Referential Intensions and Referential Synonymy: Moschovakis' Theory of Meaning

Fritz Hamm

22. Juni 2010

The Euclidian Algorithm

$$gcd(x, y) = \ \textit{if}\ (rem(x, y) = 0)\ \textit{then}\ y$$

$$\textit{else}\ gcd(y, rem(x, y)) \qquad (x \geq y \geq 1)$$

where $rem(x, y)$ is the unique number r such that for some q,

$$x = yq + r, \quad 0 \leq r < y$$

The Gaußian Algorithm

$$a_{11}x_1 + a_{12}x_2 \ldots + a_{1n}x_n = c_1$$

$$a_{21}x_1 + a_{22}x_2 \ldots + a_{2n}x_n = c_2$$

$$\vdots$$

$$a_{m1}x_1 + a_{m2}x_2 \ldots + a_{mn}x_n = c_m$$

Differential equations

$$x'' + 3x' + 2x = cos(t)$$

$$y'' + 3y' + 2y = sin(t)$$

$$e^{it} = cos(t) + i\ sin(t) \quad \text{(the Euler-de Moivre formula)}$$

$$z'' + 3z' + 2z = e^{it}$$

$$\text{Trial solution : } \ z = Ae^{it}$$

$$y'' + y = 5e^t sin(t)$$

Trial solution : $z = Ae^{(1+i)t}$

Conservativity I

Definition (Conservativity for type <1,1> quantifiers)

A type <1,1> quantifier Q is called conservative (Conserv) iff, for all M and $A, B \subseteq M$,

$$Q_M(A, B) \leftrightarrow Q_M(A, A \cap B)$$

Conservativity II

(1) a. At least four students smoke.
b. At least four students are students who smoke.

(2) a. All but five teams made it to the finals.
b. All but five teams are teams and made it to the finals.

(3) John's two bikes were stolen.
(3) John's two bikes are bikes that were stolen.

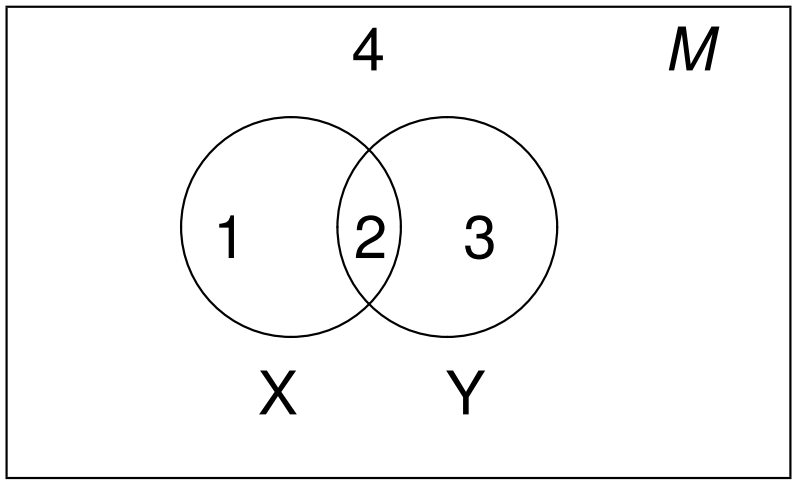
(4) a. Only bikes were stolen.
b. Many bikes were stolen.

Conservativity III

(5) $Only(A, B) \Leftrightarrow B \subseteq A$

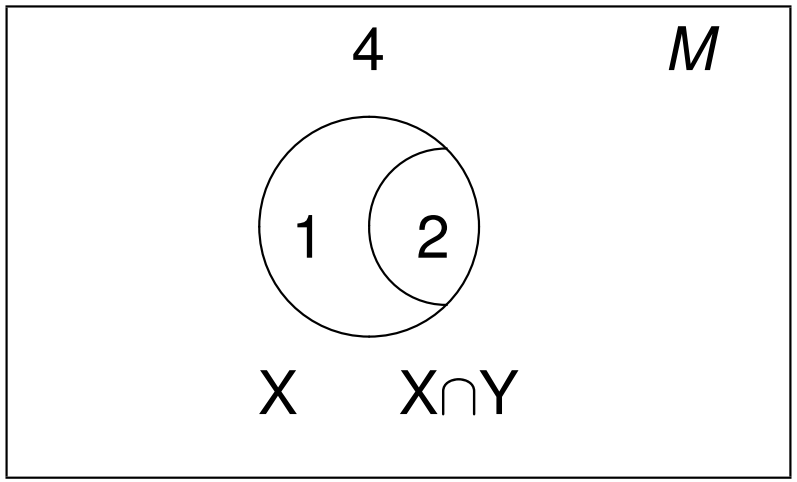
(6) $Many(A, B) \Leftrightarrow |A \cap B| > \frac{|A| \times |B|}{|M|}$

Conservativity IV



Conservativity V

$\mathbf{Q}_M XY \Longleftrightarrow \mathbf{Q}_M XX \cap Y$ Conservativity



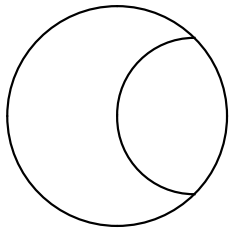
The effect of Conserv and Ext I

Definition (Ext for arbitrary quantifiers)

A quantifier Q of type $< 1, 1 >$ satisfies Ext iff the following holds:

$$\textit{If } X, Y \subseteq M, \textit{ and } M \subseteq M', \textit{then}$$

$$Q_M(X, Y) \Leftrightarrow Q_{M'}(X, Y)$$



Semantische Automaten I

(7) All poets have low self-esteem.

(8) Some dean danced nude on the table.

