

# Pure Quotation, Compositionality, and the Semantics of Linguistic Context

Peter Pagin  
Stockholm University

Dag Westerståhl  
University of Gothenburg

March 12, 2010

## 1 The problem

A straightforward treatment of quotation cannot be compositional. Consider, as we shall in this paper, the simplest case: *pure* quotation in *written language* by means of *quote marks*,<sup>1</sup> as in

- (1)
  - a. ‘Cicero’ has six letters
  - b. ‘farfalla’ is Italian for butterfly
  - c. ‘str jd e’ is not a sentence
  - d. ‘ℵ’ is a Hebrew letter

Although Cicero is Tully, and thus ‘Cicero’ and ‘Tully’ are synonymous (taking the reference of names as their meaning), (1a) is not synonymous with

- (2) ‘Tully’ has six letters

This ‘opacity’ of quotation is a direct counter-example to compositionality, as long as the *quoted phrase* is a part of *quoting phrase* (the quoted phrase surrounded by quote marks).

Indeed, non-compositionality is a main motive behind the wide variety of theories attempting to *reanalyze* quotation so that the quoted phrase is not a constituent of the quoting phrase. For example, the *proper name theory* (Tarski [30], Quine [23]) treats the quoting phrase as atomic or unanalyzable, no more containing the quoted phrase than the word ‘Quine’ contains ‘in’. Other theories have analyzed quoting phrases as *descriptions* of how to form the quoted phrase from atomic expressions (letters or words) by means of concatenation (Tarski [31], Quine [24], Geach [8]).<sup>2</sup> Especially popular among philosophers

---

<sup>1</sup>We shall consistently use single quotes.

<sup>2</sup>See Cappelen & Lepore [2] for a useful survey of theories of quotation. As these authors note, the proper name theory and the description theory fail to do justice to essential features of quotation in natural languages (this was however not the ambition of Tarski or Quine), and are now out of fashion (but see Werning [33] for a recent attempt to revive the description theory).

is Davidson’s *demonstrative* theory in [4], which takes (the occurrence of) the quote marks to *be* a demonstration of a token, the type of which is what the sentence is about.<sup>3</sup> Thus, although reference to the quoted expression occurs on this analysis, it is performed by a demonstrative device — the quote marks themselves — not by an expression containing the quoted expression as a part. There is a vast literature on the demonstrative theory; among recent works, see, for example, Cappelen & Lepore [1] and Predelli [22].

Another popular approach, called the *use theory* or — misleadingly — the *identity theory*, takes quotation to be a different *use* of words than the normal one, and quote marks (if they occur) are *indicators* of this use. The idea goes back to the medieval distinction between *suppositio materialis* and *suppositio formalis*. In modern times it occurs with Frege (see below); more recent variants can be found with Searle [29], Washington [32], Saka [28], Recanati [25], [26], and many others. This idea is not necessarily incompatible with approaches taking quoting expressions to name quoted expressions, but many of its proponents take it to be radically different.<sup>4</sup>

One may discern some recent dissatisfaction with complicated analyses of what at least seems to be — in the pure case — a very simple phenomenon: quote marks are a productive and transparent device for forming names of linguistic entities, names that refer to these entities in just the same way as other names refer to non-linguistic entities. For example, this eliminates any problem whatsoever with *iterated* quotation, something which at least is an issue with other approaches.<sup>5</sup> Intuitively, this seems correct: there is no difference between quoting expressions surrounded by quote marks and quoting any other expressions. One such *straightforward* or *disquotational* analysis is elegantly presented in Potts [21]; indeed Potts applies it not just to pure quotation but also to cases of direct and indirect discourse as well as mixed cases.

But with the straightforward analysis, the problem of compositionality returns. On this point, Potts is somewhat vague, but he does seem to claim that

---

<sup>3</sup>For example, (1b) is analyzed as

- (i) farfalla. That / The expression of which this is a token is Italian for butterfly

<sup>4</sup>E.g. Searle [29]. Recanati [25], [26] combines a pragmatic and a referential view: Pure quotations (*closed*, in his terminology) are referential singular terms, and he explicitly raises the issue of compositionality of the corresponding semantics, i.e. the issue the present paper is concerned with. *Open* quotations (e.g. indirect or mixed cases), on the other hand, do not refer; the words have their ordinary meaning but the *speech act* is different.

Some thoroughly pragmatic accounts (Clark and Gerrig [3]) deny that pure quotation is ever done by exhibiting the quoted object within some quotational device. The idea is instead that quotation *depicts selected aspects* of the object, as when I quote what someone said by using a written sentence which however leaves out his stuttering, his thick London accent, etc. We shall only deal with quotation of written text here.

<sup>5</sup>It is not obvious how Davidson’s original account indicated in footnote 3 can deal with *demonstrations within demonstrations*, as in

- (i) “farfalla” refers to ‘farfalla’,

and although Cappelen & Lepore [1], Predelli [22], and others suggest mechanisms for doing this, these are certainly more complex than the straightforward approach.

his account is compositional, despite the fact that, for the well-known reasons, it *cannot* be! Neither in Potts’ account, nor in any other account where quote marks is a syntactic operation whose semantic effect is to produce a name of the quoted expression, can you substitute the quoted expression for another one with the same meaning (referent) and expect the meaning of the complex expression (or the sentence where it occurs) to stay the same — if the account is to be empirically adequate. But precisely this fact means that straightforward accounts are not compositional.<sup>6</sup>

In this paper we show, however, that there is an extended sense of compositionality in which the straightforward account *is* compositional, a sense which we need to countenance anyway in order to give compositional accounts of *context-dependence*. To take a familiar example, the standard notion of compositionality applies directly to Kaplan’s notion of *character*. Character is assigned to expressions, so we can ask if the character of a complex expression is determined by the characters of its immediate constituents (and the mode of composition). But this doesn’t immediately apply to *content*, since content is assigned not to an expression but to an expression *and* a context. Nevertheless, no one takes this fact alone to show that content is not compositional. Indeed, Kaplan himself states explicitly in [14] that his account of content *is* compositional, and Lewis in [16] makes compositionality a necessary requirement on any assignment of semantic values, including content. But, although only implicitly acknowledged by these authors, this requires an extended notion of *contextual compositionality*, one that takes the presence of the extra context argument into

---

<sup>6</sup> Potts’ grammar deals with triples  $\langle \Pi, \Sigma, \alpha : \sigma \rangle$ , where  $\Pi$  is a phonological representation,  $\Sigma$  a syntactic representation (category), and  $\alpha$  a semantic representation of type  $\sigma$ . His grammar rules operate on such triples, although the action on each representation is independent of that on the other representations, and thus can be stated separately. He doesn’t define compositionality, but it seems plausible to take such a grammar to be compositional iff each of the component functions is compositional in the usual sense. (E.g. Kracht [15] defines compositionality for triples in this way. The relation between this notion of compositionality and the standard one is spelled out in Pagin & Westerstahl [18], Section 3.6.) In particular, the semantic interpretation function  $\llbracket \cdot \rrbracket$  must be compositional. *But it isn’t*: quotation is a syntactic operation in Potts’ grammar, with the corresponding semantic rule (p. 410)

$$\llbracket \langle \Pi, \Sigma, \alpha : \sigma \rangle \rrbracket = \langle \Pi, \Sigma, \alpha : \sigma \rangle$$

so substituting e.g. synonymous names will not in general preserve meaning.

Still, Potts says explicitly (p. 406) that his account doesn’t threaten the basic tenets of directly compositional semantics (one such tenet presumably being compositionality), a claim which is reemphasized in the conclusion of the paper (p. 426).

A possible source of confusion here might be the distinction between a *recursive* and a *compositional* semantics. Pott’s grammar is perfectly recursive: his rules allow you to *compute* the meanings of any well-formed complex expression (triple) in a finite number of steps. But this is not enough for compositionality, which requires that meanings of complex expressions be determined by the *meanings* of their immediate constituents (and the rule applied), whereas a recursive definition allows you to use not only those meanings but *the constituents themselves* as arguments of the composition operation. (In addition, it requires, in contrast with compositionality, that the composition operation is recursive; see Pagin & Westerstahl [18], Section 3.2.) This feature is indeed essential for the formulation of the semantic rule for quotation, but, as we have been at pains to emphasize, the result is that substitution of synonyms does not preserve meaning, i.e. compositionality fails.

account.<sup>7</sup>

Our claim in this paper is that with a corresponding notion of *linguistic context*, the straightforward account of quotation can be made compositional, in the extended, contextual sense. We shall not here discuss the intrinsic merits of compositionality. It is of course possible to make an exception for quotation, and content oneself with a recursive but non-compositional semantics for a straightforward treatment of this phenomenon, as Potts in fact does (see footnote 6). That would be consistent with the view that quotation is a deviant or abnormal linguistic phenomenon, without much interest. Such a view is not uncommon in the literature, but it is not Potts' view; on the contrary, he thinks quotation is a 'hugely important matter for linguistic theory' ([21], p. 425). Likewise, Cappelen and Lepore find it 'one of the most difficult and interesting topics in the philosophy of language' ([2], opening paragraph).

We agree that the way speakers 'talk about the words themselves' (see the quote from Frege below) is of significant interest, and we shall here take it for granted that a compositional account is worthwhile. And just as the presence of indexicals and demonstratives requires an extension of the notion of compositionality to handle extra-linguistic context, the presence of (pure) quotation requires an extension of the notion of compositionality to linguistic context. Moreover, the approach suggested here applies not just to quotation, but extends to several other familiar linguistic constructions, as we shall indicate.

Our approach to quotation takes inspiration from Frege's familiar comment:

If words are used in the ordinary way, what one intends to speak of is their reference. It can also happen, however, that one wishes to talk about the words themselves or their sense. This happens, for instance, when the words of another are quoted. One's own words then first designate words of the other speaker, and only the latter have their usual reference. We then have signs of signs. In writing, the words are in this case enclosed in quotation marks. Accordingly, a word standing between quotation marks must not be taken to have its ordinary reference. (Frege [6], pp. 58–9)

As Cappelen & LePore [2] note, this passage, which seems to be just about all Frege ever said about quotation, is probably too cryptic to be taken as a clear proposal.<sup>8</sup> But we shall interpret it as a *combination* of the use theory and the straightforward theory. It is straightforward in that the quoting phrase refers to the quoted phrase, but it is a use theory in that the quoted phrase itself, in the quotation context, has another meaning (reference) than it usually has: it refers to itself. So the quote marks are a syntactic operator, but they also indicate a context change. There is nothing strange, it seems to us, in a syntactic operator also having this context-changing function; in fact, we shall suggest that there are other operators working in the same way.

---

<sup>7</sup>One reading of Kaplan's principle (F2) ([14], p. 507) is a substitutional formulation of such a principle of contextual compositionality. There are actually two distinct natural versions of it; see Section 3 below.

<sup>8</sup>See Parsons [20] for a discussion of possible interpretations.

