}^{D_{\langle e \rangle}})^{D_{\langle e \rangle}}$
- Interpretation weiterhin durch V, g

Komplexe Typen für Funktionen und FA

$\langle\sigma\rangle$-Ausdrücke saturieren $\langle\sigma,\tau\rangle$-Ausdrücke zu $\langle\tau\rangle$-Ausdrücken.

- Beispiel für Saturierung durch Funktionsapplikation (FA)
  - Wenn P vom Typ $\langle e,\langle e,t\rangle\rangle$, Q vom Typ $\langle e,t\rangle$ und x, y vom Typ $\langle e\rangle$
  - dann ist $Q(x)$ vom Typ $\langle t\rangle$
  - und $P(x)$ vom Typ $\langle e,t\rangle$ sowie $P(x)(y)$ vom Typ $\langle t\rangle$

- Funktionale Typen von Funktoren
  - Negation $\neg$ | Typ $\langle t,t\rangle$
  - Andere Funktoren $\wedge, \vee, \rightarrow, \leftrightarrow$ | Typ $\langle t,\langle t,t\rangle\rangle$

Allgemeine Semantik für getypte Sprachen

Wirklich keine T-Sätze mehr!

- Semantik für $\langle e \rangle$-Typen (Terme)
  $[\![a_n]\!]^{\mathcal{M},g} = V(a_n)$
  $[\![x_n]\!]^{\mathcal{M},g} = g(x_n)$
- Ansonsten nur FA
  $[\![\delta(\alpha)]\!]^{\mathcal{M},g} = [\![\delta]\!]^{\mathcal{M},g}([\![\alpha]\!]^{\mathcal{M},g})$

Verallgemeinerung und Sprachen höherer Ordnung

Sprache höherer Ordnung = Sprache mit Variablen über höhere Typen $\langle \sigma, \tau \rangle$

- *Type* ist die Menge aller Typen
  - $\langle e \rangle, \langle t \rangle \in \mathit{Type}$
  - Wenn $\langle \sigma \rangle, \langle \tau \rangle \in \mathit{Type}$, dann $\langle \sigma, \tau \rangle \in \mathit{Type}$
  - Nichts sonst ist in *Type*.
- *ME* ist die Menge aller bedeutungsvollen Ausdrücke
  - ME_σ ist die Menge der Ausdrücke vom Typ σ | $ME = \bigcup ME_\sigma$ mit $\sigma \in \mathit{Type}$
  - *Ty* ist eine Funktion von Ausdrücken zu ihren Typen | $Ty(a) = \sigma$ *iff* $a \in ME_\sigma$
- Höhere Ordnung | Variablen über Ausdrücke von funktionalen Typen
  - $P_{\langle e,t \rangle}$ und $Q_{\langle e,\langle e,t \rangle \rangle}$ | Bekannte Konstanten höherer (=funktionaler) Typen
  - Parallel $v_{n_{\langle e,t \rangle}}$ | Die n-te Variable über einstellige Prädikate
  - Damit möglich $M = \{v_{1_{\langle e,t \rangle}} : [\![v_{1_{\langle e,t \rangle}}(m)]\!] = 1\}$
    Wenn $[\![m]\!] = \mathit{Maria}$, dann ist *M* die Menge von Marias Eigenschaften!

Systematische Interpretation zur systematischen Syntax

Zusammenfassung | Die Semantik reduziert sich auf FA und Variablenauswertung.

- Interpretation von Termen und Funktionsausdrücken
  - Nicht-logische Konstanten | α: $[\![\alpha]\!]^{\mathcal{M},g} = V(\alpha)$
  - Variablen | α: $[\![\alpha]\!]^{\mathcal{M},g} = V(\alpha)$
  - Wenn $\alpha \in \langle a, b\rangle$ und $\beta \in a$ dann $[\![\alpha(\beta)]\!]^{\mathcal{M},g} = [\![\alpha]\!]^{\mathcal{M},g}([\![\beta]\!]^{\mathcal{M},g})$
- Logische Konstanten (Typen $\langle t, t\rangle$ und $\langle t, \langle t, t\rangle\rangle$) denotieren Funktionen in $\{0, 1\}$.
- Quantoren
  - Für Variable $v_{1_{\langle\alpha\rangle}}$ und Wff $\phi \in ME_t$ ist $[\![(\forall v_1)\phi]\!]^{\mathcal{M},g} = 1$ gdw für alle $a \in D_\alpha$ $[\![\phi]\!]^{\mathcal{M},g[a/v_1]} = 1$
  - Für Variable $v_{1_{\langle\alpha\rangle}}$ und Wff $\phi \in ME_t$ ist $[\![(\exists v_1)\phi]\!]^{\mathcal{M},g} = 1$ gdw für mindestens ein $a \in D_\alpha$ $[\![\phi]\!]^{\mathcal{M},g[a/v_1]} = 1$