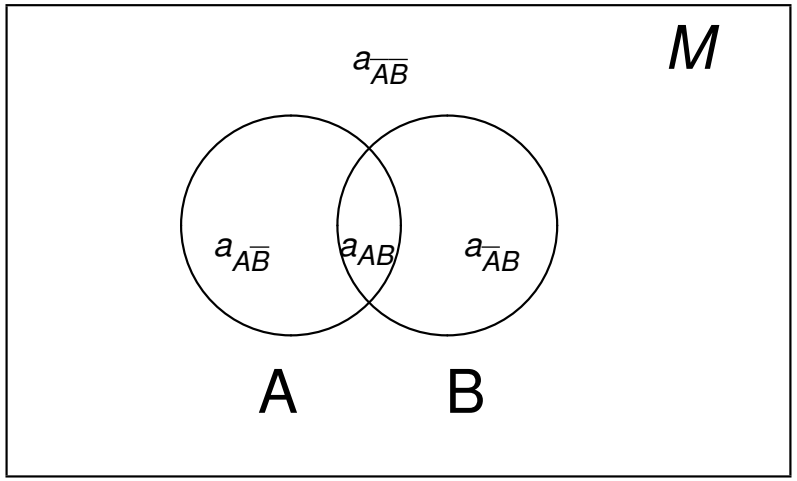
(9) At least 7 grad students prepared presentations.

(10) An even number of the students saw a ghost.

(11) Most of the students think they are smart.

(12) Less than half of the students received good grades.

Semantische Automaten II



$$\Gamma = \{a_{\overline{AB}}, a_{A\overline{B}}, a_{AB}, a_{\overline{A}B}\}$$

Example

$$\alpha_M = a_{\overline{AB}} a_{A\overline{B}} a_{AB} a_{\overline{A}B} a_{\overline{AB}}$$

$$c_1 \in \overline{AB}, c_2 \in A\overline{B}, c_3 \in AB, c_4 \in \overline{A}B, c_5 \in \overline{AB}$$

Definition

The class Q corresponding to a quantifier is represented by the set of words (language) L_Q describing all elements (models) of the class.

Theorem

A quantifier Q is first-order definable iff L_Q is accepted by an acyclic finite automaton.

Theorem

A monadic quantifier Q is definable in divisibility logic iff L_Q is accepted by a finite automaton.

Theorem

A quantifier Q of type <1,1> is definable in Presburger Arithmetic iff L_Q is accepted by a push-down automaton.

$$\chi \equiv (\forall x)(\exists y)R(x,y) \vee (\exists z)Q(z)$$

1. Step (1). Do steps (2) and (4). If one of them returns the value **t**, give the value **t**; if both of them return the value **f**, give the value **f**.
2. Step (2). For each $a \in A$, do step (3). If for every $a \in A$ the value **t** is returned, return the value **t**; if for some $a \in A$ the value **f** is returned, return the value **f**.
3. Step (3). Given $a \in A$, examine for each $b \in A$ the value $R(a,b)$. If one of these values is **t**, return the value **t**; if all these values are **f**, return the value **f**.
4. Step (4). For each $c \in A$, examine the value $Q(c)$. If one of these values is **t**, return the value **t**; if all the values are **f**, return the value **f**.

Referential Intension

$$int(\chi) \;=\; \text{the algorithm which computes the truth value of } \chi$$

Frege's Sense and Denotation

$$A \longrightarrow \mathit{sense}(A) \longrightarrow \mathit{den}(A)$$

• **Terms** (denoting objects) include sentences, which denote either 1 (truth) or 0 (falsity).

• The **sense** (meaning) of a term
"contains the mode of presentation of the denotation."

• The function $A \longrightarrow \mathit{sense}(A)$ is **compositional**.

Standard approach: possible worlds

(13) a. There are infinetly many prime numbers.
b. There are infinetly many odd numbers.

Way out: structured meanings

the sense of a complex term A can be determined from the syntactic structure of A and the senses or denotations of the basic constituents of A.

(14) Theaetetus flies.

Program **A** $\longrightarrow$ value(**A**)

- Many implementations
- To prove (or express) implementation correctness you need an independent definition of “algorithm” and “program value”.

Kaplan’s Architecture

Character $\longrightarrow$ Content $\longrightarrow$ Denotation

PART I: THE EXTENSIONAL SYSTEM

The typed λ–calculus with acyclic recursion, L^{λ}_{ar}

Types

$\sigma :\equiv e \mid t \mid s \mid (\sigma_1 \rightarrow \sigma_2)$

Ontologies

$\mathsf{T}_t = \mathsf{T}_e$

$\mathsf{T}_{(\sigma\rightarrow\tau)} =$ the set of all functions $p : \mathsf{T}_\sigma \rightarrow \mathsf{T}_\tau$

Carnap intensions

$\tilde{\sigma} \equiv (s \rightarrow \sigma)$

Examples:

$\tilde{t} \equiv (s \rightarrow t) =$ the type of Carnap intensions.

$\tilde{e} \equiv (s \rightarrow t) =$ the type of state–dependend entities.

$\tilde{q} \equiv ((\tilde{e} \rightarrow \tilde{t}) \rightarrow \tilde{t}) =$ the type of unary quantifiers.

Constants

Entities	0, 1, 2,. . . , er:	e
Names, demonstratives	I, he, him, today:	$\tilde{e}$
Common nouns	man, unicorn, temperature:	$\tilde{e} \rightarrow \tilde{t}$
Adjectives	tall, young:	$(\tilde{e} \rightarrow \tilde{t}) \rightarrow (\tilde{e} \rightarrow \tilde{t})$
Propositions	it rains:	$\tilde{t}$
Intransitive verbs	stand, run, rise:	$\tilde{e} \rightarrow \tilde{t}$
Transitive verbs	find, love, be, seek:	$(\tilde{e} \times \tilde{e}) \rightarrow \tilde{t}$
Adverbs	rapidly, allegedly:	$(\tilde{e} \rightarrow \tilde{t}) \rightarrow (\tilde{e} \rightarrow \tilde{t})$

$$(\tilde{e} \times \tilde{e}) \rightarrow \tilde{t} \equiv (\tilde{e} \rightarrow (\tilde{e} \rightarrow \tilde{t}))$$

Variables and terms

Two types of variables

- pure variables (of type σ): $v_0^\sigma, v_1^\sigma \ldots$
- recursion variables or locations (of type σ): $p_0^\sigma, p_1^\sigma \ldots$

Terms

$A :\equiv c \mid x \mid B(C) \mid \lambda(v)(B) \mid$

A_0 *where* $\{p_1 := A_1, \ldots, p_n := A_n\}$

c is a constant of type σ, $c : \sigma$
x is a variable of either kind of type σ, $x : \sigma$

$C : \sigma, B : (\sigma \to \tau)$, and $B(C) : \tau$
$B : \tau$, v a pure variable of type σ, and $\lambda(v)(B) : (\sigma \to \tau)$

$A_i : \sigma_i$, for $i \leq n, p_i, \ldots, p_n$ distinct recursion variables (locations) of type $p_i : \sigma_i$, the system $\{p_1 := A_1, \ldots, p_n := A_n\}$ is acyclic, and

$$A_0 \textit{ where } \{p_1 := A_1, \ldots, p_n := A_n\} : \sigma_0$$

Definition

A system of equations

$$\{p_1 := A_1, \ldots, p_n := A_n\}$$

is acyclic if it is possible to assign a natural number $rank(p_i)$ to each of the locations, so that

$$rank(p_i) > rank(p_j) \Leftrightarrow p_j \text{ occurs free in } A_i$$

Example

$\{f := \mathit{father}(m), m := \mathit{mother}(j), j = \mathit{John}\}$ is acyclic.