We begin, in the next section, by presenting standard compositionality at a sufficiently abstract and precise level, and, in the section after that, the extension of this concept to deal with non-linguistic context. Then we discuss linguistic context, in a setting which is more general than what is needed for quotation contexts; it also applies to belief contexts, for example. Armed with a precise notion of linguistic context, we propose a corresponding version of contextual compositionality. We also suggest a way to make precise the conditions under which an ordinary semantics can be said to generalize to a semantics which is compositional in this sense. Finally, we show that the approach to quotation we take Frege to recommend is a compositional generalization in the proper sense.<sup>9</sup>

## 2 Ordinary compositionality

Compositionality is not a demand for a certain format of the grammar or the syntax-semantics interface. It is a condition that may or may not apply to any grammar in (almost) any format. This is clear already from the informal formulation of the compositionality principle, in either its functional or its substitutional version:

- (PoC-F) The meaning of a complex expression is determined by the meanings of its immediate constituents and the mode of composition.
- (PoC-S) Appropriate substitution of synonymous constituents (not necessarily immediate) preserves meaning.

These formulations are of course relative to given specifications of *meaning*, *constituent*, etc., but they are independent of *how* those specifications are made. For any assignment of *semantic values* to any set of *structured expressions* (expressions equipped with a notion of constituent or sub-expression), the issue of compositionality can be raised. Indeed, for (PoC-S) we don't even need to specify what the values are, only when two expressions have the *same* value. It follows that general facts about compositionality, such as the ones about quotation established in this paper, are (to a large extent) independent of particular formats chosen for syntax and semantics.

### 2.1 A formal framework

To make (PoC-F) and (PoC-S) precise we shall use the following framework.<sup>10</sup> Think of (surface) expressions as *strings*, and complex expressions as *generated* from *atomic* ones via grammar rules, which we simply take to be *partial functions* from tuples of expressions to expressions. There is no requirement on what these functions are; in particular, they are not restricted to concatenation

<sup>9</sup>A sketch of our approach and its application to quotation appeared in Pagin & Westerståhl [18] and [19].

<sup>10</sup>See Hodges [13] or Westerståhl [36] for detailed accounts including proofs of the basic facts about compositionality mentioned here, and Pagin and Westerståhl [18] and [19] for further discussion and motivation.

of strings. Partiality is used to respect syntactic categories: rather than taking these as primitive, we let grammar rules be *undefined* for arguments of the wrong kind. To illustrate, rules like

$$\begin{aligned} \text{NP} &\longrightarrow \text{Det N} \\ \text{S} &\longrightarrow \text{NP VP} \end{aligned}$$

correspond to binary partial functions, say  $\alpha, \beta$ , such that, if *few*, *cat*, and *bark* are atoms, one derives the complex expression (sentence) *Few cats bark*, by first applying  $\alpha$  to *few* and *cat*, and then applying  $\beta$  to the result of that and *bark*. These functions are necessarily partial; for example,  $\beta$  is undefined whenever its second argument is *cat*.

Thus, let a *grammar*

$$\mathbf{E} = (E, A, \Sigma)$$

be a partial algebra, where  $E$  is a set of expressions (strings),  $A$  its subset of atoms, and  $\Sigma$  is a set that, for each required  $n \geq 1$ , has a subset of partial functions from  $E^n$  to  $E$ , and is such that  $E$  is generated from  $A$  via  $\Sigma$ .

One and the same expression may be generated in more than way, i.e. the grammar may allow *structural ambiguity*. Also, a semantically relevant element need not be represented in the surface expression. For these reasons, it is not the expressions in  $E$  but rather their *derivation histories* (‘analysis trees’), that are assigned semantic values. These derivation histories can be represented by the *terms* in the *partial term algebra* corresponding to  $\mathbf{E}$ . The sentence *Few cats bark* is a string, but its derivation history is represented by the term

$$(3) \quad t = \beta(\alpha(\textit{few}, \textit{cat}), \textit{bark})$$

in the term algebra. *Grammatical terms* are those where all the functions involved are defined for the respective arguments. So  $t$  is grammatical but  $\beta(\alpha(\textit{few}, \textit{cat}), \textit{cat})$  is not. Let  $GT_{\mathbf{E}}$  be the set of grammatical terms of  $\mathbf{E}$ .

We shall need to distinguish between the *functions*  $\alpha, \beta, \dots$  taking expressions to expressions and the *names* of these functions used as operations in the term algebra. The convention here will be that  $\bar{\alpha}$  is a name of  $\alpha$ . Thus,

$$\alpha(\textit{few}, \textit{cat}) = \textit{few cats}$$

is an element of  $E$  — the string *few cats*,<sup>11</sup> but the corresponding *term* is a formal expression in the term algebra. So rather than (3), we shall write, from now on,

$$(4) \quad t = \bar{\beta}(\bar{\alpha}(\overline{\textit{few}}, \overline{\textit{cat}}), \overline{\textit{bark}})$$

In other words, the  $\bar{\alpha}$  for  $\alpha \in \Sigma$  are the operations in the term algebra corresponding to  $\mathbf{E}$ . If we obey the constraints coming from the partiality of

<sup>11</sup>More correctly, we should write the string value of  $\alpha(\textit{few}, \textit{cat})$  as  $\textit{few} \hat{\ } \textit{cats}$ , where ‘ $\hat{\ }$ ’ denotes word space and ‘ $\hat{\ }$ ’ concatenation, but the simplified notation used here is easier to read.

the functions in  $\Sigma$ , we get the grammatical terms. Forgetting those constraints we obtain the set  $T_{\mathbf{E}}$  of arbitrary *terms*, obtained by successively applying the operations  $\bar{\alpha}$  to any (term) arguments.

Each grammatical term maps to a unique string in  $E$ . In other words, there is a *string value function*  $V$  from  $GT_{\mathbf{E}}$  to  $E$ . For an atomic term like  $\overline{few}$ , we have  $V(\overline{few}) = few$ . To handle lexical ambiguity, we need distinct atomic terms with the same value, say,  $V(\overline{bank_1}) = V(\overline{bank_2}) = bank$ . For a complex term  $\bar{\alpha}(t_1, \dots, t_n)$  we have

$$V(\bar{\alpha}(t_1, \dots, t_n)) = \alpha(V(t_1), \dots, V(t_n))$$

where  $\alpha$  is defined for the arguments  $V(t_1), \dots, V(t_n)$  precisely when the term  $\bar{\alpha}(t_1, \dots, t_n)$  is grammatical.<sup>12</sup> To illustrate:

$$\begin{aligned} V(\bar{\beta}(\bar{\alpha}(\overline{few}, \overline{cat}), \overline{bark})) &= \beta(V(\bar{\alpha}(\overline{few}, \overline{cat})), V(\overline{bark})) \\ &= \beta(\alpha(V(\overline{few}), V(\overline{cat})), bark) \\ &= \beta(\alpha(few, cat), bark) \\ &= \beta(few\ cats, bark) \\ &= few\ cats\ bark \end{aligned}$$

Now we can let a *semantics* for  $\mathbf{E}$  simply be a partial function  $\mu$  from  $GT_{\mathbf{E}}$  to some set  $X$  of semantic values ('meanings'). The domain of  $\mu$  can be a proper subset of  $GT_{\mathbf{E}}$  if, for example, some grammatical terms have no semantic value because no reasonable meaning fits them (eg. *colorless green ideas*, according to some), or because none has yet been found (say, the language is still partly unknown to the theorist).

As noted, for compositionality the nature of the meanings doesn't matter, only sameness of meaning.  $\mu$  induces a partial equivalence relation (synonymy) on  $E$ : for  $u, t \in E$ , define

$$u \equiv_{\mu} t \text{ iff } \mu(u), \mu(t) \text{ are both defined and } \mu(u) = \mu(t)$$

## 2.2 Compositionality

For a given grammar  $\mathbf{E}$  and semantics  $\mu$ , we render (PoC-F) as follows:

**Func $t(\mu)$**  For every rule  $\alpha \in \Sigma$  there is a meaning operation  $r_{\alpha}$  such that if  $\bar{\alpha}(u_1, \dots, u_n)$  has meaning (belongs to the domain of  $\mu$ ), then  $\mu(\bar{\alpha}(u_1, \dots, u_n)) = r_{\alpha}(\mu(u_1), \dots, \mu(u_n))$ .

Here is the formal version of (PoC-S):

**Subst $(\equiv_{\mu})$**  If  $s[u_1, \dots, u_n]$  and  $s[t_1, \dots, t_n]$  are both meaningful terms, and if  $u_i \equiv_{\mu} t_i$  for  $1 \leq i \leq n$ , then  $s[u_1, \dots, u_n] \equiv_{\mu} s[t_1, \dots, t_n]$ .

---

<sup>12</sup>I.e.  $V$  is a homomorphism from the term algebra to the expression algebra  $\mathbf{E}$ .

The notation  $s[u_1, \dots, u_n]$  indicates that the term  $s$  contains — not necessarily immediate — *disjoint* occurrences of subterms among  $u_1, \dots, u_n$ , and  $s[t_1, \dots, t_n]$  results from replacing each  $u_i$  by  $t_i$ .

Note that the formulation  $\text{Funct}(\mu)$  presupposes the *Domain Principle* (DP): Subterms of meaningful terms are meaningful. The following is well-known (e.g. Hodges [13]):

**Fact 1**

*Under DP,  $\text{Funct}(\mu)$  is equivalent to  $\text{Subst}(\equiv_\mu)$ .*

In some cases, DP seems too strong. For example, a semantics for first-order logic assigning meanings to closed but not to open formulas doesn't satisfy DP. Likewise, a semantics for quotation that allows quotation of meaningless strings, as in (1c), may appear to violate DP. We will see, however (Sections 5 and 7), that when linguistic context is taken into account, DP is a less demanding requirement, and indeed our account handles quotation of arbitrary strings while satisfying a generalized version of DP.

$\text{Funct}(\mu)$  and  $\text{Subst}(\equiv_\mu)$ , like their informal counterparts (PoC-F) and (PoC-S), express the core meaning of compositionality. For compositionality to have practical import, one also needs to require that the meaning operations  $r_\alpha$  are *computable* in some suitable sense. There are several ways in which basic compositionality can be strengthened.<sup>13</sup> But note that (PoC-F) — by far the most common informal version of the principle in the literature — only says that the meaning of a complex phrase is *determined* by the meanings of its constituents (and the mode of composition); it says nothing about *how* it is determined. And the equivalent (PoC-S) doesn't even mention functionality or determination.

Although  $\text{Funct}(\mu)$  and  $\text{Subst}(\equiv_\mu)$  without extra conditions are weak, they are not trivial: there are grammars and semantics for which they fail (e.g. Potts' semantics for quotation mentioned in footnote 6.) It is another matter that any semantics  $\mu$  for  $\mathbf{E}$  is *recoverable* from some compositional semantics  $\mu'$ ; for a simple example, let

$$(5) \quad \mu'(t) = \langle \mu(t), t \rangle \text{ (when defined)}$$

The function  $\mu'$  is one-one, so  $\text{Subst}(\equiv_{\mu'})$  holds trivially. But this in no way makes  $\text{Subst}(\equiv_\mu)$  trivial.<sup>14</sup>

---

<sup>13</sup>See Pagin and Westerståhl [18] for a survey. Compositionality can also be weakened, for example, by only requiring that the meaning of a complex expression is determined by the meanings of its *atomic* constituents.

