A theory of logical form, meaning and synonymy

(The logical form and meaning of attitudinal sentences)

Yiannis N. Moschovakis
University of California, Los Angeles
and University of Athens

Göteborg, October 24, 2014

$\mathsf{L} = \mathsf{L}(K)$: The typed λ-calculus with acyclic recursion and propositional attitudes on the lexicon K

(the typed λ-calculus with two more constructs, `where` and `that`)

- L is an interpreted formal or programming language of terms
- A fragment of natural language is rendered in L:

  natural language expression + context

  $\xrightarrow{\text{render}}$ formal expression of L [+state]

- K is partitioned into a denotational and an attitudinal part,

$$K = K_d \cup K_a$$

(1) Syntax of L (7 slides)
(2) Denotational semantics of $\mathsf{L}(K_d)$ (3 slides)
(3) Intensional semantics of $\mathsf{L}(K_d)$ (12 slides)
(4) The interpretation of L into $\mathsf{L}(K_d^*)$ for some $K^* \supseteq K$ (9 slides)

The types and typed universes of $\mathsf{L} = \mathsf{L}(K)$

- Basic types: *Entities* : e *Truth values* : t *States* : s
- Types: $\boxed{\sigma :\equiv \mathsf{e} \mid \mathsf{t} \mid \mathsf{s} \mid (\sigma_1 \to \sigma_2)}$

  $\sigma_1 \times \sigma_2 \to \tau :\equiv (\sigma_1 \to (\sigma_2 \to \tau))$,
  $\sigma_1 \times \sigma_2 \times \sigma_3 \to \tau :\equiv \sigma_1 \to (\sigma_2 \times \sigma_3 \to \tau)$, etc.
- Uniquely, every non-basic $\sigma \equiv \sigma_1 \times \cdots \times \sigma_n \to b$ with basic b
- Interpretation (standard):

  $\mathbb{T}(\mathsf{e}) =$ a given set (or class) of people, objects, etc.

  $\mathbb{T}(\mathsf{s}) =$ a given set of states

  $\{\mathsf{tt}, \mathsf{ff}, \mathit{er}\} \subseteq \mathbb{T}(\mathsf{t}) =$ a given set of truth values $\subseteq \mathbb{T}(\mathsf{e})$ (er : error)

  $\mathbb{T}(\sigma \to \tau) = (\mathbb{T}(\sigma) \to \mathbb{T}(\tau)) =$ the set of all functions $f : \mathbb{T}(\sigma) \to \mathbb{T}(\tau)$
- State $a = (\mathsf{world}(a), \mathsf{time}(a), \mathsf{location}(a), \mathsf{agent}(a), \delta)$
  where $\delta(\mathsf{He}_1) = \ldots$, $\delta(\mathsf{this}) = \ldots$, etc.
- $\boxed{x : \sigma \iff x \in \mathbb{T}(\sigma) \quad (x \text{ is an object of type } \sigma)}$

Pure and natural language types and objects

- Pure types: $\boxed{\sigma :\equiv \mathsf{e} \mid \mathsf{t} \mid (\sigma_1 \to \sigma_2)}$ (for abstract objects)
- Basic natural language types:

$$\tilde{\mathsf{t}} :\equiv (\mathsf{s} \to \mathsf{t}) \quad \text{(Carnap intensions, propositions)}$$
$$\tilde{\mathsf{e}} :\equiv (\mathsf{s} \to \mathsf{e}) \quad \text{(Carnap individual concepts)}$$

- Natural language types: $\boxed{\sigma :\equiv \tilde{\mathsf{e}} \mid \tilde{\mathsf{t}} \mid (\sigma_1 \to \sigma_2)}$

$$\mathsf{boy} : (\tilde{\mathsf{e}} \to \tilde{\mathsf{t}}), \quad \mathsf{loves} : \tilde{\mathsf{e}} \times \tilde{\mathsf{e}} \to \tilde{\mathsf{t}} \equiv (\tilde{\mathsf{e}} \to (\tilde{\mathsf{e}} \to \tilde{\mathsf{t}}))$$

- Natural quantifier type: $\mathsf{some(girl)}, \mathsf{every(boy)} : \tilde{\mathsf{q}} :\equiv ((\tilde{\mathsf{e}} \to \tilde{\mathsf{t}}) \to \tilde{\mathsf{t}})$
- *If* $x : \sigma$ *with a natural* σ*, then* $x : \tilde{\mathsf{t}}$*, or* $x : \tilde{\mathsf{e}}$*, or* x *is a function*

$$x : \mathbb{T}(\sigma_1) \times \cdots \mathbb{T}(\sigma_n) \to \mathbb{T}(\tilde{b})$$

with natural $\sigma_1, \ldots, \sigma_n$ and $b \equiv \mathsf{e}$ or $b = \mathsf{t}$

★ *The terms which render natural language phrases are (hereditarily) of natural language type*

Constants; a sample lexicon

Denotational empirical constants:

Entities	$0, 1, 2, \ldots, \mathbb{R}, \ldots, \mathtt{tt}, \mathtt{ff}, \mathit{er}$	$\mathtt{e}$
Names, indexicals	John, I, he, him	$\tilde{\mathtt{e}}$
Common nouns	man, unicorn, temperature	$\tilde{\mathtt{e}} \longrightarrow \tilde{\mathtt{t}}$
Adjectives, adverbs	tall, young, rapidly	$(\tilde{\mathtt{e}} \longrightarrow \tilde{\mathtt{t}}) \longrightarrow (\tilde{\mathtt{e}} \longrightarrow \tilde{\mathtt{t}})$
Propositions	it rains	$\tilde{\mathtt{t}}$
Intransitive verbs	stand, run, rise	$\tilde{\mathtt{e}} \longrightarrow \tilde{\mathtt{t}}$
Transitive verbs	find, love, be	$(\tilde{\mathtt{e}} \times \tilde{\mathtt{e}}) \longrightarrow \tilde{\mathtt{t}}$
Description operator	the	$(\tilde{\mathtt{e}} \longrightarrow \tilde{\mathtt{t}}) \longrightarrow \tilde{\mathtt{e}}$