$\{p := c(p)\}$ is not acyclic.

A term is *explicit* if the recursion ("where") construct does not occur in it, and *recursive* if it is of the form
A_0 *where* $\{p_1 := A_1, \ldots, p_n := A_n\}$;
it is *closed* if it has no free occurrences of variables.

Denotations

$$\mathcal{A} = (\mathsf{T}_e, \mathsf{T}_s, \{c \mid c \in K\})$$

(D1) $den(x)(g) = g(x); den(c)(g) = c.$
(D2) $den(A(B))(g) = den(A)(g)(den(B)(g)).$
(D3) $den(\lambda(v)(B)(g) = h,$ where, for all t,
$$h(t) = den(B)(g\{v := t\})$$

(D4) $den(A_0 \textit{ where } \{p_1 := A_1, \ldots, p_n := A_n\})(g) =$
$$den(A_0(g\{p_1 := \overline{p_1}, \ldots, p_n := \overline{p_n}\})$$

The values $\overline{p_i}$ are defined for $i = 1, \ldots, n$ by recursion on $rank(p_i)$:

$$\overline{p_i} = den(A_i)(g\{p_{k_1} := \overline{p_{k_1}}, \ldots, p_{k_m} := \overline{p_{k_m}}\})$$

where $p_{k_1}, \ldots, p_{k_m}$ are the variables with lower rank than $rank(p_i)$.

Example

John loves Mary and dislikes her husband

Stage 1: $j :=$ John, $m :=$ Mary

Stage 2: $h :=$ husband$(m) =$ Mary's husband
$p :=$ loves(j, m) = the truth value of
"John loves Mary"

Stage 3: $q :=$ dislikes(j, h) the truth value of
"John dislikes Mary's husband"

Stage 4: $den(A) = p \& q$ the truth value of
"John loves Mary and
dislikes her husband".

Congruence

Definition

Congruence is the smallest equivalence relation $\equiv_c$ between terms which satisfies the following conditions:

(C1) If $A \equiv_c A'$ and $B \equiv_c B'$, then $A(B) \equiv_c A'(B')$.
(C2) If $A \equiv_c B$, then

$$\lambda(u)(A) \equiv_c \lambda(v)(B\{u := v\}),$$

where $B\{u := v\}$ is the result of replacing u by v in all its free occurrences in B and v is not bound in $(B\{u := v\})$.
(C3) If $A_i \equiv_c B_i$ for $i = 0, \ldots, n$ and $q_1, \ldots, q_n$ are distinct recursion variables of the same respective types as $p_1, \ldots, p_n$, then
A_0 *where* $\{p_1 := A_1, \ldots, p_n := A_n\} \equiv_c$
$B_0\{\vec{p} :\equiv \vec{q}\}$ *where* $\{q_1 := B_1\{\vec{p} :\equiv \vec{q}\}, \ldots, q_n := B_n\{\vec{p} :\equiv \vec{q}\}\}$;

Definition

(C4) $A \equiv_c A$ *where* $\{\}$.
(C5) If π is a permutation of the set $\{1, \ldots, n\}$, then
A_0 *where* $\{p_1 := A_1, \ldots, p_n := A_n\} \equiv_c$
A_0 *where* $\{p_{\pi(1)} := A_{\pi(1)}, \ldots, p_{\pi(n)} := A_{\pi(n)}\}$

Example

A *where* $\{p := B, q := C\} \equiv_c \{q := C, p := B\}$

$\pi(1) = 2$ and $\pi(2) = 1$

States: A *state* is an quintuple
$a =< i, j, k, A, \delta >$ where i is a possible world, j a moment of time, k a point in space, A a speaker, and δ a function which assigns values to all occurrences of proper names and demonstratives.

Carnap objects of type $\tilde{\sigma} \equiv (s \rightarrow \sigma)$

it rains$(a) = 1 \Leftrightarrow$ it is raining in state a.

A Carnap object $x : \tilde{\sigma}$ is rigid if for all states a, b $x(a) = x(b)$

Common nouns and intransitive verbs of type $(\tilde{e} \rightarrow \tilde{t})$.

$$man(x, a) \Leftrightarrow x(a) \text{ is a man in state a}$$

$$temperature(x, a) \Leftrightarrow x(a) \text{ the temperature in state a is } x(a)$$

$$stand(x, a) \Leftrightarrow x(a) \text{ is standing in state a}$$

$$rise(x, a) \Leftrightarrow x \text{ is rising in state a}$$

Locality

An object p of type $(\tilde{e} \to \tilde{t})$ is local if each value $p(x, a)$ depends only on $x(a)$, i.e.

$$x(a) = x'(a) \Rightarrow p(x, a) = p(x', a).$$

$$\mathit{rise}(x, a) \Leftrightarrow \frac{\partial x(a\{j := t\})}{\partial t}(a) > 0$$

Logical constants

$$
\begin{array}{ll}
=_\sigma: & \sigma \times \sigma \rightarrow \tilde{t} \\
\neg, \Box: & \tilde{t} \rightarrow \tilde{t} \\
\&, \vee: & \tilde{t} \times \tilde{t} \rightarrow \tilde{t} \\
\textit{every}, \textit{some}: & (\tilde{e} \rightarrow \tilde{t}) \rightarrow \tilde{t} \\
\textit{the}: & (\tilde{e} \rightarrow \tilde{t}) \rightarrow \tilde{e}
\end{array}
$$

$$
(p\&q)(a) = \begin{cases}
1 & \textit{if } p(a) = q(a) = 1, \\
0, & \textit{if } p(a) = 0 \textit{ and } [q(a) = 0 \textit{ or } q(a) = 1], \\
0, & \textit{if } q(a) = 0 \textit{ and } [p(a) = 0 \textit{ or } p(a) = 1], \\
\textit{er}, & \textit{otherwise}.
\end{cases}
$$

