Hypothetical Reasoning and the Grammar of Signs

Yoad Winter

3.12.2010, DIP Colloquium, Amsterdam

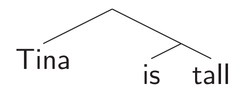
Textbook draft – www.phil.uu.nl/~yoad/efs/main.html

Minimalist Formal Semantics

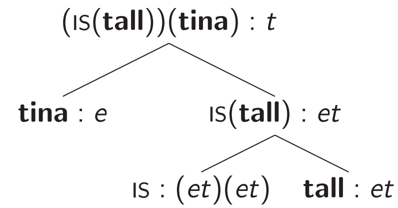
- Trees over <u>strings</u>
- Lexical <u>semantic</u> types and denotations
- Inductive interpretation of trees using <u>function application</u>

Example

A.



B.



Problem: non-local dependencies – quantifiers, scope, extraction

Towards a Saussurean grammar

- Trees over linguistic <u>signs</u> – pairs of pheno-level+semantic-level denotations
- Lexical types for such pairs
- Inductive interpretation of sign trees using function application and <u>function abstraction</u>

Insight: non-local dependencies are described within the derived signs

Plan of talk

- λ-terms as mere notation for functions
- Three classic problems in the Syntax-Semantics interface
- The Semantic Operator approach vs. the λ-Syntax approach
- Hypothetical Reasoning and the Lambek-Van Benthem Calculus
- Abstract Categorial Grammar – a Saussurean perspective

Why λ-terms?

1. Easy notation for functions

When writing "$\lambda x_\tau.\varphi$", where τ is a type, we mean: "the function sending every element x of the domain D_τ to φ".

$$[\![\text{Tina is tall}]\!] = (\textit{IS}_{(et)(et)}(\mathbf{tall}_{et}))(\mathbf{tina}_e)$$

The denotation IS of *is* =

a. the function sending every element x of the domain D_{et} to x

b. $\lambda x_{et}.x$

2. Easy notation for applying functions (β-reduction over-simplified)

The result $(\lambda x_\tau.\varphi)(a_\tau)$ of applying a function described by a lambda term $\lambda x_\tau.\varphi$ to an argument a_τ, is equal to the value of the expression φ, with all occurrences of x replaced by a.

a. the result of applying the function sending every element x of the domain D_{et} to x to the function $\mathbf{tall}_{et}$ is **tall**

b. $(\lambda x_{et}.x)(\mathbf{tall}_{et}) = \mathbf{tall}$

Thus:
$(\text{IS}_{(et)(et)}(\mathbf{tall}_{et}))(\mathbf{tina}_e) = ((\lambda x_{et}.x)(\mathbf{tall}))(\mathbf{tina}_e) = \mathbf{tall}(\mathbf{tina}_e)$

> *A priori, λ-terms and their reductions are just notation for functions and simple set-theoretic equivalences.*

Three classic problems

Quantifiers in object position

Tina praised every student

What do we do with types $e(et)$ and $(et)t$?

Quantifier scope

Some teacher praised every student

How do we derive the object wide scope reading?

Extraction

Some teacher that Mary praised smiled

How can we interpret constituents like Mary praised?

Object quantifiers – desired derivation

In our example –

	praise	$e(et)$
+	$\textit{EVERY}(\mathbf{student})$	$(et)t$
	$\lambda y_e.\textit{EVERY}(\mathbf{student})(\lambda x_e.\mathbf{praise}(x)(y))$	et

= *the function that characterizes the set of entities y s.t. $\{x \in D_e : y \text{ praised } x\}$ contains the set of students*

In general

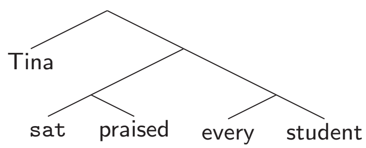
	R	$e(et)$
+	Q	$(et)t$
	$\lambda y_e.Q(\lambda x_e.R(x)(y))$	et

A *saturation* operator for binary relations and quantifiers

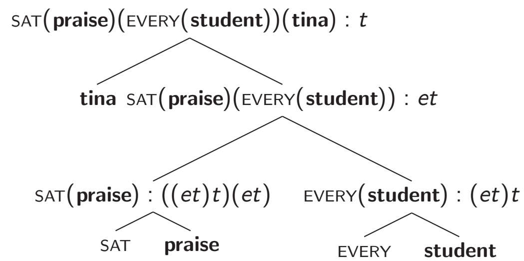
$$\textit{SAT} = \lambda R_{e(et)}.\lambda Q_{(et)t}.\lambda y_e.Q(\lambda x_e.R(x)(y))$$

Object quantifiers – the Semantic Operator approach

A.

