Two aspects of situated meaning

Eleni Kalyvianaki and Yiannis N. Moschovakis

Abstract

We introduce two structural notions of situated meaning for natural language sentences which can be expressed by terms of Montague's Language of Intensional Logic. Using the theory of referential intensions, we define for a sentence at a particular situation its *factual content* and its *local meaning* which express different abstract algorithms that compute its reference at that situation. With the use of characteristic examples, we attempt to show the distinctive roles of these two notions in any theory of meaning and to discuss briefly their relation to indexicality, propositional attitudes and translation.

1. Introduction

> If a speaker of the language can rationally believe *A* and disbelieve *B* in the same situation, then the sentences *A* and *B* do not have the same meaning—they are not synonymous.

The principle is old (Frege 1892), and it has been used both as a test for theories of meaning and a source of puzzles about belief and synonymy. We think that at least some of the puzzles are due to a confusion between two plausible and legitimate but distinct understandings of *situated meaning*, the *factual content* and the (referential) *local meaning*.

Consider, for example, the sentences

$$A \equiv \text{John loves Mary}, \text{ and } B \equiv \text{John loves her},$$

in a state (situation) *a* in which 'her' refers to Mary. They express the same information about the world in state *a* (they have the same factual content at that state); but they do not have the same meaning in that state, as they are not interchangeable in belief contexts: one may very well believe *A* but disbelieve *B* in *a*, because she does not know that 'her' refers to Mary.

We will give precise, mathematical definitions of factual content and local meaning for the fragments of natural language which can be formalized in the

Language of Intensional Logic of Montague (1973), within the mathematical theory of *referential intensions*; this is a rigorous (algorithmic), structural modeling of meanings for the typed λ-calculus developed in (Moschovakis 2006), and so the article can be viewed as a contribution to the formal "logic of meaning". We think, however, that some of our results are relevant to the discussion of these matters in the philosophy of language and in linguistics, and, in particular, to Kaplan's work on the logic of indexicals. We will discuss briefly some of these connections in Section 5.

2. Three formal languages

There are three (related) formal languages that we will deal with, the Language of Intensional Logic LIL of Montague (1973); the Two-sorted Typed λ-calculus Ty_2 of Gallin (1975); and the extension $\mathsf{L}^{\lambda}_{\mathrm{ar}}$ of Ty_2 by acyclic recursion in (Moschovakis 2006). We describe these briefly in this section, and in the next we summarize equally briefly the theory of referential intensions in $\mathsf{L}^{\lambda}_{\mathrm{ar}}$, which is our main technical tool.[1]

All three of these languages start with the same, three *basic types*

$$\mathsf{e} : \text{entities}, \quad \mathsf{t} : \text{truth values}, \quad \mathsf{s} : \text{states},$$

and, for the interpretation, three fixed, associated, non-empty sets

$$\mathbb{T}_{\mathsf{e}} = \text{the entities}, \quad \mathbb{T}_{\mathsf{s}} = \text{the states}, \quad \mathbb{T}_{\mathsf{t}} = \text{the truth values} = \{0, 1, \mathit{er}\}, \quad (1)$$

where 1 stands for truth, 0 for falsity and *er* for "error".[2] The *types* are defined by the recursion[3]

$$\sigma :\equiv \mathsf{e} \mid \mathsf{s} \mid \mathsf{t} \mid (\sigma_1 \to \sigma_2), \quad (2)$$

and a set $\mathbb{T}_\sigma$ of *objects of type* σ is assigned to each σ by adding to (1) the recursive clause[4]

$$\mathbb{T}_{(\sigma\to\tau)} = \text{the set of all functions } f : \mathbb{T}_\sigma \to \mathbb{T}_\tau. \quad (3)$$

For each type σ, there is an infinite sequence of variables of type σ

$$\mathsf{v}^{\sigma}_0, \mathsf{v}^{\sigma}_1, \ldots$$

which range over $\mathbb{T}_\sigma$.

It is also useful for our purpose here to assume a fixed (finite) set K of typed constants as in Table 1, which we will use to specify the terms of all three languages. Each constant $\mathsf{c} \in K$ stands for some basic word of natural

Names, indexicals[5]	$\mathsf{John}, \mathsf{I}, \mathsf{he}, \mathsf{thetemp}$	:	e
Sentences	$\mathsf{it\ rains}$	:	t
Common nouns	man	:	$\mathsf{e} \to \mathsf{t}$
Extensional intransitive verbs	run	:	$\mathsf{e} \to \mathsf{t}$
Intensional intransitive verbs	rise	:	$(\mathsf{s} \to \mathsf{e}) \to \mathsf{t}$
Extensional transitive verbs	love	:	$\mathsf{e} \to (\mathsf{e} \to \mathsf{t})$
The definite article	the	:	$(\mathsf{e} \to \mathsf{t}) \to \mathsf{e}$
Propositional connectives	$\&, \vee$	:	$\mathsf{t} \to (\mathsf{t} \to \mathsf{t})$
(Basic) necessity operator	$\Box$	:	$(\mathsf{s} \to \mathsf{t}) \to \mathsf{t}$
de dicto modal operators	$\mathsf{Yesterday}, \mathsf{Today}$	:	$(\mathsf{s} \to \mathsf{t}) \to \mathsf{t}$
de re modal operators	$\mathsf{Yesterday}_1, \mathsf{Today}_1$	:	$(\mathsf{s} \to (\mathsf{e} \to \mathsf{t})) \to (\mathsf{e} \to \mathsf{t})$

Table 1. Some constants with their LIL-typing.

language or logic and is assigned a type τ which (roughly) corresponds to its grammatical category; notice though, that common nouns and extensional intransitive verbs are assigned the same type $(\mathsf{e} \to \mathsf{t})$, because they take an argument of type e and (intuitively) deliver a truth value, as in the simple examples of *rendering* (formalization) in LIL,

$$\text{John is running} \xrightarrow{\text{render}} \mathsf{run}(\mathsf{John}), \quad \text{John is a man} \xrightarrow{\text{render}} \mathsf{man}(\mathsf{John}).$$

For the fixed interpretation, each constant c of type σ is assigned a function from the states to the objects of type σ:[6]

$$\text{if } \mathsf{c} : \sigma, \text{ then } \mathrm{den}(\mathsf{c}) = c : \mathbb{T}_{\mathsf{s}} \to \mathbb{T}_{\sigma}. \tag{4}$$

Thus John is interpreted in each state a by $\mathit{John}(a)$, the (assumed) specific, unique entity which is referred to by 'John' in state a.[7] The de re modal operator $\mathsf{Yesterday}_1$ is interpreted by the relation on properties and individuals

$$\mathit{Yesterday}_1(a)(p)(x) \iff p(a^-)(x),$$

where for each state a, a^- is the state on the preceding day, and similarly for Today_1.

Starting with these common ingredients, the languages LIL, Ty_2 and $\mathsf{L}^{\lambda}_{\mathrm{ar}}$ have their own features, as follows.

2.1. The language of intensional logic LIL

Montague does not admit s as a full-fledged primitive type like e and t, but uses it only as the name of the domain in the formation of function types.

This leads to the following recursive definition of types in LIL:

$$\sigma :\equiv \mathsf{e} \mid \mathsf{t} \mid (\mathsf{s} \to \sigma_2) \mid (\sigma_1 \to \sigma_2). \qquad (\mathsf{LIL}\text{-types})$$

We assume that all constants in the fixed set K are of LIL-*type*.

The terms of LIL are defined by the recursion

$$A :\equiv x \mid \mathsf{c} \mid A(B) \mid \lambda(x)(B) \mid {}^{\vee}(A) \mid {}^{\wedge}(A) \qquad (\mathsf{LIL}\text{-terms})$$

subject to some type restrictions, and each A is assigned a type as follows, where

$$A : \sigma \iff \text{the type of } A \text{ is } \sigma.$$

(LIL-T1) $x \equiv \mathsf{v}_i^\sigma$ for some LIL-type σ and some i, and $x : \sigma$.

(LIL-T2) c is a constant (of some Montague type σ), and $\mathsf{c} : \sigma$.

(LIL-T3) $A : (\sigma \to \tau), B : \sigma$ and $A(B) : \tau$.

(LIL-T4) $x \equiv \mathsf{v}_i^\sigma$ for some LIL-type σ and some i, $B : \tau$ and $\lambda(x)(B) : (\sigma \to \tau)$.

(LIL-T5) $A : (\mathsf{s} \to \tau)$ and ${}^{\vee}(A) : \tau$.

(LIL-T6) $A : \tau$ and ${}^{\wedge}(A) : (\mathsf{s} \to \tau)$.

In addition, the *free* and *bound* occurrences of variables in each term are defined as usual, and A is *closed* if no variable occurs free in it. A *sentence* is a closed term of type t.

The constructs ${}^{\vee}(A)$ and ${}^{\wedge}(A)$ are necessary because LIL does not have variables over states, and express (roughly) application and abstraction on an implicit variable which ranges over "the current state". This is explained by the semantics of LIL and made explicit in the Gallin translation of LIL into Ty_2 which we will describe in the next section.

Semantics of LIL

As usual, an *assignment* π is a function which associates with each variable $x \equiv \mathsf{v}_i^\sigma$ some object $\pi(x) \in \mathbb{T}_\sigma$. The denotation of each LIL-term $A : \sigma$ is a function

$$\mathrm{den}_{\mathsf{LIL}}(A) : \text{Assignments} \to (\mathbb{T}_{\mathsf{s}} \to \mathbb{T}_\sigma)$$

which satisfies the following, recursive conditions, where a, b range over the set of states $\mathbb{T}_{\mathsf{s}}$:[8]

(LIL-D1) $\mathrm{den}_{\mathsf{LIL}}(x)(\pi)(a) = \pi(x)$.