<sup>14</sup>See Westerståhl [34], and for further discussion of the alleged triviality of compositionality, Pagin and Westerståhl [19]. As pointed out in footnote 6, the natural condition of compositionality for semantics like  $\mu'$  is not  $\text{Funct}$  or  $\text{Subst}$ , but rather these conditions applied to each of the components.

### 3 Extra-linguistic context dependence

Context-dependence is ubiquitous in natural languages, and there is a current debate on how much of these phenomena are amenable to a systematic or compositional treatment. The simplest case is that of *basic indexicals*. A sentence  $\varphi$  containing such indexicals does not express a complete proposition, something that is true or false, but *utterances* or *occurrences* of  $\varphi$  do. The *context* of the utterance or occurrence is needed to fix the reference of the indexicals. Thus, we can distinguish the linguistic meaning of the sentence (type), and the content or proposition that various uses of it express. But what does it mean to say that this kind of linguistic meaning is or is not compositional? Without commitment to any view of what contexts are, the abstract situation is as follows.

In addition to our set  $GT_{\mathbf{E}}$  of terms corresponding to structured expressions, and our set  $X$  of semantic values, there is a set  $C$  of contexts.<sup>15</sup> We can then think of the assignment of values in two ways:

$$(6) \quad \begin{array}{l} \text{a. } \nu: GT_{\mathbf{E}} \longrightarrow [C \longrightarrow X] \\ \text{b. } \mu: GT_{\mathbf{E}} \times C \longrightarrow X \end{array}$$

Here  $f: A \longrightarrow B$  means that  $f$  is a (partial) function from the set  $A$  to the set  $B$ ,  $[A \longrightarrow B]$  is the set of such functions, and  $A \times B$  is the set of pairs  $(a, b)$  such that  $a \in A$  and  $b \in B$ . So  $\nu$  is a meaning assignment of the kind treated above: it assigns values directly to terms, although these values are now themselves functions. It corresponds to what Kaplan called *character*.  $\mu$ , on the other hand, assigns values to ‘expressions-in-context’. Is this more than a notational difference, more than two ways to think of character?

It is trivial to *define*  $\mu$  in terms of  $\nu$ , and vice versa, by the equation

$$\mu(t, c) = \nu(t)(c)$$

In mathematical parlance,  $\nu$  is the *currying* of  $\mu$ , and  $\mu$  is the *uncurrying* of  $\nu$ . Nevertheless, compositionality for the two functions is not the same. Indeed, the notion of compositionality from the preceding section applies only to  $\nu$ .  $\text{Funct}(\nu)$  is well-defined, but for  $\mu$  we must define compositionality for semantics that take an extra context argument.

There is an obvious way to do this: think of the previous condition as *parametric* in the context. Just replace  $\mu(t)$  with  $\mu(t, c)$  everywhere. We obtain:<sup>16</sup>

<sup>15</sup>Note that  $X$  may in turn consist of functions. For example, in standard possible-world semantics,  $X$  contains *intensions*, which are functions from possible worlds to ordinary extensions. In relativist variants, these functions have further arguments, such as times, locations, assessors, standards, etc. We then get analogous issues for the compositionality of intension, and how such compositionality is related to that discussed below. These questions are treated in Westerståhl [37], but they will not be important here.

<sup>16</sup>We make the simplifying assumption that the non-linguistic context  $c$  is the same for all parts of a complex term. This is sometimes violated; for example, when the sergeant, pointing in turn at three of the recruits, shouts

- (i) You, you, and you are volunteers!

C-Funct( $\mu$ ) For every rule  $\alpha \in \Sigma$  there is a meaning operation  $r_\alpha$  such that for every context  $c$ , if  $\bar{\alpha}(u_1, \dots, u_n)$  and  $u_1, \dots, u_n$  have meaning in  $c$ , then  $\mu(\bar{\alpha}(u_1, \dots, u_n), c) = r_\alpha(\mu(u_1, c), \dots, \mu(u_n, c))$ .

Now, one might think that this is simply equivalent to Funct( $\nu$ ). But it isn't. *First*, it is easy to give examples where Funct( $\nu$ ) holds but C-Funct( $\mu$ ) fails.. *Second*, there is no corresponding substitution version of C-Funct( $\mu$ ). There is one for *immediate* subterms: C-Funct( $\mu$ ) is equivalent to

$$(7) \quad \text{If } \bar{\alpha}(u_1, \dots, u_n) \text{ and } \bar{\alpha}(t_1, \dots, t_n) \text{ are both meaningful, and if } \mu(u_i, c_1) = \mu(t_i, c_2) \text{ for } 1 \leq i \leq n, \text{ then } \mu(\bar{\alpha}(u_1, \dots, u_n), c_1) = \mu(\bar{\alpha}(t_1, \dots, t_n), c_2).$$

But this does not extend to subterms that are more deeply embedded: this would require that the subterms of  $s[u_1, \dots, u_n]$  that are *not* replaced have the same meaning in  $c_1$  and  $c_2$ , and there is no guarantee for that.

*Third*, there is a way in which C-Funct( $\mu$ ) can fail *merely by changing the context*, without replacing any expressions at all, namely, if each of  $u_1, \dots, u_n$  has the same meaning in  $c_1$  and  $c_2$  but  $\bar{\alpha}(u_1, \dots, u_n)$  doesn't. This is called *context shift failure* in Pagin [17], where possible examples are given. This kind of failure of compositionality is not available for  $\nu$ .

*Fourth*, there is a weaker variant of C-Funct( $\mu$ ), which also seems quite natural:

C-Funct( $\mu$ )<sub>w</sub> For every  $\alpha \in \Sigma$  there is a meaning operation  $r_\alpha$  such that for every context  $c$ , if  $\bar{\alpha}(u_1, \dots, u_n)$  and  $u_1, \dots, u_n$  have meaning in  $c$ , then  $\mu(\bar{\alpha}(u_1, \dots, u_n), c) = r_\alpha(\mu(u_1, c), \dots, \mu(u_n, c), c)$ .

The only difference here is that the meaning operation  $r_\alpha$  takes  $c$  as an extra argument. Again, this might seem insignificant, but it isn't. Context shift failure cannot occur for this version. Various semantic theories for which C-Funct( $\mu$ )<sub>w</sub> holds but C-Funct( $\mu$ ) fails have been proposed. Furthermore, this time *there is* a simple substitution version, for arbitrary subterms:

C-Subst( $\equiv_\mu$ )<sub>w</sub> If  $s[u_1, \dots, u_n]$  and  $s[t_1, \dots, t_n]$  are both meaningful terms, and if  $\mu(u_i, c) = \mu(t_i, c)$  for  $1 \leq i \leq n$  (i.e. both sides are defined and have the same value), then  $\mu(s[u_1, \dots, u_n], c) = \mu(s[t_1, \dots, t_n], c)$ .

Moreover, we shall see that it is this version of contextual compositionality that lends itself to adaptation for linguistic contexts.

The following result explains the logical relations between the notions of compositionality discussed so far, with  $\mu$  and  $\nu$  as in (6) (see Westerståhl [37] for a proof and discussion):

**Fact 2**

C-Funct( $\mu$ ) implies C-Funct( $\mu$ )<sub>w</sub> (equivalently, C-Subst( $\equiv_\mu$ )<sub>w</sub>), which in turn

---

We shall not pursue this here. When we come to linguistic context, however, it is a crucial feature that the context of the immediate subterms of a term can be different from the context of the term itself in a way that is semantically significant (see Section 4).

implies  $\text{Funct}(\nu)$  (equivalently,  $\text{Subst}(\equiv_\nu)$ ), but none of these implications can be reversed.

## 4 Linguistic context

Often the interpretation of a linguistic expression on an occasion depends on other expressions that are used (or just have been used) on the same occasion. This can happen in two main ways: by *linguistic* context-dependence proper, and also by way of the influence of linguistic expressions (used on the occasion) on the *extra-linguistic* context. We may call the latter ‘linguistic contextual impact’. Consider

(8)            That is a big mouse.

Here it may be, or it may fail to be, the case that a standard or comparison class for size is given in the extra-linguistic context. If there is, e.g. the comparison class of animals in some particular laboratory, and it applies in the case of the utterance of (8), then *big* applies intersectively with respect to *mouse*. But the occurrence of *mouse* may also set a *new standard* of size in the context, whether or not there was a standard in the immediately preceding context. In that case the complex noun *big mouse* will apply to anything that (within some relevant domain) is big for a mouse. ‘big’ still applies intersectively, but the standards have been changed. This phenomenon is worthy of further study (see e.g. Recanati [27] for a similar view), but is mentioned here only as a contrast to the present topic, that of linguistic context dependence proper.

### 4.1 Sentential contexts

What is a linguistic context? Here we restrict attention to the context of an expression within a larger expression, notably a sentence. The basic idea is simple enough: take a well-formed complex term  $s[u]$  and knock out the constituent  $u$ . What remains,  $s[. . .]$ , is the *linguistic context* of that occurrence of  $u$  in  $s[u]$ , the environment of the argument place. This simple idea needs to be refined, and we shall see how.

Standard examples of linguistic contexts that are relevant to semantics include *belief contexts*. In a belief sentence like

(9)            John believes that Jenny is hungry

the embedded sentence

(10)          Jenny is hungry

occurs in the belief context

(11)          John believes that . . .

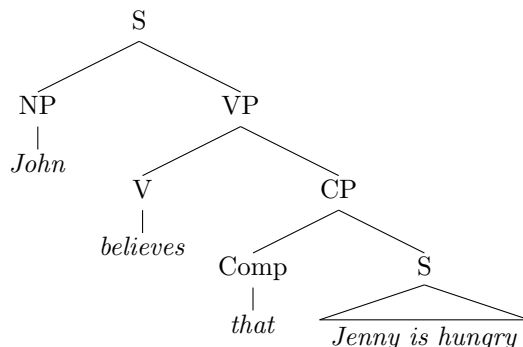


Figure 1: A phrase structure tree for (9).

According to Frege [6], that (10) occurs in a belief context has an effect on its meaning in that context: it has *indirect reference* (*ungerade Bedeutung*).

This view gives us a first reason for revising the simple idea: the occurrence of the subject *John* does not participate in defining the belief context. What matters for being in a belief context is that the expression occurs in the second argument place of the verb *believes*. What occurs in the first argument place, *John*, matters to the truth conditions of the sentence as a whole, but has no effect on the semantics for the embedded sentence itself.

For definiteness, let the construction of belief VPs to be given by two one-place rules:  $\alpha_{\text{bel}}$ , where the argument is the sentence term for the *that*-clause (Complementizer Phrase) argument, and  $\alpha_{\text{cp}}$ , which takes the embedded sentence as argument and gives the *that*-clause as value.<sup>17</sup> Figure 1 gives a simplified tree representation.