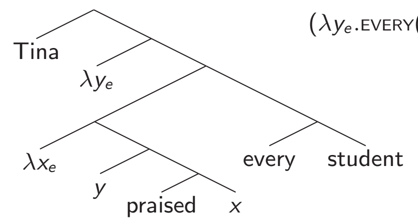


B.

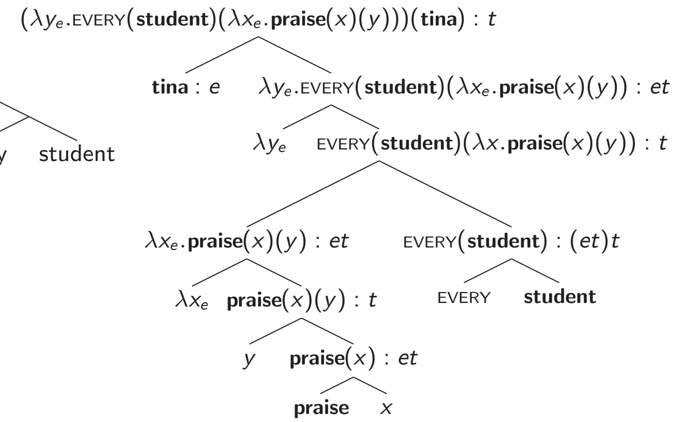


Object quantifiers – the λ-Syntax approach

A.

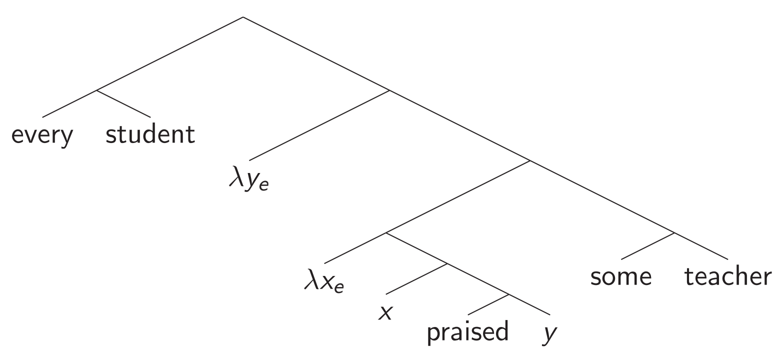


B.



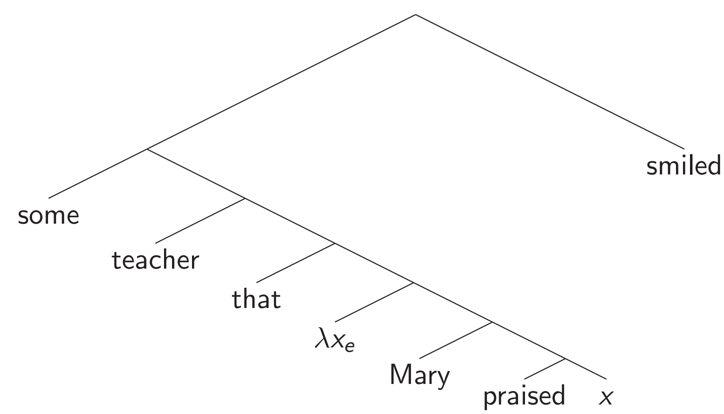
Object wide scope – the λ-Syntax approach

Some teacher praised every student



Extracted object – the λ-Syntax approach

Some teacher that praised Mary smiled



Semantic operators vs. λ-Syntax

Both approaches stipulate the desired outcome:

- using the semantic operator that encodes it

  OR
- in the syntactic derivation of the desired λ-term

We are left with the question:

> *What are the principles that get us the denotation below from the denotations* **mary** *and* **praise**?

$$\lambda x_e.((\mathbf{praise}_{e(et)}(x))\,(\mathbf{mary}_e))$$

$$PBM = \{x \in D_e : \textit{Mary praised } x\ \}$$

Proposal (Saussure, Lambek, Curry, Van Benthem, De Groote, Muskens): Hypothetical Reasoning + a Grammar of Signs