Attitudinal constants: Known, Claims, Believes_of : $(\tilde{\mathtt{e}} \times \cdots \times \tilde{\mathtt{e}} \times \tilde{\mathtt{t}}) \longrightarrow \tilde{\mathtt{t}}$

Pure type logical constants:

$$=_\sigma \ : \ \sigma \times \sigma \longrightarrow \mathtt{t}$$
$$\neg \ : \ \mathtt{t} \longrightarrow \mathtt{t}$$
$$\&, \vee, \Rightarrow \ : \ \mathtt{t} \times \mathtt{t} \longrightarrow \mathtt{t}$$
$$\forall_\sigma, \exists_\sigma \ : \ (\sigma \longrightarrow \mathtt{t}) \longrightarrow \mathtt{t}$$

Natural type logical constants

$$\text{not}, \Box, \text{in the future} \ : \ \tilde{\mathtt{t}} \longrightarrow \tilde{\mathtt{t}}$$
$$\text{and, or, if .. then ..} \ : \ \tilde{\mathtt{t}} \times \tilde{\mathtt{t}} \longrightarrow \tilde{\mathtt{t}}$$
$$\text{every, some} \ : \ (\tilde{\mathtt{e}} \longrightarrow \tilde{\mathtt{t}}) \longrightarrow \tilde{\mathtt{q}}$$
$$\text{the} \ : \ (\tilde{\mathtt{e}} \longrightarrow \tilde{\mathtt{t}}) \longrightarrow \tilde{\mathtt{e}}$$

Typed variables

★ *Two kinds of typed variables*

- Pure variables of type σ: $\mathtt{v}_0^\sigma, \mathtt{v}_1^\sigma \ldots$
- Recursion variables or locations of type σ: $\mathtt{V}_0^\sigma, \mathtt{V}_1^\sigma \ldots$

- Both $\mathtt{v}_i^\sigma$ and $\mathtt{V}_i^\sigma$ range over arbitrary objects $x : \sigma$
  ... but they are treated differently in the syntax

★ Pure variables are bound by λ (as in the typed λ-calculus)

★ Locations are used for equations or (formal) assignments

$$\mathtt{P} := A$$

and are bound by the recursion construct where

★ *There are no variables over entities in the natural language fragment, variables over individual concepts are used instead*

Terms (defined recursively, with natural restrictions)

$$A :\equiv \mathtt{x} \mid c \mid A_1(A_2) \mid \lambda(\mathtt{u})(A_1) \qquad \text{(explicit, } \lambda\text{-calculus)}$$
$$\mid A_0 \text{ where } \{\mathtt{P}_1 := A_1, \ldots, \mathtt{P}_n := A_n\} \qquad \text{(acyclic recursion)}$$
$$\mid C(B_1, \ldots, B_m, \text{that } A) \qquad \text{(attitudinal application)}$$

Restrictions, typing and binding (except for recursive terms)

- $\mathtt{x}$ is a variable of either kind and $\mathtt{u}$ is a pure variable
- c is a typed denotational constant
- Four formations rules: application, λ-abstraction
  acyclic recursion (where) and attitudinal application (that)
- C is a typed attitudinal constant
- The obvious type restrictions are observed, and each term is assigned a type, $A : \text{type}(A)$
  - If $B_1, \ldots, B_m : \tilde{\mathrm{e}}$ and $A : \tilde{\mathrm{t}}$, then $C(B_1, \ldots, B_m, \text{that } A) : \tilde{\mathrm{t}}$
- Free and bound occurrences of variables are specified in the obvious way, with C not binding any variables

- *A is explicit* (a λ-calculus term) *if there is no* "where" *or* "that" *in it*

Acyclic recursion

$$\boxed{A \equiv A_0 \text{ where } \{\mathrm{P}_1 := A_1, \ldots, \mathrm{P}_n := A_n\}}$$

- We allow $n = 0$, so that if A is a term, then so is A where $\{\ \}$
- The locations $\mathrm{P}_1, \ldots, \mathrm{P}_n$ are distinct
  and the system of equations (term assignments)

  $$\{\mathrm{P}_1 := A_1, \ldots, \mathrm{P}_n := A_n\}$$

  is correctly typed and syntactically acyclic, i.e.,

  $$\text{type}(\mathrm{P}_i) = \text{type}(A_i) \text{ for } i = 1, \ldots, n,$$

  and there are numbers $\text{rank}(\mathrm{P}_1), \ldots, \text{rank}(\mathrm{P}_n)$, such that

  $$\text{if } \text{rank}(\mathrm{P}_j) \leq \text{rank}(\mathrm{P}_i), \text{ then } \mathrm{P}_i \text{ does not occur free in } A_j$$
- $A : \text{type}(A_0)$
- All occurrences of $\mathrm{P}_1, \ldots, \mathrm{P}_n$ are bound in A
- If a variable x is not in the list $\mathrm{P}_1, \ldots, \mathrm{P}_n$, then an occurrence of in any A_i is free in A exactly if it is free in A_i

John loves Mary and dislikes her husband

$$A \equiv \mathtt{P} \text{ and } \mathtt{Q} \text{ where } \{\mathtt{J} := \text{John}, \mathtt{M} := \text{Mary}, \mathtt{H} := \text{husband}(\mathtt{M}), \mathtt{P} := \text{loves}(\mathtt{J}, \mathtt{M}), \mathtt{Q} := \text{dislikes}(\mathtt{J}, \mathtt{H})\} : \tilde{\mathrm{t}}$$