The belief context is the context of the position of the dominated S node in the tree, where the embedded sentence goes. In the term notation, this is  $\bar{\alpha}_{\text{bel}}(\bar{\alpha}_{\text{cp}}(\dots))$ . The example illustrates that we should not rule out that what matters to determining the linguistic context can extend beyond the immediate syntactic rule. In this case, two rules are required.

There is a further important complication. On a standard conception, if a complex expression occurs in a belief context, then the constituents of that expression also occur in the belief context. For instance, the name *Jenny* occurs in the belief context just like the entire sentence (10). So on this conception, being in a particular linguistic context is like being in the scope of an operator. But this is not quite right, or at least highly misleading, as can be seen by considering iterated belief contexts. In

(12) Emma believes that John believes that Jenny is hungry

the name *John* occurs in a simple belief context, but *Jenny* occurs in an iterated, and more precisely second degree, belief context. This can matter to seman-

<sup>17</sup>Various other formats are possible.

tics. On a common interpretation of Frege, expressions in second degree belief contexts have a doubly indirect reference and a doubly indirect sense.<sup>18</sup> This shows that in general we need to take into account several constituent levels in the syntactic term, and for each level the relevant argument place. Therefore it is appropriate to let linguistic context be defined in terms of operators and argument positions down to the target argument place. We now give a precise version of this idea.

## 4.2 A formal notion of linguistic context

The notion of linguistic context applies to *occurrences* of (sub)terms. We let  $o$  (with subscripts) range over occurrences.<sup>19</sup> For any term  $s$ , and any occurrence  $o$  of a subterm of  $s$ , we define inductively *the (linguistic) context of  $o$  in  $s$* ,  $cxt(o, s)$ , as a finite sequence  $\xi = \langle (\alpha_1, i_1), \dots, (\alpha_n, i_n) \rangle$  of pairs consisting of a grammar rule and a natural number. ( $\wedge$  is now concatenation of sequences.)

- ( $L_{cxt}$ )
- (i)  $cxt(s, s) = \langle \rangle$  (the *null context*).
  - (ii) If  $cxt(\bar{\alpha}(o_1, \dots, o_k), s) = \xi$  then  $cxt(o_i, s) = \xi \wedge \langle (\alpha, i) \rangle$ ,  $1 \leq i \leq k$ .
  - (iii)  $CXT_{\mathbf{E}} = range(cxt)$  (the set of *contexts* in  $\mathbf{E}$ ).

So the context of  $o$  in  $s$  encodes the *path* in the tree corresponding to the term  $s$  that starts with the node  $s$  and ends with the node  $o$ . Clearly, every subterm occurrence in  $s$  is in a unique context in  $s$ , and distinct subterm occurrences are in distinct contexts in  $s$ . The same context  $\xi$  can be a context in many different terms,<sup>20</sup> but for any grammatical term  $s$ ,  $\xi$  determines at most one subterm occurrence. If  $cxt(o, s) = \xi$ , and we replace  $o$  by another term, we obtain a term  $s'$  (which may or may not be grammatical), differing from  $s$  in having the new term in the *same* context as  $o$ , i.e.  $\xi$ .

We say that  $\xi$  is a *possible context* for  $t$  if there is a term  $s$  which has a subterm occurrence  $o$  of  $t$  such that  $cxt(o, s) = \xi$ . Thus,  $CXT_{\mathbf{E}}$  is the set of possible contexts of grammatical terms in  $\mathbf{E}$ . If  $\xi$  is a possible context for some term  $\bar{\alpha}(u_1, \dots, u_n)$ , then  $\xi \wedge \langle (\alpha, i) \rangle$  is a possible context for  $u_i$ . In this case, we say that  $\xi$  *allows extension* by  $\alpha$ .

$CXT_{\mathbf{E}}$  may be large — usually infinite — but when studying the effect of linguistic context on meaning we may be interested only in a small number of *types* of contexts. For example, Frege in the quote earlier in effect isolates

<sup>18</sup>There is some textual evidence that this, at least at some point, was Frege's own view. The following is a quote from a letter to Russell 28 December 1902:

Man kann sagen, dass wir im doppelt Unterstrichenen die ungerade Bedeutung zweiten Grades haben, in dem einmal Unterstrichenen dagegen die ungerade Bedeutung ersten Grades ([7], p. 236).

Here Frege explicitly refers to “an indirect reference of second degree”.

<sup>19</sup>We use an informal notion of a term occurrence here. It could be redefined later in terms of the function  $cxt$  introduced below.

<sup>20</sup>For example,  $\xi = \langle (\alpha, 1), (\beta, 2) \rangle$  is a context in each of the terms  $\bar{\alpha}(\bar{\beta}(t, u), \bar{\gamma}(t))$ ,  $\bar{\alpha}(\bar{\beta}(t, u), t')$ , and  $\bar{\alpha}(\bar{\beta}(t, \bar{\delta}(u)), \bar{\gamma}(t))$ . The subterm occurring at context  $\xi$  is  $u$  in the first two cases, and  $\bar{\delta}(u)$  in the third case.

three types of context: an ‘ordinary’ (null) context type (when talking about the references of words), a quotation context type (talking about the words themselves), and an ‘intensional’ context type (talking about their senses). This motivates the following definition:

( $L_{\text{ctype}}$ ) A *context typing* is a partition  $C$  of  $CXT_{\mathbf{E}}$  satisfying the following requirement, where  $[\xi]$  is the equivalence class of  $\xi$ :

- (i) If  $[\xi] = [\xi']$ , and  $\xi$  and  $\xi'$  both allow extension by an  $n$ -ary grammar rule  $\alpha$ , then,  $[\xi \wedge \langle(\alpha, i)\rangle] = [\xi' \wedge \langle(\alpha, i)\rangle]$  for  $1 \leq i \leq n$ .

The elements of  $C$  are called *context types*.

The requirement (i) says that if two contexts are of the same type, and can be extended in the same way, then the corresponding extensions also have the same type. This means that the context type of a subterm occurrence  $\bar{\alpha}(o_1, \dots, o_k)$  in  $s$  determines the context types of  $o_1, \dots, o_k$  in a way which is *independent of*  $s$ , according to the partial function  $\Phi_C$ , which takes a  $n$ -ary grammar rule  $\alpha$ , a natural number  $i$  (an argument place of  $\alpha$ ), and a context type  $c$  to a context type  $\Phi_C(\alpha, i, c)$ , and is defined as follows:

$$(13) \quad \Phi_C(\alpha, i, [\xi]) = \begin{cases} [\xi \wedge \langle(\alpha, i)\rangle] & 1 \leq i \leq n, \text{ if } \xi \text{ allows extension by } \alpha \\ \text{undefined} & \text{otherwise} \end{cases}$$

Thus,  $\Phi_C(\alpha, i, c)$  is defined if and only if some  $\xi \in c$  is a possible context for some term of the form  $\bar{\alpha}(u_1, \dots, u_n)$ .

Finally, it is straightforward to see that using  $\Phi_C$ , one can define by induction a function  $\Theta_C$  taking a context type  $c$  and a context  $\xi$  as arguments, such that:

- (14) If  $s$  occurs in context type  $c$ , and  $\xi$  is the context of an occurrence of  $t$  in  $s$ , then the context type of that occurrence is  $\Theta_C(c, \xi)$ .

## 5 Compositionality for linguistic context

The first main idea for an extension of a semantic framework to handle linguistic context dependence is that when a semantic function  $\mu$  is applied to a term  $s$ , the value depends on the relevant context type. The second main idea is that the context type of an immediate subterm of a term  $s$  may be different from the context type of  $s$  itself. As we shall see, there are two equivalent styles of implementing this: either with a semantic function that takes a context type as a second argument, or instead with a set of ordinary semantic functions, one for each context type.

### 5.1 A formulation in terms of formal contexts

The first style builds on the formal notion of context just introduced. Assume a context typing  $C$  of  $CXT_{\mathbf{E}}$  according to ( $L_{\text{ctype}}$ ) is given. Just as for extralinguistic context (Section 3), a semantics  $\mu$  now takes an additional argument:

it is a partial function from  $GT_{\mathbf{E}} \times C$ .

For simplicity, we shall also require that a Generalized Domain Principle, holds for  $\mu$ :

(GDP $_{\mu}$ ) If  $\mu(\bar{\alpha}(u_1, \dots, u_n), c)$  is defined, then  $\mu(u_i, \Phi_C(\alpha, i, c))$  is defined for  $1 \leq i \leq n$ .

In Section 2.2 we noted that DP is sometimes too demanding. When meaning is allowed to depend on linguistic context, however, the case for a (generalized) domain principle is stronger. Rather than not assigning any meaning at all to some subterms of a complex term, we can assign them a *different* meaning when they are in that context (type). In Section 7 we will see an example how this works out, in a way consistent with GDP $_{\mu}$ .

Now, the compositionality requirement is similar to C-Funct( $\mu$ ) $_w$ , the difference being that the context types of the arguments of complex terms may change; indeed the function  $\Phi_C$  assigns those types. The linguistic contexts of the arguments are different from the context of the term itself, and that difference may be semantically relevant. For example, the context type of the complex term may be a quotation type.

LC-Funct( $\mu, C$ ) For every  $\alpha \in \Sigma$  there is an operation  $r_{\alpha}$  such that for every  $c \in C$ , if  $\mu(\bar{\alpha}(u_1, \dots, u_n), c)$  is defined, then

- (i)  $\mu(\bar{\alpha}(u_1, \dots, u_n), c) = r_{\alpha}(\mu(u_1, c_1), \dots, \mu(u_n, c_n), c)$ ,  
where  $c_i = \Phi_C(\alpha, i, c)$ ,  $1 \leq i \leq n$ .

## 5.2 A formulation with sets of semantic functions

In the second style, we keep the standard notion of a semantics as an assignment of meanings to grammatical terms, but we use several such semantic functions. Intuitively, there is one function for each context type, but this formulation doesn't require a formal notion of context. We simply suppose that a *set*  $S$  of (ordinary) semantic functions is given, together with a *selection function*  $\Psi$ ; a partial function from triples of a grammar rule, a natural number, and an element of  $S$ , to  $S$ . The Generalized Domain Principle now takes the form:

(GDP $_S$ ) If  $\mu \in S$  and  $\mu(\bar{\alpha}(u_1, \dots, u_n))$  is defined, then  $\Psi(\alpha, i, \mu)(u_i)$  is defined,  $1 \leq i \leq n$ .

We also assume that there is a designated (null) function  $\mu_d \in S$ , the default semantics corresponding to the null context type.  $\mu_d$ , however, plays no explicit role in the compositionality condition, which, for  $S$  satisfying GDP $_S$ , becomes:

LC-Funct( $S, \Psi$ ) For every  $\alpha \in \Sigma$  and every  $\mu \in S$  there is an operation  $r_{\alpha, \mu}$  such that if  $\mu(\bar{\alpha}(u_1, \dots, u_n))$  is defined, then

- (ii)  $\mu(\bar{\alpha}(u_1, \dots, u_n)) = r_{\alpha, \mu}(\mu_1(u_1), \dots, \mu_n(u_n))$ ,  
where  $\mu_i = \Psi(\alpha, i, \mu)$ ,  $1 \leq i \leq n$ .