(LIL-D2) $\mathrm{den}_{\mathsf{LIL}}(\mathsf{c})(\pi)(a) = c(a)$, as in (4).

(LIL-D3) $\mathrm{den}_{\mathsf{LIL}}(A(B))(\pi)(a) = \Big(\mathrm{den}_{\mathsf{LIL}}(A)(\pi)(a)\Big)\big(\mathrm{den}_{\mathsf{LIL}}(B)(\pi)(a)\big)$.

(LIL-D4) $\mathrm{den}_{\mathsf{LIL}}(\lambda(x)(B))(\pi)(a) = \big(t \mapsto \mathrm{den}_{\mathsf{LIL}}(B)(\pi\{x := t\})(a)\big)$,
where $x : \sigma$ and t ranges over the objects in $\mathbb{T}_\sigma$ (with σ a LIL-type).

(LIL-D5) $\mathrm{den}_{\mathsf{LIL}}(\check{}(A))(\pi)(a) = \Big(\mathrm{den}(A)(\pi)(a)\Big)(a)$.

(LIL-D6) $\mathrm{den}_{\mathsf{LIL}}(\hat{}(A))(\pi)(a) = (b \mapsto \mathrm{den}_{\mathsf{LIL}}(A)(\pi)(b))\ \ (= \mathrm{den}_{\mathsf{LIL}}(A)(\pi))$.

Consider the following four, simple and familiar examples which we will use throughout the paper to illustrate the various notions that we introduce; these are sentences whose denotations are independent of any assignment π, and so we will omit it.

$$\text{John loves her} \xrightarrow{\text{render}} \mathsf{love}(\mathsf{John}, \mathsf{her}) \tag{5}$$

$$\mathrm{den}_{\mathsf{LIL}}\Big(\mathsf{love}(\mathsf{John}, \mathsf{her})\Big)(a) = \mathit{love}(a)(\mathit{John}(a), \mathit{her}(a))$$

$$\text{John loves himself} \xrightarrow{\text{render}} \Big(\lambda(x)\mathsf{love}(x,x)\Big)(\mathsf{John}) \tag{6}$$

$$\begin{aligned}\mathrm{den}_{\mathsf{LIL}}\Big(\big(\lambda(x)\mathsf{love}(x,x)\big)(\mathsf{John})\Big)(a) &= (t \mapsto \mathit{love}(a)(t,t))(\mathit{John}(a))\\ &= \mathit{love}(a)(\mathit{John}(a), \mathit{John}(a))\end{aligned}$$

The President is necessarily American

$$\xrightarrow{\text{render}} \Box\Big(\hat{}(\mathsf{American}(\mathsf{the}(\mathsf{president})))\Big) \tag{7}$$

$$\begin{aligned}&\mathrm{den}_{\mathsf{LIL}}\Big(\Box(\hat{}(\mathsf{American}(\mathsf{the}(\mathsf{president}))))\Big)(a)\\ &\qquad = \mathrm{Nec}(a)\Big(b \mapsto \mathit{American}(b)(\mathit{the}(b)(\mathit{president}(b)))\Big)\end{aligned}$$

$$\text{I was insulted yesterday} \xrightarrow{\text{render}} \mathsf{Yesterday}_1(\hat{}\mathsf{be\_insulted}, \mathsf{I}) \tag{8}$$

$$\begin{aligned}&\mathrm{den}_{\mathsf{LIL}}\Big(\mathsf{Yesterday}_1(\hat{}\mathsf{be\_insulted}, \mathsf{I})\Big)(a)\\ &\quad = \mathit{Yesterday}_1(a)(\mathrm{den}_{\mathsf{LIL}}(\hat{}\mathsf{be\_insulted})(a), \mathrm{den}_{\mathsf{LIL}}(\mathsf{I})(a))\\ &\qquad\qquad = \mathit{Yesterday}_1(a)(b \mapsto \mathit{be\_insulted}(b), I(a))\end{aligned}$$

The temperature is ninety and rising

$$\xrightarrow{\text{render}} \Big(\lambda(x)[\mathsf{ninety}(\check{}x)\,\&\,\mathsf{rise}(x)]\Big)(\hat{}(\mathsf{thetemp})) \qquad (9)$$

For (9), a computation similar to those in the examples above gives the correct, expected denotation.

2.2. The two-sorted, typed, λ-calculus Ty_2

The assumption that every term is interpreted in "the current state" and the lack of state variables are natural enough when we think of the terms of LIL as rendering expressions of natural language, but they are limiting and technically awkward. Both are removed in the two-sorted typed λ-calculus Ty_2, whose characteristic features are that it admits all types as in (2), and interprets terms of type σ by objects in $\mathbb{T}_\sigma$. We fix a set of constants

$$K^G = \{\mathsf{c}^G \mid \mathsf{c} \in K\}$$

in one-to-one correspondence with the constants K of LIL[9]. In accordance with the interpretation (rather than the formal typing) of LIL,

$$\text{if } \mathsf{c} : \sigma, \text{ then } \mathsf{c}^G : (\mathsf{s} \to \sigma) \text{ and } \mathrm{den}(\mathsf{c}^G)(a) = c^G(a) = c(a) \quad (a \in \mathbb{T}_s),$$

i.e., the object c^G in Ty_2 which interprets c^G is exactly the object c which interprets c in LIL. The terms of Ty_2 are defined by the recursion

$$A :\equiv x \mid \mathsf{c}^G \mid A(B) \mid \lambda(x)(B) \qquad (\mathsf{Ty}_2\text{-terms})$$

where now x can be a variable of any type as in (2), and the typing of terms is obvious. Assignments interpret all variables, including those of type s, and denotations are defined naturally:

(Ty_2–D1) $\mathrm{den}(x)(\pi) = \pi(x)$.

(Ty_2–D2) $\mathrm{den}(\mathsf{c}^G)(\pi) = c^G$.

(Ty_2–D3) $\mathrm{den}(A(B))(\pi) = \big(\mathrm{den}(A)(\pi)\big)(\mathrm{den}(B)(\pi))$.

(Ty_2–D4) $\mathrm{den}(\lambda(x)A(x))(\pi) = \big(t \mapsto \mathrm{den}(A)(\pi\{x := t\})\big)$.

We notice the basic property of the Ty_2-typing of terms: for every assignment π,

$$\text{if } A : \sigma, \text{ then } \mathrm{den}(A)(\pi) \in \mathbb{T}_\sigma. \qquad (10)$$

The Gallin translation

For each LIL-term A and each state variable u representing “the current state”, the Gallin translation $A^{G,u}$ of A in Ty_2 is defined by the following recursive clauses:[10]

$$\begin{aligned}
[x]^{G,u} &:\equiv x\\
[\mathsf{c}]^{G,u} &:\equiv \mathsf{c}^G(u)\\
[A(B)]^{G,u} &:\equiv A^{G,u}(B^{G,u})\\
[\lambda(x)(A)]^{G,u} &:\equiv \lambda(x)(A^{G,u})\\
[\check{}A]^{G,u} &:\equiv A^{G,u}(u)\\
[\hat{}A]^{G,u} &:\equiv \lambda(u)A^{G,u}
\end{aligned}$$

By an easy recursion on the LIL-terms, $A^{G,u}$ has the same (LIL-) type as A, and for every assignment π,

$$\text{if } \pi(u) = a, \text{ then } \mathrm{den}(A^{G,u})(\pi) = \mathrm{den}_{\mathsf{LIL}}(A)(\pi)(a).$$

In effect, the Gallin translation expresses formally (within Ty_2) the definition of denotations of LIL.

Here are the Gallin translations of the standard examples above:[11]

$$[\mathsf{love}(\mathsf{John},\mathsf{her})]^{G,u} \equiv \mathsf{love}^G(u)(\mathsf{John}^G(u),\mathsf{her}^G(u)) \tag{11}$$

$$\Big[\lambda(x)\Big(\mathsf{love}(x,x)\Big)(\mathsf{John})\Big]^{G,u} \equiv \Big(\lambda(x)\mathsf{love}^G(u)(x,x)\Big)(\mathsf{John}^G(u)) \tag{12}$$

$$\Big[\Box\Big(\hat{}(\mathsf{American}(\mathsf{the}(\mathsf{president})))\Big)\Big]^{G,u}$$
$$\equiv \Box^G(u)\Big(\lambda(u)\mathsf{American}^G(u)(\mathsf{the}^G(u)(\mathsf{president}^G(u)))\Big) \tag{13}$$

$$\Big[\mathsf{Yesterday}_1(\hat{}\mathsf{be\_insulted},\mathsf{I})\Big]^{G,u}$$
$$\equiv \mathsf{Yesterday}_1{}^G(u)\Big(\lambda(u)\mathsf{be\_insulted}^G(u),\mathsf{I}^G(u)\Big) \tag{14}$$

$$\Big[\Big(\lambda(x)[\mathsf{ninety}(\check{}x)\,\&\,\mathsf{rise}(x)]\Big)\Big(\hat{}(\mathsf{thetemp})\Big)\Big]^{G,u}$$
$$\equiv \Big(\lambda(x)[\mathsf{ninety}^G(u)(x(u))\,\&\,\mathsf{rise}^G(u)(x)]\Big)\Big(\lambda(u)\mathsf{thetemp}^G(u)\Big) \tag{15}$$

Notice that the selected formal variable u occurs both free and bound in these Gallin translations—which may be confusing, but poses no problem.