Beispiel | Quantifikation über Prädikate

$\forall v_{0_{\langle e,t\rangle}} \left[v_{0_{\langle e,t\rangle}}(j) \rightarrow v_{0_{\langle e,t\rangle}}(d) \right]$

- Eine quantifizierbare Variable vom Typ $\langle e, t\rangle$ | $v_{0_{\langle e,t\rangle}}$
- Zwei Individuenkonstanten | $j, d \in ME_{\langle e\rangle}$ z. B. John und Dorothy
- *Für alle einstelligen Prädikate gilt: Wenn j die vom Prädikat beschriebene Eigenschaft hat, hat d auch diese Eigenschaft.*
- Wann ist diese Wff wahr?
  - Wenn *j* und *d* alle benennbaren Eigenschaften teilen?
  - Eine Eigenschaft jedes Objekts |
    CF der Menge $\{x : x \textit{ is the sole member of this set}\}$ (*union set*)
  - Einzige Möglichkeit für Wahrheit der Wff also j=d

Beispiel | Wortbildung mit Präfix *non*

non in Sätzen wie *This function is non-continuous*.

- Produktives Suffix im Englischen, wie *nicht-* im Deutschen
- Bedeutungsbeitrag | Invertiert die CF eines Adjektivs
- Komplementbildung der Ursprungsmenge in $D_{\langle e,t\rangle}$
- Syntax und Semantik von *non*
  - Adjektiv *continuous* | Typ $\langle e,t\rangle$
  - Typ von *non* | In: Adjektiv /Out: Adjektiv | $\langle\langle e,t\rangle,\langle e,t\rangle\rangle$
  - $[\![non]\!]^{\mathcal{M},g} = h$ s. t. $h \in D_{\langle\langle e,t\rangle,\langle e,t\rangle\rangle}$ and for every $k \in D_{\langle e,t\rangle}$ and every $d \in D_{\langle e\rangle}$ $(h(k))(d) = 1$ iff $k(d) = 0$ and $(h(k))(d) = 0$ iff $k(d) = 1$

Beispiel | Argumentunterdrückung

Optionale Argumente wie in *I eat.* oder *Vanity kills.*

- Zweistellige Verben wie *eat* in $ME_{\langle e,\langle e,t\rangle\rangle}$
- Aus einem zweistelligen Verb ein einstelliges machen
  - Phonologisch leere lexikalische Konstante | $R_0 \in ME_{\langle\langle e,\langle e,t\rangle\rangle,\langle e,t\rangle\rangle}$
    Ähnlich wie lexikalische Regeln in HPSG
  - Semantik | $[\![R_0]\!]^{\mathcal{M},g} = h$ s. t. $h \in D_{\langle\langle e,\langle e,t\rangle\rangle,\langle e,t\rangle\rangle}$ and for all $k \in D_{\langle e,\langle e,t\rangle\rangle}$ and all $d \in D_{\langle e\rangle}$ $(h(k))(d) = 1$ iff there is some $d' \in D_{\langle e\rangle}$ s. t. $k(d')(d) = 1$

λ-Sprachen

Sie kennen bereits λ-Abstraktionen!

Was bedeutet $f(x) = 3x^2 + 5x + 8$?

- $3x^2 + 5x + 8$ ist eine Wff mit einer ungebundenen Variable.
- Die Variable wird gebunden und die Wff wird damit zur Funktion
  x wird zur Eingabevariable und muss bei Anwendung durch Eingabewert ersetzt werden.
- Außerdem wird die Funktion f genannt.