- A is closed, $\text{den}(A) : \mathbb{T}(s) \to \mathbb{T}(\mathrm{t})$, and we compute it as follows:

  Stage 1: $\overline{\mathtt{J}} := \text{John} : \tilde{\mathrm{e}}$, $\overline{\mathtt{M}} := \text{Mary} : \tilde{\mathrm{e}}$

  Stage 2: $\overline{\mathtt{H}} := \text{husband}(\overline{\mathtt{M}}) := \text{Mary's husband} : \tilde{\mathrm{e}}$

  $\overline{\mathtt{P}} := \text{loves}(\overline{\mathtt{J}}, \overline{\mathtt{M}}) : \tilde{\mathrm{t}}$

  Stage 3: $\overline{\mathtt{Q}} := \text{dislikes}(\overline{\mathtt{J}}, \overline{\mathtt{H}}) : \tilde{\mathrm{t}}$

  Stage 4: $\text{den}(A) := \overline{\mathtt{P}} \text{ and } \overline{\mathtt{Q}} : \tilde{\mathrm{t}}$

For any state u,

$\text{den}(A)(u) = (\overline{\mathtt{P}} \text{ and } \overline{\mathtt{Q}})(u) = \overline{\mathtt{P}}(u) \text{ and } \overline{\mathtt{Q}}(u)$

= the truth value of John loves Mary and dislikes her husband in state u

(= *er* if Mary does not have exactly one husband in state u)

Congruence and extended terms

- Two terms are congruent if one can be obtained from the other by alphabetic changes of bound variables, re-ordering of the equations in recursive subterms and deleting empty systems. We write

$$A \equiv_c B \iff A \text{ is congruent with } B,$$

so in particular, $A \equiv_c A$ where $\{\ \}$

- An extended term is a pair

$$A[\mathtt{x}] \equiv (A, \mathtt{x})$$

of a term A and a sequence $\mathtt{x} \equiv \mathtt{x}_1, \ldots, \mathtt{x}_k$ of distinct variables (of both kinds) which includes all the free variables of A

- $\mathtt{x}$ is the list of putative free variables of $A[\mathtt{x}]$
- Formal substitution: $A[B_1, \ldots, B_k] :\equiv A\{\mathtt{x}_i :\equiv B_i, i = 1, \ldots k\}$
- Typing and congruence for extended terms: $A[\mathtt{x}] : \mathrm{type}(A)$ and

$$A[\mathtt{x}] \equiv_c B[\mathtt{y}] \iff A \equiv_c B \,\&\, \mathtt{x} \equiv \mathtt{y}$$

Denotational semantics for $\mathsf{L}(K_d)$

- We assume given with each $c : \sigma$ an object $\overline{c} : \sigma$
- If $A[\mathtt{x}]$ is an extended term with $\mathtt{x} \equiv \mathtt{x}_1, \ldots, \mathtt{x}_k$ and $x = (x_1, \ldots, x_k)$ with $x_i : \mathrm{type}(\mathtt{x}_i)$ for $i = 1, \ldots, k$, we define

$$A[x] = \text{the value of } A \text{ when } \mathtt{x} = x$$

by the natural (structural) recursion on A:

- $\mathtt{x}_i[x] = x_i, \quad c[x] = \overline{c},$
- $A(B)[x] = A[x](B[x]),$
- $\lambda(\mathtt{x}_1)(A)[x_2, \ldots, x_k] = h : \mathbb{T}(\mathrm{type}(\mathtt{x}_1)) \to \mathbb{T}(\mathrm{type}(A))$, where for each $t : \mathrm{type}(\mathtt{x}_1)$, $h(t) = A[t, x_2, \ldots, x_k]$,
- A_0 where $\{\mathtt{P}_1 := A_1, \ldots, \mathtt{P}_n := A_n\}[x] = A_0[\overline{\mathtt{P}}_1(x), \ldots, \overline{\mathtt{P}}_n(x), x]$, where $\overline{\mathtt{P}}_1(x), \ldots, \overline{\mathtt{P}}_n(x)$ are the unique solutions of the system

$$\mathtt{P}_i := A_i[\mathtt{P}_1, \ldots, \mathtt{P}_n, x] \quad (i = 1, \ldots, n)$$

guaranteed by the acyclicity of the term

The denotation of an extended term

- If $A[\mathtt{x}]$ is an extended term with $\mathtt{x} \equiv \mathtt{x}_1, \ldots, \mathtt{x}_k$, let

$$X = \mathbb{T}(\mathrm{type}(\mathtt{x}_1) \times \cdots \times \mathbb{T}(\mathrm{type}(\mathtt{x}_n))$$

be the space associated with the sequence of putative free variables $\mathtt{x}$ and define

$$\mathrm{den}(A[\mathtt{x}]) : X \to \mathbb{T}(\mathrm{type}(A)) \quad \text{by} \quad \mathrm{den}(A[\mathtt{x}])(x) = A[x]$$

- Notice that if A is in the natural fragment of L, then $\mathrm{den}(A[\mathtt{x}])$ is always a function; for example, with empty $\mathtt{x}$

$$\mathrm{den}(\mathrm{likes}(\mathrm{the}(\mathrm{blond}), \mathrm{John})) : \mathbb{T}(\mathtt{s}) \to \mathbb{T}(\mathtt{t}),$$

so that for each state u, $\mathrm{den}(\mathrm{likes}(\mathrm{the}(\mathrm{blond}), \mathrm{John}))(u)$ is a truth value

- *Every* $A[\mathtt{x}]$ *is denotationally equal to some* λ*-calculus term* $A^*[\mathtt{x}]$

Some (oversimplified) remarks on Montague semantics

• In Montague's *Language of Intensional Logic* LIL, every sentence of the fragment of English that he studies is rendered by a closed term A of truth-value type t and interpreted by a Carnap intension

$$\mathsf{CI}_{\mathsf{Mon}}(A) : \mathbb{T}(\mathsf{s}) \to \{\mathsf{tt}, \mathsf{ff}\}$$