2.3. The λ-calculus with acyclic recursion $\mathsf{L}^{\lambda}_{\mathrm{ar}}$

We now add to Ty_2 an infinite sequence of *recursion variables* or *locations*

$$\mathsf{p}^{\sigma}_0, \mathsf{p}^{\sigma}_1, \ldots$$

for each type σ. In the semantics of the extended language $\mathsf{L}^{\lambda}_{\mathrm{ar}}$, these will vary over the corresponding universe $\mathbb{T}_{\sigma}$ just as the usual (*pure*) variables $\mathsf{v}^{\sigma}_0, \mathsf{v}^{\sigma}_1, \ldots$, but they will be assigned-to rather than quantified in the syntax, and so they will be treated differently by the semantics. The terms of $\mathsf{L}^{\lambda}_{\mathrm{ar}}$ are defined by the following extension of the recursive definition of the Ty_2-terms:

$$\begin{aligned} A :\equiv x \mid p \mid \mathsf{c}^{G} \mid A(B) \mid \lambda(x)(B) & \\ \mid A_0 \ \mathsf{where}\ \{p_1 := A_1, \ldots, p_n := A_n\} & \quad (\mathsf{L}^{\lambda}_{\mathrm{ar}}\text{-terms}) \end{aligned}$$

where x is a pure variable of any type; p is a location of any type; and the restrictions, typing and denotations are defined exactly as for Ty_2 for all but the last, new *acyclic recursion construct*, where they are as follows.

Acyclic recursive terms

For $A \equiv A_0$ where $\{p_1 := A_1, \ldots, p_n := A_n\}$ to be well-formed, the following conditions must be satisfied:

(i) $p_1, \ldots, p_n$ are distinct locations, such that the type of each p_i is the same as that of the term A_i, and

(ii) the system of *simultaneous assignments* $\{p_1 := A_1, \ldots, p_n := A_n\}$ is *acyclic*, i.e., there are no cycles in the *dependence relation*

$$i \succ j \iff p_j \text{ occurs free in } A_i$$

on the index set $\{1, \ldots, n\}$.

All the occurrences of the locations $p_1, \ldots, p_n$ in the *parts* $A_0, \ldots, A_n$ of A are bound in A, and the type of A is that of its *head* A_0. The *body* of A is the system $\{p_1 := A_1, \ldots, p_n := A_n\}$.

To define the denotation function of a recursive term A, we notice first that by the acyclicity condition, we can assign a number $\mathrm{rank}(p_i)$ to each of the locations in A so that

$$\text{if } p_j \text{ occurs free in } A_i, \text{ then } \mathrm{rank}(p_i) > \mathrm{rank}(p_j).$$

For each assignment π then (which now interprets all pure and recursive variables), we set by induction on $\text{rank}(p_i)$,

$$\overline{p}_i(\pi) = \text{den}(A_i)(\pi\{p_{j_1} := \overline{p}_{j_1}, \dots, p_{j_m} := \overline{p}_{j_m}\}),$$

where $p_{j_1}, \dots, p_{j_m}$ is an enumeration of the locations with $\text{rank}(p_{j_k}) < \text{rank}(p_i)$, $(k = 1, \dots, m)$, and finally,

$$\text{den}(A)(\pi) = \text{den}(A_0)(\pi\{p_1 := \overline{p}_1, \dots, p_n := \overline{p}_n\}).$$

For example, if

$$\begin{aligned} A \equiv \Big(\lambda(x)(p(x)\,\&\,q(x))\Big)(t) \text{ where } \Big\{& p := \lambda(x)\mathsf{ninety}^G(u)(r(x)), \\ & r := \lambda(x)x(u),\ \ q := \lambda(x)\mathsf{rise}^G(u)(x),\ \ t := \lambda(u)\mathsf{thetemp}^G(u)\Big\}, \end{aligned} \tag{16}$$

we can compute $\text{den}(A)(\pi) = \text{den}(A)$ in stages, corresponding to the ranks of the parts, with $a = \pi(u)$:

Stage 1. $\overline{r} = (x \mapsto x(a))$, so $\overline{r}(x) = x(a)$,
$\overline{q} = (x \mapsto \mathit{rise}^G(a)(x)) = \mathit{rise}^G(a)$, so that
$\overline{q}(x) = 1 \iff x$ is rising in state a, and $\overline{t} = \mathit{thetemp}^G$.

Stage 2. $\overline{p} = (x \mapsto \mathit{ninety}^G(a)(\overline{r}(x)))$, so $\overline{p}(x) = 1 \iff x(a) = 90$.

Stage 3. $\text{den}(A) = \overline{p}(\overline{t})\,\&\,\overline{q}(\overline{t})$, so

$$\text{den}(A) = 1 \iff \mathit{thetemp}^G(a) = 90\,\&\,\mathit{rise}^G(a)(\mathit{thetemp}^G).$$

We will use the familiar model-theoretic notation for denotational equivalence,

$$\models A = B \iff \text{ for all assignments } \pi, \text{den}(A)(\pi) = \text{den}(B)(\pi).$$

It is very easy to check that *every* $\mathsf{L}^{\lambda}_{\text{ar}}$*-term A is denotationally equivalent with a* Ty_2*-term A^**, and so $\mathsf{L}^{\lambda}_{\text{ar}}$ is no-more expressive than Ty_2 as far as denotations go; it is, however, *intensionally* more expressive than Ty_2, as we will see.

Congruence

Two $\mathsf{L}^{\lambda}_{\mathrm{ar}}$-terms are congruent if one can be obtained from the other by alphabetic changes of bound variables (of either kind) and re-orderings of the parts in the bodies of recursive subterms, so that, for example, assuming that all substitutions are free,

$$\begin{aligned}&\lambda(x)(A\{z :\equiv x\}) \equiv_c \lambda(y)(A\{z :\equiv y\}),\\ &\qquad A\{p :\equiv q\} \text{ where } \{q := B\{p :\equiv q\}\} \equiv_c A \text{ where } \{p := B\},\\ &\qquad\qquad A \text{ where } \{p := B, q := C\} \equiv_c A \text{ where } \{q := C, p := B\}.\end{aligned}$$

All the syntactic and semantic notions we will define respect congruence, and so it will be convenient on occasion to identify congruent terms.

Since Ty_2 is a sublanguage of $\mathsf{L}^{\lambda}_{\mathrm{ar}}$, we can think of the Gallin translation as an interpretation of LIL into $\mathsf{L}^{\lambda}_{\mathrm{ar}}$; and so we can apply to the terms of LIL the theory of meaning developed for $\mathsf{L}^{\lambda}_{\mathrm{ar}}$ in (Moschovakis 2006), which we describe next.

3. Referential intension theory

The *referential intension* $\mathrm{int}(A)$ of a $\mathsf{L}^{\lambda}_{\mathrm{ar}}$-term A is a mathematical (set-theoretic) object which purports to represent faithfully "the natural algorithm" (process) which "computes" $\mathrm{den}(A)(\pi)$ for each π. It models an intuitive notion of *meaning* for $\mathsf{L}^{\lambda}_{\mathrm{ar}}$-terms (and the natural language expressions which they render), and it provides a precise relation $\approx$ of *synonymy* between terms which can be tested against our intuitions and other theories of meaning that are similarly based on "truth conditions". Roughly:

$$A \approx B \iff \mathrm{int}(A) = \mathrm{int}(B) \quad (A, B \text{ in } \mathsf{L}^{\lambda}_{\mathrm{ar}}), \tag{17}$$

where "A in $\mathsf{L}^{\lambda}_{\mathrm{ar}}$" naturally means that A is a $\mathsf{L}^{\lambda}_{\mathrm{ar}}$-term. To facilitate the discussion of meaning in LIL, we also set

$$A \approx_{\mathsf{LIL}} B \iff A^{G,\mathsf{u}} \approx B^{G,\mathsf{u}} \quad (A, B \text{ in } \mathsf{LIL}). \tag{18}$$

This relation models quite naturally (global) synonymy for terms of LIL.

The operation $A \mapsto \mathrm{int}(A)$ and the relation of referential synonymy are fairly complex, and their precise definitions in (Moschovakis 2006) require the establishment of several technical facts. Here we will confine ourselves to a brief summary of the main results of referential intension theory, primarily so that this article can be read independently of (Moschovakis 2006).

There are two important points to keep in mind.

First, variables and some very simple, *immediate* (variable-like) terms are not assigned referential intensions: they denote *directly* and *immediately*, without the mediation of a meaning. Thus (17) is not exactly right: it holds for *proper* (non-immediate terms), while for immediate terms synonymy coincides with denotational equality or (equivalently for these terms) congruence. The distinction between *direct* and *immediate* reference is precise but not just technical: it lies at the heart of the referential intension approach to modeling meaning, and it plays an important role in our analysis of examples from natural language. We will discuss it in Section 3.2.

Second, the denotational rule of β-conversion

$$\models \Big(\lambda(x)A\Big)(B) = A\{x :\equiv B\}$$

does not preserve referential synonymy, so that, for example,

$$\Big(\lambda(x)\mathsf{love}(x,x)\Big)(\mathsf{John}) \not\approx_{\mathsf{LIL}} \mathsf{love}(\mathsf{John},\mathsf{John}).$$

This is common in structural theories of meaning in which the meaning of a term A codes (in particular) the logical form of A; see (Moschovakis 2006) for a related extensive discussion. It is good to remember this here, especially as we render natural language phrases into LIL and then translate these terms into Ty_2 and so into $\mathsf{L}^{\lambda}_{\mathrm{ar}}$: we want rendering to preserve (intuitive) meaning, so that we have a chance of capturing it with the precisely defined referential intension of the end result, and so we should not lose it by carelessly applying β-conversions in some step of the rendering process.