### 5.3 Some facts

The two generalized versions of contextual compositionality have standard compositionality as a special case, in the sense of the next, easily verified, fact.

**Fact 3**

If  $\mu$  is a partial function from  $GT_{\mathbf{E}}$  such that  $DP$  holds, then  $\text{Funct}(\mu)$  is equivalent to each the following:

- (a)  $LC\text{-Funct}(\{\mu\}, \Psi_0)$ , where  $\Psi_0(\alpha, i, \mu) = \mu$ .
- (b)  $LC\text{-Funct}(\mu^*, \{CXT_{\mathbf{E}}\})$ , relative to the trivial context typing of  $CXT_{\mathbf{E}}$  that has  $CXT_{\mathbf{E}}$  itself as the only context type, and the trivial context-sensitive version  $\mu^*$  of  $\mu$  given by  $\mu^*(t, CXT_{\mathbf{E}}) = \mu(t)$  (if  $\mu(t)$  is defined).

Next, we can show that  $LC\text{-Funct}(\mu, C)$  and  $LC\text{-Funct}(S, \Psi)$  are equivalent in the following sense:

**Proposition 4**

- (a) If  $LC\text{-Funct}(\mu, C)$  holds, define, for  $c \in C$ ,  $\mu^c(t) = \mu(t, c)$  (if defined), and let  $\Psi(\alpha, i, \mu^c) = \mu^{\Phi_C(\alpha, i, c)}$  (if  $\Phi_C(\alpha, i, c)$  is defined),  $1 \leq i \leq n$ . Then  $GDP_S$  and  $LC\text{-Funct}(S, \Psi)$  hold for  $S = \{\mu^c : c \in C\}$ .
- (b) Conversely, suppose  $LC\text{-Funct}(S, \Psi)$  holds. Define a partial mapping  $F$  from  $CXT_{\mathbf{E}}$  to  $S$  inductively by:

$$\begin{cases} F(\langle \rangle) & = \mu_d \text{ (the null function in } S\text{)} \\ F(\xi \hat{\ } \langle \alpha, i \rangle) & = \Psi(\alpha, i, F(\xi)), \text{ if for some term } \bar{\alpha}(u_1, \dots, u_n), \\ & F(\xi)(\bar{\alpha}(u_1, \dots, u_n)) \text{ is defined (otherwise undefined)} \end{cases}$$

The functions in  $S$  themselves are taken to be the context types, or more exactly, their inverse images under  $F$ : let

$$[\xi] = \{\xi' : F(\xi') \text{ is defined iff } F(\xi) \text{ is defined, and then } F(\xi') = F(\xi)\}$$

and let  $C = \{[\xi] : \xi \in CXT_{\mathbf{E}}\}$ . Define a semantics  $\nu$ , taking a term  $t$  and a context type  $[\xi]$  as arguments, by

$$\nu(t, [\xi]) = F(\xi)(t), \text{ if both } F(\xi) \text{ and } F(\xi)(t) \text{ are defined}$$

(otherwise undefined). Then  $GDP_{\nu}$  is satisfied,  $C$  is a context typing of  $CXT_{\mathbf{E}}$ , and  $LC\text{-Funct}(\nu, C)$  holds.

The proof of this result is given in the Appendix. In view of Proposition 4, we shall refer to  $LC\text{-Funct}(\mu, C)$  and  $LC\text{-Funct}(S, \Psi)$  indiscriminately as *general compositionality*, or *g-compositionality*.

## 5.4 Synonymy

With the general versions of functional compositionality, we obtain a variety of synonymy relations. With respect to  $\text{LC-Funct}(S, \Psi)$ , clearly for every  $\mu \in S$  there is a corresponding synonymy relation  $\equiv_\mu$ . This gives synonymy *relative* to members of  $S$ . But we can also distinguish three absolute notions. First, define

$$u_i \equiv_{S_{\text{des}}} u_j \quad \text{iff} \quad u_i \equiv_{\mu_d} u_j \quad (\mu_d \text{ is the designated function in } S)$$

Call this *designated synonymy*. Another natural candidate is *total synonymy*:

$$u_i \equiv_{S_{\text{tot}}} u_j \quad \text{iff for every } \mu \in S, \quad u_i \equiv_\mu u_j$$

As is intuitively clear, and as will follow from the semantics of the following section, if there is a quotation context type in  $\mathbf{E}$ , then there will be no non-trivial *total* synonymy pairs: a term  $s$  is totally synonymous only with itself. Because of this, we may be interested in a concept of *total synonymy except for quotation*, i.e. synonymy in all pure *use-contexts* (as opposed to quotation contexts, where expressions are both used and mentioned). Call this strong notion *use synonymy* (which for languages without quotation coincides with total synonymy); like the others, it is a partial equivalence relation on the set of grammatical terms.

Finally, let us consider the substitution version of  $\text{LC-Funct}(\mu, C)$ . To this end, let us modify the notation  $s[u_1, \dots, u_n]$  as follows: let

$$s = s[(u_1, \xi_1), \dots, (u_n, \xi_n)]$$

indicate that, for  $1 \leq i \leq n$ , at context  $\xi_i$  in  $s$  there is an occurrence of  $u_i$ , where  $\xi_i$  and  $\xi_j$  are pairwise *disjoint* for  $i \neq j$ , i.e. neither is an extension of the other. Then

$$t = s[(t_1, \xi_1), \dots, (t_n, \xi_n)]$$

is the unique term obtained from  $s$  by replacing the occurrence of  $u_i$  at  $\xi_i$  by  $t_i$ .<sup>21</sup>

We also need recourse to the function  $\Theta_C$  from (14) in Section 4.2. Then we can state:

LC-Subst( $\mu, C$ )     If  $s = s[(u_1, \xi_1), \dots, (u_n, \xi_n)]$  and  $t = s[(t_1, \xi_1), \dots, (t_n, \xi_n)]$  are both  $\mu$ -meaningful in the context type  $c \in C$ , and if, for  $1 \leq i \leq n$ ,  $\mu(u_i, c_i) = \mu(t_i, c_i)$ , where  $c_i = \Theta_C(c, \xi_i)$ , then  $\mu(s, c) = \mu(t, c)$ .

This says that, for any  $c$ , the semantic values of two terms are the same in context type  $c$  if one results from the other by means of substitution of subterms that are pairwise  $\mu$ -equivalent in the type of context where they occur.

<sup>21</sup> $t$  is a well-defined term in  $T_{\mathbf{E}}$ ; it need not be a *grammatical* term. The disjointness assumption entails that the result is independent of whether the replacements are done simultaneously or sequentially.

**Fact 5**

If  $(GDP_\mu)$  holds,  $LC\text{-Funct}(\mu, C)$  is equivalent to  $LC\text{-Subst}(\mu, C)$ .

The left-to-right direction of Fact 5 is proved by means of induction over term complexity. The right-to-left direction follows immediately from the special case when the substitution terms are the immediate subterms.

## 6 Compositional generalization

$LC\text{-Funct}(\mu, C)$  holds for a pair  $(\mu, C)$  of a binary function  $\mu$  and a context typing  $C$ . Similarly,  $LC\text{-Funct}(S, \Psi)$  holds for a pair  $(S, \Psi)$  of a set  $S$  of unary functions and a selection function  $\Psi$  (for  $LC\text{-Funct}(\mu, C)$  the function  $\Phi_C$  is determined by the typing itself according to (13)).

Typically, however, we are initially given a grammar  $\mathbf{E}$  and an initial single unary semantic function  $\mu$  for  $\mathbf{E}$ . In some cases, for instance with a language that contains quotation and with a semantics that respects ordinary intuitions about synonymy,  $\mu$  is non-compositional, as we have seen. We are then interested in whether there is a proper corresponding  $\mu$ -related pair  $(\mu', C)$  or  $(S, \Psi)$  that satisfies  $LC\text{-Funct}(\mu, C)$  or  $LC\text{-Funct}(S, \Psi)$ , respectively. We use the following terminology:

- (15)  $(\mu', C)$  *generalizes*  $\mu$  iff for all  $t$ ,  $\mu(t) = \mu'(t, [\langle \rangle])$ . Similarly,  $(S, \Psi)$  *generalizes*  $\mu$  iff  $\mu$  is the designated function in  $S$ .

Now we are asking when  $\mu$  is in a relevant sense *compositionally generalizable*. At first blush, the relevant sense seems easy to spell out. In the  $LC\text{-Funct}(S, \Psi)$  format, the first suggestion would be:

- (16)  $\mu$  is compositionally generalizable iff  $LC\text{-Funct}(S, \Psi)$  holds for some pair  $(S, \Psi)$  generalizing  $\mu$ .

But this requirement is *empty*: in the sense of (16), *any* semantic function  $\mu$  is generalizable. To see this, define the function  $\nu$  by  $\nu(t) = \langle \mu(t), t \rangle$ . Let  $S = \{\mu, \nu\}$ , and  $\Psi(\alpha, i, \mu) = \Psi(\alpha, i, \nu) = \nu$ , for any  $\alpha$  and  $i$ . Then  $LC\text{-Funct}(S, \Psi)$  holds.<sup>22</sup>

In the above definition of  $\nu$ , general compositionality is achieved by way of *hyperdistinctness*: if  $\nu(t) = \nu(u)$ , then  $t = u$ . This also ensures ordinary compositionality, in a degenerate way (cf. (5) in Section 2.2). It is reasonable to

<sup>22</sup> We must find, for each  $\alpha$ , meaning operations  $r_{\alpha, \mu}$  and  $r_{\alpha, \nu}$  that satisfy the compositional recursion equations. Let

$$\begin{aligned} r_{\alpha, \mu}(m_1, \dots, m_n) &= \mu(\bar{\alpha}(\pi_2(m_1), \dots, \pi_2(m_n))) \\ r_{\alpha, \nu}(m_1, \dots, m_n) &= \langle \mu(\bar{\alpha}(\pi_2(m_1), \dots, \pi_2(m_n))), \bar{\alpha}(\pi_2(m_1), \dots, \pi_2(m_n)) \rangle \end{aligned}$$

where  $\pi_2$  is a right projection function:  $\pi_2(\langle x, y \rangle) = y$ . Then, with  $\nu$ ,  $S$ , and  $\Psi$  as above:

- (i) For any initial  $\mu$ ,  $LC\text{-Funct}(S, \Psi)$  holds.

block this move by requiring that a semantic function for a particular context type *respect* the semantic equivalences in that context for the original function  $\mu$ . Staying in the  $\text{LC-Funct}(S, \Psi)$  format, recall from Proposition 4 that each function in  $S$  has the form  $\mu^c$ , for some context type  $c$  in the corresponding typing. The requirement can then be stated as follows (with the notation from Section 5.4):

(MAX) If  $\xi$  is a context such that for all terms  $s$  containing  $\xi$ ,  $s[(u, \xi)] \equiv_{\mu} s[(t, \xi)]$ , then  $u \equiv_{\mu[\xi]} t$ .

