Pure quotation[*]

Emar Maier
University of Groningen

(to appear in *Philosophy Compass*)

Abstract Pure quotation, as in *'cat' has three letters*, is a linguistic device designed for referring to linguistic expressions. I present a uniform reconstruction of the four classic philosophical accounts of the phenomenon: the proper name theory, the description theory, the demonstrative theory, and the disquotational theory. I evaluate the strengths and weaknesses of each proposal with respect to fundamental semantic properties like compositionality, productivity, and recursivity.

Keywords: quotation; use–mention distinction; metalinguistic reference; compositionality; syntax–semantics interface

1 Pure quotation: a challenge for the Fregean Program

In semantics and philosophy of language we typically assume that the primary use of language is to refer to objects, truth values, possible worlds and various sets thereof. Meanings are conceptualized as mappings from expressions of certain types to the corresponding semantic domains. Furthermore, we typically assume that language is compositional, that is, that the meaning of a complex linguistic expression depends only on the meanings of its constituents. The idea is that this may allow us to derive all infinitely many, knowable sentence meanings (say in terms of sets of possible worlds) from a finite, and therefore in some sense cognitively plausible, system consisting of (i) a number of syntactic/semantic composition rules, and (ii) a finite lexicon.[1]

Quotation poses a significant threat to this so-called Fregean Program in semantics. I will restrict attention to pure quotation:

[*] I thank Markus Werning, Peter Pagin, Hans-Martin Gärtner and the anonymous reviewer for valuable comments and discussion. My research is supported by the EU under FP7, ERC Starting Grant 263890-BLENDS.

[1] This reasoning is usually associated with Frege. Cf. Pagin & Westerståhl (2010a,b) and Werning et al. (2012).

(1) ‘cat’ is a three-letter word.

The quotation *‘cat’* does not denote a regular semantic object, say a set of cats, but rather a linguistic object, viz., the word *cat*. Intuitively, putting quotation marks around a word turns it into a referential expression that refers to that word. However, turning this intuitive characterization into a composition rule would render the language noncompositional: the meaning of *‘cat’* is not determined by the meaning of the constituent *cat* (rather, it *is* the constituent).

To bring out the tension between compositionality and pure quotation we can look at substitution failures. Clearly, quotation is referentially opaque in the sense that substitution of coreferential terms, and even synonyms, inside a quotation need not preserve truth or meaning (Quine 1953). In other words, (2) is clearly invalid.

(2) ‘Cicero’ has six letters
Cicero = Tully
———————————
∴ ‘Tully’ has six letters

This type of opacity seems to contradict compositionality: since *Cicero* and *Tully* have the same meaning, so should the complex expressions *‘Cicero’* and *‘Tully’*, and the otherwise identical statements that contain them.

To avoid giving up compositionality, the first three classic quotation theories surveyed below simply deny that *‘Cicero’* is a complex expression containing *Cicero* as a constituent. The proper name theory (§3) states that *‘cat’* is a proper name of *cat*. The closely related description theories (§4) analyze the quotation as a description of the referent in terms of smaller quotation units. Davidson’s celebrated demonstrative theory (§5) analyzes the quotation marks themselves as a referring expression. Finally, the disquotational theory (§6) returns to the naïve intuition laid out above, that a quotation is syntactically composed of quotation marks and an expression.

I will investigate below if and to what extent these theories can be said to satisfy compositionality. Moreover, I will evaluate them with respect to a related fundamental desideratum for semantic theory, viz. productivity: Can we capture the semantics of quotation with a finite set of rules and lexical items, that, when acquired, allow us to form and interpret the infinitely many new quotation expressions, viz. for each expression of the language its quotation?[2] In addition, I’ll consider more quotation specific desiderata

[2] Cf. Werning (2005) for a definition of productivity in terms of computability that captures roughly the notion used here, and a demonstration that it is in fact logically independent of compositionality (defined as homomorphism between syntax and semantics). Cf. also

such as opacity (substitution failure), recursivity (or, more specifically, the ability to deal with iterated quotation and interpretation), and autonymy (the self-referential character of quotations).

2 A toy grammar formalism

To discuss pure quotation in any detail it will be vitally important to carefully separate the different levels of linguistic analysis, such as phonological surface form, syntax, logical form, and modeltheoretic interpretation. For concreteness, and to ensure a fair comparison of quotation theories, I propose here a simplified Montagovian grammar system to be used throughout.

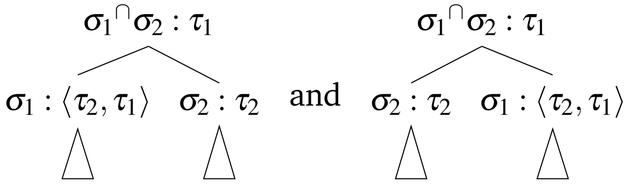
Language is described syntactically in terms of trees, generated by a finite set of grammatical rules. The nodes of the tree are typically thought of as expressions, decorated with category labels (e.g. *[$_{VP}$ loves Mary]*). To facilitate the formulation of the interfaces with phonology and semantics, I take the node expressions to be finite sequences of letters from an alphabet (e.g. John, has six letters, loves), and the category labels are just logical types (e.g. $\langle e,t\rangle$, $\langle e,\langle e,t\rangle\rangle$).[3] I restrict attention to written language, i.e., phonological forms are strings of letters in some extension of the Roman alphabet.

More precisely, let A^* denote the set of finite strings of letters from alphabet A; let $^\frown$ denote string concatenation; and let TY denote the set of functional types generated from the basic types e(ntities) and t(ruthvalues). The syntax $\mathscr{L}_{syn}$ consists of all binary trees with nodes from $A^* \times TY$, generated by a finite given lexicon of terminal nodes as in (3a), and the tree composition rules in (3b).

(3) a. lexicon: Cicero : e, orator : $\langle e,t\rangle$, walks : $\langle e,t\rangle$, $\ldots \in \mathscr{L}_{syn}$

b. composition: if $\underset{\triangle}{\sigma_1 : \langle\tau_2,\tau_1\rangle}$ and $\underset{\triangle}{\sigma_2 : \tau_2}$ are in $\mathscr{L}_{syn}$,

then so are

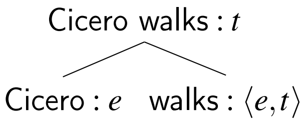

$\sigma_1{}^\frown\sigma_2 : \tau_1$
$\sigma_1 : \langle\tau_2,\tau_1\rangle$
$\sigma_2 : \tau_2$
and
$\sigma_1{}^\frown\sigma_2 : \tau_1$
$\sigma_2 : \tau_2$
$\sigma_1 : \langle\tau_2,\tau_1\rangle$

Pagin & Westerståhl (2010b) for careful distinctions between the various related properties of interpreted languages often associated with compositionality (learnability, systematicity, creativity, etc.).

[3] This purely type-driven syntax is a gross oversimplification leading to massive overgeneration, but this paper is not about natural language syntax.