3.1. Reduction, irreducibility, canonical forms

The main technical tool of (Moschovakis 2006) is a binary relation of *reduction* between $\mathsf{L}^{\lambda}_{\mathrm{ar}}$-terms, for which (intuitively)

$$\begin{aligned} A \Rightarrow B \iff\ & A \equiv_c B \\ & \text{or } A \text{ and } B \text{ have the same meaning} \\ & \quad \text{and } B \text{ expresses that meaning “more simply”}. \end{aligned}$$

The disjunction is needed because the reduction relation is defined for all pairs of terms, even those which do not have a meaning, for which, however, the relation is trivial. We set

$$A \text{ is irreducible} \iff \text{for all } B, \text{ if } A \Rightarrow B, \text{ then } A \equiv_c B, \tag{19}$$

so that the irreducible terms which have meaning, express that meaning "as simply as possible".

Theorem 1 (Canonical form) *For each term A, there is a unique* (up to congruence) *recursive, irreducible term*

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

such that $A \Rightarrow \mathrm{cf}(A)$. We write

$$A \Rightarrow_{\mathrm{cf}} B \iff B \equiv_c \mathrm{cf}(A).$$

If $A \Rightarrow B$, then $\models A = B$, and, in particular, $\models A = \mathrm{cf}(A)$.

The reduction relation is determined by ten, simple *reduction rules* which comprise the *Reduction Calculus*, and the computation of $\mathrm{cf}(A)$ is effective. The parts A_i of $\mathrm{cf}(A)$ are *explicit*[12], *irreducible terms*; they are determined uniquely (up to congruence) by A; and they code the basic facts which are needed to compute the denotation of A, in the assumed fixed interpretation of the language. If $A : \mathsf{t}$ and $\mathrm{den}(A) = 1$, then the irreducible parts of $\mathrm{cf}(A)$ can be viewed as the *truth conditions* which ground the truth of A.

Variables and constants are irreducible, and so is the more complex-looking term $\lambda(x)\mathsf{love}^G(u)(x,x)$. On the other hand, the term expressing John's self-love in the current state is not:

$$\begin{aligned}\Big(\lambda(x)\mathsf{love}^G(u)(x,x)\Big)&(\mathsf{John}^G(u))\\ &\Rightarrow_{\mathrm{cf}} \Big(\lambda(x)\mathsf{love}^G(u)(x,x)\Big)(j) \text{ where } \{j := \mathsf{John}^G(u)\}.\end{aligned} \quad (20)$$

For a more complicated example, the canonical form of the Gallin translation of the Partee term in (15) is the term (16). So canonical forms get very complex, as do their explicit, irreducible parts—which is not surprising, since they are meant to express directly the meanings of complex expressions.

The specific rules of the Reduction Calculus are at the heart of the matter, of course, and they deliver the subtle differences in (formal) meaning with which we are concerned here. It is not possible to state or explain them in this article—they are the main topic of (Moschovakis 2006); but the most important of them will be gleaned from their applications in the examples below.

3.2. Direct vs. immediate reference

An important role in the computation of canonical forms is played by the *immediate terms*. These are defined by

$$X :\equiv v \mid p \mid p(\vec{v}) \mid \lambda(\vec{u})p(\vec{v}), \qquad \text{(Immediate terms)}$$

where $\vec{v} = (v_1, \dots, v_n), \vec{u} = (u_1, \dots, u_m)$ and $v, v_1, \dots, v_n, u_1, \dots, u_m$ are pure variables, while p is a location. Immediate terms are treated like variables in the Reduction Calculus; this is not true of constants (and other irreducible terms) which contribute in a non-trivial way to the canonical forms of the terms in which they occur. For example, $\mathsf{run}^G(u)(p(v))$ is irreducible, because $p(v)$ is immediate, while $\mathsf{run}^G(u)(\mathsf{John}^G(u))$ is not:

$$\mathsf{run}^G(u)(\mathsf{John}^G(u)) \Rightarrow_{\text{cf}} \mathsf{run}^G(u)(j) \text{ where } \{j := \mathsf{John}^G(u)\}.$$

In the intensional semantics of $\mathsf{L}^{\lambda}_{\text{ar}}$ to which we will turn next, immediate terms refer *directly* and *immediately*: they are not assigned meanings, and they contribute only their reference to the meaning of larger (proper) terms which contain them. Irreducible terms also refer directly, in the sense that their meaning is completely determined by their reference; but they are assigned meanings, and they affect in a non-trivial (structural) way the meanings of larger terms which contain them.

3.3. Referential intensions

If A is not immediate and

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

then $\text{int}(A)$ is the abstract algorithm which intuitively computes $\text{den}(A)(\pi)$ for each assignment π as indicated in the remarks following (16), as follows:

(i) Solve the system of equations

$$d_i = \text{den}(A_i)(\pi\{p_1 := d_1, p_2 := d_2, \dots, p_n := d_n\}) \quad (i = 1, \dots, n),$$

(which, easily, has unique solutions by the acyclicity hypothesis).

(ii) If the solutions are $\overline{p}_1, \dots, \overline{p}_n$, set

$$\text{den}(A)(\pi) = \text{den}(A_0)(\pi\{p_1 := \overline{p}_1, \dots, p_n := \overline{p}_n\}).$$

So how can we define precisely this “abstract algorithm”? The idea is that it must be determined completely by the head of A and the system of equations in its body, and it should not depend on any particular method of solving the system; so it is most natural to simply identify it with the tuple of functions

$$\mathrm{int}(A) = (f_0, f_1, \dots, f_n) \tag{21}$$

defined by the parts of A, i.e.,

$$f_i(d_1, \dots, d_n, \pi) = \mathrm{den}(A_i)(\pi\{p_1 := d_1, p_2 := d_2, \dots, p_n := d_n\}) \quad (i \le n).$$

Tuples of functions such as (21) are called *recursors*.

For a concrete example, which also illustrates just how abstract this notion of meaning is, the referential intension of the Partee example (15) is determined by its canonical form $A^{G,u}$ in (16), and it is the recursor

$$\mathrm{int}(A) = (f_0, f_1, f_2, f_3, f_4),$$

where

$$\begin{aligned}
f_0(p,r,q,t,\pi) &= \big(x \mapsto (p(x) \,\&\, q(x))(t)\big),\\
f_1(p,r,q,t,\pi) &= \big(x \mapsto \mathit{ninety}^G(\pi(u))(r(x))\big),\\
f_2(p,r,q,t,\pi) &= \big(x \mapsto x(\pi(u))\big),\\
f_3(p,r,q,t,\pi) &= \big(x \mapsto \mathit{rise}^G(\pi(u))(x)\big),\\
f_4(p,r,q,t,\pi) &= \mathit{thetemp}^G.
\end{aligned}$$

Theorem 2 (Compositionality) *The operation $A \mapsto \mathrm{int}(A)$ on proper (not immediate) terms is compositional, i.e., $\mathrm{int}(A)$ is determined from the referential intensions of the proper subterms of A and the denotations of its immediate subterms.*

This does not follow directly from the definition of referential intensions that we gave above, via canonical forms, but it is not difficult to prove.

3.4. Referential synonymy

Two terms A and B are *referentially synonymous* if either $A \equiv_c B$, or $\mathrm{int}(A)$ and $\mathrm{int}(B)$ are *naturally isomorphic*. Now this is tedious to make precise, but, happily, we don’t need to do this here because of the following

Theorem 3 (Referential Synonymy) *For any two terms A, B of $\mathsf{L}^{\lambda}_{\mathrm{ar}}$, A is referentially synonymous with B if and only if there exist suitable terms $A_0, \ldots$, $B_0, \ldots$ such that*

$$A \Rightarrow_{\mathrm{cf}} A_0 \text{ where } \{p_1 := A_1, \ldots, p_n := A_n\},$$
$$B \Rightarrow_{\mathrm{cf}} B_0 \text{ where } \{p_1 := B_1, \ldots, p_n := B_n\},$$

and for $i = 0, 1, \ldots, n, \models A_i = B_i$, i.e., for all π, $\mathrm{den}(A_i)(\pi) = \mathrm{den}(B_i)(\pi)$.

Thus the referential synonymy relation $A \approx B$ is grounded by a system of denotational identities between explicit, irreducible terms. It is important, of course, that the formal identities

$$A_i = B_i, \qquad i = 0, \ldots, n$$

can be computed from A and B (using the Reduction Calculus), although their truth or falsity depends on the assumed, fixed structure of interpretation and cannot, in general, be decided effectively.

4. Two notions of situated meaning

We can now make precise the two, promised notions of situated meaning for terms of LIL, after just a bit more preparation.

State parameters

Intuitively, a notion of "situated meaning" of a LIL-term $A : \tau$ is a way that we understand A in a given state a; and so it depends on a, even when A is closed, when its semantic values do not depend on anything else. To avoid the cumbersome use of assignments simply to indicate this state dependence, we introduce a *parameter* $\overline{a}$ for each state a, so that the definition of the terms of $\mathsf{L}^{\lambda}_{\mathrm{ar}}$ now takes the following form:

$$A :\equiv x \mid \overline{a} \mid p \mid \mathsf{c}^G \mid A(B) \mid \lambda(x)(B)$$
$$\mid A_0 \text{ where } \{p_1 := A_1, \ldots, p_n := A_n\} \quad (\mathsf{L}^{\lambda}_{\mathrm{ar}}\text{-terms})$$

Parameters are treated like free pure variables in the definition of immediate terms and in the Reduction Calculus; in fact, the best way to think of $\overline{a}$ is as a free variable with *preassigned* value

$$\mathrm{den}(\overline{a})(\pi) = a$$

which does not depend on the assignment π.