Informally, if  $u$  and  $t$  make the same contribution to the meaning of complex expressions in which they occur, *in a certain linguistic context*, then their meaning *in* the type of that context is the same. This is similar to Hodges' notion of a *Fregean extension* (or full abstraction; see Hodges [13]), which can be seen as expressing a version of Frege's Context Principle, but this time for meanings that are sensitive to linguistic context.

Suppose the trivial generalization of  $\mu$  into  $S = \{\mu, \nu\}$  above satisfies MAX. Since for all distinct terms  $t, u$ ,  $\nu(t) \neq \nu(u)$ , this means that in the  $\nu$  context type, to which every embedded (non-null) context belongs according to the associated  $\Psi$ , distinct terms are semantically non-equivalent. By MAX, it follows that for any distinct terms  $t, u$ , and for any embedded context  $\xi$  where  $t$  and  $u$  can occur, there is a term  $s$  such that

$$s[(u, \xi)] \not\equiv_{\mu} s[(t, \xi)]$$

As a consequence, if for any two terms  $t$  and  $u$ ,  $\mu(t) = \mu(u)$ , then there is substitution failure *in every embedded linguistic context* (although not for every term that contains it).  $\mu$  is then non-compositional, not just with respect to *some* operator  $\alpha$ , but with respect to *every* syntactic operator. Although such semantic functions are possible, we seem not to find them in natural language. Generating a language  $L'$  by adding quotation to a language  $L$  does make  $L'$  non-compositional if there are non-trivial synonymies in  $L$ , but only because of the quotation contexts. It does not make all the operators in the  $L$  fragment non-compositional as well. So, for an ordinary semantics  $\mu$ , generalizing it into  $S = \{\mu, \nu\}$  as above does not satisfy MAX. MAX seems to be a reasonable requirement on generalization.

However, even under MAX, any unary semantic function  $\mu$  turns out to be generalizable to *some*  $(S, \Psi)$  satisfying LC-Funct, provided there is no limitation on the number of context types:

**Fact 6**

*For any unary semantics  $\mu$ , there is  $(S, \Psi)$  generalizing  $\mu$  such both MAX and LC-Funct( $S, \Psi$ ) hold.*

The key to the proof, which is given in the Appendix, is to use many context types; in fact, we will let each context be its own type. Even though that is hardly a situation one encounters in practice, there is thus reason to introduce a second requirement on compositional generalizability.

The type of a context  $\xi$  may in principle depend on all the operators in  $\xi$  (recall that the formal notion of a context is a sequence starting with the null context, followed by operator-index pairs), in the order in which they occur, or on any particular subset of the set of operators in  $\xi$ , again in their order in  $\xi$ . In the simplest case, however, only one operator matters. We can spell this out as follows.

Call an operator-index pair  $(\alpha, i)$  a *switcher* (relative to a context typing  $C$ ) iff there are at least two context types, and for any two contexts  $\xi, \xi'$  in which a term  $\bar{\alpha}(t_1, \dots, t_n)$  can occur,

$$[\xi^\wedge \langle (\alpha, i) \rangle] = [\xi'^\wedge \langle (\alpha, i) \rangle]$$

This means that there is a context type  $c$  such that whatever the context type of (a grammatical occurrence of)  $\bar{\alpha}(t_1, \dots, t_n)$ , the context type of the corresponding occurrence of  $t_i$  is  $c$ . Let us also say that  $(\alpha, i)$  is a *keeper* iff there are at least two context types, and for any admissible context  $\xi$ ,

$$[\xi^\wedge \langle (\alpha, i) \rangle] = [\xi]$$

This means that  $(\alpha, i)$  never changes context type. We can now define a *first-grade* g-compositional semantics  $(S, \Psi)$  to be a g-compositional semantics where every operator-index pair is either a switcher or a keeper. In such a semantics, the context type of a context  $\xi$  is determined by the *last* switcher pair in the sequence that constitutes  $\xi$ , or else is the default type if there is no switcher in the sequence. If all pairs are switcher pairs, then the type of a context  $\xi'^\wedge \langle (\alpha, i) \rangle$  is always determined by  $(\alpha, i)$ , irrespective of  $\xi'$ .

We can analogously define a *second-grade* g-compositional semantics by defining *switcher pairs* of operator-index pairs. In this case, the type of a context  $\xi$  is determined by the last switcher *pair* of operator-index pairs in the sequence that constitutes  $\xi$ . Similarly, switcher triples determine context type in a *third-grade* semantics. And so on. A Fregean semantics for belief contexts, where each iteration of the belief operator switches to a new type in an infinite hierarchy of indirect context types, is an  $\omega$ -grade semantics.

Clearly, along this parameter, the simplest g-compositional semantics are the first-grade ones. The number of context types for such semantics is at most equal to the number of operator-index pairs. A particular operator-index pair is responsible for the context type. The number of rules can be kept low, and no more than one operator-index pair needs to be kept in memory at any single recursion step of processing the semantics. This indicates that the following terminology is appropriate:<sup>23</sup>

---

<sup>23</sup>In Pagin and Westerståhl [18], we distinguish *1st level compositionality*, where the meaning of a complex term depends on the meanings of its immediate subterms, from *2nd level compositionality*, where also the meanings of the immediate subterms of the immediate subterms may be required, and so on. This is somewhat analogous to the grade distinction here, and standard compositionality is 1st level, which is yet another reason for choosing the formulation in GEN. Also, many semantics where linguistic context is allowed to play a role in fact satisfy GEN; see the next section, and note 24. The  $\omega$ -grade Fregean semantics, on the other hand, requires infinitely many context types, which threatens to trivialize the compositionality

(GEN)  $\mu$  is *compositionally generalizable* iff there is a first-grade pair  $(S, \Psi)$  generalizing  $\mu$  such that  $\text{LC-Funct}(S, \Psi)$  and  $\text{MAX}$  hold.

GEN is *not* an empty requirement. A proof of the following is outlined in the Appendix:

**Fact 7**

*For each  $n$  there is a grammar  $\mathbf{E}$  and a semantics  $\mu$  for  $\mathbf{E}$  which is not generalizable by any  $n$ th-grade pair  $(S, \Psi)$  satisfying  $\text{LC-Funct}$  and  $\text{MAX}$ .*

The g-compositional semantics for pure quotation given in the next section satisfies GEN. The one-place quotation operator always triggers a quotation context type. All other operator-index pairs are keepers, preserving either the quotation context type or the default type. In particular, there is no switcher that takes the semantics *out* of a pure quotation context, which seems in accordance with ordinary intuitions (but see note 29).<sup>24</sup>

## 7 A g-compositional account of pure quotation

Our final task is to show that a semantics in which quotation is a syntactic operator, and quoting expressions have the quoted expressions as their semantic value, is compositionally generalizable in the sense of GEN. We formulate this as a *compositional extension* result: *given* any semantics  $\mu$  which is compositional in the ordinary sense, there is a natural way to extend it to semantics which handles pure quotation and is compositionally generalizable.<sup>25</sup> We use the formulation  $\text{LC-Funct}(S, \Psi)$ , but Proposition 4 provides an immediate translation to the version with explicit context types.

Let, then,  $\mathbf{E} = (E, A, \Sigma)$  be a given grammar and  $\mu$  a semantics for  $\mathbf{E}$  such that  $\text{Funct}(\mu)$  holds. As explained in Section 2.1, we then have a string value function  $V$  from  $GT_{\mathbf{E}}$  to  $E$ . The idea of what we called a *straightforward* semantics for quotation in Section 1 is to add a new syntactic operation  $\kappa$  that

---

requirement.

<sup>24</sup>General compositional semantics appear in earlier work by Kathrin Glüer and Peter Pagin. [9] presents a semantics accounting for the behavior of proper names in modal contexts. The idea is to keep proper names as non-rigid designators, with non-constant intensions, and achieve the rigidity-like effect by means of the interaction of the modal operator with the term. The modal operator then acts as a switcher, creating an *actualist* context where the name reference is the value of its intension in the actual world.

The resulting semantics is equivalent to a standard semantics with rigid designators w.r.t. truth, and almost equivalent w.r.t. logical consequence, as shown in [10]. In [9], an appendix considers the addition of a naive belief operator that switches from an actualist context type back to the default *possibilist* context type.

The idea has also been extended to handle *general terms* in modal contexts in [11]. In unpublished work, Pagin develops a more adequate g-compositional account of belief sentences. In all these cases, GEN is in fact satisfied, although this condition is not mentioned.

<sup>25</sup>A compositional extension result has the form: if  $\mu$  is a compositional semantics for  $\mathbf{E}$ , there is a (possibly unique) extension  $\mu'$  of  $\mu$  satisfying a certain property. Hodges [13] and Westerståhl [36] prove theorems concerning the extension from a partial to a total semantics. Westerståhl [35] gives various extension results for adding idiomatic expressions to a language.

puts quote marks around any expression  $e$ , and interprets the result as referring to  $e$ . That is:

- (17)      a.  $\kappa(e) = 'e'$  (i.e. the string `leftquote $e$ rightquote`)  
             b.  $\mu'(\bar{\kappa}(t)) = V'(t)$

where  $\mu'$  is  $\mu$  extended to the new terms, and  $V'$  the corresponding string value function. Roughly, the claim is now that the set  $S = \{\mu', V'\}$  (with a suitable selection function) is g-compositional and satisfies GEN with respect to  $\mu'$ .

However, as indicated with sentences (1) in Section 1, we may want to quote not only well-formed expressions but also, say, single letters of some alphabet, or arbitrary concatenations of such letters, even non-sensical ones. For concreteness, let us assume that the strings we need are composed of *letters* belonging to a (normally finite) set  $L$ ; these could be Roman letters, and some letters from other alphabets; we also assume that a space symbol and the quote marks are in  $L$ . Thus,  $A \subseteq E \subseteq L^*$ , where  $L^*$  is the set of all finite strings of letters. Some strings in  $L^*$  are well-formed expressions, most strings aren't, but in principle we should be able to quote all of them.

One way to achieve this would be to add all strings of the form ' $u$ ', when  $u \in L^*$ , as new atoms. This is the *proper name theory* of quotation, and we indicated in Section 1 why it is unsatisfactory: (a) we get infinitely many primitive expressions, and (b) quoting expressions have no structure. To remove (a), we shall generate all strings from letters and the concatenation operation. And (b) is removed by using  $\kappa$  whenever we want to quote something (grammatical or not).<sup>26</sup>

In more detail, we proceed to extend the grammar **E** as follows. Let the new set of atoms be

$$(18) \quad A' = A \cup L$$

(assuming  $A \cap L = \emptyset$ ). The new rules are

$$(19) \quad \Sigma' = \{\alpha' : \alpha \in \Sigma\} \cup \{cc, \kappa\}$$

where  $cc$  is a binary concatenation operation generating, we assume,  $L^*$  from  $L$ , and  $\kappa$  is the function in (17a), taken to be *total*. Each  $\alpha'$  extends  $\alpha$  in some way that naturally incorporates the new expressions into the grammar. Lacking a detailed description of **E**, we cannot specify exactly how this is done, but the idea should be clear. Using  $\kappa$ , we want to generate strings like *John likes 'Mary'* (meaning that he likes the name), or with iterated quotation as in

$$(20) \quad \text{'John likes 'Mary'' is a sentence}$$

but not, for example, *John 'likes' Mary* (we deal only with pure quotation), or *'John likes' Mary*. This could be done by treating all quoting expressions

---

<sup>26</sup>Thus, the task of quoting arbitrary strings leads us to something like the *description theory* of quotation, and in this respect our account is similar to the one in Werning [33]. But the crucial difference is that we combine this with a standard syntactic quotation operator.