Descriptions

$$
\textit{the}(p)(a) = \begin{cases}
\text{the unique } y \in \mathtt{I}_e \text{ such that } p(\textit{dere}(\tilde{y}, a), a), \\
\text{if it exists,} \\
\textit{er}, \text{ otherwise.}
\end{cases}
$$

Modal operators

$$\Box(p)(a) \Leftrightarrow (\forall b)p(b).$$

$$\Box_{dere}(p, x) \Leftrightarrow x \text{ necessarily has property } p \ (p : (\tilde{e} \rightarrow \tilde{t}, x : \tilde{e}).$$

$$\Box_1(p, x)(a) = \Box(p(dere(x, a)))(a).$$

$$\Box_n(p, x_1, \ldots, x_n)(a) = \Box(p(dere(x_1, a), \ldots, dere(x_n, a)))(a)$$

(15) I am necessarily here.

$\Box(reside(I, here))$ $\Box_1(\lambda(x)reside(x, here), I)$

$\Box_1(\lambda(y)reside(I, y), here)$ $\Box_2(reside, I, here)$

$$reside : \tilde{e} \times \tilde{e} \rightarrow \tilde{t}$$

(16) John kissed his wife $\longrightarrow$ $kissed(John, wife(his))$
$\longrightarrow_\lambda$ $(\lambda(j)kissed(j, wife(j)))(John)$

(17) John loves his wife and he honors her
$\longrightarrow$ $loves(John, wife(his))\&honors(he, her)$
$\longrightarrow_\lambda$ $\lambda(j)[loves(j, wife(j))\&honors(j, her)(John)]$
$\longrightarrow_\lambda$
$\lambda(j)[\lambda(w)(loves(j, w)\&honor(j, w)(wife(j)))](John)$

(18) If $X_i \longrightarrow \overline{X_i} : \tilde{e}$, set
X_1 *and* $X_2 \longrightarrow \lambda(r)(r(x_1)\&r(x_2))$
where $\{x_1 = \overline{X_1}, x_2 = \overline{X_2}\}$

John and Mary
$\longrightarrow \lambda(r)(r(x_1)\&r(x_2)$ *where* $\{x_1 = John, x_2 = Mary\}$

(19) If $X \longrightarrow \overline{X} : \tilde{e}$ *and* $Q \longrightarrow \overline{Q} : \tilde{q}$, set
X *and* $Q \longrightarrow \lambda(r)(r(x)\&q(r)$
where $\{x = \overline{X}, q = \overline{Q}\}$

the teacher and every student
$\longrightarrow \lambda(r)(r(x)\&q(r))$ *where* $\{x =$ *the teacher*$, q =$
every student$\}$

(20) If $Q_i \longrightarrow \overline{Q_i} : \tilde{q}$, set
Q_1 *and* $Q_2 \longrightarrow \lambda(r)(q_1(r)\&q_2(r)$
where $\{q_1 = \overline{Q_1}, q_2 = \overline{Q_2}\}$

some boy and every girl
$\longrightarrow \lambda(r)(b(x)\&g(r))$ *where* $\{b = some(boy), g = every(girl)\}$

(21) If $P_i \longrightarrow \overline{X_i} : \tilde{e} \rightarrow \tilde{t}$, set
P_1 *and* $P_2 \longrightarrow \lambda(i)(p_1(i)\&p_2(i))$
where $\{p_1 = \overline{P_1}, p_2 = \overline{P_2}\} : \tilde{e} \rightarrow \tilde{t}$

the temperature is ninty and rising
$\longrightarrow (\lambda(t)(n(t)\&r(t)$ *where* $\{n = \lambda(x)[x = 90], r =$ *rises*$\})$(*the temperature*)

(22) John loves himself. $\longrightarrow$ $loves(John, himself)$
$\longrightarrow_{ara}$ $loves(j, j)$ $where$ $\{j = John\}$

(23) John kissed his wife.
$\longrightarrow$ $kissed(John, wife(his))$
$\longrightarrow_{ar}$ $kissed(j, wife(j))$ $where$ $\{j = John\}$

(24) John loves his wife and he honors her
$\longrightarrow$ $loves(John, wife(his))\&honors(he, her)$
$\longrightarrow_{ar}$ $loves(j, wife(j))\&honors(j, her)$ $where$ $\{j = John\}$
$\longrightarrow_{ar}$ $(loves(j, w)\&honor(j, w)$ $where$ $\{w = wife(j)\})$
$where$ $\{j = John\}$

$loves(j, w)\&honors(j, w)$ $where$ $\{w = wife(j), j = John\}$

PART II: REFERENTIAL INTENSIONS AND REFERENTIAL SYNONYMY

I. Reduction, irreducability, canonical forms

$A \Rightarrow B \Leftrightarrow A \equiv_c B$ (A is congruent with B) or A and B have the same meaning and B expresses that meaning "more simply".

A is irreducible $\Leftrightarrow$ for all B, if $A \Rightarrow B$, then $A \equiv_c B$.

Theorem (Canonical Form Theorem)

For each term A, there is an irreducible term

$$cf(A) \equiv A \text{ or } cf(A) \equiv A_0 \text{ where } \{p_1 := A_1, \ldots, p_n := A_n\},$$

such that $A \Rightarrow cf(A)$*; moreover,* $cf(A)$ *is the unique (up to congruence) irreducible term to which A can be reduced, i.e.*

$$\text{if } A \Rightarrow B \text{ and } B \text{ is irreducible, then } B \equiv_c cf(A)$$

II. Referential intensions

$$cf(A) \equiv A_0 \textit{ where } \{p_1 := A_1, \ldots, p_n := A_n\}$$

The *referential intension* of A $int(A)$ is (intuitively) the abstract algorithm which computes for each assignment g the denotation $den(A)(g)$, as described in case (D4).