4.1. Factual content

The term

$$A^{G,\bar{a}} \equiv A^{G,u}\{u :\equiv \bar{a}\} \tag{22}$$

expresses the Gallin translation of A at the state a—not only its denotation, but also its meaning or at least one aspect of it. Thus, for each proper LIL-term A and each state a, we set

$$\mathrm{FC}(A,a) = \mathrm{int}\big(A^{G,\bar{a}}\big). \tag{23}$$

This is the (referential) *factual content* of A at the state a. For proper terms A, B and states a, b, we also set[13]

$$\begin{aligned}(A,a) \text{ is } \textit{factually synonymous} \text{ with } (B,b) \iff {} & \mathrm{FC}(A,a) = \mathrm{FC}(B,b) \\ \iff {} & A^{G,\bar{a}} \approx B^{G,\bar{b}}.\end{aligned}$$

By the Referential Synonymy Theorem 3 then, we can read factual synonymy by examining the canonical forms of terms. For example,

$$\begin{aligned}[\text{John loves her}]^{G,\bar{a}} & \xrightarrow{\text{render}} \mathsf{love}^G(\bar{a})(\mathsf{John}^G(\bar{a}), \mathsf{her}^G(\bar{a})) \\ & \Rightarrow_{\mathrm{cf}} \mathsf{love}^G(\bar{a})(j,h) \text{ where } \{j := \mathsf{John}^G(\bar{a}), h := \mathsf{her}^G(\bar{a})\}\end{aligned}$$

and

$$\begin{aligned}[\text{John loves Mary}]^{G,\bar{a}} & \xrightarrow{\text{render}} \mathsf{love}^G(\bar{a})(\mathsf{John}^G(\bar{a}), \mathsf{Mary}^G(\bar{a})) \\ & \Rightarrow_{\mathrm{cf}} \mathsf{love}^G(\bar{a})(j,h) \text{ where } \{j := \mathsf{John}^G(\bar{a}), h := \mathsf{Mary}^G(\bar{a})\},\end{aligned}$$

so that

$$\begin{aligned}& \text{if } \mathrm{den}\big(\mathsf{Mary}^G(\bar{a})\big) = \mathrm{den}\big(\mathsf{her}^G(\bar{a})\big), \\ & \qquad \text{then } [\text{John loves her}]^{G,\bar{a}} \approx [\text{John loves Mary}]^{G,\bar{a}},\end{aligned}$$

which expresses formally the fact that ‘John loves her’ and ‘John loves Mary’ convey the same “information” about the world at this state a. These two sentences are not, of course, synonymous, as it is easy to verify by the definition of $\approx_{\mathsf{LIL}}$ in (18) and Theorem 3.

Next consider example (8) which involves the indexical ‘Yesterday’. In (Frege 1918), Frege argues that

If someone wants to say today what he expressed yesterday using the word 'today', he will replace this word with 'yesterday'. Although the thought is the same, its verbal expression must be different in order that the change of sense which would otherwise be effected by the differing times of utterance may be cancelled out.

It appears that Frege's "thought" in this case is best modeled by the factual content of the uttered sentence in the relevant state.

In detail, suppose that at state a the speaker is DK and the time is 27 June 2005. If we consider the sentence 'I am insulted today' uttered at a state $b = a^-$ when the time is 26 June 2005, the speaker is again DK and nothing else has changed, then, according to Frege's remark above, it should be that

$$\big[\text{I was insulted yesterday}\big]^{G,\bar{a}} \approx \big[\text{I am insulted today}\big]^{G,\bar{b}}.$$

This is indeed the case:

$$\begin{aligned}
&\big[\text{I was insulted yesterday}\big]^{G,\bar{a}}\\
&\qquad\xrightarrow{\text{render}} \mathsf{Yesterday}_1{}^G(\bar{a})\big(\lambda(u)\mathsf{be\_insulted}^G(u), \mathsf{I}^G(\bar{a})\big)\\
&\Rightarrow_{\text{cf}} \mathsf{Yesterday}_1{}^G(\bar{a})(p,q) \text{ where } \{p := \lambda(u)\mathsf{be\_insulted}^G(u), q := \mathsf{I}^G(\bar{a})\}
\end{aligned}$$

$$\begin{aligned}
&\big[\text{I am insulted today}\big]^{G,\bar{b}} \xrightarrow{\text{render}} \mathsf{Today}_1{}^G(\bar{b})\big(\lambda(v)\mathsf{be\_insulted}^G(v), \mathsf{I}^G(\bar{b})\big)\\
&\quad\Rightarrow_{\text{cf}} \mathsf{Today}_1{}^G(\bar{b})(p,q) \text{ where } \{p := \lambda(v)\mathsf{be\_insulted}^G(v), q := \mathsf{I}^G(\bar{b})\},
\end{aligned}$$

and the canonical forms of these sentences at these states satisfy the conditions of Theorem 3 for synonymy—assuming, of course, that $\mathsf{Yesterday}_1$ and Today_1 are interpreted in the natural way, so that for these a and b,

$$\mathit{Yesterday}_1(a)(p)(x) \iff \mathit{Today}_1(b)(p)(x) \quad (p : \mathbb{T}_s \to (\mathbb{T}_\mathsf{e} \to \mathbb{T}_\mathsf{t}), x \in \mathbb{T}_\mathsf{e}).$$

On the other hand, example (7) shows that, in some cases, the factual content is independent of the state and incorporates the full meaning of the term:

$$\begin{aligned}
&[\text{The President is necessarily American}]^{G,\bar{a}}\\
&\qquad\xrightarrow{\text{render}} \Box^G(\bar{a})\Big(\lambda(u)\mathsf{American}^G(u)(\mathsf{the}^G(u)(\mathsf{president}^G(u)))\Big)\\
&\qquad\quad\Rightarrow_{\text{cf}} \Box^G(\bar{a})(q) \text{ where } \{q := \lambda(u)\mathsf{American}^G(u)(t(u)),\\
&\qquad\qquad\qquad t := \lambda(u)\mathsf{the}^G(u)(p(u)), p := \lambda(u)\mathsf{president}^G(u)\}.
\end{aligned}$$

Notice that the state parameter $\bar{a}$ occurs only in the head of the relevant canonical form, and so, with the "necessarily always" interpretation of $\Box$ that we have adopted, the factual content of this term is independent of the state a.

4.2. Referential (global) meaning

A plausible candidate for the (global) *referential meaning* of a LIL-term A is the operation

$$a \mapsto \mathrm{int}\big(A^{G,\bar{a}}\big)$$

which assigns to each state a the factual content of A at a. We can understand this outside the formal system, as an operation from states to recursors; but we can also do it within the system, taking advantage of the abstraction construct of the typed λ-calculus and setting

$$\mathrm{M}(A) = \mathrm{int}\big(\lambda(u)A^{G,u}\big). \tag{24}$$

It follows by Theorem 3 and the Reduction Calculus that for proper terms A, B,

$$\mathrm{M}(A) = \mathrm{M}(B) \iff \lambda(u)A^{G,u} \approx \lambda(u)B^{G,u} \iff A \approx_{\mathsf{LIL}} B,$$

and so there is no conflict between this notion of global meaning and the referential synonymy relation between LIL-terms defined directly in terms of the Gallin translation.

The recursor $\mathrm{M}(A)$ is expressed directly by the canonical form of $\lambda(u)A^{G,u}$, which gives some insight into this notion of formal meaning. For example:

$$\begin{aligned}
\lambda(u)[\text{John loves her}]^{G,u} &\xrightarrow{\text{render}} \lambda(u)\mathsf{love}^G(u)(\mathsf{John}^G(u), \mathsf{her}^G(u))\\
\Rightarrow_{\mathrm{cf}} \lambda(u)\mathsf{love}^G(u)(j(u), h(u)) &\ \mathsf{where}\ \{j := \lambda(u)\mathsf{John}^G(u), h := \lambda(u)\mathsf{her}^G(u)\}\\
\approx \lambda(u)\mathsf{love}^G(u)(j(u), h(u)) &\ \mathsf{where}\ \{j := \mathsf{John}^G, h := \mathsf{her}^G\}
\end{aligned}$$

while

$$\begin{aligned}
\lambda(u)[\text{John loves Mary}]^{G,u} &\xrightarrow{\text{render}} \lambda(u)\mathsf{love}^G(u)(\mathsf{John}^G(u), \mathsf{Mary}^G(u))\\
\Rightarrow_{\mathrm{cf}} \lambda(u)\mathsf{love}^G(u)(j(u), h(u)) &\ \mathsf{where}\ \{j := \lambda(u)\mathsf{John}^G(u),\\
&\qquad h := \lambda(u)\mathsf{Mary}^G(u)\}\\
\approx \lambda(u)\mathsf{love}^G(u)(j(u), h(u)) &\ \mathsf{where}\ \{j := \mathsf{John}^G, h := \mathsf{Mary}^G\}.
\end{aligned}$$

To "grasp" the meanings of these two sentences, as Frege would say, we need the functions *love*, *John*, *Mary* and *her*—not their values in any one, particular state, but their range of values in all states; and to realize that they are not synonymous, we need only realize that 'her' is not 'Mary' in all states.