The slogan (not Montague's) is that all language is situated

• Montague calls $\mathsf{CI}_{\mathsf{Mon}}(A)$ the sense of A; which then implies that all true mathematical sentences are Montague synonymous

• Gallin interpreted LIL into the typed λ-calculus and hence into L; and we can use this embedding to derive non-trivial referential intensions for the sentences in the fragment of English that is rended in LIL—e.g., mathematical sentences

• However: These meanings derived by mapping English into L via LIL and Gallin are rarely in the spirit of the work of Montague, whose renderings disregarded meaning and were often eccentric

The route not taken

- The intensional semantics of $\mathrm{L}(K_d)$ associates with each extended term $A[\mathtt{x}]$ its referential intension, an acyclic recursor

$$\mathrm{int}(A[\mathtt{x}]) : X \rightsquigarrow \mathbb{T}(\mathrm{type}(A))$$

which computes the denotation of $A[\mathtt{x}]$; i.e.,

$$X = \mathbb{T}(\mathrm{type}(\mathtt{x}_1) \times \cdots \times \mathbb{T}(\mathrm{type}(\mathtt{x}_n))$$

is the space associated with the list of putative free variables $\mathtt{x}$, and

$$\overline{\mathrm{int}(A[\mathtt{x}])}(x) = \mathrm{den}(A[\mathtt{x}])(x) \quad (x \in X)$$

- $\mathrm{int}(A[\mathtt{x}])$ can be defined by a direct recursion on A just like $\mathrm{den}(A[\mathtt{x}])$; but we follow a circuitous route which is technically simpler and in the end more illuminating

★ Immediate and direct reference

★ *Variables have no meaning*, they denote immediately

and the same is true of immediate terms defined by

$$Im :\equiv \mathtt{u} \mid \mathtt{P} \mid \mathtt{P}(\mathtt{u}_1, \ldots, \mathtt{u}_n) \mid \lambda(\mathtt{u})Im$$

- *Immediate terms act like generalized variables*
- Terms which are not immediate are proper

★ *Denotational constants are proper and denote directly*
i.e., their meaning is exhausted by their reference

They are the simplest directly referring proper terms, which include

$$\lambda(\mathtt{u})\mathsf{love}(\mathtt{u},\mathtt{v}), \quad \lambda(\mathtt{u})\mathsf{love}(\mathtt{u},\mathtt{u}), \quad \text{etc}$$

- *We will characterize syntactically all directly denoting terms*

- The distinctions between immediate, direct and complex reference (via a non-trivial meaning) are basic to this theory —and they come up in various ways in every theory of meaning

⋆ The Reduction Calculus for $\mathsf{L}(K_d)$

We define a reduction relation between terms so that intuitively

$$A \Rightarrow B \iff A \equiv_c B \quad (A \text{ is congruent with } B)$$
or A and B have the same meaning
and B expresses that meaning "more directly"

- For $\mathsf{L}(K_d)$, $A \Rightarrow B$ is defined by ten rules, like a proof system
- Compositionality: $C_1 \Rightarrow C_2 \implies A\{\mathrm{u} :\equiv C_1\} \Rightarrow A\{\mathrm{u} :\equiv C_2\}$
- *Reduction respects free occurrences of variables*, and so it extends trivially to extended terms,

$$A[\mathrm{x}] \Rightarrow B[\mathrm{y}] \iff A \Rightarrow B \,\&\, \mathrm{x} \equiv \mathrm{y}$$

- $\Rightarrow$ respects den: $A[\mathrm{x}] \Rightarrow B[\mathrm{x}] \implies \mathrm{den}(A[\mathrm{x}]) = \mathrm{den}(B[\mathrm{x}])$,

⋆ but it only compiles terms, it does not compute any values

⋆ A term A is irreducible if $A \Rightarrow B \implies A \equiv_c B$

⋆ Canonical and logical forms

Canonical Form Theorem. *For each term A, there is a unique (up to congruence) recursive, irreducible term*

$$\boxed{\mathrm{cf}(A) \equiv A_0 \text{ where } \{\mathtt{P}_1 := A_1, \ldots, \mathtt{P}_n := A_n\}}$$

such that $A \Rightarrow \mathrm{cf}(A)$. Each A_i is explicit and irreducible

⋆ $\mathrm{cf}(A)$ models the logical form of A

⋆ The parts $A_0, A_1, \ldots, A_n$ of A act like truth conditions for A

• $\mathrm{cf}(A)$ *can be computed effectively by applying (successively, in any order) the reduction rules to subterms of A, using compositionality*

• Notation: $\boxed{A \Rightarrow_{\mathrm{cf}} B \iff B \equiv_c \mathrm{cf}(A)}$

• Jos Tellings is nearly done writing an implementation of the reduction calculus for L which computes the canonical forms of L-terms

★ Referential intensions

- Notation convention: to interpret $\lambda(\mathtt{V})$ with a location $\mathtt{V}$, we set

$$\lambda(\mathtt{V})A :\equiv \lambda(\mathtt{v})A\{\mathtt{V} :\equiv \mathtt{v}\}$$

where $\mathtt{v}$ is a fresh pure variable of type$(\mathtt{V})$

★ Let $\mathtt{x} \equiv \mathtt{x}_1, \ldots, \mathtt{x}_k$ and $\mathtt{P} \equiv \mathtt{P}_1, \ldots, \mathtt{P}_n$. If $A[\mathtt{x}]$ is proper and

$$A[\mathtt{x}] \Rightarrow_{\mathsf{cf}} A_0[\mathtt{x}, \mathtt{P}] \text{ where } \{\mathtt{P}_1 := A_1[\mathtt{x}, \mathtt{P}], \ldots, \mathtt{P}_n := A_n[\mathtt{x}, \mathtt{P}]\},$$

then the referential intension of $A[\mathtt{x}]$ is the acyclic recursor

$$\alpha(x) = A_0[x, \mathtt{P}] \ \underline{\textit{where}} \ \{\mathtt{P}_1 := A_1[x, \mathtt{P}], \ldots, \mathtt{P}_n := A_n[x, \mathtt{P}]\},$$

and it computes den$(A[\mathtt{x}])$. As a tuple of functions,

$$\boxed{\mathsf{int}(A[\mathtt{x}]) = \Big(\mathsf{den}(\lambda(\mathtt{x})\lambda(\mathtt{P})A_0), \ldots, \mathsf{den}(\lambda(\mathtt{x})\lambda(\mathtt{P})A_n)\Big)}$$

