Higher-Order Logic for Natural Language Semantics

Yoad Winter

Utrecht Institute of Linguistics

May 15, 2018

WOL Departementsmiddag, Mathematisch Instituut, Utrecht

Inferences in language

Classical syllogisms:

Every human is mortal, and Socrates is human $\Rightarrow$ *Socrates is mortal*

Other inferences with quantifiers:

More teachers than students snore, and at most five teachers snore $\Rightarrow$ *At most four students snore*

Plurals and reciprocity:

Dan and Sue killed each other $\Rightarrow$ *Dan killed Sue and Sue killed Dan*

Spatial and temporal inferences:

The bird is above the house and between the house and the cloud $\Rightarrow$ *The bird is below the cloud*

Events:

Carthage was destroyed in 146 BC $\Rightarrow$ *Carthage's destruction occurred in 146 BC*

Presupposition:

If John stops smoking he'll be able to run the marathon $\Rightarrow$ *John smokes*

Traditional view

Inferences in language are translated to formal logic.

Every human is mortal, and Socrates is human $\Rightarrow$ *Socrates is mortal*

$(\forall x.H(x) \rightarrow M(x)) \wedge H(\mathbf{s}) \Rightarrow M(\mathbf{s})$

Translation procedure is left undefined.

Modern view

Since 1970s:

- Inferences as *empirical facts* – psychology, linguistics, computer science.
- *Applied* logic.
- In Linguistics –
  Formal Semantics: study of (in)valid inferences in language
  Formal Syntax: study of (un)grammaticality in language
  **mortal every is human*
- **Challenge**: the scientific study of grammar as mediating between syntactic structures of natural language and logical inferences

Plan of Talk

1. Introduction
2. Type-logical semantics (1970s-2000s)
3. Abstract type-logical grammar (2000s-)

Common thread: Higher-Order Logics and λ-Calculus

A friendly intro

Farmer, William M. The seven virtues of simple type theory. *Journal of Applied Logic* 6 (2008):267–286.

Types and lexical typing

e: entities

t: truth-values

Inductively: et – functions from entities to truth-values

$e(et)$ – functions from entities to et functions

$(e(et))(et)$ – functions from $e(et)$ functions to et functions

...

Types over e and t:

$$\mathcal{T}^{e,t} \stackrel{def}{=} \{e\} \cup \{t\} \cup \{(\tau\sigma) \mid \tau, \sigma \in \mathcal{T}\}$$

Words:

W = finite non-empty set

Lexical typing over W:

$T_0 : W \to \mathcal{T}$ mapping words to types

Example: $T_0(Socrates) = e$; $T_0(mortal) = et$

Expressions and their typing

EXP = the *expressions* over W, typed by a function $T : EXP \to \mathcal{T}$

- $W \subseteq EXP \subseteq W^*$

  T extends T_0
- Concatenation is possible when function application is.
- **Notation** $\varphi : \tau$ or φ_τ "φ is an expression of type τ"
- **Example**: $$\frac{Socrates : e \quad mortal : et}{Socrates \cdot mortal : t}$$

Expressions over W and T_0

Base: $w : T_0(w)$

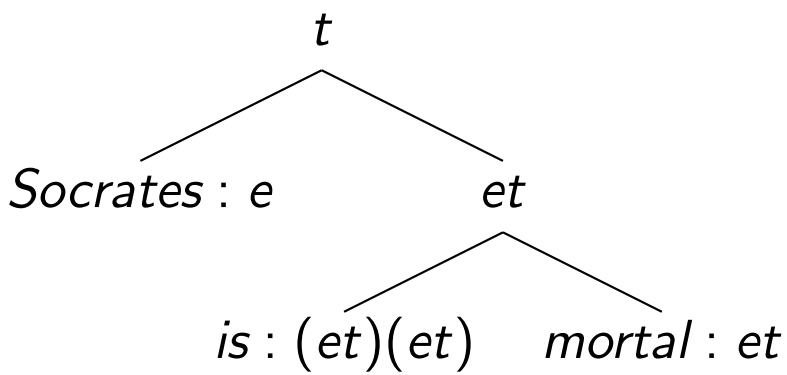
every $w \in W$ is an expression of type $T_0(w)$