4.3. Local meaning

Once we have a global meaning of *A*, we can compute its *local meaning* at a state *a* by evaluation, and, again, we could do this outside the system by defining in a natural way an operation of application of a recursor to an argument; but since we already have application in the typed λ-calculus, we set, within the system,

$$\mathrm{LM}(A,a) = \mathrm{int}\big((\lambda(u)A^{G,u})(\overline{a})\big). \tag{25}$$

This is the (referential) *local meaning* of *A* at *a*. For proper terms A,B and states a,b, we set

$$\begin{aligned}(A,a) \text{ is } \textit{locally synonymous} \text{ with } (B,b) &\iff \mathrm{LM}(A,a) = \mathrm{LM}(B,b)\\ &\iff \Big(\lambda(u)A^{G,u}\Big)(\overline{a}) \approx \Big(\lambda(v)B^{G,v}\Big)(\overline{b}).\end{aligned}$$

It is important to recall here that, in general,

$$\Big(\lambda(u)C^{G,u}\Big)(\overline{a}) \not\approx C^{G,\overline{a}},$$

because β-conversion does not preserve referential synonymy.

The three synonymy relations we have defined are related as one would expect:

Lemma 1 (a) *Referential synonymy implies local synonymy at any state, that is*

$$\lambda(u)A^{G,u} \approx \lambda(u)B^{G,u} \Longrightarrow \big(\lambda(u)A^{G,u}\big)(\overline{a}) \approx \big(\lambda(u)B^{G,u}\big)(\overline{a})$$

(b) *Local synonymy at a state implies factual synonymy at that state,*

$$\big(\lambda(u)A^{G,u}\big)(\overline{a}) \approx \big(\lambda(u)B^{G,u}\big)(\overline{a}) \Longrightarrow A^{G,\overline{a}} \approx B^{G,\overline{a}}.$$

Both parts of the lemma are easily proved using Theorem 3 and some simple denotational equalities between the parts of the relevant canonical forms.

In the following sections, we consider some examples which (in particular) show that neither part of the Lemma has a valid converse. Perhaps most interesting are those which distinguish between factual and local synonymy, and show that the latter is a much more fine-grained relation, very close in fact to (global) referential synonymy.

4.4. Factual content vs. local meaning

In Section 4.1, we showed that for any state a,

$$\text{if } her(a) = Mary(a), \text{ then [John loves her]}^{G,\bar{a}} \approx \text{[John loves Mary]}^{G,\bar{a}}.$$

To check for local synonymy, we compute the canonical forms of these terms:

$$\begin{aligned}
&\Big(\lambda(u)\text{[John loves her]}^{G,u}\Big)(\bar{a}) \\
&\quad\xrightarrow{\text{render}} \Big(\lambda(u)\mathsf{love}^G(u)(\mathsf{John}^G(u),\mathsf{her}^G(u))\Big)(\bar{a}) \\
&\qquad\Rightarrow_{\text{cf}} \Big(\lambda(u)\mathsf{love}^G(u)(j(u),h(u))\Big)(\bar{a}) \\
&\qquad\qquad \mathsf{where}\ \{j := \lambda(u)\mathsf{John}^G(u), h := \lambda(u)\mathsf{her}^G(u)\} \\
&\qquad\approx \mathsf{love}^G(\bar{a})(j(\bar{a}),h(\bar{a}))\ \mathsf{where}\ \{j := \mathsf{John}^G, h := \mathsf{her}^G\}
\end{aligned}$$

while

$$\begin{aligned}
&\Big(\lambda(u)\text{[John loves Mary]}^{G,u}\Big)(\bar{a}) \\
&\quad\xrightarrow{\text{render}} \Big(\lambda(u)\mathsf{love}^G(u)(\mathsf{John}^G(u),\mathsf{Mary}^G(u))\Big)(\bar{a}) \\
&\qquad\Rightarrow_{\text{cf}} \Big(\lambda(u)\mathsf{love}^G(u)(j(u),h(u))\Big)(\bar{a}) \\
&\qquad\qquad \mathsf{where}\ \{j := \lambda(u)\mathsf{John}^G(u), h := \lambda(u)\mathsf{Mary}^G(u)\} \\
&\qquad\approx \mathsf{love}^G(\bar{a})(j(\bar{a}),h(\bar{a}))\ \mathsf{where}\ \{j := \mathsf{John}^G, h := \mathsf{Mary}^G\}
\end{aligned}$$

But $her^G \neq Mary^G$, and so these two sentences are not locally synonymous at a—although they have the same factual content at a.

The example illustrates the distinction between factual content and local meaning: to grasp the factual content $\text{FC}(\text{John loves her}, a)$ we only need know who 'her' is at state a; on the other hand, to grasp the local meaning $\text{LM}(\text{John loves her}, a)$ we need to understand 'her' as a function on the states. This is what we also need in order to grasp the (global) referential meaning of 'John loves her', which brings us to the more difficult comparison between local and global meaning.

4.5. Local vs. global synonymy

By the Reduction Calculus, if

$$A^{G,u} \Rightarrow_{\text{cf}} A_0\ \mathsf{where}\ \{p_1 := A_1, \ldots, p_n := A_n\}$$

then

$$\begin{aligned}\lambda(u)A^{G,u} \Rightarrow_{\text{cf}} \quad & \lambda(u)(A_0\{p_1 :\equiv q_1(u), \dots, p_n :\equiv q_n(u)\}) \\ & \text{where } \big\{q_1 := \lambda(u)A_1\{p_1 :\equiv q_1(u), \dots, p_1 :\equiv q_n(u)\}, \\ & \qquad \vdots \\ & \qquad q_n := \lambda(u)A_n\{p_1 :\equiv q_1(u), \dots, p_n :\equiv q_n(u)\}\big\}\end{aligned}$$

and

$$\begin{aligned}\big(\lambda(u)A^{G,u}\big)(\overline{a}) \Rightarrow_{\text{cf}} \quad & \big(\lambda(u)(A_0\{p_1 :\equiv q_1(u), \dots, p_n :\equiv q_n(u)\})\big)(\overline{a}) \\ & \text{where } \big\{q_1 := \lambda(u)A_1\{p_1 :\equiv q_1(u), \dots, p_1 :\equiv q_n(u)\}, \\ & \qquad \vdots \\ & \qquad q_n := \lambda(u)A_n\{p_1 :\equiv q_1(u), \dots, p_n :\equiv q_n(u)\}\big\}\end{aligned}$$

The computations here are by the most complex—and most significant—λ-*rule* of the Reduction Calculus, which, unfortunately, we cannot attempt to motivate here. The formulas do imply, however, that for any term B,

$$\big(\lambda(u)A^{G,u}\big)(\overline{a}) \approx \big(\lambda(u)B^{G,u}\big)(\overline{a})$$

if and only if

$$B^{G,u} \Rightarrow_{\text{cf}} B_0 \text{ where } \{p_1 := B_1, \dots, p_n := B_n\},$$

for suitable $B_0, \dots, B_n$, so that:
For any $i = 1, \dots, n$, (1)

$$\begin{aligned}\models \lambda(u)(A_i\{p_1 :\equiv q_1(u), \dots, p_n :\equiv q_n(u)\}) \\ = \lambda(u)(B_i\{p_1 :\equiv q_1(u), \dots, p_n :\equiv q_n(u)\}),\end{aligned}$$

and

$$\begin{aligned}\models A_0\{u :\equiv \overline{a}\}\{p_1 :\equiv q_1(\overline{a}), \dots, p_n :\equiv q_n(\overline{a})\} \\ = B_0\{u :\equiv \overline{a}\}\{p_1 :\equiv q_1(\overline{a}), \dots, p_n :\equiv q_n(\overline{a})\}. \quad (2)\end{aligned}$$

On the other hand, $A \approx_{\text{LIL}} B$ if (1) holds and instead of (2) the stronger

$$\begin{aligned}\models \lambda(u)(A_0\{p_1 :\equiv q_1(u), \dots, p_n :\equiv q_n(u)\}) \\ = \lambda(u)(B_0\{p_1 :\equiv q_1(u), \dots, p_1 :\equiv q_n(u)\}) \quad (2^*)\end{aligned}$$

is true.

Thus, local synonymy is very close to global synonymy, the only difference being that for global synonymy we need the heads of the two terms to be denotationally equal for all states, while for local synonymy at a state *a*, we only need their heads to be denotationally equal at *a*. This explains why, by Lemma 1, the former implies the latter while the converse may fail.

Natural examples which illustrate this distinction are hard to find, but the following one may, at least, be amusing.

Consider a particular state *a* at which two common nouns are co-extensive – for example, 'man' and 'human'. This was the case at the time just after God had created Adam but not yet Eve. At that state *a*, then, the sentences 'Adam is a man' and 'Adam is a human' are locally synonymous, since

$$\begin{aligned}&\Big(\lambda(u)[\text{Adam is a man}]^{G,u}\Big)(\overline{a})\\ &\qquad\xrightarrow{\text{render}} \Big(\lambda(u)\mathsf{man}^G(u)(\mathsf{Adam}^G(u))\Big)(\overline{a})\\ &\qquad\Rightarrow_{\text{cf}} \Big(\lambda(u)\mathsf{man}^G(u)(j(u))\Big)(\overline{a}) \text{ where } \{j := \lambda(u)\mathsf{Adam}^G(u)\}\end{aligned}$$

$$\begin{aligned}&\Big(\lambda(u)[\text{Adam is human}]^{G,u}\Big)(\overline{a})\\ &\qquad\xrightarrow{\text{render}} \Big(\lambda(u)\mathsf{human}^G(u)(\mathsf{Adam}^G(u))\Big)(\overline{a})\\ &\qquad\Rightarrow_{\text{cf}} \Big(\lambda(u)\mathsf{human}^G(u)(j(u))\Big)(\overline{a}) \text{ where } \{j := \lambda(u)\mathsf{Adam}^G(u)\}\end{aligned}$$

and

$$\models \mathsf{man}^G(\overline{a})(j(\overline{a})) = \mathsf{human}^G(\overline{a})(j(\overline{a})).$$

These sentences, of course, are not referentially synonymous, as they are not even factually synonymous in any reasonable state.