(D4) $den(A_0 \ where \ \{p_1 := A_1, \ldots, p_n := A_n\})(g) = den(A_0(g\{p_1 := \overline{p_1}, \ldots, p_n := \overline{p_n}\})$

The values $\overline{p_i}$ are defined for $i = 1, \ldots, n$ by recursion on $rank(p_i)$:

$$\overline{p_i} = den(A_i)(g\{p_{k_1} := \overline{p_{k_1}}, \ldots, p_{k_m} := \overline{p_{k_m}}\})$$

where $p_{k_1}, \ldots, p_{k_m}$ are the variables with lower rank than $rank(p_i)$.

II. Referential synonymy

Two non–immediate terms are *referentially synonymous* if their referential intensions are *naturally isomorphic*, so that they model – they are, from the mathematical point of view – identical algorithms.

$$A \approx B \Leftrightarrow A \text{ and } B \text{ are referentially synonymous.}$$

Theorem (Referential Synonymy Theorem)

Two terms A, B are referentially synonymous if and only if

$$A \Rightarrow_{nf} A_0 \text{ where } \{p_1 := A_1, \ldots, p_n := A_n\},$$

$$B \Rightarrow_{nf} B_0 \text{ where } \{p_1 := B_1, \ldots, p_n := B_n\},$$

for some $n \geq 0$ *and suitable,* $A_0, A_1, \ldots, A_n, B_0, B_1, \ldots, B_n$, *so that*

$$\models A_i = B_i$$

Reduction rules: part I

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

(tans) 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 $\lambda(u)(A) \Rightarrow \lambda(u)(B)$

(rep3) If $A_i \Rightarrow B_i$ for $i = 0, \ldots, n$, then
A_0 *where* $\{p_1 := A_1, \ldots, p_n := A_n\} \Rightarrow$
B_0 *where* $\{p_1 := B_1, \ldots, p_n := B_n\}$

The reduction calculus: rules for recursion

(head) $(A_0 \ \mathit{where}\ \{\vec{p} := \vec{A}\})\ \mathit{where}\ \{\vec{q} := \vec{B}$
$\Rightarrow A_0 \ \mathit{where}\ \{\vec{p} := \vec{A}, \vec{q} := \vec{B}\}$

(B–S) $A_0 \ \mathit{where}\ \{p := (B_0 \ \mathit{where}\ \{\vec{q} := \vec{B}\}), \vec{p} := \vec{A}\}$
$\Rightarrow A_0 \ \mathit{where}\ \{p := B_0, \vec{q} := \vec{B}, \vec{p} := \vec{A}\}$

(recap) $(A_0 \ \mathit{where}\ \{\vec{p} := \vec{A}\})(B)$
$\Rightarrow A_0(B) \ \mathit{where}\ \{\vec{p} := \vec{A}\}$

(ap) $A(B) \Rightarrow A(b) \ \mathit{where}\ \{b := B\}$

Example

(head rule)
$(loves(j, w)\ where\ \{w := wife(p), p :=\ \text{Paul}\})\ where$
$\{j :=\ \text{John}\} \Rightarrow$
$loves(j, w)\ where\ \{w := wife(p), p :=\ \text{Paul}, j :=\ \text{John}\}$

The Bekič–Scott rule (B–S)

A_0 *where* $\{p := (B_0$ *where* $\{\vec{q} := \vec{B}\}), \vec{p} := \vec{A}\}$

$\Rightarrow A_0$ *where* $\{p := B_0, \vec{q} := \vec{B}, \vec{p} := \vec{A}\}$

Example

((B–S)–rule)
loves(j, w) *where* $\{w :=$ (*wife*(p) *where* $\{p :=$ Paul$\}), j :=$ John$\}$
$\Rightarrow$ *loves*(j, w) *where* $\{w :=$ (*wife*$(p), p :=$ Paul$, j :=$ John$\}$

The recursion application rule (recap)

$(A_0 \textit{ where } \{\vec{p} := \vec{A}\})(B) \Rightarrow A_0(B) \textit{ where } \{\vec{p} := \vec{A}\}$

Example

(recap–rule)
Let $A \equiv (h = s) \textit{ where } \{h := \text{ He}, s := \text{ Scott}\}$.

$A(a) \Rightarrow B$ where

$B \equiv (h = s)(a) \textit{ where } \{h := \text{ He}, s := \text{ Scott}\}$

Immediate terms and the application rule

$X :\equiv x \mid p(v_1, \ldots, v_n) \mid \lambda(u_1, \ldots, u_m)(p(v_1, \ldots, v_n)) \mid$
$\lambda(u_1, \ldots, u_m)(p(v_1, \ldots, v_n)$

(p a location, v_i, u_j pure)

$A(B) \Rightarrow A(b)$ *where* $\{b := B\}$

(B proper, b fresh)

Example

John is tall $\longrightarrow$ $tall(John) \Rightarrow tall(j)$ *where* $\{j :=$ John$\}$

$tall(j) \Rightarrow tall(j')$ *where* $\{j' := j\}$

$tall(John) \Rightarrow tall(j)$ *where* $\{j :=$ John$\}$
$\Rightarrow (tall(j')$ *where* $\{j' := j\})$ *where* $\{j :=$ John$\}$
$tall(j')$ *where* $\{j' := j, j :=$ John$\}$.

$tall(j)$ *where* $\{j :=$ John$\} \approx tall(j')$ *where* $\{j' := j, j :=$ John$\}$.