By way of illustration, with (3) we derive the following syntactic representation of *Cicero walks*:

(4)

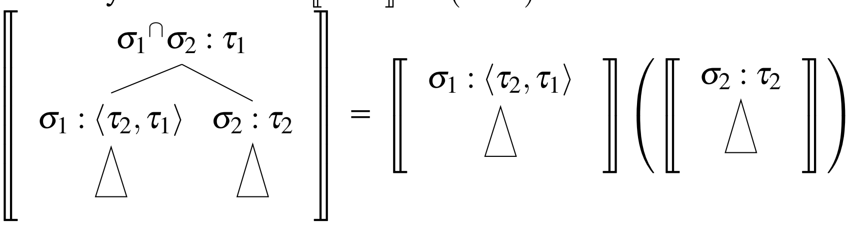


We can analyze the syntax–phonology interface as a simple mapping $|\ |$ from syntactically well-formed trees ($\mathscr{L}_{syn}$) to strings of letters (A^*): simply take the first element of the tree's top node. The syntax–semantics interface is a mapping $[\![\ ]\!]$ from $\mathscr{L}_{syn}$ into the semantic domain provided by the model. For any type τ, $[\![\ ]\!]$ maps syntactic expressions of type τ to elements of the corresponding domain D_τ. Both the domains and $[\![\ ]\!]$ are defined recursively on the basis of a given model. First we define a model for a language, with a given lexicon and set of types, as a basic domain of entities paired with a lexical interpretation function, i.e. $\langle D_e, \mathbb{I}\rangle$, for which the following hold:

(5) a. D_e is a set of entities. $D_t = \{0,1\}$. $D_{\langle\tau_1,\tau_2\rangle}$ is the set of functions from D_{τ_1} to D_{τ_2}.
b. $\mathbb{I}$ is a function that maps any lexical item $\sigma : \tau \in \mathscr{L}_{syn}$ to an element of D_τ

Semantic interpretation of the whole language in the model is defined as follows:

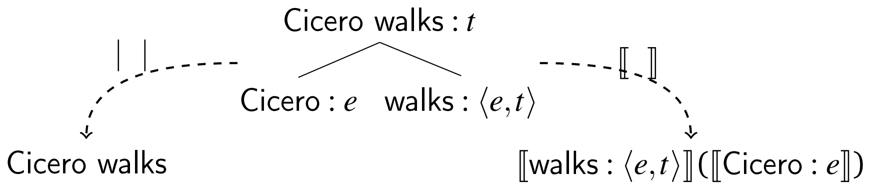
(6) a. For every lexical item: $[\![\sigma : \tau]\!] = \mathbb{I}(\sigma : \tau)$
b.


$\sigma_1{}^\frown\sigma_2 : \tau_1$
$\sigma_1 : \langle\tau_2,\tau_1\rangle$
$\sigma_2 : \tau_2$
=
$\sigma_1 : \langle\tau_2,\tau_1\rangle$
$\sigma_2 : \tau_2$

These definitions constitute an execution of the Fregean program. The meaning of any complex syntactic tree is fully determined by the meanings of its constituents, i.e., the subtrees immediately below the top node (its daughters). These daughters' meanings in turn are determined by their daughters' meanings, and so on down to the leaves, the meanings of which are specified in the model's lexical interpretation function $\mathbb{I}$. What's more, complex meanings here are defined uniformly in terms of function application – apply the meaning of one subtree to that of the other.

In (7) we see the two interface functions applied to our minimal example tree from (4):

(7)


Cicero walks : t
$|\ |$
Cicero : e
walks : $\langle e,t\rangle$
$[\![\ ]\!]$
Cicero walks
$[\![$walks : $\langle e,t\rangle]\!]([\![$Cicero : $e]\!])$

The expression tree is composed of two lexical items. The meanings of the lexical items are given by the model: $\mathsf{walks} : \langle e,t\rangle$ denotes the characteristic function of the set of walkers, and $\mathsf{Cicero} : e$ denotes Cicero. The meaning of the whole tree then is the result of applying the function to the individual, i.e. 1 if Cicero walks, 0 if he doesn't.

Below I reconstruct the major theories of pure quotation within this grammatical framework by adding different composition and interpretation rules, extending the lexicon, and/or changing the models. For broad introductions to the various forms and theories of quotation, I refer to Cappelen & Lepore (2012) and Saka (2013). Sacrificing wide and historically accurate coverage for depth and precision, I focus here only on pure quotation, and moreover discuss only the four classic approaches. [4]

Some notes on notation: I use single quotes for pure quotation, both in formulas and informally in natural language examples and running text. Double quotes therefore mean quotations of quotations (or, occasionally, "scare quotes"). Informally, I use italics both for emphasis and for all types of quotation and emphasis in the running text. I'm adapting quotations from the literature to these conventions. For strings I freely add or suppress spaces and concatenation symbols for readability: $\mathsf{h}^\frown\mathsf{e}^\frown\mathsf{w}^\frown\mathsf{a}^\frown\mathsf{l}^\frown\mathsf{k}^\frown\mathsf{s} = \mathsf{he}^\frown\mathsf{walks} = \mathsf{he\ walks}$. I don't use corner quotes, so *for every string α, 'α' is a well-formed expression*, means that the result of concatenating a left quote, a string α, and a right quote is always a well-formed expression.

[4] In addition, I'm ignoring much of the debate about what exactly quotations refer to (utterance tokens? written inscriptions? expression types?). We can quote nonsense, letters, gestures, facial expressions, animal sounds, or newly invented letter-like symbols (cf. Partee 1973, Clark & Gerrig 1990). Most of this is beyond the scope of the paper. I'm assuming that quotations can refer to (i) phonological expression types, formalized as strings of letters, including gibberish letter combinations, and (ii) syntactically wellformed expression types of the language under consideration.

3 The proper name theory

In philosophical logic, the classic reaction to the substitution failures with pure quotation is to deny that quotations are syntactically structured at all (Quine 1940, Tarski 1933).

> Quotation mark names may be treated like single words of a language, and thus like syntactically simple expressions. The single constituents of these names—quotation marks and the expressions standing between them—fulfill the same function as the letters and complexes of successive letters in single words. [...] Every quotation-mark name is [...] a name of the same nature as the proper name of a man [Tarski 1933: 159]

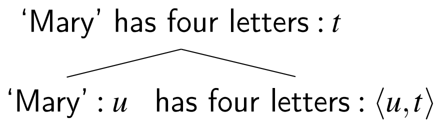
In other words, just like *Mary* is a name of Mary, so *'Mary'* is a name of her name. And therefore, just like we don't consider *ar* a syntactic constituent of *Mary*, we don't consider *Mary* a syntactic constituent of *'Mary'*. Substitution failures are correctly predicted. Given that quotations are lexical atoms, without constituents, potential problems with respect to compositionality or substitutability, which can only be formulated in terms of a quotation's constituents, never even arise.