4.6. Local synonymy across different states

Things get more complicated when we try to trace local synonymy between sentences at different states. In Section 4.1, it was shown that 'Yesterday I was insulted' uttered at a state *a* where the time is 27 June 2005 and 'Today I am insulted' uttered at state *b* where the time is 26 June 2005 are factually synonymous, provided that the two states are otherwise identical. By the Reduction Calculus,

$$\begin{aligned}
&\Big(\lambda(u)\big[\text{Yesterday I was insulted}\big]^{G,u}\Big)(\overline{a}) \\
&\quad\xrightarrow{\text{render}} \Big(\lambda(u)\mathsf{Yesterday_1}^G(u)\big(\lambda(u)\mathsf{be\_insulted}^G(u),\mathsf{I}^G(u)\big)\Big)(\overline{a}) \\
&\qquad\Rightarrow_{\text{cf}} \Big(\lambda(u)\mathsf{Yesterday_1}^G(u)(p(u),q(u))\Big)(\overline{a}) \text{ where} \\
&\qquad\qquad \{p := \lambda(u)\lambda(u)\mathsf{be\_insulted}^G(u), q := \lambda(u)\mathsf{I}^G(u)\}
\end{aligned}$$

and

$$\begin{aligned}
&\Big(\lambda(u)\big[\text{Today I am insulted}\big]^{G,u}\Big)(\overline{b}) \\
&\quad\xrightarrow{\text{render}} \Big(\lambda(u)\mathsf{Today_1}^G(u)\big(\lambda(u)\mathsf{be\_insulted}^G(u),\mathsf{I}^G(u)\big)\Big)(\overline{b}) \\
&\qquad\Rightarrow_{\text{cf}} \Big(\lambda(u)\mathsf{Today_1}^G(u)(p(u),q(u))\Big)(\overline{b}) \text{ where} \\
&\qquad\qquad \{p := \lambda(u)\lambda(u)\mathsf{be\_insulted}^G(u), q := \lambda(u)\mathsf{I}^G(u)\},
\end{aligned}$$

and these two canonical forms have the same bodies, so they will be locally synonymous if and only if their heads are denotationally equal. But

$$\not\models \mathsf{Yesterday_1}^G(\overline{a})(p(\overline{a}),q(\overline{a})) = \mathsf{Today_1}^G(\overline{b})(p(\overline{b}),q(\overline{b}))$$

on the possible assumption that John is running while Mary sleeps (today and yesterday), taking

$$\pi(p)(a) = \pi(p)(b) = \text{runs}(a), \quad \pi(q)(a) = \text{John}, \quad \pi(q)(b) = \text{Mary}$$

and computing the denotations of the two sides of this equation for the assignment π; so

$$\text{LM}(\text{Yesterday I was insulted}, a) \neq \text{LM}(\text{Today I am insulted}, b).$$

This argument is a little unusual, and it may be easier to understand in the next example, where we compare the local meanings of the same sentence, with no modal operator, on two different but nearly identical states. Consider 'John runs', in *a* and *b*:

$$\begin{aligned}
&\Big(\lambda(u)\mathsf{run}^G(u)(\mathsf{John}^G(u))\Big)(\overline{a}) \\
&\qquad\Rightarrow_{\text{cf}} \Big(\lambda(u)\mathsf{run}^G(u)(j(u))\Big)(\overline{a}) \text{ where } \{j := \lambda(u)\mathsf{John}^G(u))\}
\end{aligned}$$

$$\Big(\lambda(u)\mathsf{run}^G(u)(\mathsf{John}^G(u))\Big)(\overline{b})$$
$$\Rightarrow_{\mathrm{cf}} \Big(\lambda(u)\mathsf{run}^G(u)(j(u))\Big)(\overline{b}) \text{ where } \{j := \lambda(u)\mathsf{John}^G(u))\}$$

Local synonymy requires that

$$\models \mathsf{run}^G(\overline{a})(j(\overline{a})) = \mathsf{run}^G(\overline{b})(j(\overline{b})), \tag{26}$$

which means that for all functions $j : \mathbb{T}_{\mathsf{s}} \to \mathbb{T}_{\mathsf{e}}$,

$$run^G(a)(j(a)) = run^G(b)(j(b)).$$

Suppose further that the two states *a* and *b* are exactly the same except that the speaker is different—an aspect of the state that, intuitively, should not affect the meaning of 'John runs'. In particular, the interpretation $run^G : \mathsf{s} \to (\mathsf{e} \to \mathsf{t})$ of the constant run^G is the same in the two states, i.e., that whoever runs at state *a* also runs at state *b* and conversely. But unless either everyone or nobody runs in these states, (26) fails: just take an assignment π such as $\pi(j)(a) \neq \pi(j)(b)$ and such that in both states $\pi(j)(a)$ runs whereas $\pi(j)(b)$ does not run.

The examples suggest that synonymy across different states is a complex relation, and that when we feel intuitively that sentences may have the same "meaning" in different states, it is factual synonymy that we have in mind.

5. Situated meaning in the philosophy of language

In this section we will investigate briefly the connection of the proposed aspects of situated meaning to indexicality, propositional attitudes and translation.

5.1. Kaplan's treatment of indexicals

Indexicality is a phenomenon of natural language usage, closely connected to situated meaning. Among its many proposed treatments, Kaplan's theory of direct reference (Kaplan 1989) reaches some very interesting results which we can relate to the aspects of situated meaning introduced in this paper.

Kaplan's theory is expressed formally in the Logic of Demonstratives (LD), where each term or formula has two semantic values, *Content* and *Character*. The Content of a term or a formula is given with respect to a context, considered as *context of utterance*, and it is a function from possible circumstances, considered as *contexts of evaluation*, to denotations or truth values, respectively. The Character of a term or a formula A is the function which assigns to each context the Content of A in that context.

In (Kaplan 1978), it is argued that

> Thus when I say
>
> I was insulted yesterday (K4)
>
> specific content–*what I said*–is expressed. Your utterance of the same sentence, or mine on another day, would not express the same content. What is important to note is that it is not just the truth value that may change; what is said is itself different. Speaking today, my utterance of (K4) will have a content roughly equivalent to that which
>
> David Kaplan is insulted on 20 April 1973 (K5)
>
> would have, spoken by you or anyone at any time.

Kaplan gives formal definitions of these notions in LD, from which it follows that at a context of utterance where the speaker is David Kaplan and the time is 21 April 1973, the two sentences (K4) and (K5) have the same Content, that is the same truth value for every possible circumstance: but, of course, they have different Characters—(K5)'s Character is a constant function, whereas (K4)'s clearly depends on the context of utterance.

In $\mathsf{L}^{\lambda}_{\mathrm{ar}}$, these intuitively plausible semantic distinctions can be made with the use of the factual content and the global meaning which, roughly speaking, correspond to Kaplan's Content and Character respectively.

Suppose a is a state[14] where the speaker is David Kaplan (or DK for short) and the time is 21 April 1973. As example (8) suggests, (K4) and (K5) are factually synonymous at a, that is

$$\big[\text{I was insulted yesterday}\big]^{G,\bar{a}} \approx \big[\text{David Kaplan is insulted on 20 April 1973}\big]^{G,\bar{a}}.$$

Moreover, for any state b where $\mathit{DavidKaplan}^G(b)$ is David Kaplan,

$$\big[\text{I was insulted yesterday}\big]^{G,\bar{a}} \approx \big[\text{David Kaplan is insulted on 20 April 1973}\big]^{G,\bar{b}}.$$

These two sentences, however, are not referentially (globally) synonymous,

$$\lambda(u)\big[\text{I was insulted yesterday}\big]^{G,u} \\ \not\approx \lambda(u)\big[\text{David Kaplan is insulted on 20 April 1973}\big]^{G,u},$$

since I^G is not denotationally equivalent with $\mathsf{DavidKaplan}^G$ nor is $\mathsf{Yesterday}_1{}^G$ with $\mathsf{on20April1973}^G$. Notice that the indexical 'I' occurs within the scope of the modal operator 'Yesterday' in (K4), and in any such example in order to account for the directly referential usage of indexicals, we choose the de-re reading of the operators, thus translating 'Yesterday' as $\mathsf{Yesterday}_1$.

Kaplan argues that (K4) has another characteristic as well. Consider two contexts c_1 and c_2 which differ with respect to agent and/or moment of time—that is, the aspects of the context of utterance which are relevant to this particular sentence. Then, its Content with respect to c_1 is different with its Content with respect to c_2. Similarly, in $\mathsf{L}^{\lambda}_{\text{ar}}$,

$$\big[\text{I was insulted yesterday}\big]^{G,\bar{a}} \not\approx \big[\text{I was insulted yesterday}\big]^{G,\bar{b}}$$

for states a and b which differ in the same way as c_1 and c_2 do.

It is clear from the examples that at least some of the aspects of indexicality which Kaplan (1989) seeks to explain can also be understood using factual content, with no need to introduce "contexts of utterance" and "contexts of evaluation", which (to some) appear somewhat artificial. There are two points that are worth making.

First, in $\mathsf{L}^{\lambda}_{\text{ar}}$, synonymy is based on the isomorphism between two recursors, which is a structural condition, whereas in LD the identity of Contents or Characters is defined as a simple equality between two functions.

For example, consider 'I am insulted', a simpler version of (K4) with no modal operator, and suppose (as above) that there are states a and b which differ only in the identity of the speakers, call them Agent_a and Agent_b. Suppose also that both utterances of the sentence by the two agents are true.