Canonical forms and the λ–rule

$tall(John) \Rightarrow_{nf} tall(j)$ *where* $\{j :=$ John$\}$

John loves Mary

$loves(John) \Rightarrow loves(j)$ *where* $\{j :=$ John$\}$ (ap)
$loves(John)(Mary) \Rightarrow (loves(j)$ *where*
$\{j :=$ John$\})$(Mary) (rep_1)
$\Rightarrow$ $loves(j)$(Mary) *where* $\{j :=$ John$\}$ (recap)
$\Rightarrow$ $(loves(j)(m)$ *where* $\{m :=$ Mary$\})$ *where*
$\{j :=$ John$\}$ (ap)
$\Rightarrow_{nf} loves(j)(m)$ *where* $\{m :=$ Mary$, j :=$ John$\}$ (head)

$\lambda(u)(A_0 \textit{ where } \{p_1 := A_1, \ldots, p_n := A_n\}$
$\Rightarrow \lambda(u)(A'_0 \textit{ where } \{p'_1 := \lambda(u)A'_1, \ldots, p'_n := \lambda(u)A'_n\}$

where for $i = 1, \ldots, m$, p'_i is a fresh location and A'_i is defined by the replacement

$$A'_i :\equiv A_i\{p_1 :\equiv p'_1(u), \ldots, p_n :\equiv p'_n(u)\}$$

Example

(λ–rule)

$\lambda(u)\mathit{danced}(u, w)$ *where* $\{w := \text{wife}(u)\}$

$\Rightarrow \lambda(u)\mathit{danced}(u, w'(u))$ *where* $\{w' := \lambda(u)\text{wife}(u)\}$

THEOREMS

Theorem

If $A \Rightarrow B$, *then* $\models A = B$.

Theorem

a. *Constants and immediate terms are irreducible.*

b. *An application term* $A(B)$ *is irreducible if and only if* B *is immediate and* A *is explicit (up to congruence) and irreducible.*

c. *A* λ*–term* $\lambda(u)(A)$ *is irreducible if and only if* A *is explicit (up to congruence) and irreducible.*

d. *A recursive term*

$$A_0 \text{ where } \{p_1 := A_1, \ldots, p_n := A_n\}$$

is irreducible if and only if all the parts $A_0, \ldots, A_n$ *are explicit (up to congruence) and irreducible.*

($CF1$) $cf(c) :\equiv c(\equiv c \textit{ where } \{\}),\ cf(x) :\equiv (\equiv x \textit{ where } \{\}$
($CF2$) suppose $cf(A) \equiv A_0 \textit{ where } \{p_1 := A_1, \ldots, p_n := A_n\}$ $(n \geq 0)$. If X is immediate, then

$$cf(A(X) :\equiv A_0(X) \textit{ where } \{p_1 := A_1, \ldots, p_n := A_n\}$$

and if B is proper and $cf(B) \equiv B_0\{q_1 := B_1, \ldots, q_n := B_m\}$, then

$$cf(A(B)) := A_0(q_0) \textit{ where } \{p_1 := A_1, \ldots, p_n := A_n,$$

$$q_0 := B, q_1 := B_1, \ldots, q_m := B_m\}$$

($CF3$) For any pure variable u, if

$$cf(A) \equiv A_0 \textit{ where } \{p_1 := A_1, \ldots, p_n := A_n\ (n \geq 0)\}$$

then,

$$cf(\lambda(u)A) \equiv \lambda(u)A'_0 \textit{ where } \{p'_1 := \lambda(u)A'_1, \ldots, p'_n := \lambda(u)A'_n\}$$

where (as in the λ-rule for reduction) each p'_i is a fresh location and

$$A'_i \equiv A_i\{p := p'_i, \ldots, p_n := p'_n(u)\}\ (k_i \geq 0)$$

(*CF4*) *If* $A \equiv A_0$ *where* $\{p_1 := A_1, \ldots, p_n := A_n\}$ with $n \geq 0$ and if , for $i = 0, \ldots, n$,

$$cf(A_i) \equiv A_{i,0} \textit{ where } \{p_{i,1} := A_{i,1}, \ldots, p_{i,k} := A_{i,k_i}\}$$

then

$$cf(A) :\equiv A_{0,0} \textit{ where } \{p_{0,1} := A_{0,1}, \ldots, p_{0,k_0} := A_{0,k_0}, \}$$

$$p_1 := A_{1,0}, p_{1,1} := A_{1,1}, \ldots, p_{1,k_1} := A_{1,k_1}$$

$$\vdots$$

$$p_n := A_{n,0}, p_{n,1} := A_{n,1}, \ldots, p_{n,k_n} := A_{n,k_n}\}$$

Theorem

For every term A:

1. *The canonical form of A is a term*

$$cf(A) \equiv A_0 \text{ where } \{p_1 : A_1, \ldots, p_n := A_n\} \ (n \geq 0)$$

   with explicit, irreducible parts $A_0, A_1, \ldots, A_n$, so that its is irreducible. A constant c or a variable x occur (free) in $cf(A)$ if and only if it occurs (free) in A.

2. $A \Rightarrow cf(A)$
3. *If A is irreducible, then $cf(A) = A$.*
4. *If $A \Rightarrow B$, then $cf(A) \equiv_c cf(B)$.*
5. *If $A \Rightarrow B$ and B is irreducible, then $B \equiv_c cf(A)$.*

Theorem

1. *If $A \Rightarrow D$ and $B \Rightarrow D$ for some D, then $A \approx_s B$; and similarly, if $D \Rightarrow A$ and $D \Rightarrow B$ for some D, then $A \approx_s B$.*
2. *If $A \approx_s X$ for some immediate term X, then A is also immediate and $A \equiv_c X$.*
3. *$\approx_s$ is an equivalence relation on terms which extends congruence and respects application, λ-abstraction and the formation of recursive terms, i.e.,*

$$\frac{A_1 \approx_s B_1 \; A_2 \approx_s B_2}{A_1(A_2) \approx_s B_1(B_2)} \qquad \frac{A \approx_s B}{\lambda(u)A \approx_s \lambda(u)B}$$

$$\frac{A_0 \approx_s B_0,\; A_1 \approx_s B_1, \ldots, A_n \approx_s B_n}{A_0 \text{ where } \{p_1 := A_1, \ldots, p_n := A_n\} \approx_s B_0 \text{ where } \{p_1 := B_1, \ldots, p_n := B_n\}}$$

4. *If $z : \sigma$ is a constant c, or a variable of either kind, $C : \sigma$ is a proper term of the same type and the substitution $\{z :\equiv C\}$ is free in A then*

$$A\{z :\equiv C\} \approx_s (cf(A))\{z :\equiv C\}.$$

Definition

Let A be a non–immediate term with canonical form

$$cf(A) \equiv A_0 \text{ where } \{p_1 := A_1, \ldots, p_n := A_n\},$$

and for $i = 0, \ldots, n$ set

$$\alpha_i(g, d_1, \ldots, d_n) = den(A_i)(g\{p_1 := d_1, \ldots, p_n := d_n\}).$$