Induction: $$\frac{\varphi : \tau\sigma \quad \psi : \tau}{\varphi \cdot \psi : \sigma} \qquad \frac{\varphi : \tau\sigma \quad \psi : \tau}{\psi \cdot \varphi : \sigma}$$

Example

$$\dfrac{\mathit{Socrates} : e \qquad \dfrac{\mathit{is} : (et)(et) \quad \mathit{mortal} : et}{\mathit{is}\cdot\mathit{mortal} : et}}{\mathit{Socrates}\cdot\mathit{is}\cdot\mathit{mortal} : t}$$

Tree notation:



Models and denotations

$D_e \quad = \quad$ non-empty set

$D_t \quad = \quad \{0,1\}_{\leq}$

$D_{\tau\sigma} \quad = \quad D_\sigma{}^{D_\tau}$

Interpretation over words W and lexical typing T_0:

$I : W \to \bigcup_{\tau \in \mathcal{T}} D_\tau$, s.t. for every $w \in W$: $I(w) \in D_{T_0(w)}$

– mapping each word to an element of the domain of its type

Model over W and typing $T_0 \quad = \quad$ a pair $\langle D_e, I \rangle$

– the **denotation** in a model $M = \langle D_e, I \rangle$ extends I for expressions

Expression denotation over $M = \langle D_e, I \rangle$

Base: $[\![w]\!]^M = I(w)$

every $w \in W$ denotes its interpretation in M

Induction:

$$\frac{[\![\varphi]\!]^M = A_{\tau\sigma} \quad [\![\psi]\!]^M = B_\tau}{[\![\varphi \cdot \psi]\!]^M = A(B)} \qquad \frac{[\![\varphi]\!]^M = A_{\tau\sigma} \quad [\![\psi]\!]^M = B_\tau}{[\![\psi \cdot \varphi]\!]^M = A(B)}$$

Example: denotations in a model

Model:

$I(\textit{Socrates}) = \mathbf{s} \in D_e \qquad I(\textit{mortal}) = \mathbf{mortal} \in D_{et}$

$I(\textit{is}) = \textsc{is} \in D_{(et)(et)}$

the *identity function* of type $(et)(et)$

denoted "$\lambda P_{et}.P$"

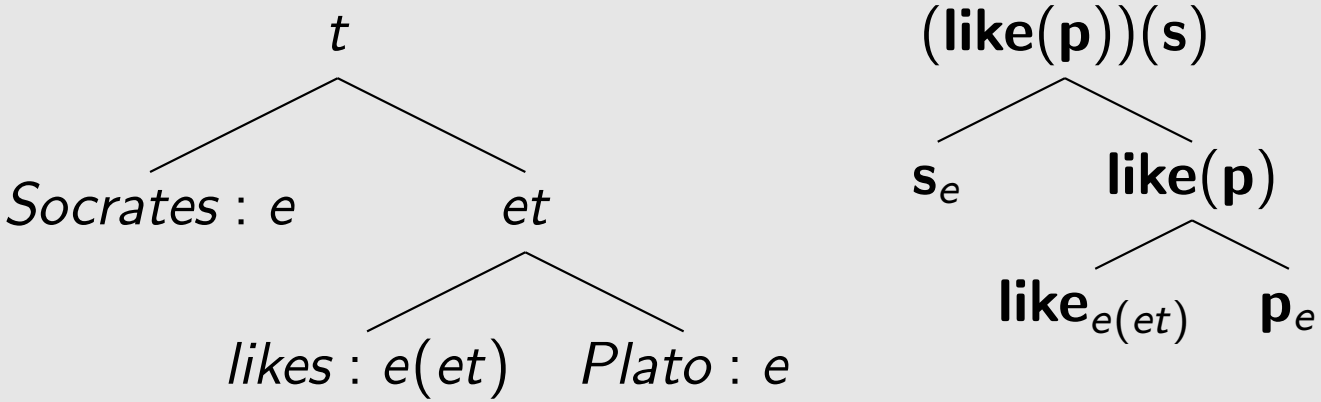
Denotations:

$[\![\textit{is mortal}]\!]^M = \textsc{is}(\mathbf{mortal}) = (\lambda P.P)(\mathbf{mortal}) = \mathbf{mortal}$

$[\![\textit{Socrates is mortal}]\!]^M = [\![\textit{is mortal}]\!]^M([\![\textit{Socrates}]\!]^M) = \mathbf{mortal}(\mathbf{s})$

Example: simple transitive sentences

Socrates likes Plato



$e(et)$ functions $\equiv$ binary relations over D_e

$\mathbf{like}_{e(et)} \equiv LIKE = \{\langle x, y\rangle \in D_e^2 \mid (\mathbf{like}(y))(x)\}$

$\mathbf{like}(\mathbf{p}) \equiv \{x \in D_e \mid LIKE(x, \mathbf{p})\}$

λ-calculus quick and dirty

Abstraction: $\lambda x_\tau.\varphi_\sigma$

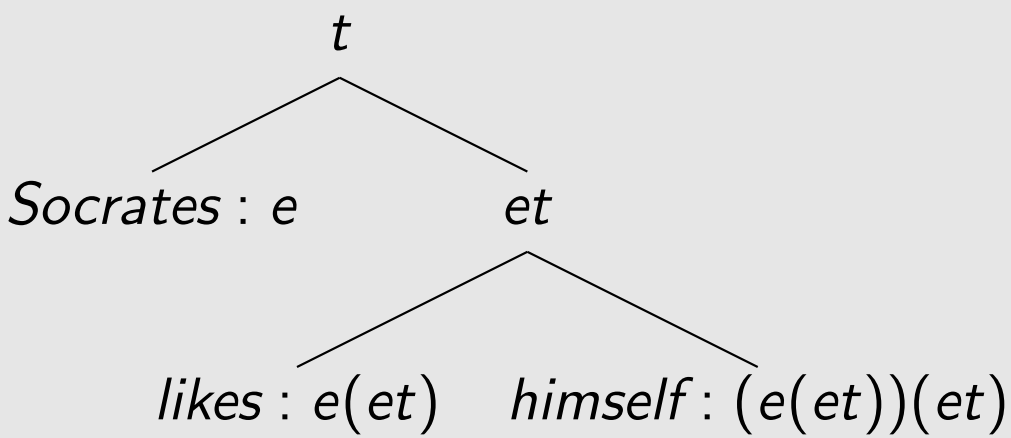
describes a function from D_τ to D_σ: substitute value in D_τ for every free x in φ

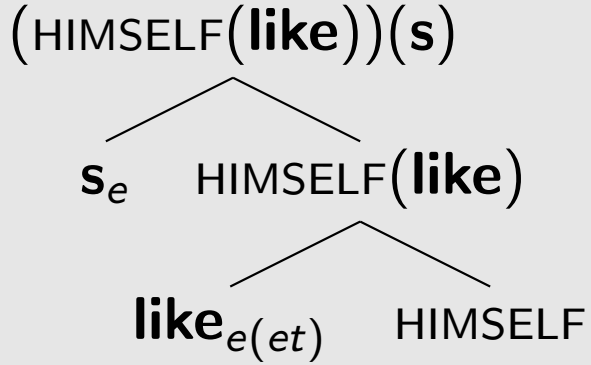
Application: $\varphi_{\tau\sigma}(\psi_\tau)$

describes an object in D_σ: the result of applying the function described by φ to the object described by ψ