To show in LD that the two relevant Contents are different, we need to consider their values in contexts of evaluation other than that determined by the states a and b: the argument being that the interpretation function of the constant $\mathsf{be\_insulted}$ evaluated on some circumstances for the two different agents is not the same (because there are circumstances at which Agent_a is insulted while Agent_b is not), and so the two Contents are not identical. On the other hand, the factual content of this sentence in state a is expressed by the canonical form

$$\mathsf{be\_insulted}^G(\bar{a})(\mathsf{I}^G(\bar{a})) \Rightarrow_{\text{cf}} \mathsf{be\_insulted}^G(\bar{a})(p) \text{ where } \{p := \mathsf{I}^G(\bar{a})\},$$

and the one for b is the same, but with $\overline{b}$ in place of $\overline{a}$. So, in $\mathsf{L}^{\lambda}_{\mathrm{ar}}$,

$$\mathrm{FC}(\text{I am insulted}, a) \neq \mathrm{FC}(\text{I am insulted}, b)$$

simply because

$$\not\models \mathsf{I}^{G}(\overline{a}) = \mathsf{I}^{G}(\overline{b}).$$

There is no need to consider the values of the function *be_insulted*^{G} in any state, not even at a and b.

Second, in $\mathsf{L}^{\lambda}_{\mathrm{ar}}$ there is the possibility to compare, in addition, the local meanings of the two sentences (K4) and (K5) at the specific state a. As one would expect, these are not locally synonymous at a, i.e.,

$$\big(\lambda(u)\big[\text{I was insulted yesterday}\big]^{G,u}\big)(\overline{a})$$
$$\not\approx \big(\lambda(u)\big[\text{David Kaplan is insulted on 20 April 1973}\big]^{G,u}\big)(\overline{a}).$$

This accounts for the fact that although “what is said” (the factual content) is the same, to understand this one must know that ‘I’ and ‘yesterday’ refer to DK and 20 April 1973 respectively; two sentences are locally synonymous in a state only when the fact that “they say the same thing” can be realized by a language speaker who does not necessarily know the references of the indexicals in them.

5.2. What are the belief carriers?

The objects of belief must be situated meanings of some sort: can we model them faithfully by *factual contents* or *local meanings*, the two versions of situated meaning that we introduced?

There are several well-known paradoxes which argue against taking factual contents as the objects of belief,[15] but our pedestrian example (5) can serve as well. If belief respected factual content, then in a state a in which ‘her’ is Mary, an agent would equally believe ‘John loves her’ as she would ‘John loves Mary’; but we can certainly imagine situations in which the agent does not know that ‘her’ refers to Mary—we make this sort of factual error all the time, and it certainly affects our beliefs. Thus factual synonymy is not preserved by belief attribution.

Local meanings are more promising candidates for belief carriers, especially as they eliminate this sort of belief paradox which depends on the agent’s mistaking the values of indexicals. Moreover, the discussion in Section 4.5 suggests that the local meaning $\mathrm{LM}(A, a)$ models what has sometimes been called “the sentence *A* under the situation *a*” and has been proposed as the object of belief. So it would appear that of the known candidates, local meanings may be the best, formal representations of belief carriers.

5.3. What is preserved under translation?

Faithful translation should also preserve some aspect of meaning, so which is it? It is clear, again, that it cannot be factual content, as ‘John loves her’ would never be translated as ‘Jean aime Marie’, whatever the state. Perhaps referential (global) synonymy or local synonymy are translation invariants, but there are no good arguments for one or the other of these—or for preferring one over the other, given how closely related they are. The question is interesting and we have no strong or defensible views on it—but it should be raised in the evaluation of any theory of meaning, and it almost never is.

Acknowledgements

The research of Eleni Kalyvianaki for this article was co-funded by the European Union - European Social Fund & National Resourses - EPEAEK II.

Notes

1. We will generally assume that the reader is reasonably familiar with Montague’s LIL and thus it will be clear to her that we employ a rather “simplified” version of this language, at least in what concerns the way natural language is translated into it. On the other hand, we will describe the basic ideas of Gallin (1975) and Moschovakis (2006), so that this article is largely independent of the details of those papers.
2. In fact, it is convenient (and harmless) to take $\mathbb{T}_t \subseteq \mathbb{T}_e$, i.e., simply to assume that the truth values $0, 1, er$ are in $\mathbb{T}_e$, so that the denotations of sentences are treated like common entities, with Frege’s approval. The extra truth value is useful for dealing with simple cases of presupposition, but it will not show up in this article.
3. Pedantically, types are finite sequences (strings) generated by the distinct “symbols” $\mathsf{e}, \mathsf{s}, \mathsf{t}, (, \rightarrow$ and $)$, and terms (later on) will similarly be strings from a larger alphabet. We use ‘$\equiv$’ to denote the identity relation between strings.
4. We assume for simplicity this *standard* (largest) *model* of the typed λ-calculus built from the universes $\mathbb{T}_s$ and $\mathbb{T}_e$.
5. We have assumed, for simplicity, a constant $\mathsf{thetemp} : \mathsf{e}$ which we grouped with names and indexicals, because of its typing.
6. For the examples from natural language, we will assume some plausible properties of the interpretations of these constants, e.g., that there are states in which

some people are running while others sit, that John loves Mary in some states while he dislikes her in others, etc. None of these assumptions affect the logic of meaning which is our primary concern.

7. This ‘John’ is just a formal expression (a string of symbols), which, quite obviously, may refer to different objects in different states. “Proper names” which should be *rigid designators* by the prevailing philosophical view are more than strings of symbols, and in any case, the logic of meaning which concerns us here does not take a position on this (or any other) philosophical view.
8. We denote by $\pi\{x := t\}$ the *update* of an assignment π which changes it only by re-assigning to the variable $x : \sigma$ the object $t \in \mathbb{T}_\sigma$:

$$\pi\{x := t\}(\mathsf{v}_i^\tau) = \begin{cases} t, & \text{if } \mathsf{v}_i^\tau \equiv x, \\ \pi(\mathsf{v}_i^\tau), & \text{otherwise.} \end{cases}$$

9. An alternative would be to re-type the same constants in K and distinguish between the LIL-typing and the Ty_2-typing of c. The method we adopted is (probably) less confusing, and it makes it easier to express the Gallin translation of LIL into Ty_2 below.
10. We use ‘G’ instead of Gallin’s ‘*’ and we make the state variable explicit.
11. In example (15), for simplicity, the logic constant & is not translated into $\&^G(u)$ since its denotation is independent of the state.
12. A term A is *explicit* if the constant where does not occur in it.
13. Factual synonymy can be expressed without the use of parameters by the following, simple result: $A^{G,\bar{a}} \approx B^{G,\bar{b}}$ *if and only if there exist terms* $A_0, \ldots, B_0, \ldots,$ *as in the Referential Synonymy Theorem 3, such that for all assignments* π,

$$\text{if } \pi(u) = a \text{ and } \pi(v) = b, \text{ then } \mathrm{den}(A_i{}^{G,u})(\pi) = \mathrm{den}(B_i{}^{G,v})(\pi), \quad (i = 0, \ldots n).$$

The same idea can be used to define the recursor $\mathrm{FC}(A,a)$ directly, without enriching the syntax with state parameters.

14. A state in $\mathsf{L}_{\mathrm{ar}}^{\lambda}$ acts both as context of utterance, thus disambiguating all occurrences of indexicals, names etc, and as context of evaluation, thus evaluating the denotations of verbs, adjectives etc.
15. See for example Russell’s “author of Waverley” example as presented in the Introduction of (Salmon and Soames 1988) or in (Church 1982).

References

Church, Alonzo

1982 A remark conerning Quine’s paradox about modality. Spanish version in *Analisis Filosófico* 25–32, reprinted in English in (Salmon and Soames 1988).

Frege, Gottlob

1892 On sense and denotation. *Zeitschrift für Philosophie und philosophische Kritik* 100. Translated by Max Black in (Frege 1952) and also by Herbert Feigl in (Martinich 1990).

1918 Der Gedanke - Eine Logische Untersuchung. *Beiträge zur Philosophie des deutschen Idealismus I* 58–77. Translated as "Thoughts" and reprinted in (Salmon and Soames 1988).

1952 *Translations from the Philosophical Writings of Gottlob Frege*. Oxford: Blackwell. Edited by Peter Geach and Max Black.

Gallin, Daniel

1975 *Intensional and higher-order modal logic*. Number 19 in North-Holland Mathematical Studies. Amsterdam, Oxford, New York: North-Holland, Elsevier.

Kaplan, David

1978 On the logic of demonstratives. *Journal of Philosophical Logic* 81–98, reprinted in (Salmon and Soames 1988).

1989 Demonstratives An Essay on the Semantics, Logic, Metaphysics, and Epistemology of Demonstratives and Other Indexicals & Afterthoughts. In Joseph Almog, John Perry, and Howard Wettstein, (eds.), *Themes from Kaplan*, 481–614. Oxford University Press.

Martinich, Aloysius P., (ed.)

1990 *The Philosophy of Language*. New York, Oxford: Oxford University Press, second edition.

Montague, Richard

1973 The Proper Treatment of Quantification in Ordinary English. In Jaakko Hintikka et al., (eds.), *Approaches to Natural Language: Proceedings of the 1970 Stanford Workshop on Grammar and Semantics*, 221–224, reprinted in (Montague 1974). Dordrecht: D. Reidel Publishing Co.

1974 *Formal Philosophy*. New Haven and London: Yale University Press. Selected papers of Richard Montague, edited by Richmond H. Thomason.

Moschovakis, Yiannis N.

2006 A logical calculus of logic and synonymy. *Linguistics and Philosophy* 29: 27–89.

Salmon, Nathan and Scott Soames, (eds.)

1988 *Propositions and attitudes*. Oxford: Oxford University Press.