Although the proper name theory is typically regarded as "an utter failure" (Saka 1998: 114), I will discuss it in some detail. The reason being that while properly formalizing, evaluating and extending the proper name theory, I lay the necessary foundation for introducing (and evaluating) more fashionable theories like Davidson's demonstrative theory or the disquotational analysis.

To build this proper name theory into our formal grammar we leave the composition rules intact and only add a series of quotation names as lexical primitives in $\mathscr{L}_{syn}$. For instance, in addition to $\mathsf{Mary} : e$, we add $\text{'}\mathsf{Mary}\text{'} : e$. Semantically we enrich the model's lexicon by adding that $\mathbb{I}(\text{'}\mathsf{Mary}\text{'} : e) = \mathsf{Mary}$.

Accommodating quotation in this way presupposes that phonological strings like Mary are part of the domain of entities. To avoid conflating domains of quantification here it seems appropriate to postulate a dedicated semantic domain and corresponding logical type for such linguistic entities. Following Potts (2007) I'm adding u(tterance) as the new basic type, in addition to e and t. We then get $\mathbb{I}(\text{'}\mathsf{Mary}\text{'} : u) = \mathsf{Mary} \in D_u$. We can now construct and correctly interpret a tree like (8).

(8)


‘Mary’ has four letters : t
‘Mary’ : u
has four letters : $\langle u,t\rangle$

Interpreting the tree gives: $[\![(8)]\!] = 1$ iff $\mathbb{I}(\textsf{has four letters} : \langle u,t\rangle)(\mathbb{I}([\![\textsf{‘Mary’} : u]\!]))$ $= 1$ iff Mary has four letters.

As pointed out above, quotations have no parts so compositionality is not affected. The problem with the theory is rather that we’re putting too much in the lexicon and thereby fail to account for the productivity of quotation.

Productivity was characterized in section 1 as the reduction of an infinite number of interpretable expressions to a finite, learnable set of rules and lexical items. Competent speakers of English evidently know how to name any of the infinite number of expressions of their language by quoting them. In fact even phonological strings that do not correspond to wellformed expressions can be so quoted, as is evident from the intuitive wellformedness and interpretability of assertions as in (9).

(9) a. ‘walks hit Mary’ is not grammatical
b. ‘FgHj’ is not a word

The lexicon extension needed to cover quotation thus can be summarized as follows:

(10) For every $\sigma \in A^*$, there is a lexical item ‘σ’ : u with interpretation given by $\mathbb{I}(\text{‘}\sigma\text{’} : u) = \sigma \ (\in D_u)$

This extension would be (at least) as big as the, presumably infinite, original language as a whole. The only way to square this with productivity is to introduce some kind of constituent structure within the part of the lexicon dealing with quotation. But then our supposed quotation names are really not semantic units, like proper names, anymore. This suggestion brings from the proper name theory to the description theory, cf. section 4.

As for recursivity, since there is no rule of quotation formation we obviously cannot recursively apply the output of such a rule to itself to form and interpret quotations of quotations and the like. However, there is a nonrecursive way to form quotations of quotations of .... If we add quotation marks to the alphabet, then, say, ‘‘John’’ is itself a string and by (10) should have a name ‘‘‘John’’’ : u in $\mathscr{L}_{syn}$. Thus, we can indeed interpret an (apparent) triple quotation as referring to the enclosed double quotation string.

On closer inspection there is something missing in this account of iteration. We do not just want to be able to interpret iterated quotations, but also iterate the interpretation of quotations. Intuitively, *'John'* refers to the name *John*, which in turn refers to the person John: $\llbracket\llbracket\text{`John'} : u\rrbracket\rrbracket = \text{John}$. However, we've been assuming throughout that quotations refer to phonological expressions rather than syntactic expressions. As a result we can't iterate interpretation: $\llbracket\llbracket\text{`John'} : u\rrbracket\rrbracket = \llbracket\text{John}\rrbracket$, which is undefined because the mere string John is not a well-formed tree (it doesn't have a type).

To fix this we put full-fledged expressions in D_u. The quotational lexical items could stay the same, we just adjust the interpretations, i.e., instead of the uniform rule (10) we stipulate in the lexicon that $\mathbb{I}(\text{`John'} : u) = \text{John} : e$, $\mathbb{I}(\text{``John''} : u) = \text{`John'} : u$, etc.[5] For ungrammatical strings we could fall back on the phonological interpretations provided by (10): $\mathbb{I}(\text{`Hglljd'} : u) = \text{Hglljd}$. I conclude that we *can* make the proper name theory pseudo-recursive in the sense that we can (i) interpret quotations of quotations, and (ii) interpret interpretations of quotations. It all relies on how we define the lexicon and its interpretation – the syntax and semantics of complex expressions in the language remain altogether unaffected by any of these tweaks.

All in all, although it unites opacity and compositionality, and can even be made to deal with iterations of quotation and interpretation, the proper name theory fails to provide an adequate account of at least the productivity of quotation.

4 The description theory

Description theorists give up the idea that quotations are just lexical items without internal structure. Geach (1957) proposes the following refinement for dealing with quoted sentences, which for the strict proper name theorist would be names just like any other quoted string of letters.

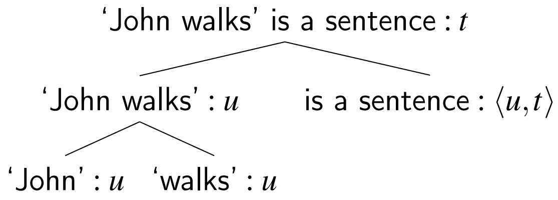
> If we use an ampersand & to mean *followed by*, then *man* & *is* & *mortal* is just the expression *man is mortal*; I should maintain that the quotation *'man is mortal'* is rightly understood only if we read it as meaning the same as *'man'* & *'is'* & *'mortal'*, i.e. read it as describing the quoted expression in terms of the expressions it contains.
>
> [Geach 1957:82]

So, just as sentences are complex expressions syntactically built out of words, we analyze quoted sentences as syntactically complex expressions, built out

[5] If we capture this lexicon extension in a general rule, we end up with something very close to the syntactic quotation rule in (23) that characterizes the disquotational theory. See section 6 below.

of quoted words. In our grammar:

(11)

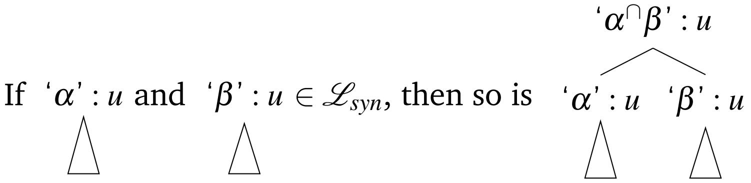

‘John walks’ is a sentence : t
‘John walks’ : u
is a sentence : $\langle u,t\rangle$
‘John’ : u
‘walks’ : u

We’re assuming that the interpretation of quoted *words* is given:

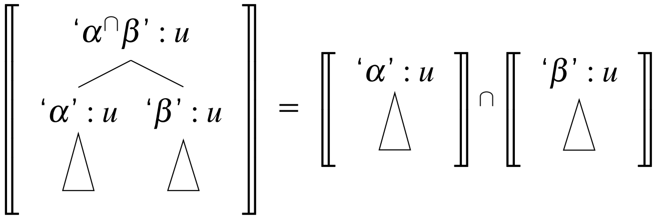
(12) For each lexical item $\sigma:\tau$ ($\tau \neq u$) there is another lexical item ‘σ’ : u, with $\mathbb{I}(\text{‘}\sigma\text{’} : u) = \sigma$

What we need to add to our grammar is a rule that says we can combine expressions of type u by concatenating them (and deleting the two adjacent quotation marks ’‘ from the middle).