Function Application and Modus Ponens

Function Application (FA) Rule

$$\frac{\tau\sigma \qquad \tau}{\sigma}$$

Interpretation

$$\frac{A \qquad B}{A(B)}$$

Implication Elimination (Modus Ponens)

$$\frac{\varphi \to \psi \qquad \varphi}{\psi}$$

If Mary is tall then Tina is tall,
and Mary is tall
$\Rightarrow$ Tina is tall

Hypothetical Reasoning – Two Equivalent Patterns

(A) Tina is taller than Mary
$\Rightarrow$ If Mary is tall then Tina is tall

(B) Tina is taller than Mary
and Mary is tall
$\Rightarrow$ Tina is tall

Proving (B) using (A)

$$\dfrac{\dfrac{\text{Tina is taller than Mary}}{\text{If Mary is tall then Tina is tall}}\,(A) \qquad \text{Mary is tall}}{\text{Tina is tall}}\,\text{MP}$$

Proving (A) using (B)

$$\dfrac{\dfrac{\text{Tina is taller than Mary} \qquad [\text{Mary is tall}]^1}{\text{Tina is tall}}\,(B)}{\text{If Mary is tall then Tina is tall}}\,\text{discharge hypothesis 1}$$

Implication Introduction

Implication Introduction

$$\dfrac{\dfrac{\dfrac{\ldots \qquad [\varphi]^1}{\vdots}}{\psi}}{\varphi \to \psi}\,\text{discharge hypothesis 1}$$

Example

$$\dfrac{\dfrac{\dfrac{\varphi_1 \to (\varphi_2 \to \psi) \qquad [\varphi_1]^1}{\varphi_2 \to \psi}\,\text{MP} \qquad \varphi_2}{\psi}\,\text{MP}}{\varphi_1 \to \psi}\,\text{discharge hypothesis 1}$$

Function Abstraction

Function Introduction

$$\dfrac{\begin{matrix} \ldots & [\tau]^1 \\ & \vdots \end{matrix}}{\dfrac{\sigma}{\tau\sigma}\ \text{discharge hypothesis 1}}$$

Example

$$\dfrac{\dfrac{e(et) \quad [e]^1}{et}\ \text{APP} \qquad e}{\dfrac{t}{et}\ \text{discharge hypothesis 1}}\ \text{APP}$$

Function Abstraction – Interpretation

Function Introduction

$$\dfrac{\begin{matrix} \ldots & [u : \tau]^1 \\ & \vdots \end{matrix}}{\dfrac{z : \sigma}{\lambda u.z : \tau\sigma}\ \text{discharge hypothesis 1}}$$

Example

$$\dfrac{\dfrac{\dfrac{\textbf{praise} : e(et) \quad [u : e]^1}{\textbf{praise}(u) : et}\ \text{FA} \qquad \textbf{mary} : e}{\textbf{praise}(u)(\textbf{mary}) : t}\ \text{FA}}{\lambda u_e.\textbf{praise}(u)(\textbf{mary}) : et}\ \text{discharge hypothesis 1}$$

Intermediate summary

- Semantic engine: type-logical, over denotations
- Interpreted objects: trees over strings
- No syntactic variables

Using Application: **praise**(**mary**)

- **mary** **praise**

Using Abstraction: $\lambda u.$**praise**(u)(**mary**)

- **mary** **praise**

Application (Ajdukiewicz):
undergeneration – object quantifiers, wide scope, extraction
overgeneration – extraction

Application + Abstraction (Lambek-Van Benthem):
less undergeneration
more overgeneration

Using signs

"The linguistic sign unites, not a thing and a name, but a concept and a sound-image."
(de Saussure 1916)

> *A linguistic sign, or in short a* **sign**, *is a pair* $\langle P, C\rangle$, *where P stands for a* perceptual *representation of sensory input and C stands for a* conceptual *representation of meaning.*

Sign composition:

MARY (sign)	`mary`	(perception)
	mary	(concept)
+		
PRAISE (sign)	`praise`	(perception)
	praise	(concept)

=... (two possibilities)

Pheno-level interpretation

The domain of phonetic entities $D_f = F$ satisfies:

- *Closure under concatenation.* For all phonetic entities $a, b \in F$, the concatenation $a \cdot b$ is also in F.
- *Neutral element for concatenation.* F contains an element ϵ that satisfies for every $x \in F$: $x \cdot \epsilon = \epsilon \cdot x = x$.

Pheno-types: f is a pheno-type. If σ and τ are pheno-types then $(\sigma\tau)$ is a pheno-type as well.