★ If $A[\mathtt{x}]$ is explicit and irreducible, then it denotes directly, because

$$A[\mathtt{x}] \Rightarrow_{\mathsf{cf}} A[\mathtt{x}] \text{ where } \{\ \}, \text{ and so } \mathsf{int}(A[\mathtt{x}]) = \Big(\mathsf{den}(A[\mathtt{x}])\Big)$$

John loves Mary and (he) dislikes her husband

Coindexing "he" with "John" and "her" with "Mary", we render this by

$$A \equiv \text{loves}(\mathtt{J}, \mathtt{M}) \text{ and } \text{dislikes}(\mathtt{J}, \text{husband}(\mathtt{M})) \quad \text{where } \{\mathtt{J} := \text{John}, \mathtt{M} := \text{Mary}\}$$

$$\Rightarrow_{\text{cf}} \mathtt{P} \text{ and } \mathtt{Q} \text{ where } \{\mathtt{J} := \text{John}, \mathtt{M} := \text{Mary}, \mathtt{H} := \text{husband}(\mathtt{M}), \mathtt{P} := \text{loves}(\mathtt{J}, \mathtt{M}), \mathtt{Q} := \text{dislikes}(\mathtt{J}, \mathtt{H})\}$$

$$\text{int}(A) = \mathtt{P} \text{ and } \mathtt{Q} \ \underline{\textit{where}} \ \{\mathtt{J} := \text{John}, \mathtt{M} := \text{Mary}, \mathtt{H} := \text{husband}(\mathtt{M}), \mathtt{P} := \text{loves}(\mathtt{J}, \mathtt{M}), \mathtt{Q} := \text{dislikes}(\mathtt{J}, \mathtt{H})\}$$

or, as a tuple of functions, with $\mathtt{u} \equiv (\mathtt{j}, \mathtt{m}, \mathtt{h}, \mathtt{p}, \mathtt{q})$,

$$\text{int}(A) = \Big(\text{den}(\lambda \mathtt{u}\,(\mathtt{p} \text{ and } \mathtt{q})), \text{den}(\lambda \mathtt{u}\,\text{John}), \text{den}(\lambda \mathtt{u}\,\text{Mary}), \text{den}(\lambda \mathtt{u}\,\text{husband}(\mathtt{m})), \text{den}(\lambda \mathtt{u}\,\text{loves}(\mathtt{j}, \mathtt{m})), \text{den}(\lambda \mathtt{u}\,\text{dislikes}(\mathtt{j}, \mathtt{h}))\Big)$$

⋆ Referential synonymy

• Two proper extended terms are referentially synonymous if their referential intensions are equal (naturally isomorphic) recursors,

$$A[\mathrm{x}] \approx B[\mathrm{x}] \iff \mathrm{int}(A[\mathrm{x}]) = \mathrm{int}(B[\mathrm{x}])$$

Referential Synonymy Theorem. *Two proper extended terms* $A[\mathrm{x}], B[\mathrm{x}]$ *are referentially synonymous if and only if*

$$A \Rightarrow_{\mathrm{cf}} A^* \equiv A_0 \text{ where } \{\mathrm{P}_1 := A_1, \ldots, \mathrm{P}_n := A_n\}$$
$$B \Rightarrow_{\mathrm{cf}} B^* \equiv B_0 \text{ where } \{\mathrm{P}_1 := B_1, \ldots, \mathrm{P}_n := B_n\}$$

for suitable A^*, B^* *so that for* $i = 0, \ldots, n$,

$$\mathrm{den}(\lambda(\mathrm{x})\lambda(\mathrm{P}_1, \ldots, \mathrm{P}_n)A_i) = \mathrm{den}(\lambda(\mathrm{x})\lambda(\mathrm{P}_1, \ldots, \mathrm{P}_n)B_i)$$

⋆ $A[\mathrm{x}] \approx B[\mathrm{x}]$ is determined by their logical forms and denotational equality on their explicit, irreducible parts — their truth conditions

• Conjecture: *If* K_d *is finite, then referential synonymy is decidable on terms of* $L(K_d)$

This is a problem in the typed λ-calculus

Reduction Calculus: the basic rules

Congruence, Transitivity, Compositionality

(cong) If $A \equiv_c B$, then $A \Rightarrow B$

(trans) If $A \Rightarrow B$ and $B \Rightarrow C$, then $A \Rightarrow C$

(rep1) If $A \Rightarrow A'$ and $B \Rightarrow B'$, then $A(B) \Rightarrow A'(B')$

(rep2) If $A \Rightarrow B$, then $\lambda(\mathrm{u})(A) \Rightarrow \lambda(\mathrm{u})(B)$

(rep3) If $A_i \Rightarrow B_i$ for $i = 0, \ldots, n$, then

$$A_0 \text{ where } \{\mathrm{P}_1 := A_1, \ldots, \mathrm{P}_n := A_n\} \Rightarrow B_0 \text{ where } \{\mathrm{P}_1 := B_1, \ldots, \mathrm{P}_n := B_n\}$$