(13) a.


If ‘α’ : u and ‘β’ : $u \in \mathscr{L}_{syn}$, then so is

b.



The interpretation of the complex tree is now evidently compositional. In (13b) the meaning of a complex quotation is given as the concatenation of the meanings of its daughters. in example (11) these daughters are lexical quotations, the interpretations of which are strings so concatenation does indeed give the right result: $[\![(11)]\!] = 1$ iff John$^\frown$walks is a sentence.

Since the fact that *‘John’* refers to *John* remains a lexical stipulation, we can consider Geach’s description theory a relatively minor variant of the proper name theory. However, if there are but finitely many words, as the Fregean assumes, this will be enough to account for productivity, i.e., we’ve succeeded in reducing the task of understanding an infinity of quotations to a finite, manageable task. And we didn’t have to give up compositionality either.

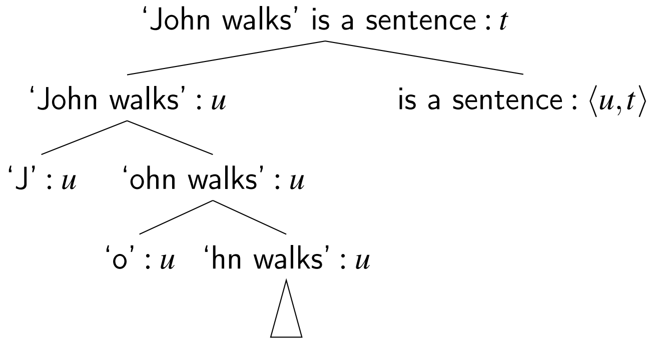
But Geach’s strategy has some serious limitations. While the syntactic composition of words into sentences is type-driven, quoted words and expressions all have the same type: they are metalinguistic names (type u). The Geach composition rule (13) cannot *see* that *‘John’* is a name of a name (type e), and *‘walks’* a name of an intransitive verb ($\langle e,t\rangle$). In other words, since all we see are the types u, anything goes. In a way this is a good thing. We want to be able to name ungrammatical complexes (*‘John hits walks’*) and predicate things of them (*is ungrammatical*). But given that we want pure quotation to extend to meaningless nonsense, why stop at words? After all, as remarked earlier, we can and do quote letters and arbitrary strings of them too.

Taking the names of sublexical symbols as atomic elements while keeping the Geachian quote composition rule results in the so-called spelling or phonological theory, typically traced back to (Tarski 1944: 344) and more recently revived by Werning (2005). Such an analysis maximally reduces the primitive metalinguistic lexicon. All we need to add to the lexicon of the quotation-free language is the names of the letters in the alphabet A, i.e. we stipulate that *‘J’*, *‘o’* etc are primitive terms referring to the letters J, o, respectively.

(14) For each letter $\lambda \in A$ there is a lexical item ‘λ’ : u with $\mathbb{I}($‘λ’ : $u) = \lambda$

Together with quote composition we then get the following logical form:

(15)

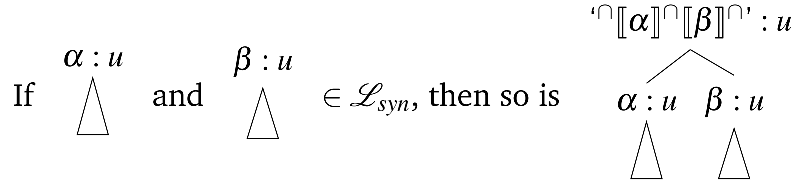


To further motivate an already rather minimal lexicon extension (compared with the proper name theorist’s (10) or Geach’s (12)), spelling theorists can point out that the practice of naming letters is already an integral part of learning (to read and write) a language. Except, we typically start with more pronounceable names, like *doubleyu* for the letter w. Indeed, for the spelling theorist, as for any proper name theorist, nothing hinges on the primitive names (in this case the letter names) containing quotation marks. Quine formulates his spelling theory as follows:

> Instead of [*'Tully was a Roman'*] we might as well say *tee-yu-ell-ell-wye-space-doubleyu-ay-ess-space-ar-oh-em-ay-en* [(Quine 1960: 143)]

Note that working with such names requires some modification of the quote composition rule (13a). I propose the following more general syntactic composition rule for quotations:[6]

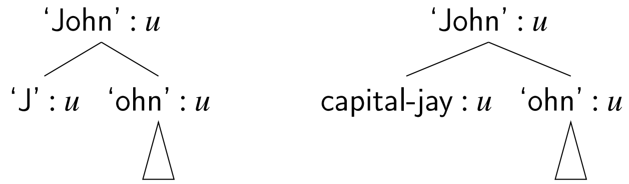
(16)


If
$\alpha : u$
and
$\beta : u$
$\in \mathscr{L}_{syn}$, then so is
$`{}^\frown[\![\alpha]\!]^\frown[\![\beta]\!]^\frown' : u$
$\alpha : u$
$\beta : u$

The corresponding interpretation rule (13b) remains essentially the same.

The description theories now account for both compositionality and productivity. But there's a price to pay. First, we lose opacity. Take the spelling theory. The letter name *'J'* is a genuine constituent of the complex quotation expression *'John'*. Therefore, this *'J'* should be substitutable with other names of that same letter, like *capital-jay*, say. Somewhat counterintuitively, at least according to (Davidson 1979: 35), we thus predict that *capital-jay*$^\frown$*'ohn'* is a wellformed expression referring to *John*. However, on closer inspection, using the generalized Geach rule in (16), what we predict is that the following two trees are well-formed and coreferential, which seems rather harmless:

(17)

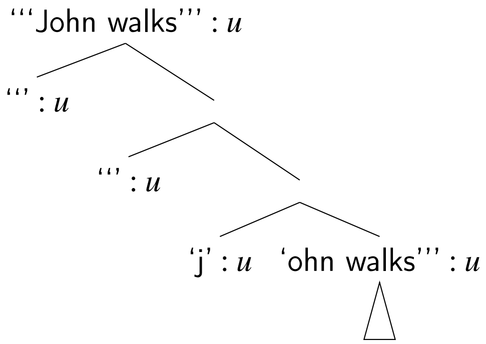


In addition, real recursion is still unavailable due to the ultimately lexical nature of quotation introduction. But let's try and emulate the account of iteration presented for the original proper name theory above. In the Geach variant we need to adjust the composition rule to take care of multiple quotation marks: combining two iterated quotations of type u, say *"'John'"* and *"'walks'"*, should yield a tree with top node *"'John walks'"*. Then, at the lexical level, we follow the proper name theory and add iterated word quotations to the lexicon, which of course causes the lexicon to explode. That is, we'd trade iteration for productivity.