The referential intension of A is the tuple of functions

$$int(A) = (\alpha_0, \alpha_1, \ldots, \alpha_n)$$

Example

$int(loves(John, Mary) = (\alpha_0, \alpha_1, \alpha_2)$ where,

$$\alpha_0(g, j, m) = loves(j, m)$$

$$\alpha_1(g, j, m) = John$$

$$\alpha_2(g, j, m) = Mary$$

(25) $\alpha_0 \ : \ G \times T_{\sigma_1} \times \ldots T_{\sigma_n} \rightarrow T_{\sigma},$

(26) *for* $i = 1, \ldots, n$
$\alpha_i \ : \ G \times T_{\sigma_1} \times \ldots T_{\sigma_n} \rightarrow T_{\sigma_i}.$

(27) For all $g, d_1, \ldots, d_n, d'_1, \ldots, d'_n$ with $rank(j) = rank(p_j)$:
If $d_j = d'_j$ for all j such that $rank(j) < rank(i)$, *then*

$$\alpha_i(g, d_1, \ldots, d_n) = \alpha_i(g, d'_1, \ldots, d'n)$$

Acyclic recursors

A tuple of functions $(\alpha_0, \alpha_1, \ldots, \alpha_n)$ which satisfies conditions (25) – (27) with some

$$\mathit{rank} : \{1, \ldots, n\} \to \{1, \ldots, n\}$$

is called an acyclic recursor on G to T_σ.

$$\alpha = (\alpha_0, \alpha_1, \ldots, \alpha_n) : G \rightsquigarrow \mathsf{T}_\sigma$$

$$\overline{\alpha} : G \to \mathsf{T}_\sigma$$

$$\overline{\alpha}(g) = \alpha_0(g, \overline{d_1}, \ldots, \overline{d_n})$$

$$d_i = \alpha_i(g, d_1, \ldots, d_n) \qquad (i = 1, \ldots, n)$$

Intensions

$$int(A) : G \rightsquigarrow \mathsf{T}_\sigma \qquad A : \sigma$$

$$\overline{int(A)}(g) = den(A)(g)$$

Definition

Two acyclic recursors

$$\alpha = (\alpha_0, \alpha_1, \ldots, \alpha_n),\ \beta = (\beta_0, \beta_1, \ldots, \beta_n) : G \rightsquigarrow \mathsf{T}_\sigma$$

of respective types $\sigma_1 \times, \ldots, \times \sigma_n$ and $\tau_1 \times, \ldots, \times \tau_n$ and into the same output set T_σ are naturally isomorphic, if they have the same dimension ($m = n$) and there is a permutation

$$\pi\ \ \{0, 1, \ldots, n\} \rightarrow \{0, 1, \ldots, n\},\ \ \textit{with}\ \pi(0) = 0$$

such that $\sigma_{\pi(i)} = \tau_i$ for $i = 1, \ldots, n$ and

$$\alpha_{\pi(i)}(g, d_1, \ldots, d_n) = \beta_i(g, d_{\pi(1)}, \ldots, d_{\pi(n)})$$

for ($g \in G, d_i \in \mathsf{T}_{\sigma_i}, i = 0, \ldots, n$)

$\alpha \simeq \beta \Leftrightarrow \alpha$ and β are naturally isomorphic.

$A \approx B \Leftrightarrow$ A, B are immediate and for all g, $den(A)(g) = den(B)(g)$.

or A and B are proper and $int(A) \simeq int(B)$.

A Logical Calculus for Meaning and Synonymy

$$A \approx A \qquad \frac{A \approx B}{B \approx A} \qquad \frac{A \Rightarrow B}{A \approx B} \qquad \frac{A \approx B \qquad B \approx C}{A \approx C}$$

$$\frac{A_1 \approx B_1 \qquad A_2 \approx B_2}{A_1(A_2) \approx B_1(B_2)} \qquad \frac{A \approx B}{\lambda(u)A \approx \lambda(u)B}$$

$$\frac{A_0 \approx B_0, \quad A_1 \approx B_1, \quad \ldots \quad A_n \approx B_n}{A_0 \text{ where } \{p_1 := A_1, \ldots, p_n := A_n\} \approx B_0 \text{ where } \{p_1 := B_1, \ldots, p_n := B_n\}}$$

$$\frac{\models A = B}{A \approx B} \qquad \frac{}{(\lambda(u)A(u))(v) \approx A\{u := v\}}$$

$$\models A = B \Leftrightarrow \text{ for all assignments } g, \mathit{den}(A)(g) = \mathit{den}(B)(g)$$

Compositionality

Theorem

For all terms A, B, C and every variable x such that type(x) = type(B) = type(C),

$$\text{if } B \approx C, \text{ then } A\{x :\equiv B\} \approx A\{x :\equiv C\},$$

assuming that the substitution is free.

Theorem

a. *No location occurs in more than one part of an explicit term.*

b. *Suppose a location p occurs in two parts A_k and A_l of a term A, and neither A_k nor A_l denotes a function which is independent of p, i.e., for some assignment to the variables g and objects r, r',*

$$den(A_k)(g\{p := r\}) \neq den(A_k)(g\{p := r'\})$$

and similarly with A_l. It follows that A is not referentially synonymous with any explicit term.

John loves his wife and he honors her.

Example

$p \& q$ *where* $\{p := loves(j, w), q := honors(j, w), j :=$ John$, w := wife(j)\}$

Definition (Utterance)

An utterance is a pair (A, a) of a closed Carnap intension $A : \tilde{t}$ and a state a.

$$A \Rightarrow_{cf} A_0 \textit{ where } \{p_1 := A_1, \ldots, p_n := A_n\} : \tilde{t}$$

$$\downarrow \text{recap}$$

$$A(\overline{a}) \Rightarrow_{cf} A_0(\overline{a}) \textit{ where } \{p_1 := A_1, \ldots, p_n := A_n\} : t$$

Example

$\textit{loves}(\textit{John}, \textit{Mary})(\overline{a}) \Rightarrow_{cf} \textit{loves}(j, m)(\overline{a})$ *where* $\{j := \textit{John}, m := \textit{Mary}\}$

Petros emigrated from his native Greece to the United States at a rather advanced age, and immediately fell in love with the city of Los Angeles, where he settled. Every chance he gets he declares proudly

(A) I live in Los Angeles.