Example: reflexives in transitive sentences

Socrates likes himself





HIMSELF $= \lambda R_{e(et)}.\lambda x_e.(R(x))(x)$

map binary relation $R' \subseteq E \times E$ to set $\{x \in E \mid \langle x, x \rangle \in R'\}$

$$
\begin{aligned}
&(\text{HIMSELF}(\textbf{like}))(\textbf{s}) \\
&= ((\lambda R_{e(et)}.\lambda x_e.(R(x))(x))(\textbf{like}))(\textbf{s}) \\
&= (\lambda x_e.(\textbf{like}(x))(x))(\textbf{s}) && \textbf{s} \in \{x \in D_e \mid \langle x, x \rangle \in \textbf{like}'\} \\
&= (\textbf{like}(\textbf{s}))(\textbf{s}) && \langle \textbf{s}, \textbf{s} \rangle \in \textbf{like}'
\end{aligned}
$$

Back to logical inferences in language

Socrates likes himself $\Rightarrow$ *Socrates likes Socrates*

- Both expressions of type t – by derivation
- We only consider models where
  $I(\textit{himself}) = \textsc{himself} = \lambda R_{e(et)}.\lambda x_e.(R(x))(x)$
- In these models:
  $[\![\textit{Socrates likes himself}]\!]^M \leq [\![\textit{Socrates likes Socrates}]\!]^M$

Combinators

Defined in "pure" λ-calculus.

No reliance on properties of basic domains like D_t or D_e

Map any function $f : (A \times A) \rightarrow B$ to $g : A \rightarrow B$ s.t. $g(x) = f(x, x)$.

Equality and higher-order logic

For any two λ-terms φ and ψ:

- $\varphi = \psi$ is a (impure) λ-term of type t
- $[\![\varphi = \psi]\!]^M = 1$ iff $[\![\varphi]\!]^M = [\![\psi]\!]^M$

Equality turns λ-calculus into an expressive higher-order logic

Logical sugaring

$\top_t$	$\lambda x_t.x = \lambda x_t.x$	
$\bot_t$	$\lambda x_t.\top = \lambda x_t.x$	
$\neg\varphi_t$	$\varphi = \bot$	
$\neg A_{et}$	$\lambda x_e.\neg A(x)$	(polymorphism)
$\varphi_t \wedge \psi_t$	$\lambda f_{t(tt)}.f(\varphi)(\psi) = \lambda f_{t(tt)}.f(\top)(\top)$	
	relies on $\lvert D_t\rvert = 2$	($\vee$, $\rightarrow$)
$A_{et} \wedge B_{et}$	$\lambda x_e.A(x) \wedge B(x)$	(polymorphism)
$\forall x_\tau.\varphi_t$	$\lambda x_\tau.\varphi_t = \lambda x_\tau.\top$	
	substituting any value for x in φ gives 1	(higher-order, $\exists$)
$\textsc{inj}(F_{\tau\sigma}, A_{\tau t}, B_{\sigma t})$	F is an injection from A to $B\ldots$	
$\lvert A_{\tau t}\rvert \leq \lvert B_{\sigma t}\rvert$	$\exists F_{\tau\sigma}.\textsc{inj}(F, A, B)$	

Example: at least half

At least half of the students are snoring
= there are at least as many snoring students as non-snoring students

$$\text{AT\_LEAST\_HALF} = \lambda A_{et}.\lambda B_{et}.|A \wedge \neg B| \leq |A \wedge B|$$

$$(\text{AT\_LEAST\_HALF}(\textbf{student}))(\textbf{snore})$$
$$= |\textbf{student} \wedge \neg \textbf{snore}| \leq |\textbf{student} \wedge \textbf{snore}|$$

Inferences in language – made easier

Classical syllogisms:
Every human is mortal, and Socrates is human ⇒ Socrates is mortal

Other inferences with quantifiers:
More teachers than students snore, and at most five teachers snore ⇒ At most four students snore

Plurals and reciprocity:
Dan and Sue killed each other ⇒ Dan killed Sue and Sue killed Dan

Spatial and temporal inferences:
The bird is above the house and between the house and the cloud ⇒ The bird is below the cloud

Events:
Carthage was destroyed in 146 BC ⇒ Carthage's destruction occurred in 146 BC

Presupposition:
If John stops smoking he'll be able to run the marathon ⇒ John smokes

Further reading and references

Winter, Yoad. *Elements of Formal Semantics: An Introduction to the Mathematical Theory of Meaning in Natural Language*. Edinburgh University Press, 2016.