[6] The use of semantic interpretation ($[\![\;]\!]$) in a syntactic rule seems harmless, but it does entail that we cannot state syntax independently of semantics.

The same strategy (add names of names of names of letters etc. to the lexicon) is available for the spelling theory, but there we have another option as well. Just add two symbol names: ‘‘’ : *u*, the name of the left quote symbol, and ‘’’ : *u*, the name of the right quote symbol.

(18)



Our spelling theory would assign the correct interpretation here, $[\![(18)]\!]$ = ‘‘John’’. So, the spelling theory can deal with iterated quotations, without giving up productivity.

Iterated interpretation however is problematic. For the proper name theorist and her vast lexicon it was no problem to stipulate interpretations in $\mathscr{L}_{syn}$. The spelling theorist too could stipulate whatever meanings she wants for her atomic elements, but her atoms correspond to individual letters rather than plausibly meaningful syntactic expressions.

In section 6 I present a truly recursive account, which naturally deals with iterations of quotation and interpretation, at the cost of giving up full compositionality. But first we'll see how Davidson criticizes proper name and description theories alike for their reliance on a primitive naming relation.

5 The demonstrative theory

We've seen how description and proper name theories ultimately ground the reference of quotations in a lexical naming relation between one expression and another. *‘John’* refers to *John* (or, for the spelling theorist, *‘t’* refers to *t*), in exactly the same way that *John* refers to John, viz. by virtue of the lexical interpretation function $\mathbb{I}$.

Davidson (1979) argues that this cannot be right. Quotations don't name but *picture* their referents. An occurrence of *‘John’* refers to the expression *John* because the thing between the quotation marks resembles, or is a token of that expression. The demonstrative theory is one way to make this intuition precise (the disquotational theory is another).

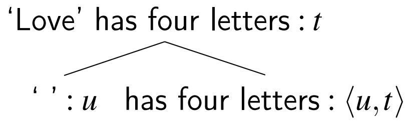
An essential feature of Davidson's proposal is that the material inside the quotation marks is treated as completely semantically and syntactically inert – it's merely a (concrete) inscription token, salient in the context.[7] The actual linguistic referring is done by the quotation marks:

> On the demonstrative theory, neither the quotation as a whole (quotes plus filling) nor the filling alone is, except by accident, a singular term. The singular term is the quotation marks, which may be read *the expression a token of which is here*. [Davidson 1979:37-38]

In other words, the quotation marks are a special type of demonstrative, meant to point at a concrete inscription or utterance token and thereby refer to the corresponding expression.[8]

In our framework, the bare quotation marks are the type u expression that is supposed to refer to the quoted word. The material enclosed in quotation marks must be kept out of the logical form altogether.

(19)


'Love' has four letters : t
' ' : u
has four letters : $\langle u,t\rangle$

Following Predelli's (2008) lead, we'll capture the semantics of these Davidsonian quotation marks in Kaplan's (1989) *Logic of Demonstratives*.

In a Kaplanian system, the semantic values of expressions are relativized to both a context (c) and an index (w) parameter. Indexical expressions like *I* or *tomorrow* and demonstratives depend solely on the context parameter: $[\![\mathsf{I}:e]\!]^c_w =$ the speaker of c ($\in D_e$). Other expressions like *the president of the U.S.* or *walks* depend on the index w. Some additional remarks: (i) we import the type system as is, i.e. expressions of type $\langle e,t\rangle$ denote (relative to c and w) sets of entities; (ii) D_u is still the set of strings; and (iii), given the basic

[7] This reading of Davidson is incompatible with a literal interpretation of the so-called paratactic analysis which treats *'I am an idiot' is a sentence* as meaning the same as the two-sentence discourse *I am an idiot. That is a sentence*. The paratactic analysis (apparently endorsed by (Davidson 1979: 90) as well as his main defenders Cappelen & Lepore (1997)) immediately runs into trouble with quotations containing indexicals, false statements, and ungrammatical elements, among other things: *John keeps repeating (the words) 'I am an idiot'* has John referring to himself, while in a discourse like *I am an idiot. John keeps repeating that.* he is talking about me.

[8] A distinct but related theory is the so-called demonstration theory (Clark & Gerrig 1990, Recanati 2001), which likewise takes as its starting point a relation of depiction between the quotation and the quoted. Since this theory is primarily designed for dealing with other varieties of quotation, like direct and mixed/open quotation I will not discuss it further here.

extensional values, we can define intensions expressed relative to an utterance context by abstracting over possible worlds ($[\![\varphi]\!]^c : w \mapsto [\![\varphi]\!]^c_w$), and also so-called characters by additionally abstracting over contexts ($[\![\varphi]\!] : c \mapsto [\![\varphi]\!]^c$).[9]

(20) a. $[\![\text{has four letters} : \langle u,t\rangle]\!]^c_w$ = the set of strings ($\in D_u$) that have four letters.[10]

b. $[\![\text{‘ ’} : u]\!]^c_w = \mathbb{I}(\text{‘ ’} : u)(c,w)$ = the most salient quotation-marked string ($\in D_u$) tokened in context c.

Take as utterance the written sentence token labeled (19) and call the context in which this utterance took place $c_{(19)}$. In $c_{(19)}$ the string Love is the most salient quotation-marked string. The proposition expressed by (19) in $c_{(19)}$, $[\![(19)]\!]^{c(19)}$, is the (characteristic function of the) set of worlds in which Love has four letters, i.e. the tautological proposition.

This seems the correct prediction, but consider what happens when we abstract over contexts and look at the character, $[\![(19)]\!]$, i.e. the function from contexts to propositions expressed by utterances in those contexts. This character no longer involves the quoted expression: it maps any context c with its contextually provided string σ_c to the proposition that σ_c has four letters. As a result, (21a) and (21b) both have the same characters as (19):

(21) a. ‘Hate’ has four letters.

b. ‘Cat’ has four letters.

For Kaplan that would lead to the highly counterintuitive prediction that (19), (21a), and (21b) are epistemically equivalent and linguistically synonymous. Perhaps even worse, on the current proposal, (21b) uttered in context $c_{(19)}$ would express the proposition that the string Love has four letters.

A solution for these problematic predictions is based on the idea that $c_{(19)}$ (with demonstrated string Love) is not a suitable or “proper” context for uttering sentences that do not contain quotations of that string. Departing from Predelli’s use-conditional account, I would take a cue from dynamic semantics and turn this around: uttering a sentence containing quotation marks leads to an *update* of the context of utterance with the quoted strings. In other words, a sentence containing n quotations can only be semantically evaluated against a context if that context has first undergone n of these updates, but let’s focus on $n = 1$.