Example: In a given model –

- $\texttt{tina}_f = \mathit{tina}$
- $\texttt{mary}_f = \mathit{mary}$
- $\texttt{praise}_{f(ff)} = \lambda x_f.\lambda y_f.\ y \cdot \mathit{praised} \cdot x$

$$\texttt{praise}_{f(ff)}(\texttt{mary}_f)(\texttt{tina}_f) = \mathit{tina} \cdot \mathit{praised} \cdot \mathit{mary}$$
$$\texttt{praise}_{f(ff)}(\texttt{tina}_f)(\texttt{mary}_f) = \mathit{mary} \cdot \mathit{praised} \cdot \mathit{tina}$$

Application and Abstraction using signs

Application

$$\frac{\langle \texttt{praise}_{f(ff)}, \textbf{praise}_{e(et)} \rangle \quad \langle \texttt{mary}_f, \textbf{mary}_e \rangle}{\langle \texttt{praise}(\texttt{mary}), \textbf{praise}(\textbf{mary}) \rangle} \ APP$$

In our model: $\langle \lambda y_f.\ y \cdot \mathit{praised} \cdot \mathit{mary}$, entities that praised Mary$\rangle$

Abstraction

$$\frac{\dfrac{\dfrac{\langle \texttt{praise}_{f(ff)}, \textbf{praise}_{e(et)} \rangle \quad [\langle u_f, u_e \rangle]^1}{\langle \texttt{praise}(u_f), \textbf{praise}(u_e) \rangle}\ \text{FA} \quad \langle \texttt{mary}_f, \textbf{mary}_e \rangle}{\langle \texttt{praise}(u_f)(\texttt{mary}),\ \textbf{praise}(u_e)(\textbf{mary}) \rangle}\ \text{FA}}{\langle \lambda u_f.\texttt{praise}(u_f)(\texttt{mary}),\ \lambda u_e.\textbf{praise}(u_e)(\textbf{mary}) \rangle}\ \text{discharge hypothesis 1}$$

In our model: $\langle \lambda u_f.\mathit{mary} \cdot \mathit{praise} \cdot u$, entities that Mary praised$\rangle$

Hypothetical reasoning without overgeneration!

Hypothesis: The Lambek-Van Benthem Calculus is a suitable logical apparatus for manipulating the composition of signs in natural language grammar.

Back to relative clauses (1)

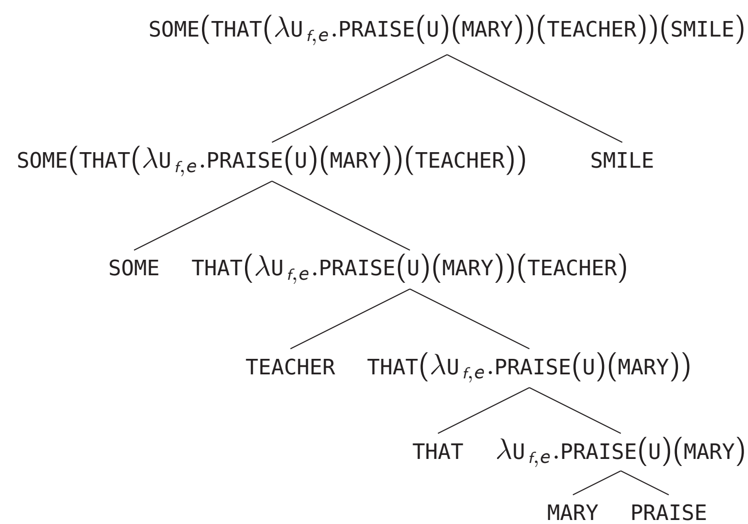
$$\texttt{that} = \lambda P_{ff}.\lambda y_f.\ y \cdot \mathit{that} \cdot P(\epsilon)$$

$$\texttt{mp} = \lambda u_f.\mathit{mary} \cdot \mathit{praised} \cdot u$$

$\texttt{that}(\texttt{mp})$
$= (\lambda P_{ff}.\lambda y_f.\ y \cdot \mathit{that} \cdot P(\epsilon))(\texttt{mp})$
$= \lambda y_f.\ y \cdot \mathit{that} \cdot \texttt{mp}(\epsilon)$
$= \lambda y_f.\ y \cdot \mathit{that} \cdot ((\lambda u_f.\mathit{mary} \cdot \mathit{praised} \cdot u)(\epsilon))$
$= \lambda y_f.\ y \cdot \mathit{that} \cdot (\mathit{mary} \cdot \mathit{praised} \cdot \epsilon)$
$= \lambda y_f.\ y \cdot \mathit{that} \cdot \mathit{mary} \cdot \mathit{praised}$