Semantics, BA *Kunstmatige Intelligentie*, UU.

Relative clauses

(1) Some man that saw Dan ran.

(2) Some man that Dan saw ran.

First-order translations:

(1') $\exists x.\mathbf{man}(x) \wedge \mathbf{see}(x, \mathbf{d}) \wedge \mathbf{run}(x)$

(2') $\exists x.\mathbf{man}(x) \wedge \mathbf{see}(\mathbf{d}, x) \wedge \mathbf{run}(x)$

How do we reach such translations?

Higher-order treatment

(1) Some man that saw Dan ran.

(2) Some man that Dan saw ran.

$\textsc{that} = \lambda A_{et}.\lambda B_{et}.\lambda x_e.B(x) \wedge A(x)$

map char. functions of two sets to char. function of intersection

man that ran: $(\textsc{that}(\textbf{run}))(\textbf{man}) = \lambda x.\textbf{man}(x) \wedge \textbf{run}(x)$

man that saw Dan: $(\textsc{that}(\textbf{see}(\textbf{d})))(\textbf{man}) = \lambda x.\textbf{man}(x) \wedge (\textbf{see}(\textbf{d}))(x)$

$\textsc{some} = \lambda A_{et}.\lambda B_{et}.\exists x_e.A(x) \wedge B(x)$

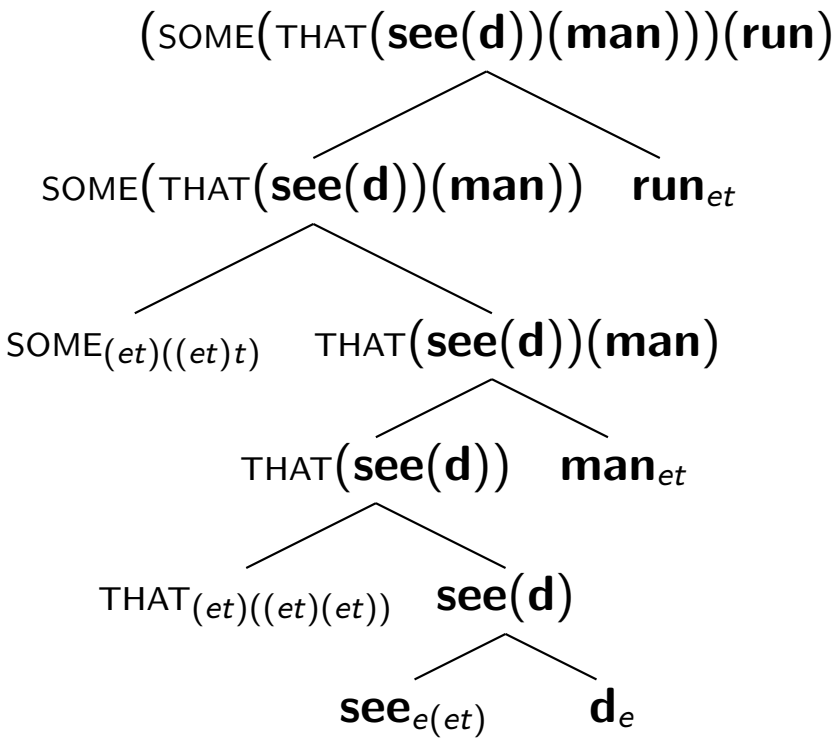
map char. functions of two sets to 1 iff intersection is not empty

some man ran: $(\textsc{some}(\textbf{man}))(\textbf{run}) = \exists x.\textbf{man}(x) \wedge \textbf{run}(x)$

some man that saw Dan ran: $(\textsc{some}(\textsc{that}(\textbf{see}(\textbf{d})))(\textbf{man}))(\textbf{run}) = \exists x.\textbf{man}(x) \wedge (\textbf{see}(\textbf{d}))(x) \wedge \textbf{run}(x)$

Higher-order treatment (cont.)