Formally, we first need to add a stack of strings as a context parameter:

[9] The reader unfamiliar with the Kaplanian definitions of context–index and character–intension–extension should consult Zimmermann (1991) or Schlenker (2011).

[10] Having four letters is a property that is arguably neither index nor context dependent.

$c = \langle s_c, t_c, w_c, \sigma_c \rangle$ (i.e., the speaker, time, world and string-stack[11] of c, respectively). Our demonstrative quotation marks now simply refer to the top of the current stack: $\mathbb{I}(\text{‘ ’} : u)(c,w)$ = the top element of the contextual stack σ_c. Next, uttering a quotation triggers adding the relevant string inside the quotation to the context's stack: $c + \mathsf{Love} = \langle s_c, t_c, w_c \langle \sigma_c, \mathsf{Love} \rangle \rangle$. With the help of this mechanism we can turn the regular Kaplanian notion of a character, $[\![\varphi]\!]$, into a quotation-sensitive character $[\![\![\varphi]\!]\!]$, by updating the contextual string stack with any quoted strings occurring in the expression to be interpreted:

(22) $$[\![\![\varphi]\!]\!]^c = \begin{cases} [\![\varphi]\!]^{c+\tau} & \text{if } \varphi \text{ contains ‘}^\frown\tau^\frown\text{’} \\ [\![\varphi]\!]^c & \text{otherwise.} \end{cases}$$

On this extension of the Predelli/Davidson analysis, an utterance of (19) will, correctly, express the (tautological) proposition that Love has four letters, regardless of the context of utterance. The quotation-sensitive characters of the examples in (21) are appropriately different, e.g., for every c, $[\![\![(21b)]\!]\!]^c$ = the (false) proposition that Cat has four letters.

However, with the move from $[\![\varphi]\!]$ to $[\![\![\varphi]\!]\!]$ we have made the quoted expression itself part of the conventional, linguistic meaning of a sentence with a quotation. In effect then we have drifted away from Davidson's starting point, and arrived at something closer to the so-called disquotational analysis, which builds the fact that a quotation refers to the quoted expression (autonymously) into the syntax/semantics from the start.

For now, let's review the main selling points of the demonstrative over the proper name and description accounts: (i) we need no extension of the lexicon or any other part of the syntax/semantics, except for a new demonstrative (and a "pragmasemantic" context update mechanism); and (ii) the fact that a quotation like *'John'* refers to the word *John* is no longer a lexical accident but falls out of the mechanism of context update plus demonstrative reference.[12]

I finish the section by evaluating the demonstrative analysis with respect to compositionality, productivity, and recursivity.

First, compositionality. At first sight we expect no problems, since a sentence with a quotation is syntactically represented with just the bare pair of quotation marks, the interpretation of which is determined by a lexical interpretation rule. The crucial difference with the proper name theory however is that the lexical interpretation in question takes the extralinguistic context of

[11] A stack is recursively defined as either just a top element or an ordered pair containing a stack and a top element.

[12] Whether it will count as a logical, analytical and/or a priori truth depends on the precise definition of such notions in our extended Kaplanian logic.

utterance as an additional input in determining reference. To avoid counting vanilla indexicality as evidence of non-compositionality, we would typically consider only the character level when evaluating a two-dimensional Kaplanian system. Compositionality then demands that the character of a complex expression be determined by the characters of its syntactic constituents. In this case, given that quotations have no syntactic constituents, the question becomes: is the character of a quotation fully determined by the lexicon? The answer is yes for our initial, basic character notion $[\![\text{‘ ’} : u]\!]$, but no for the eventual, quotation-sensitive alternative $[\![\![\text{‘ ’} : u]\!]\!]$, since the latter notion depends on the quoted string. Nonetheless, in either version, quotations have no syntactic constituents, so it is easily verified that substitutions are blocked and we predict full opacity.

Second, productivity. The semantics of quotation proper consists of a single lexical item and its interpretation rule. But, as shown above, what goes on outside the abstract expression in $\mathscr{L}_{syn}$, the pragmasemantic mechanism of context update and indexical interpretation, is essential for properly interpreting quotations in natural language. Since the pragmasemantic components involved, i.e. the contextual update mechanism, is also finitely statable, I take it that productivity is captured satisfactorily.

Finally, recursion. The principal objection against the demonstrative account in the literature is that it fails to account for iterated quotation and interpretation. However, as before we can make some sense of iterated quotations. We treat an utterance of *“‘John’”* as a quotation ‘ ’ : u triggering a context update with the string ‘‘John’’. Analyzed in this way, we correctly predict that the apparent triple quotation refers to the enclosed double quotation ($[\![\text{‘ ’} : u]\!]^{c+\text{‘‘John’’}} = \text{‘‘John’’}$).

The real problem, it seems, lies with iterated interpretation. A quotation picks out the string that occurred in quotation marked form in the utterance context, and strings as such are not interpretable. However, as observed before, if the original quotation contains a word or phrase, we would want to say that the interpretation of the quotation should itself be interpretable. For the proper name theory we incorporated this fact by adding genuine expressions to D_u and letting certain quotations pick out those. This won’t work for the demonstrative analysis, where we have but a single lexical entry to play with.[13] In principle, we could relegate the distinctions between quotations referring to expressions and quotations referring to mere strings to the pragmasemantic component of the theory. The pragmasemantic mechanism should then tell us that when we encounter an utterance of *‘love’ has four*

[13] Recall that we ran into a similar problem with the spelling theory.

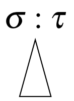
letters the filling of the quotation marks corresponds not just to a phonological expression, a string, but to the wellformed syntactic expression $\mathsf{love} : \langle e, \langle e,t \rangle \rangle$. It is then this expression that we add to the stack, and that, consequently, is the referent of the quotation. In such a case we can indeed iterate semantic interpretation, as desired.

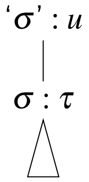
Unfortunately, the combination, iterated interpretation for iterated quotation, shows that we might we asking too much of our pragmasemantic mechanism. Consider the double quotation *“love”*. Double interpretation requires updating the stack not with the string ‘love’, but with the corresponding syntactic expression. But syntax doesn’t “see” quotation fillings, so we’re stuck with $‘\ ’ : u$. Interpreting *“love”* comes down to $[\![‘\ ’ : u]\!]^{c+‘\ ’:u} = ‘\ ’ : u$. We could interpret the result, but it’s not quite clear where the intuitively needed update with love is supposed to come from.

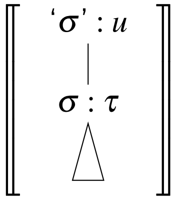
In sum, Davidson’s demonstrative theory requires only very minimal extensions to the syntax and semantics, but does come with a heavy pragmasemantic mechanism. A careful analysis shows that we thereby effectively lose compositionality, although we can maintain that the system as a whole is productive. Some forms of iteration of quotation and interpretation may be accounted for.