In λ-Notation: $f \stackrel{def}{=} \lambda x \left[3x^2 + 5x + 8\right]$

Nur ein neuer Variablenbinder

Mit λ bildet man ad hoc anonyme Funktionen.

- Abstraktion über Wffs beliebiger Komplexität
- λ-Bindung der Variable | Gebundene Variable als Eingabevariable der Funktion
- Sehr ähnlich wie Listendefinition
  - Menge | $\{x : x \; mod \; 2 = 0\}$ | allgemein $\{x : \phi\}$
  - CF dieser Menge | $\lambda x \, [x \; mod \; 2 = 0]$ | allgemein $\lambda x \, [\phi]$

Formale Erweiterung von L_{Type}

Nur wenige Erweiterungen in L_{Type}

- Für jede Wff ϕ mit $Ty(\phi) = \langle t \rangle$ und jede $x \in Var$ und jede $a \in Con$
  - Abstraktion | $\phi \implies \lambda x \left[\phi^{[a/x]}\right]$
    Definition $\phi^{[a/x]}$ | Wff ϕ in der alle a durch x getauscht wurden
  - Anwendung der Funktion (λ-Konversion) | $\lambda x \left[\phi^{[a/x]}\right](a) = \phi$
- Es gilt $\lambda x \left[\phi^{a/x}\right](a) \equiv \phi$ für jede Wff ϕ, jede $a \in Con$ und jede $x \in Var$
- x kann von einem beliebigen Typ σ sein.
- Es gilt für $\lambda x [\phi]$ mit $x \in ME_{\langle \sigma \rangle}$ stets $\phi \in ME_{\langle t \rangle}$ sowie $\lambda x [\phi] \in ME_{\langle \sigma, t \rangle}$

Zwei Beispiele

Abstraktion über Individuenvariable und Prädikatsvariable

- Individuenvariable $x_{\langle e \rangle}$ alternativ $v_{1_{\langle e \rangle}}$
  - $\lambda x_{\langle e \rangle}\ [L(x)]$
  - Mit L z. B. für *laughs*
  - Die CF der Menge von Individuen $d \in D_{\langle e \rangle}$ mit die Eigenschaft L (alle Lachenden)
  - Mengendefinition dazu $\{x : L(x)\}$
- Prädikatsvariable $P_{\langle e,t \rangle}$ alternativ $v_{1_{\langle e,t \rangle}}$
  - $\lambda P_{\langle e,t \rangle}\ [P(l)]$
  - Mit l z. B. für *Horst Lichter*
  - Die CF aller Eigenschaften $k \in D_{\langle e,t \rangle}$ von l (alle Eigenschaften Horst Lichters)
  - Mengendefinition dazu $\{P : P(l)\}$

Als wäre das jetzt nicht schon klar …

Die vollen Regeln aus Dowty u. a. (1981: 102) (Syn C.10 and Sem 10)

- If $\alpha \in ME_\alpha$ and $u \in Var_b$, then $\lambda u\,[\alpha] \in ME_{\langle b,a \rangle}$.
- If $\alpha \in ME_a$ and $u \in Var_b$ then $[\![\lambda u\,[\alpha]]\!]^{\mathcal{M},g}$ is that function h from D_b into D_a ($h \in D_a^{D_b}$) s. t. for all objects k in D_b, $h(k)$ is equal to $[\![\alpha]\!]^{\mathcal{M},g[k/u]}$.

Konversionen

Konversionen/Reduktionen | Arten, λ-Ausdrücke umzuschreiben

- α-Konversion | Umbenennung von Variablen
  - $\lambda x\,[\phi] \stackrel{\alpha}{\equiv} \lambda y\,\left[\phi^{[x/y]}\right]$ gdw y in ϕ nicht vorkommt
- β-Reduktion | Funktionsapplikation
  - $\lambda x\,[\phi]\,(a) \stackrel{\beta}{\equiv} \phi^{[x/a]}$
  - Ausdrucke mit nicht realisierten, aber möglichen β-Reduktionen: β-Redex
- η-Reduktion | Entfernen von leeren Abstraktionen
  - $\lambda x\,[F(x)] \stackrel{\eta}{\equiv} F$ gdw $Ty(F) = Ty(\lambda x\,[F(x)])$ (und x nicht frei in F ist)
  - Ausdruck mit nicht realisierten, aber möglichen η-Reduktionen: η-Redex
  - Mäßige Semantiker | η-Redex-Fetisch mit $\lambda x \lambda y \lambda z\,[gibt'(x, y, z)]$ usw.