- These do not produce any non-trivial reductions

Reduction Calculus: the rules for recursion

For distinct locations $\mathtt{P}_1, \ldots, \mathtt{P}_n$ and $\mathtt{Q}_1, \ldots, \mathtt{Q}_m$, and terms A_i, B_j, set

$$\vec{\mathtt{P}} := \vec{A} \quad \text{for} \quad \mathtt{P}_1 := A_1, \ldots, \mathtt{P}_n := A_n,$$
$$\vec{\mathtt{Q}} := \vec{B} \quad \text{for} \quad \mathtt{Q}_1 := B_1, \ldots, \mathtt{Q}_m := B_m,$$

(head) $\Big(A_0 \text{ where } \{\vec{\mathtt{P}} := \vec{A}\}\Big) \text{ where } \{\vec{\mathtt{Q}} := \vec{B}\}$

$$\Rightarrow A_0 \text{ where } \{\vec{\mathtt{P}} := \vec{A}, \vec{\mathtt{Q}} := \vec{B}\}$$

(B-S) $A_0 \text{ where } \{\mathtt{R} := \Big(B_0 \text{ where } \{\vec{\mathtt{Q}} := \vec{B}\}\Big), \vec{\mathtt{P}} := \vec{A}\}$

$$\Rightarrow A_0 \text{ where } \{\mathtt{R} := B_0, \vec{\mathtt{Q}} := \vec{B}, \vec{\mathtt{P}} := \vec{A}\}$$

- These just allow the "parallel" combination of assignments

(recap) $\boxed{\Big(A_0 \text{ where } \{\vec{\mathtt{P}} := \vec{A}\}\Big)(B) \Rightarrow A_0(B) \text{ where } \{\vec{\mathtt{P}} := \vec{A}\}}$

provided no $\mathtt{P}_i$ is free in B

- (recap) is one the two main rules that produce non-trivial reductions

Reduction Calculus: the proper application rule

(ap) $\boxed{A(B) \Rightarrow A(\mathtt{B}) \text{ where } \{\mathtt{B} := B\}}$ (B proper, $\mathtt{B}$ fresh)

- John is tall $\xrightarrow{\text{render}}$ tall(John) $\Rightarrow_{\text{cf}}$ tall(J) where $\{\mathtt{J} := \text{John}\}$

- The blond likes John $\xrightarrow{\text{render}}$ likes(the(blond))(John)

  $\Rightarrow$ likes(the(blond))(J) where $\{\mathtt{J} := \text{John}\}$ (ap)

  $\Rightarrow \Big(\text{likes}(\mathtt{B}) \text{ where } \{\mathtt{B} := \text{the}(\text{blond})\}\Big)(\mathtt{J}) \text{ where } \{\mathtt{J} := \text{John}\}$ (ap)

  $\Rightarrow$ likes(B)(J) where $\{\mathtt{B} := \text{the}(\text{blond}), \mathtt{J} := \text{John}\}$ (recap)

  $\Rightarrow$ likes(B)(J) where $\{\mathtt{B} := \text{the}(\mathtt{B}_1) \text{ where } \{\mathtt{B}_1 := \text{blond}\}, \mathtt{J} := \text{John}\}$

  $\Rightarrow$ likes(B)(J) where $\{\mathtt{B} := \text{the}(\mathtt{B}_1), \mathtt{B}_1 := \text{blond}, \mathtt{J} := \text{John}\}$

- The real work is done by the application rules (ap) and (recap)

The Reduction Calculus (minus the λ-rule) – summary

(cong) If $A \equiv_c B$, then $A \Rightarrow B$

(trans) If $A \Rightarrow B$ and $B \Rightarrow C$, then $A \Rightarrow C$

(rep1) If $A \Rightarrow A'$ and $B \Rightarrow B'$, then $A(B) \Rightarrow A'(B')$

(rep2) If $A \Rightarrow B$, then $\lambda(\mathtt{u})(A) \Rightarrow \lambda(\mathtt{u})(B)$

(rep3) If $A_i \Rightarrow B_i$ for $i = 0, \ldots, n$, then

$$A_0 \text{ where } \{\vec{\mathtt{P}} := \vec{A}\} \Rightarrow B_0 \text{ where } \{\vec{\mathtt{P}} := \vec{B}\}$$

(head) $\Big(A_0 \text{ where } \{\vec{\mathtt{P}} := \vec{A}\}\Big) \text{ where } \{\vec{\mathtt{Q}} := \vec{B}\} \Rightarrow A_0 \text{ where } \{\vec{\mathtt{P}} := \vec{A}, \vec{\mathtt{Q}} := \vec{B}\}$

(B-S) $A_0 \text{ where } \{\mathtt{R} := \Big(B_0 \text{ where } \{\vec{\mathtt{Q}} := \vec{B}\}\Big), \vec{\mathtt{P}} := \vec{A}\}$

$$\Rightarrow A_0 \text{ where } \{\mathtt{R} := B_0, \vec{\mathtt{Q}} := \vec{B}, \vec{\mathtt{P}} := \vec{A}\}$$

(recap) $\Big(A_0 \text{ where } \{\vec{\mathtt{P}} := \vec{A}\}\Big)(B) \Rightarrow A_0(B) \text{ where } \{\vec{\mathtt{P}} := \vec{A}\}$ (no $\mathtt{P}_i$ free in B)

(ap) $A(B) \Rightarrow A(\mathtt{B}) \text{ where } \{\mathtt{B} := B\}$ (B proper, $\mathtt{B}$ fresh)

(λ-rule) $\lambda(\mathtt{u})(A_0 \text{ where } \{\vec{\mathtt{P}} := \vec{A}\}) \Rightarrow \lambda(\mathtt{u})A'_0 \text{ where } \{\overrightarrow{\mathtt{P}'} := \overrightarrow{\lambda(\mathtt{u})A'}\}$