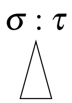
6 The disquotational theory

The starting point of the disquotational analysis is that quotation is a genuine syntactic operation that turns an expression into an NP referring to that expression (Richard 1986, Pagin & Westerståhl 2011, Gaskin & Hill 2013). In our framework, this means that we add the following syntactic rule, and its corresponding semantic interpretation. Note that type u now corresponds to the domain of expressions, i.e. $D_u = \mathscr{L}_{syn}$.

(23) a. If 

$\in \mathscr{L}_{syn}$, then so is 

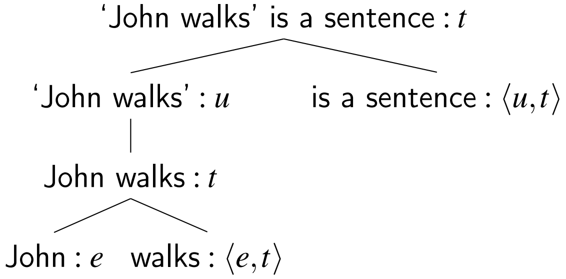
b. 

= 

The semantic rule clearly shows the self-referential character of quotation. Moreover, the opacity of quotation is trivially satisfied.

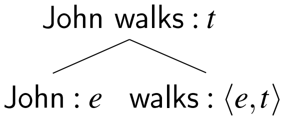
To compare this with the previous analyses consider the new analysis of one of the earlier examples (11):

(24)


‘John walks’ is a sentence : t
‘John walks’ : u
is a sentence : $\langle u,t\rangle$
John walks : t
John : e
walks : $\langle e,t\rangle$

The truth conditions we derive are as follows:

(25) $[\![(24)]\!] = 1$ iff



is a sentence

In contrast to the description and proper name theories we've left the lexicon intact, we're only adding a composition rule. Since quotations are no longer lexical items but complex expressions, compositionality becomes a valid concern. As indicated in section 1, this theory is not compositional: the interpretation of the complex expression *'John walks'* is completely independent of the meaning of its sole constituent *John walks*, i.e. that John walks. Instead it depends on, or rather is equal to, the expression *John walks* itself.[14]

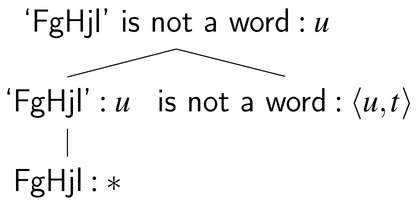
Another important disadvantage of the current proposal is that quotations here contain grammatical expressions. This means that we cannot represent quotations of ungrammatical expressions. The simplest way to overcome this limitation is to add ungrammatical strings to $\mathscr{L}_{syn}$. One way to do this is by adding a special nonsense type $*$, associated with an empty semantic domain $D_* = \emptyset$. We extend the lexicon as follows:

(26) For every $\sigma \in A^*$, there is a lexical item $\sigma : * \in \mathscr{L}_{syn}$, and $\mathbb{I}(\sigma : *)$ is undefined.

[14] Pagin & Westerståhl (2011) formulates a workable notion of generalized compositionality which, inspired by Frege (1892), relativizes compositionality to the linguistic context in which an expression occurs. In particular, compositionality in a quotational context need amount to the same requirement as in an extensional or intensional context. Under this definition the disquotational analysis does count as compositional.

Furthermore, we may suppose that items of type $*$ are extra-grammatical in the sense that they do not compose with any other expressions in $\mathscr{L}_{syn}$. The tree in (27) illustrates the use of "nonsense expressions":

(27)



Although with (26) we extend our lexicon considerably, quotations are still always generated by a syntactic rule, with corresponding uniform interpretation procedure, and hence not treated as unanalyzable wholes. Hence, the interpretation of a quotation is not given by a lexical stipulation or an extralinguistic context, but can be read off directly from its syntactic form.

A more tangible benefit of this "syntactic transparency" is that the disquotational analysis captures the productivity of quotation. To learn the meaning and use of quotation we need only learn a single syntactic composition rule and its semantic counterpart. Admittedly, in order to fully capture the application of quotation to arbitrary strings, we did switch to an infinite lexicon. This is however entirely harmless from a productivity point of view, because (i) the infinite addition is generated recursively from a finite alphabet with a single rule of concatenation, and (ii) we are not assigning any semantic interpretations to these added lexical items.

The main benefit of the disquotational analysis over the previous accounts is that it is genuinely recursive and hence allows us to model iterated quotation and interpretation out of the box. For example, since the interpretation of a quoted expression is itself always an expression, we are able to interpret that in turn (unless it's of type $*$). In addition we can represent and interpret quotations of quotations. Combined:

$$(28)\qquad \left[\!\!\left[\left[\!\!\left[\left[\!\!\left[\begin{array}{c}\text{``John''} : u\\ | \\ \text{`John'} : u \\ | \\ \text{John} : e\end{array}\right]\!\!\right]\right]\!\!\right]\right]\!\!\right] = \left[\!\!\left[\left[\!\!\left[\begin{array}{c}\text{`John'} : u \\ | \\ \text{John} : e\end{array}\right]\!\!\right]\right]\!\!\right] = [\![\text{John} : e]\!] = \text{John}$$

In the other direction, we can also still quote interpretations of quoted expressions. In short, quotation and interpretation are (almost) each other's inverse operations. We can prove that the following schemas are logically valid. (For readability (29) uses a sloppy one-dimensional notation, with types suppressed).

(29) a. for all $\alpha \in \mathscr{L}_{syn}$: $[\![`\alpha']\!] = \alpha$
b. for all $\alpha \in \mathscr{L}_{syn}$ of type u: $`[\![\alpha]\!]' = \alpha$

That these schemas hold for all α in all models, means that there is a fundamental difference between the statement that *'John'* refers to *John*, and the statement that *John* refers to John. The former is true by virtue of the grammar of quotation, i.e. the rule of disquotation; the latter is true by virtue of the particular chosen model and it's at best a necessary truth (following Kripke 1980), not a logical or a priori truth. We could see this as further proof that the disquotational theory satisfies Davidson's desideratum that quotations do not merely name or describe their referents by lexical stipulation.

7 Conclusion

Let's review the results of our investigations.

Proper name theory Analyzing quotations as names gives a straightforward account of the opacity of quotation without giving up compositionality. However, putting everything in an infinite lexicon we lose productivity and hence learnability. With minor adjustments we can make sense of iterated quotations, and iterated interpretation of quotations (e.g. $[\![[\![[\![\text{“}\textit{John}\text{”}]\!]]\!]]\!] = [\![[\![\text{‘}\textit{John}\text{’}]\!]]\!] = \ldots = \text{John}$).