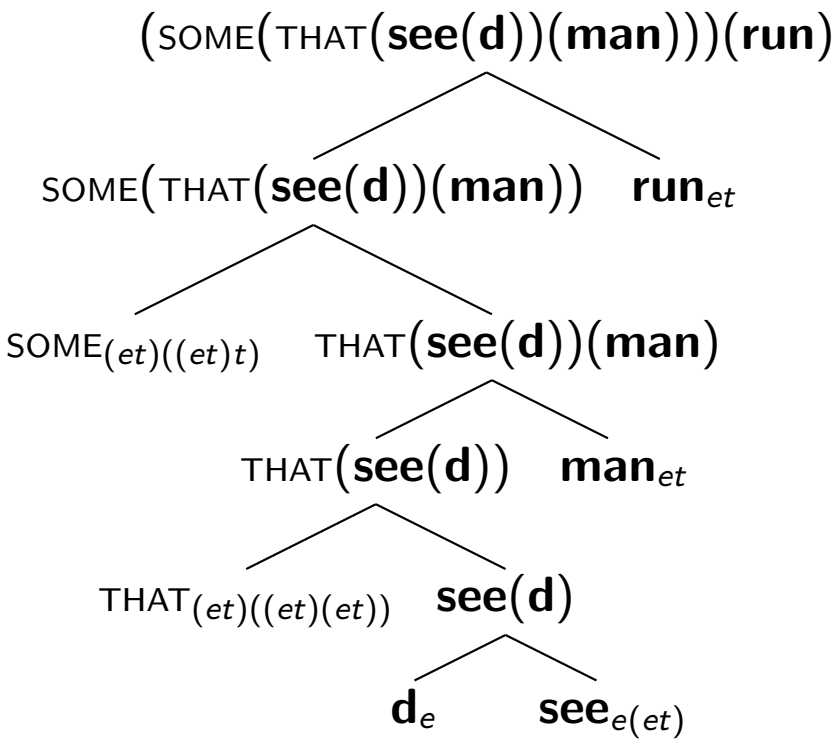
(1) Some man that saw Dan ran



$\exists x.\mathbf{man}(x) \wedge \mathbf{see}(\mathbf{d})(x) \wedge \mathbf{run}(x)$ ✓

Higher-order treatment (cont.)

(2) Some man that Dan saw ran



$$\exists x.\mathbf{man}(x) \wedge \mathbf{see}(\mathbf{d})(x) \wedge \mathbf{run}(x) \quad \text{✗}$$

Two sides of a problem

Expressions "that saw Dan" and "that Dan saw" are treated as equivalent.

- Correct meaning missing = undergeneration
  Solution: add dual principle to function application
  - **d** can appear in either argument of **see** of type $e(et)$
  - either **see**(**d**) or $\lambda x.$**see**(x)(**d**)
- Incorrect meaning present = overgeneration
  Solution: involve word order in deriving expressions and their denotations
  - *saw Dan*: **see**(**d**)
  - *Dan saw*: $\lambda x.$**see**(x)(**d**)

What is "a dual principle to FA"?

Prawitz, Dag. *Natural Deduction: A Proof-Theoretical Study*, Almqvist & Wiksell, Stockholm, 1965

Function Application and Modus Ponens

Function Application (FA)

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

Interpretation

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

Implication Elimination (Modus Ponens)

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

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

Dual operations to FA and MP

What logical principle allows us to *deduce* a function or a conditional statement?

Hypothetical Reasoning with conditional statements

(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)

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

Proving (A) using (B)

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

Implication Introduction

$$\dfrac{\dfrac{\cdots \quad [\varphi]^1}{\vdots}}{\dfrac{\psi}{\varphi \to \psi}}\ \text{discharge hypothesis 1}$$

Example

Given: $\varphi_1 \to (\varphi_2 \to \psi)$ and φ_2

Prove: $\varphi_1 \to \psi$

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

Function Introduction (abstraction)

$$\dfrac{\begin{array}{c}\ldots \quad [\tau]^1 \\ \vdots \\ \sigma\end{array}}{\tau\sigma}\ \text{discharge hypothesis 1}$$

Example

Given: $e(et)$ and e

Prove: et

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

Lambek, Joachim. The mathematics of sentence structure. *American Mathematical Monthly* (65), 154-169, 1958.

van Benthem, Johan. *Essays in Logical Semantics*, D. Reidel, Dordrecht, 1986.