(expressions of the form  $\kappa(e)$ ) as *noun phrases*, and adapting the old rules accordingly.

Finally  $E'$  is the closure of  $A'$  under the functions in  $\Sigma'$ , and

$$\mathbf{E}' = (E', A', \Sigma')$$

is the extended grammar. As to the term algebra, we now have a primitive grammatical term

$$\bar{l}$$

for each  $l$  in  $L$ . Let the set of *string terms*,  $ST$ , be defined inductively by

$$(21) \quad \begin{aligned} \bar{l} &\in ST \text{ for } l \in L \\ \overline{cc}(t, u) &\in ST \text{ when } t, u \in ST \end{aligned}$$

The new set of grammatical terms,  $GT_{\mathbf{E}'}$ , is the closure of  $\{\bar{a} : a \in A'\}$  under the functions  $\overline{cc}$ ,  $\bar{\kappa}$ , and  $\bar{\alpha}'$  for  $\alpha \in \Sigma$ , where  $\overline{cc}$  is only defined on  $ST$ , whereas  $\bar{\kappa}$  is total (every term can be quoted). The new string value function  $V'$  is as usual:<sup>27</sup>

$$(22) \quad \begin{cases} V'(\bar{a}) &= a, \text{ for } a \in A' \\ V'(\bar{\delta}(t_1, \dots, t_n)) &= \delta(V'(t_1), \dots, V'(t_n)), \text{ for } \delta \in \Sigma' \end{cases}$$

It is now fairly clear how the given semantics  $\mu$  can be extended to a semantics  $\mu'$  suitable for  $\mathbf{E}'$ . We adjoin  $L^*$  to the given domain of interpretation  $M$  (assuming  $M \cap L^* = \emptyset$ ),<sup>28</sup> and adapt the given composition functions  $r_\alpha$  for  $\alpha \in \Sigma$  to functions  $r_{\alpha'}$  that work for the new terms. The meaning of terms of the form  $\bar{\kappa}(t)$  was given in (17b), but  $\mu'$  is *undefined* for string terms.

As we have emphasized, the extended semantics  $\mu'$  is *not compositional*: one cannot substitute synonymous terms in the range of  $\bar{\kappa}$  and expect meaning to be preserved. However,  $(S, \Psi)$  is g-compositional, where  $S = \{\mu', V'\}$ , and the selection function  $\Psi$  is defined, for  $\delta \in \Sigma'$  and  $\nu \in S$ , by

$$(23) \quad \Psi(\delta, i, \nu) = \begin{cases} V' & \text{if } \delta = \kappa \\ \nu & \text{otherwise} \end{cases}$$

That is,  $\mu'$  only switches to  $V'$  when something is quoted, and then every more deeply embedded subterm is also evaluated with  $V'$ .<sup>29</sup>

Next,  $\text{LC-Funct}(S, \Psi_S)$  requires us to define operations  $r_{\delta, \nu}$  for each  $\delta \in \Sigma'$  and  $\nu \in S$ . Specifically, we must satisfy

$$(24) \quad V'(\bar{\delta}(t_1, \dots, t_n)) = r_{\delta, V'}(V'(t_1), \dots, V'(t_n))$$

<sup>27</sup>In a complete account,  $\mathbf{E}'$  and  $GT_{\mathbf{E}'}$  would be defined with one big inductive definition.

<sup>28</sup>In a type-theoretic universe built from a set of individuals, one could add a new primitive type for the elements of  $L^*$ .

<sup>29</sup>So the semantics is first-grade in the sense of Section 6. In a slightly different, but still first-grade semantics we can add an operator that switches back from  $V'$  to  $\mu'$ , e.g. if one wants a mechanism for quantifying into quotation contexts.

so it follows from (22) that we should set

$$(25) \quad r_{\delta, V'} = \delta$$

for all  $\delta \in \Sigma'$ . In particular, when  $\delta = \kappa$  we then have

$$(26) \quad r_{\kappa, V'}(V'(t)) = \kappa(V'(t)) = 'V'(t)'$$

this will apply only in iterated quotation contexts (see the example below). As to  $r_{\delta, \mu'}$ , we set

$$(27) \quad \begin{cases} r_{\alpha', \mu'}(m_1, \dots, m_n) & = r_{\alpha'}(m_1, \dots, m_n), \text{ for } \alpha \in \Sigma \\ r_{\kappa, \mu'}(m) & = m \end{cases}$$

(We don't need to define  $r_{cc, \mu'}$ , since  $\mu'(\overline{cc}(t, u))$  is always undefined.)

In particular, we get

$$(28) \quad \mu'(\overline{\kappa}(t)) = r_{\kappa, \mu'}(V'(t)) = V'(t)$$

in accordance with (17).

To see how  $S$  and  $\Psi$  work, consider sentence (20), and suppose, for simplicity of illustration, that this string is the value of the term

$$t = \overline{isa'}(\overline{\kappa}(\overline{\alpha'}(\overline{John}, \overline{\beta'}(\overline{like}, \overline{\kappa}(\overline{Mary}))))), \overline{sentence})$$

(so sentences of the form 'NP is a N' are analyzed as  $isa(\text{NP}, \text{N})$ ). Its designated meaning is calculated, using (22) – (28), as shown in Figure 2. Here we have

$$\begin{aligned} \mu'(t) &= r_{isa', \mu'}(\mu'(\overline{\kappa}(\overline{\alpha'}(\overline{John}, \overline{\beta'}(\overline{like}, \overline{\kappa}(\overline{Mary}))))), \mu'(\overline{sentence})) \\ &= r_{isa'}(\mu'(\overline{\kappa}(\overline{\alpha'}(\overline{John}, \overline{\beta'}(\overline{like}, \overline{\kappa}(\overline{Mary}))))), \text{SENT}) \\ &= r_{isa'}(V'(\overline{\alpha'}(\overline{John}, \overline{\beta'}(\overline{like}, \overline{\kappa}(\overline{Mary}))))), \text{SENT}) \\ &= r_{isa'}(\alpha'(V'(\overline{John}), V'(\overline{\beta'}(\overline{like}, \overline{\kappa}(\overline{Mary}))))), \text{SENT}) \\ &= r_{isa'}(\alpha'(\overline{John}, \beta'(V'(\overline{like}), V'(\overline{\kappa}(\overline{Mary}))))), \text{SENT}) \\ &= r_{isa'}(\alpha'(\overline{John}, \beta'(\overline{like}, \kappa(V'(\overline{Mary}))))), \text{SENT}) \\ &= r_{isa'}(\alpha'(\overline{John}, \beta'(\overline{like}, 'Mary'))), \text{SENT}) \\ &= r_{isa'}(\alpha'(\overline{John}, \overline{like} 'Mary)'), \text{SENT}) \\ &= r_{isa'}(\overline{John} \overline{likes} 'Mary'), \text{SENT}) \\ &= \text{T iff } \overline{John} \overline{likes} 'Mary' \in \text{SENT} \end{aligned}$$

Figure 2: Evaluation of a quotation example

assumed that in the given grammar, a string like *John likes Mary* is the value of the term  $\overline{\alpha}(\overline{John}, \overline{\beta}(\overline{like}, \overline{Mary}))$ , so *John likes 'Mary'* has the same form in

the extended grammar, except that the quote marks are introduced by  $\kappa$ , and  $\alpha$  and  $\beta$  are extended to  $\alpha'$  and  $\beta'$  which apply to quoted strings as well.

We can summarize our extension result for quotation as follows:

**Fact 8**

Suppose  $\text{Funct}(\mu)$  holds, and  $\mu$  is extended to a semantics  $\mu'$  for  $\mathbf{E}'$ , which is generalized to  $(S, \Psi)$  as above.

- (a)  $\mu'$  handles pure quotation correctly, i.e. it gives the intended meanings to sentences containing such quotation.
- (b)  $\mu'$  extends  $\mu$  in the sense that when  $t \in GT_{\mathbf{E}}$ ,  $\mu'(t) = \mu(t)$ .
- (c) If  $\mu$  satisfies  $DP$ , then  $GDP_S$  and  $LC\text{-Funct}(S, \Psi)$  hold, even though  $\text{Funct}(\mu')$  in general fails.
- (d)  $(S, \Psi)$  satisfies  $GEN$ : it is first-grade and satisfies  $MAX$  (the semantics is hyperdistinct in quotation contexts while  $\mu'$  itself applies in the null context type.)

In the absence of a detailed specification of the given grammar  $\mathbf{E}$ , we have only been able to indicate certain parts of the extension, but it should be fairly clear how to apply these indications to specific grammars.

The extended semantics allows quotation of an arbitrary string  $z$  in  $L^*$ , using  $\bar{\kappa}(s)$ , where  $s$  is a string term such that  $V'(s) = z$ . So it can handle all the examples (1). It may of course happen that  $z$  is also the string value of some  $\mu'$ -meaningful grammatical term  $t$ . Perhaps in some sense there is a distinction between quoting, say, *John likes 'Mary'* as a meaningful sentence and quoting it as a mere string of letters. But in the present extensional semantics it is a distinction without a difference: we have a structural ambiguity with no semantic import, since it is clear that  $\mu'(\bar{\kappa}(t)) = \mu'(\bar{\kappa}(s))$ .

## 8 Further directions

The work begun here could be extended in at least two directions. First, one could try to extend account to cover other uses of quotation, such as the interplay between direct and indirect discourse, or mixed quotation (as in *Quine says that quotation 'has the overwhelming practical convenience of visible reference'*). Second, as we have already indicated, linguistic context-dependence as outlined here could be exploited for other constructions, notably various kinds of intensional contexts.