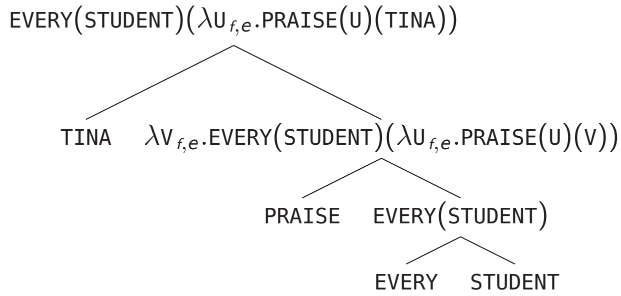
Back to relative clauses (2)

some · teacher · that · mary · praised · smiled
SOME(THAT(λu_e.**praise**(u)(**mary**))(**teacher**))(**smile**)



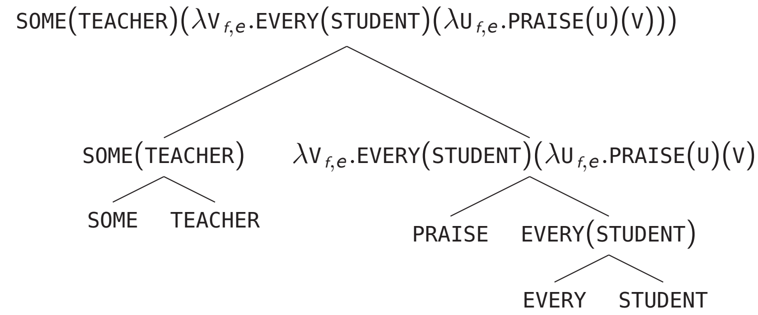
Back to quantificational object noun phrases

tina · praised · every · student
$\textsc{every}(\textbf{student})(\lambda u_e.\textbf{praise}(u)(\textbf{tina}))$



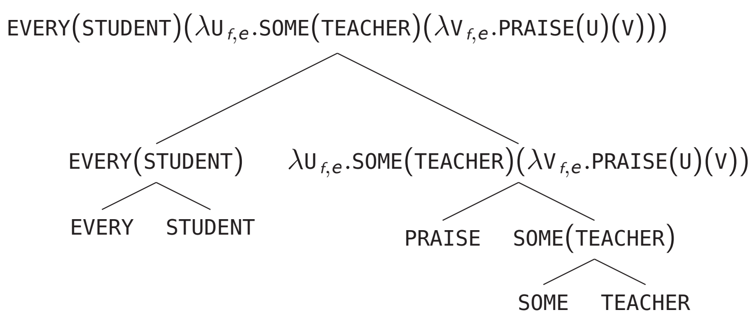
Back to quantifier scope (1)

some · teacher · praised · every · student
$\textsc{some}(\textbf{teacher})(\lambda v_e.\textsc{every}(\textbf{student})(\lambda u_e.\textbf{praise}(u)(v)))$



Back to quantifier scope (2)

some · teacher · praised · every · student
$\textsc{every}(\textbf{student})(\lambda u_e.\textsc{some}(\textbf{teacher})(\lambda v_e.\textbf{praise}(u)(v)))$



Two parameters:

- Order of composition of signs – determines semantic scope
- Sign argument saturated – determines syntactic position

Summary

- Lambek-Van Benthem Calculus – flexibility of hypothetical reasoning
- Directionality is not in tecto-level syntax, but in the pheno-level objects that it manipulates
- Saussurean signs – avoiding overgeneration
- Implications:
  - Modeltheoretic phonology
  - Free variables in grammar, not in meaning
  - Syntax and semantics hand in hand

References

Curry, H. B. (1961), Some logical aspects of grammatical structure, *in* R. O. Jakobson, ed., 'Structure of Language and its Mathematical Aspects', Vol. 12 of *Symposia on Applied Mathematics*, American Mathematical Society, Providence.

de Groote, P. (2001), Towards abstract categorial grammars, *in* 'Proceedings of the 39th annual meeting of the Association for Computational Linguistics (ACL)'.

de Saussure, F. (1959), *Course in General Linguistics*, Philosophical Library, New York. Translation of *Cours de Linguistique Générale*, Payot & Cie, Paris, 1916.

Lambek, J. (1958), 'The mathematics of sentence structure', *American Mathematical Monthly* **65**, 154–169.

Muskens, R. (2003), Language, Lambdas, and Logic, *in* G.-J. Kruijff & R. Oehrle, eds, 'Resource Sensitivity in Binding and Anaphora', Studies in Linguistics and Philosophy, Kluwer, pp. 23–54.

van Benthem, J. (1991), *Language in Action: categories, lambdas and dynamic logic*, North-Holland, Amsterdam.