Function Introduction – Interpretation

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

Example

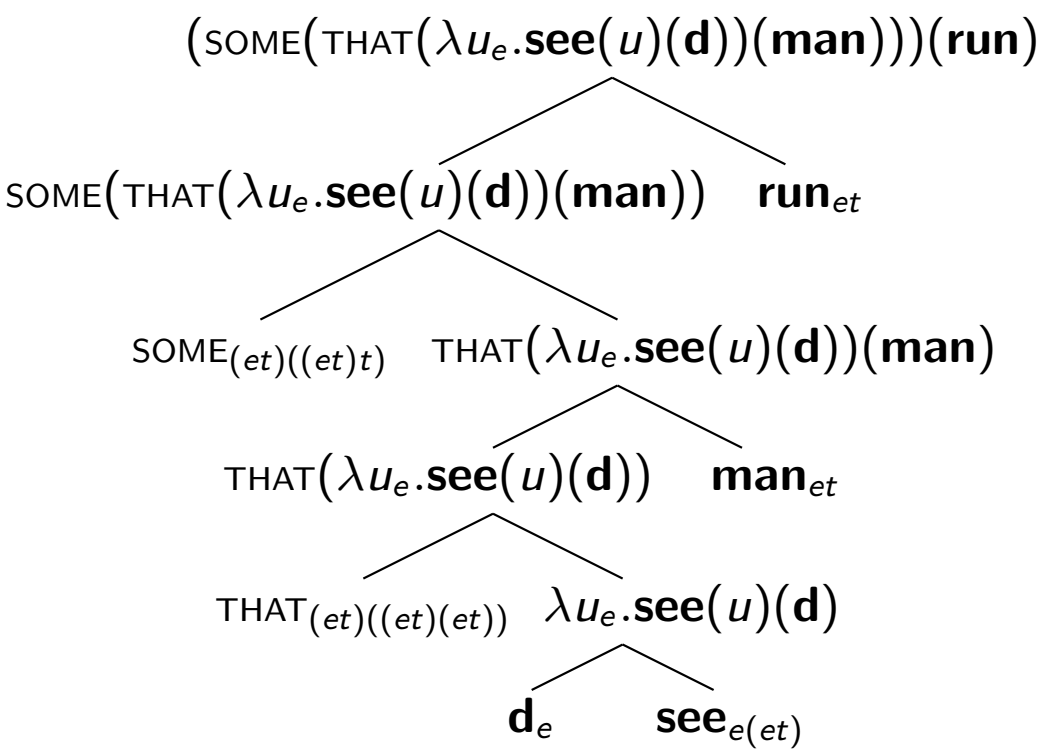
Given: $\mathbf{see}_{e(et)}$ and $\mathbf{dan}_e$