The *non* example revised (Dowty et al., 104)

- $\forall x \forall v_{0\langle e,t\rangle} \left[(\mathbf{non}(v_{0\langle e,t\rangle}))(x) \leftrightarrow \neg(v_{0\langle e,t\rangle}(x)) \right]$
- $\forall v_{0\langle e,t\rangle} \left[\lambda x \left[(\mathbf{non}(v_{0\langle e,t\rangle}))(x) \right] = \lambda x \left[\neg(v_{0\langle e,t\rangle}(x)) \right] \right]$
- $\forall v_{0\langle e,t\rangle} \left[\mathbf{non}(v_{0\langle e,t\rangle}) = \lambda x \left[\neg(v_{0\langle e,t\rangle}(x)) \right] \right]$

  (since $\lambda x\,[\mathbf{non}(v)(x)]$ is unnecessarily abstract/η reduction)
- $\lambda v_{0\langle e,t\rangle} \left[\mathbf{non}(v_{0\langle e,t\rangle}) = \lambda v_{0\langle e,t\rangle} \left[\lambda x \left[\neg(v_{0\langle e,t\rangle}(x)) \right] \right] \right]$
- and since that is about all assignments for $\lambda v_{0\langle e,t\rangle}$:

  $\mathbf{non} = \lambda v_{0\langle e,t\rangle} \left[\lambda x \left[\neg v_{0\langle e,t\rangle}(x) \right] \right]$

Mary is non-adjacent.

(translate ‘adjacent’ as $c_{0_{\langle e,t\rangle}}$, ‘Mary’ as $c_{0_{\langle e\rangle}}$, ignore the copula)

Ausblick auf Quantifikation bei Montague

The behavior of quantified NPs

- syntactically like referential NPs
- semantically like PC quantifiers
- *Every student walks.*: $\forall v_{0_{\langle e \rangle}} \left[c_{0_{\langle e,t \rangle}}(v_{0_{\langle e \rangle}}) \rightarrow c_{1_{\langle e,t \rangle}}(v_{0_{\langle e \rangle}}) \right]$
- *Some student walks.*: $\forall v_{0_{\langle e \rangle}} \left[c_{0_{\langle e,t \rangle}}(v_{0_{\langle e \rangle}}) \wedge c_{1_{\langle e,t \rangle}}(v_{0_{\langle e \rangle}}) \right]$
- making referential NPs and QNPs the same type?

A higher type

- $\lambda v_{0_{\langle e,t\rangle}} \forall v_{0_{\langle e\rangle}} \left[c_{0_{\langle e,t\rangle}}(v_{0_{\langle e\rangle}}) \rightarrow v_{0_{\langle e,t\rangle}}(v_{0_{\langle e\rangle}}) \right]$
- a second order function
- characterizes the set of all predicates true of every student
- equally: $\lambda v_{0_{\langle e,t\rangle}} \exists v_{0_{\langle e\rangle}} \left[c_{0_{\langle e,t\rangle}}(v_{0_{\langle e\rangle}}) \wedge v_{0_{\langle e,t\rangle}}(v_{0_{\langle e\rangle}}) \right]$

Combining with some predicate

Aufgaben I

- Wie kann man Passiv in L_{Type} modellieren?

Literatur I

Chierchia, Gennaro & Sally McConnell-Ginet. 2000. *Meaning and grammar: An introduction to semantics.* 2. Aufl. Cambridge, MA: MIT Press.

Dowty, David R., Robert E. Wall & Stanley Peters. 1981. *Introduction to Montague semantics.* Dordrecht: Kluwer.

Autor

Kontakt

Prof. Dr. Roland Schäfer
Institut für Germanistische Sprachwissenschaft
Friedrich-Schiller-Universität Jena
Fürstengraben 30
07743 Jena

https://rolandschaefer.net
roland.schaefer@uni-jena.de

Lizenz