⋆ Utterances, local meanings and local synonymy

- For each state u, we add to $\mathrm{L}(K_d)$ a parameter $\bar{u}$ which names u

  State parameters are treated in the syntax like free pure variables, e.g., $\bar{u}$, $\mathrm{P}(\bar{u})$ are immediate terms;

  . . . and they can be avoided, at some cost in notational complexity

- An extended utterance is a pair $(A[\mathrm{x}], u)$, where $A[\mathrm{x}] : \tilde{\mathrm{t}}$ is an extended Carnap intension and u is a state; it is expressed in L by the extended term $A(\bar{u})[\mathrm{x}] : \mathrm{t}$

- The local meaning of $A[\mathrm{x}]$ at a state u is $\mathrm{int}(A(\bar{u})[\mathrm{x}])$, and

$$A[\mathrm{x}] \approx_u B[\mathrm{x}] \iff \mathrm{int}(A(\bar{u})[\mathrm{x}]) = \mathrm{int}(B(\bar{u})[\mathrm{x}]) \iff A(\bar{u})[\mathrm{x}] \approx B(\bar{u})[\mathrm{x}]$$

- Local synonymy $\approx_u$ is a a very strong equivalence relation, very close to global synonymy

- Basic principle: $\boxed{\textit{The objects of belief are local meanings}}$

For closed A, *John believes in state u that A*

means that in state u, *John believes the utterance $A(\bar{u})$*

Is he Scott? (Scott-Soames, after Russell, Quine, Church, ...)

In the state u of a book presentation, Sir Walter Scott is disguised and

George IV does not believe that He is Scott,

but certainly

George IV believes that Scott is Scott

which appears to be a paradox. However, He is Scott $\not\approx_u$ Scott is Scott:

$$(\text{He is Scott}, u) \xrightarrow{\text{render}} \text{eq}(\text{He}, \text{Scott})(\bar{u})$$

$$\Rightarrow_{\text{cf}} \boxed{\text{eq}(\mathtt{H}, \mathtt{S})(\bar{u}) \text{ where } \{\mathtt{H} := \text{He}, \mathtt{S} := \text{Scott}\}}$$

$$(\text{Scott is Scott}, u) \xrightarrow{\text{render}} \text{eq}(\text{Scott}, \text{Scott})(\bar{u})$$

$$\Rightarrow_{\text{cf}} \boxed{\text{eq}(\mathtt{S}_1, \mathtt{S}_2)(\bar{u}) \text{ where } \{\mathtt{S}_1 := \text{Scott}, \mathtt{S}_2 := \text{Scott}\}}$$

- These are obviously not synonymous, and so the good king can believe one and not the other; he is muddled but not incoherent
- The paradox is more complex if the king has a *de re* belief about *him*

Believing a closed utterance

George believes that the blond likes John $\xrightarrow{\text{render}}$ Believes(George, that A)

where the belief of George is the closed term

$$A :\equiv \text{likes}(\text{the}(\text{blond}), \text{John}))$$

$$\Rightarrow_{\text{cf}} \text{likes}(\text{B}, \text{J}) \text{ where } \{\text{B} := \text{the}(\text{B}_1), \text{B}_1 := \text{blond}, \text{J} := \text{John}\}\Big)$$

- By Frege, Believes operates on the meaning of A, i.e., int(A)
  ... which is a quadruple of functions
  ... and whose formal version is the quadruple of closed terms

$$\text{fint}(A) \equiv \Big(\lambda \text{BB}_1\text{J}\ \text{likes}(\text{B}, \text{J}),\ \lambda \text{BB}_1\text{J}\ \text{the}(\text{B}_1),\ \lambda \text{BB}_1\text{J}\ \text{blond},\ \lambda \text{BB}_1\text{J}\ \text{John}\Big)$$

★ We add a new, denotational constant Believes^A and the reduction

$$\text{Believes}(\text{George}, \text{that } A) \Rightarrow \text{Believes}^A(\text{George}, \text{fint}(A))$$

$$\equiv \text{Believes}^A(\text{George}, \lambda \text{PBB}_1\ \text{likes}(\text{B}, \text{J}),$$

$$\lambda \text{BB}_1\text{J}\ \text{the}(\text{B}_1),\ \lambda \text{BB}_1\text{J}\ \text{blond},\ \lambda \text{BB}_1\text{J}\ \text{John})$$

- What is the denotation of Believes^A?

Coindexing (and quantifying) in

John hopes that the blond likes him

$$\xrightarrow{\text{render}} \text{Hopes}(\text{John}, \text{that Likes}(\text{the}(\text{blond}), \text{him}))$$

$$\xrightarrow{\text{coindex}} \text{Hopes}(\text{J}, \text{that likes}(\text{the}(\text{blond}), \text{J})) \text{ where } \{\text{J} := \text{John}\}$$

$$\Rightarrow \text{Hopes}(\text{J}, \text{that } A[\text{J}]) \text{ where } \{\text{J} := \text{John}\}$$

with $\boxed{A \Rightarrow_{\text{cf}} \text{Likes}(\text{B}, \text{J}) \text{ where } \{\text{B} := \text{the}(\text{B}_1), \text{B}_1 := \text{blond}\}}$

$$\text{fint}(A[\text{J}]) = \Big(\lambda \text{BB}_1 \text{J Likes}(\text{B}, \text{J}), \lambda \text{BB}_1 \text{J the}(\text{B}_1), \lambda \text{BB}_1 \text{J blond}\Big)$$

J is bound in $\text{fint}(A[\text{J}])$, but $\text{int}(A[\text{J}])$ needs its argument; so we set

$$\boxed{\text{Hopes}(\text{J}, \text{that } A[\text{J}]) \Rightarrow \text{Hopes}^{A[\text{J}]}(\text{J}, \text{J}, \text{fint}(A[\text{J}]))}$$

with a new denotational constant $\text{Hopes}^{A[\text{J}]}$

where the first argument J refers to the hopeful person,

. . . the second refers to the argument of the algorithm $\text{int}(A[\text{J}])$,