Prove: $\mathbf{dan}_e$ used as second argument of $\mathbf{see}_{e(et)}$

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

Relatives and Hypothetical Reasoning

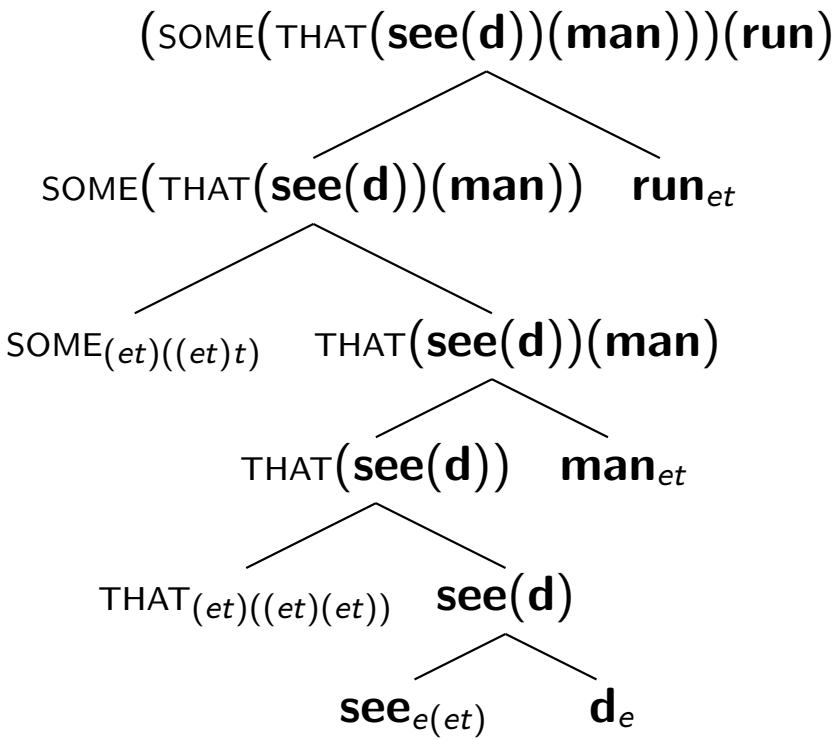
(2) Some man that Dan saw ran



$$\exists x.\mathbf{man}(x) \wedge \mathbf{see}(x)(\mathbf{d}) \wedge \mathbf{run}(x) \quad \checkmark$$

Relatives and Hypothetical Reasoning (cont.)

(1) Some man that saw Dan ran



$\exists x.\mathbf{man}(x) \wedge \mathbf{see}(\mathbf{d})(x) \wedge \mathbf{run}(x)$ ✓

Using signs

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

de Saussure, Ferdinand. *Cours de Linguistique Générale*, Payot & Cie, Paris, 1916.

Curry, Haskell B. Some logical aspects of grammatical structure. *Structure of Language and its Mathematical Aspects*, ed. by R. O. Jakobson. American Mathematical Society, Providence. 1961.

de Groote, Philippe. Towards Abstract Categorial Grammars. In *Proceedings of the 39th annual meeting of the Association for Computational Linguistics*. 2001.

Muskens, Reinhard. Language, lambdas and logic. *Resource Sensitivity in Binding and Anaphora*, ed. by G.-J. Kruijff and R. Oehrle. Kluwer, Dordrecht. 2003.

Sign composition

$\textsc{dan}$ (sign) + $\textsc{see}$ (sign) =

$$\left\{ \begin{array}{ll} \mathit{Dan}_f & \text{(form)} \\ \mathbf{d}_e & \text{(meaning)} \end{array} \right\} + \left\{ \begin{array}{ll} \lambda x_f.\lambda y_f.y \cdot \mathit{see} \cdot x & \text{(form)} \\ \mathbf{see}_{e(et)} & \text{(meaning)} \end{array} \right\} =$$

Solution 1:

$\textsc{see}(\textsc{dan})$

$$= \left\{ \begin{array}{lll} (\lambda x.\lambda y.y \cdot \mathit{see} \cdot x)(\mathit{Dan}) & = & \lambda y.y \cdot \mathit{see} \cdot \mathit{Dan} \quad \text{(form)} \\ \mathbf{see}(\mathbf{dan}) & & \text{(meaning)} \end{array} \right\}$$

Solution 2:

$\lambda U.\textsc{see}(U)(\textsc{dan})$

$$= \left\{ \begin{array}{lll} \lambda u_f.(\lambda x.\lambda y.y \cdot \mathit{see} \cdot x)(u)(\mathit{Dan}) & = & \lambda u.\mathit{Dan} \cdot \mathit{see} \cdot u \quad \text{(form)} \\ \lambda u_e.\mathbf{see}(u)(\mathbf{dan}) & & \text{(meaning)} \end{array} \right\}$$

Summary

Three virtues of higher-order logic for natural language semantics:

- Expressivity
- Types and function application
- Extendable to complex structures