Description theory Systematically breaking a larger quotation up into smaller quotations, down to the level of words or even letters, allows for a more manageable, finite lexicon extension. In addition to compositionality and iterated quotation, such description theories do justice to productivity. Iterated interpretation however becomes problematic.

Demonstrative theory Davidson argues that *'John'* is not just some name given to the expression *John* in the same way that *John* is a name given to John. Rather, quotation marks themselves are referential terms, referring to an expression by pointing at a token of it. Formalizing this in a Kaplanian system with an additional context update mechanism reveals some hurdles which can be overcome by essentially incorporating the quoted expression in the conventional meaning of a quotation. Compositionality would then fail, but productivity is still satisfied. One major drawback of the resulting system is that it can't handle iterated interpretation well.

Disquotational theory The disquotational principle is a recursive, syntactic rule that says that we can turn any well-formed expression into a complex expression referring to that expression by putting quotation marks around it. Taken as an analysis of quotation it immediately accounts for opacity, productivity and recursivity, but is not strictly speaking compositional.

So which analysis is best? If compositionality is the be all and end all of semantics, we had best go with a proper name theory, or, even better, a spelling theory. But on the basis of Davidson's criticism about the obvious non-arbitrary nature of quotational reference as opposed to proper name reference, we should opt for a demonstrative or disquotational analysis.

Ultimately, the answer may depend on how well the analysis can be extended to other quotational phenomena. Cappelen & Lepore (1997) argue that only the demonstrative account extends naturally to the ubiquitous phenomenon of mixed quotation (*John complained that he was being "misunderestimated"*). However, more recent accounts of mixed quotation in formal semantics rely on a disquotational theory (Geurts & Maier 2005, Potts 2007, Maier 2014). And pure quotation and mixed quotation are just the beginning – ultimately we have to consider also direct discourse, free indirect discourse, scare quotes, . . .

References

Cappelen, Herman & Ernest Lepore. 1997. Varieties of quotation. *Mind* 106(423). 429–450. http://dx.doi.org/10.1093/mind/106.423.429.

Cappelen, Herman & Ernest Lepore. 2012. Quotation. *The Stanford Encyclopedia of Philosophy* http://plato.stanford.edu/archives/spr2012/entries/quotation/.

Clark, Herbert & Richard Gerrig. 1990. Quotations as Demonstrations. *Language* 66(4). 764–805. http://www.jstor.org/stable/414729.

Davidson, Donald. 1979. Quotation. *Theory and Decision* 11(1). 27–40. http://dx.doi.org/10.1007/BF00126690.

Frege, Gottlob. 1892. Über Sinn und Bedeutung. *Zeitschrift fur Philosophie und philosophische Kritik* 100(1). 25–50.

Gaskin, Richard & Daniel J. Hill. 2013. Reach's Puzzle and Mention. *Dialectica* 67(2). 201–222. http://dx.doi.org/10.1111/1746-8361.12021.

Geach, Peter Thomas. 1957. *Mental acts: their content and their objects*. Taylor & Francis.

Geurts, Bart & Emar Maier. 2005. Quotation in Context. *Belgian Journal of Linguistics* 17(1). 109–128. http://dx.doi.org/10.1075/bjl.17.07geu.

Kaplan, David. 1989. Demonstratives. In Joseph Almog, John Perry & Howard Wettstein (eds.), *Themes from Kaplan*, 481–614. New York: Oxford University Press.

Kripke, Saul. 1980. *Naming and Necessity*. Cambridge: Harvard University Press.

Maier, Emar. 2014. Mixed Quotation: The Grammar of Apparently Transparent Opacity. *Semantics and Pragmatics* (to appear). http://semprag.org.

Pagin, Peter & Dag Westerståhl. 2010a. Compositionality I: Definitions and Variants. *Philosophy Compass* 5(3). 250–264. http://dx.doi.org/10.1111/j.1747-9991.2009.00228.x.

Pagin, Peter & Dag Westerståhl. 2010b. Compositionality II: Arguments and Problems. *Philosophy Compass* 5(3). 265–282. http://dx.doi.org/10.1111/j.1747-9991.2009.00229.x.

Pagin, Peter & Dag Westerståhl. 2011. Pure quotation and general compositionality. *Linguistics and Philosophy* 33(5). 381–415. http://dx.doi.org/10.1007/s10988-011-9083-8.

Partee, Barbara. 1973. The syntax and semantics of quotation. In S. Anderson & Paul Kiparsky (eds.), *A Festschrift for Morris Halle*, 410–418. New York: Holt, Rinehart and Winston.

Potts, Christopher. 2007. The Dimensions of Quotation. In Chris Barker & Pauline Jacobson (eds.), *Direct compositionality*, 405–431. New York: Oxford University Press.

Predelli, Stefano. 2008. The demonstrative theory of quotation. *Linguistics and Philosophy* 31(5). 555–572.

Quine, Willard Van Orman. 1940. *Mathematical Logic*. Cambridge: Harvard University Press.

Quine, Willard Van Orman. 1953. Three grades of modal involvement. *Proceedings of the XIth International Congress of Philosophy* .

Quine, Willard Van Orman. 1960. *Word and object*. Cambridge: MIT press.

Recanati, François. 2001. Open Quotation. *Mind* 110(439). 637–687. http://dx.doi.org/10.1093/mind/110.439.637.

Richard, Mark. 1986. Quotation, grammar, and opacity. *Linguistics and Philosophy* 9(3). 383–403. http://dx.doi.org/10.1007/BF00630275.

Saka, Paul. 1998. Quotation and the use-mention distinction. *Mind* 107(425). 113. http://dx.doi.org/10.1093/mind/107.425.113.

Saka, Paul. 2013. Quotation. *Philosophy Compass* 8(10). 935–949. http://dx.doi.org/10.1111/phc3.12069.

Schlenker, Philippe. 2011. Indexicality and De Se Reports. In Klaus von

Heusinger, Claudia Maienborn & Paul Portner (eds.), *Semantics: An international handbook of natural language meaning*, 1561–1604. The Hague: De Gruyter. http://dx.doi.org/10.1515/9783110255072.1561.

Tarski, Alfred. 1933. The concept of truth in formalized languages. In J. Corcoran (ed.), *Logic, semantics, metamathematics*, 152–278. Indianapolis: Hackett.

Tarski, Alfred. 1944. The semantic conception of truth and the foundations of semantics. *Philosophy and Phenomenological Research* 4(3). 341–376.

Werning, Markus. 2005. Right and wrong reasons for compositionality. *The Compositionality of Meaning and Content: Foundational Issues* 1. 285–309.

Werning, Markus, Wolfram Hinzen & Edouard Machery. 2012. *The Oxford Handbook of Compositionality*. Oxford: OUP.

Zimmermann, Thomas Ede. 1991. Kontextabhängigkeit. In Arnim von Stechow & Dieter Wunderlich (eds.), *Semantik: Ein internationales Handbuch der zeitgenössischen Forschung*, 156–229. Berlin/New York: Walter de Gruyter.