. . . and both are set to John (in this example)

⋆ The reduction rule (attap) for attitudinal application

- Suppose $A[\mathrm{x}]$ is a proper, extended term, with $\mathrm{x} \equiv \mathrm{x}_1, \ldots, \mathrm{x}_k$,

$$A \Rightarrow_{\mathrm{cf}} A_0 \text{ where } \{\mathrm{P}_1 := A_1, \ldots, \mathrm{P}_n := A_n\},$$

let $\mathrm{P} \equiv \mathrm{P}_1, \ldots, \mathrm{P}_n$ and let the $(n+1)$-tuple of closed terms

$$\mathrm{fint}(A([\mathrm{x}]) \equiv \left(\lambda \mathrm{x} \lambda \mathrm{P}\, A_0, \ldots, \lambda \mathrm{x} \lambda \mathrm{P}\, A_n\right)$$

be the formal referential intension of $A[\mathrm{x}]$

(attap) If $C(B_1, \ldots, B_m, \text{that } A)$ is an attitudinal term whose free variables are all in the list x, then

$$\boxed{C(B_1, \ldots, B_m, \text{that } A)[\mathrm{x}] \Rightarrow C^{A[\mathrm{x}]}(B_1, \ldots, B_m, \mathrm{x}, \mathrm{fint}(A[\mathrm{x}]))[\mathrm{x}]}$$

where $C^{A[\mathrm{x}]}$ is a denotational constant associated with C and $A[\mathrm{x}]$

Look at all these constants; what do they all denote?

• We assume given $\overline{c} : \sigma$ for every denotational constant $c : \sigma$, including $\overline{\text{red}}, \overline{\text{good}}, \overline{\text{loves}}, \ldots$, not entirely without controversy or philosophical argument—which, however, is *not a matter of logic*

• We need to assume similarly that we are given an interpretation $\overline{C}$ for every attitudinal constant C, including $\overline{\text{Believes}}, \overline{\text{Claims}}, \ldots$ —not as a matter of logic but of language

• These interpretations are not objects in the typed universe of L: they are operators on this universe and its acyclic recursors, and there is a (small but fussy) technical problem in specifying exactly what kind of objects they are

The (acyclic) L-recursors

If $\alpha(x) = \alpha_0(x)$ *where* $\{d_1 := \alpha_1(x,d), \ldots, d_n := \alpha_n(x,d)\} : X \rightsquigarrow \mathbb{T}(\tilde{\mathrm{t}})$,

set $\alpha(x)(u) = \alpha_0(x)(u)$ *where* $\{d_1 := \alpha_1(x,d), \ldots, d_n := \alpha_n(x,d)\}$

- $\mathbb{T}^* = \left(\bigcup_\sigma \mathbb{T}(\sigma)\right)^*$ = the set of all finite sequences of typed objects
- $\mathcal{A}$ = the collection of all acyclic recursors $\alpha : Z \rightsquigarrow \mathbb{T}(\tilde{\mathrm{t}})$ with $Z \subset \mathbb{T}^*$

★ An attitudinal constant $C : \tilde{\mathrm{e}}^m \times \tilde{\mathrm{t}} \to \tilde{\mathrm{t}}$ is interpreted by some

$$\overline{C} : \mathbb{T}^* \times \mathcal{A} \to \mathbb{T}(\tilde{\mathrm{t}})$$

such that for all $\vec{y} \in \mathbb{T}(\tilde{\mathrm{e}})^m, \vec{z} \in \mathbb{T}^*$ and $\alpha, \beta : Z \rightsquigarrow \mathbb{T}(\tilde{\mathrm{t}})$ in $\mathcal{A}$

$$\alpha(\vec{z})(u) = \beta(\vec{z})(u) \implies \overline{C}(\vec{y}, \vec{z}, \alpha)(u) = \overline{C}(\vec{y}, \vec{z}, \beta)(u)$$

★ $C(B_1, \ldots, B_m, \text{that } A)[\mathrm{x}] \Rightarrow C^{A[\mathrm{x}]}(B_1, \ldots, B_m, \mathrm{x}, \mathrm{fint}(A[\mathrm{x}]))[\mathrm{x}]$, with

$$\boxed{\overline{C}^{A[\mathrm{x}]}(y_1, \ldots, y_m, x_1, \ldots, x_k, \alpha) = \overline{C}(y_1, \ldots, y_m, x_1, \ldots, x_k, \alpha)}$$

What would be nice to do

- The logic allows True(that A), True($\mathtt{x}$, that $A[\mathtt{x}]$) as propositional attitudes (and in more than one ways), but also "unusual" constructs like

$$\text{Direct(that } A) \iff A \text{ denotes directly,}$$
$$\text{Even(that } A) \iff \text{the number of parts in int}(A) \text{ is even}$$

- Find natural conditions for the attitudinal operations which actually occur in natural language
- . . . and these should be such that they would make it possible to prove the Decidability of Synonymy Conjecture for the full language, not just its denotational part

Back to the confused king George IV

Suppose u is the state of Scott's book signing, and set

$A :\equiv$ The king believes of Scott that he is Scott

$B :\equiv$ The king does not believe of him that he is Scott

- With the correct rendering, we have

Scott is Scott $\approx_u$ He is Scott (in the context of a de re belief)

and so $B \equiv \neg A$ and they cannot both be true
We assume that A is true and so B is false

- Why do we see a paradox in this? Most likely because
- *the king claims, believes, swears . . .* that A *and also* that B
  . . . but $A \not\approx_u B$, and so the king may claim, believe, swear . . .
  that A and also that B without causing a formal contradiction