When, however, a new acquaintance who had heard of this tried to start conversation with an innocent “I hear you live in LA”, Petros looked puzzled, declarded again that he lives in Los Angeles, and added emphatically:

(B) I do not live in LA.

Assume: *Los Angeles*, *LA*: $\tilde{e}$
and: $\models$ *Los Angeles* = *LA*

this implies: *Los Angeles* $\approx$ *LA*

By compositionality: *reside(I Los Angeles)* $\approx$ *reside(I, LA)*

(28) a. He doesn't know that Los Angeles is LA.
b. He doesn't know that Los Angeles is Los Angeles.

(29) a. Los Angeles is between the desert and the sea.
b. Los Angeles is between the sea and the desert.

$$\models \mathit{between}(x, y, x) = \mathit{between}(z, y, x)$$

Main Conjecture: If the set of constants is finite, then the relation of referential synonymy between closed term $L^{\lambda}_{ar}(K)$ is decidable.

Let us suppose that in a book-signing ceremony given by “the author of Waverly”, a cleverly disguised Scott autographs King Georges’s copy of Waverly. King George, being fooled by Scott’s disguise, concludes that Waverly was written by someone other than Scott. He sincerely declares

(D) He is not Scott.

pointing at the disguised author: yet King George surely disbelieves, and would vigorously deny that

(E) Scott is not Scott.

Consider: $He(\overline{a}) = Scott(\overline{a})$

This implies: $He(\overline{a}) \approx Scott(\overline{a})$

???: $(He\ is\ Scott)(\overline{a}) \approx (Scott\ is\ Scott)(\overline{a})$

$$(\mathit{He\ is\ Scott})(\overline{a}) \longrightarrow_{\mathit{render}} (\mathit{He}\ =\ \mathit{Scott})(\overline{a})$$
$$\Rightarrow_{\mathit{cf}} (h = s)(\overline{a})\ \mathit{where}\ \{h := \mathit{He}, s := \mathit{Scott}\}$$
$$\approx h(\overline{a}) = s(\overline{a})\ \mathit{where}\ \{h := \mathit{He}, s := \mathit{Scott}\}$$

$$(\mathit{Scott\ is\ Scott})(\overline{a}) \longrightarrow_{\mathit{render}} (\mathit{Scott}\ =\ \mathit{Scott})(\overline{a})$$
$$\Rightarrow_{\mathit{cf}} (s' = s)(\overline{a})\ \mathit{where}\ \{s' := \mathit{Scott}, s := \mathit{Scott}\}$$
$$\approx s'(\overline{a}) = s(\overline{a})\ \mathit{where}\ \{s' := \mathit{Scott}, s := \mathit{Scott}\}$$

By referential synonymy

$$(\neg(He = Scott)(\bar{a}) \approx (Scott = Scott)(\bar{a}))$$

Since

$$\models He \neq Scott$$

The Euclidian Algorithm

$$gcd(x, y) = \textit{if } (rem(x, y) = 0) \textit{ then } y$$
$$\textit{else } gcd(y, rem(x, y)) \; (x \geq y \; \geq 1)$$

The Euclidian Algorithm: explicit form

$$gcd(x,y) = p(x,y) \; where \; \{p := \lambda(x)\lambda(y)C(q_1(x,y), y, r(x,y)\},$$

$$q_1 := \lambda(x)\lambda(y)rem(x,y),$$

$$r := \lambda(x)\lambda(y)p(y, q_2(x,y)),$$

$$q_2 := \lambda(x)\lambda(y)rem(x,y),$$

$$\epsilon = (\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4)$$

$$\alpha_0(x, y, p, q_1, r, q_2) = p(x, y),$$
$$\alpha_1(x, y, p, q_1, r, q_2) = \lambda(x)\lambda(y)C(q_1(x, y), y, r(x, y)),$$
$$\alpha_2(x, y, p, q_1, r, q_2) = \lambda(x)\lambda(y)rem(x, y) = rem,$$
$$\alpha_3(x, y, p, q_1, r, q_2) = \lambda(x)\lambda(y)p(y, q_2(x, y),$$
$$\alpha_4(x, y, p, q_1, r, q_2) = \lambda(x)\lambda(y)rem(x, y) = rem,$$

$$(N, C, rem)$$

where N are the natural numbers, C is the construct

$$C(u, s, t) = \ \textit{if}\ (u = 0)\ \textit{then}\ s\ \textit{else}\ t$$

and *rem* is reminder.

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

(tans) 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 $\lambda(u)(A) \Rightarrow \lambda(u)(B)$

(rep3) If $A_i \Rightarrow B_i$ for $i = 0, \ldots, n$, then
A_0 *where* $\{p_1 := A_1, \ldots, p_n := A_n\} \Rightarrow$
B_0 *where* $\{p_1 := B_1, \ldots, p_n := B_n\}$

(head) $(A_0$ *where* $\{\vec{p} := \vec{A}\})$ *where* $\{\vec{q} := \vec{B}$
$\Rightarrow A_0$ *where* $\{\vec{p} := \vec{A}, \vec{q} := \vec{B}\}$

(B–S) A_0 *where* $\{p := (B_0$ *where* $\{\vec{q} := \vec{B}\}), \vec{p} := \vec{A}\}$
$\Rightarrow A_0$ *where* $\{p := B_0, \vec{q} := \vec{B}, \vec{p} := \vec{A}\}$

(recap) $(A_0$ *where* $\{\vec{p} := \vec{A}\})(B)$
$\Rightarrow A_0(B)$ *where* $\{\vec{p} := \vec{A}\}$

(ap) $A(B) \Rightarrow A(b)$ *where* $\{b := B\}$

$\lambda(u)(A_0$ *where* $\{p_1 := A_1, \ldots, p_n := A_n\}$
$\Rightarrow \lambda(u)(A'_0$ *where* $\{p'_1 := \lambda(u)A'_1, \ldots, p'_n := \lambda(u)A'_n\}$
where for $i = 1, \ldots, m$, p'_i is a fresh location and A'_i is defined by the replacement

$$A'_i :\equiv A_i\{p_1 :\equiv p'_1(u), \ldots, p_n :\equiv p'_n(u)\}$$