## Appendix: Some proofs

*Proof of Proposition 4:* (a): Assume that  $\mu$  is a partial function from  $GT_{\mathbf{E}} \times C$  satisfying  $GDP_{\mu}$ , where  $C$  is a context typing of  $CXT_{\mathbf{E}}$  such that  $LC\text{-Funct}(\mu, C)$  holds, and that  $\mu^c$  for  $c \in C$  and  $\Psi$  are as described in (a), with

$S = \{\mu^c : c \in C\}$ . Take  $\alpha \in \Sigma$  and  $\mu^c \in S$  such that  $\mu^c(\bar{\alpha}(u_1, \dots, u_n))$  is defined. Using  $\text{GDP}_\mu$ , we see that  $\Psi(\alpha, i, \mu^c)(u_i)$  is defined,  $1 \leq i \leq n$ . This shows that  $\text{GDP}_S$  holds. Also, by  $\text{LC-Funct}(\mu, C)$  we have

$$\begin{aligned}
\mu^c(\bar{\alpha}(u_1, \dots, u_n)) &= \mu(\bar{\alpha}(u_1, \dots, u_n), c) \\
&= r_\alpha(\mu(u_1, \Phi_C(\alpha, 1, c)), \dots, \mu(u_n, \Phi_C(\alpha, n, c)), c) \\
&= r_\alpha(\mu^{\Phi_C(\alpha, 1, c)}(u_1), \dots, \mu^{\Phi_C(\alpha, n, c)}(u_n), c) \\
&= r_\alpha(\Psi(\alpha, 1, \mu^c)(u_1), \dots, \Psi(\alpha, n, \mu^c)(u_n), c) \\
&= s_{\alpha, \mu^c}(\Psi(\alpha, 1, \mu^c)(u_1), \dots, \Psi(\alpha, n, \mu^c)(u_n)),
\end{aligned}$$

where  $s_{\alpha, \mu^c}$  is defined by

$$\begin{aligned}
s_{\alpha, \mu^c}(m_1, \dots, m_n) &= r_\alpha(m_1, \dots, m_n, c) \\
&\text{if there are } t_1, \dots, t_n \text{ such that } \mu^c(\bar{\alpha}(t_1, \dots, t_n)) \text{ is defined and } m_i = \\
&\Psi(\alpha, i, \mu^c)(t_i), 1 \leq i \leq n \text{ (undefined otherwise)}.
\end{aligned}$$

Note that  $s_{\alpha, \mu^c}$  is well-defined, since it follows from the above that if  $\mu^c = \mu^{c'}$ , then  $r_\alpha(m_1, \dots, m_n, c) = r_\alpha(m_1, \dots, m_n, c')$ . This proves  $\text{LC-Funct}(S, \Psi)$ .

(b): Now suppose  $S$  is a set of functions and  $\Psi$  a selection function satisfying  $\text{GDP}_S$  such that  $\text{LC-Funct}(S, \Psi)$  holds. First, note that  $F$  as defined in (b) is a well-defined partial function, by induction over the length of  $\xi$ : if it is clear for which contexts of length at most  $k$   $F$  is defined and what its values are, and  $\eta$  is a context of length  $k + 1$ , then  $\eta$  has the form  $\xi \hat{\wedge} \langle \alpha, i \rangle$ . If  $F(\xi)$  is undefined, or if  $F(\xi)$  is an element of  $S$  which is not defined for any term of the form  $\bar{\alpha}(u_1, \dots, u_n)$ , then  $F(\eta)$  is undefined. Otherwise,  $F(\eta) = \Psi(\alpha, i, F(\xi))$ ; this is defined by  $\text{GDP}_S$ .

Next, note that the definition of the sets  $[\xi]$  is formulated in such a way that if  $F(\xi)$  is defined,  $[\xi] = \{\xi' : F(\xi') = F(\xi)\}$ , whereas if  $F(\xi)$  is undefined,  $[\xi] = \{\xi' : F(\xi') \text{ is undefined}\}$ . Thus,  $C$  is a partition of  $\text{CXT}_{\mathbf{E}}$ . We must check that  $C$  is a context typing, i.e. that condition (i) in  $(\text{L}_{\text{ctype}})$  holds.

So suppose  $[\xi] = [\xi']$ . If  $F(\xi)$  is defined, then so is  $F(\xi')$  and  $F(\xi') = F(\xi)$ . If there is a term  $\bar{\alpha}(u_1, \dots, u_n)$  such that  $F(\xi)(\bar{\alpha}(u_1, \dots, u_n))$  is defined, then  $F(\xi \hat{\wedge} \langle \alpha, i \rangle)$  is defined, and  $\Psi(\alpha, i, F(\xi)) = \Psi(\alpha, i, F(\xi'))$ , so  $F(\xi \hat{\wedge} \langle \alpha, i \rangle) = F(\xi' \hat{\wedge} \langle \alpha, i \rangle)$ , i.e.  $[\xi \hat{\wedge} \langle \alpha, i \rangle] = [\xi' \hat{\wedge} \langle \alpha, i \rangle]$ , as desired. If there is no such term, then  $F(\xi \hat{\wedge} \langle \alpha, i \rangle)$  and  $F(\xi' \hat{\wedge} \langle \alpha, i \rangle)$  are both undefined, so again  $[\xi \hat{\wedge} \langle \alpha, i \rangle] = [\xi' \hat{\wedge} \langle \alpha, i \rangle]$ . Finally, if  $F(\xi)$  is undefined, so is  $F(\xi')$ , and again  $F(\xi \hat{\wedge} \langle \alpha, i \rangle)$  and  $F(\xi' \hat{\wedge} \langle \alpha, i \rangle)$  are both undefined. This proves (i).

Next, it clearly follows from our definitions that the function  $\nu$  is well-defined, i.e. that if  $[\xi] = [\xi']$ , then, for any term  $t$ ,  $\nu(t, \xi)$  is defined iff  $\nu(t, \xi')$  is defined, and when both are defined they have the same value.

We must also verify that  $\text{GDP}_\nu$  holds. Suppose  $\nu(\bar{\alpha}(u_1, \dots, u_n), [\xi])$  is defined. Then  $F(\xi)$  is defined and  $\nu(\bar{\alpha}(u_1, \dots, u_n), [\xi]) = F(\xi)(\bar{\alpha}(u_1, \dots, u_n))$ . By  $\text{GDP}_S$  and the definition of  $F$ ,  $\Psi(\alpha, i, F(\xi))(u_i) = F(\xi \hat{\wedge} \langle \alpha, i \rangle)(u_i)$  is defined. Thus, each  $\nu(u_i, [\xi \hat{\wedge} \langle \alpha, i \rangle])$  is defined. Again, this reasoning is independent of the choice of context in  $[\xi]$ . Thus,  $\text{GDP}_\nu$  holds.

Finally, take a rule  $\alpha$  and a context type  $c \in C$  such that  $\nu(\bar{\alpha}(u_1, \dots, u_n), c)$  is defined. Let  $c = [\xi]$ . Using  $\text{LC-Funct}(S, \Psi)$  and the definitions of  $F$ ,  $\nu$ , and  $\Phi_C$ , we calculate:

$$\begin{aligned}
\nu(\bar{\alpha}(u_1, \dots, u_n), [\xi]) &= F(\xi)(\bar{\alpha}(u_1, \dots, u_n)) \\
&= r_{\alpha, F(\xi)}(\Psi(\alpha, 1, F(\xi))(u_1), \dots, \Psi(\alpha, n, F(\xi))(u_n)) \\
&= r_{\alpha, F(\xi)}(F(\xi^\wedge \langle (\alpha, 1) \rangle)(u_1), \dots, F(\xi^\wedge \langle (\alpha, n) \rangle)(u_n)) \\
&= r_{\alpha, F(\xi)}(\nu(u_1, [\xi^\wedge \langle (\alpha, 1) \rangle]), \dots, \nu(u_n, [\xi^\wedge \langle (\alpha, n) \rangle])) \\
&= s_\alpha(\nu(u_1, [\xi^\wedge \langle (\alpha, 1) \rangle]), \dots, \nu(u_n, [\xi^\wedge \langle (\alpha, n) \rangle]), [\xi]) \\
&= s_\alpha(\nu(u_1, \Phi_C(\alpha, 1, [\xi])), \dots, \nu(u_n, \Phi_C(\alpha, n, [\xi])), [\xi])
\end{aligned}$$

where the operation  $s_\alpha$  is defined by

$$s_\alpha(m_1, \dots, m_n, [\xi]) = r_{\alpha, F(\xi)}(m_1, \dots, m_n)$$

These calculations are independent of the choice of  $\xi$ , as long as  $F(\xi)$  is defined, which is an assumption. Therefore,  $s_\alpha$  is well-defined. This shows that  $\text{LC-Funct}(\nu, C)$  holds.  $\square$

*Proof of Fact 6:* Let  $\mu$  be given, and let the context typing  $C$  be such that each context is its own type, i.e.  $[\xi] = \{\xi\}$ . Define, for each  $\xi$ , the relation  $\equiv^\xi$  by

$$(29) \quad u \equiv^\xi t \text{ iff for all terms } s \text{ containing } \xi, s[(u, \xi)] \equiv_\mu s[(t, \xi)]$$

$\equiv^\xi$  is a partial equivalence relation on  $GT_{\mathbf{E}}$ , with  $\text{dom}(\equiv^\xi) = \{u : u \equiv^\xi u\}$ . Let  $[[t]]^\xi = \{t' : t' \equiv^\xi t\}$  be the equivalence class of  $t$  under  $\equiv^\xi$ . Note that, since for all terms  $s, t$ ,  $s[(t, \langle \rangle)] = t$ , we have

$$(30) \quad \equiv^{\langle \rangle} = \equiv_\mu$$

Then let, for each  $\xi$ ,  $\mu^{[\xi]}$  be a semantics whose corresponding synonymy relation is  $\equiv^\xi$ , i.e.  $\equiv_{\mu^{[\xi]}} = \equiv^\xi$ . For definiteness, put

$$(31) \quad \mu^{[\xi]}(t) = \begin{cases} \mu(t) & \text{if } \xi = \langle \rangle \text{ and } \mu(t) \text{ is defined} \\ [[t]]^\xi & \text{if } \xi \neq \langle \rangle \text{ and } t \in \text{dom}(\equiv^\xi) \\ \text{undefined} & \text{otherwise} \end{cases}$$

Then (cf. Hodges [13], Lemma 1), we have

$$\equiv_{\mu^{[\xi]}} = \equiv^\xi$$

Also,

$$\mu^{\langle \rangle} = \mu$$

so  $(S, \Psi)$  generalizes  $\mu$ , with  $S = \{\mu^{[\xi]} : \xi \in CXT_{\mathbf{E}}\}$ , and

$$\Psi(\alpha, i, \mu^{[\xi]}) = \mu^{[\xi^\wedge \langle (\alpha, i) \rangle]}$$

Now it is immediate from (29) that MAX holds. To show compositionality, we prove:

$$(32) \quad \text{If } s \text{ contains } \xi, \text{ and if } t \equiv^{\xi' \wedge \xi} u, \text{ then } s[(t, \xi)] \equiv^{\xi'} s[(u, \xi)].$$

For let  $s'$  be any term containing the context  $\xi'$ . Then, for all terms  $v$ ,  $s'' = s'[(s[(v, \xi)], \xi')]$  contains the context  $\xi' \wedge \xi$ , and

$$s'[(s[(v, \xi)], \xi')] = s''[(v, \xi' \wedge \xi)]$$

By assumption and (29),  $s''[(t, \xi' \wedge \xi)] \equiv_{\mu} s''[(u, \xi' \wedge \xi)]$ , and so  $s'[(s[(t, \xi)], \xi')] \equiv_{\mu} s'[(s[(u, \xi)], \xi')]$ . Since  $s'$  was arbitrary, it follows by (29) that  $s[(t, \xi)] \equiv^{\xi'} s[(u, \xi)]$ . This proves (32).

Now, in the chosen context typing  $C$ , the function  $\Theta_C$  from (14) in Section 4.2 is simply given by

$$\Theta_C([\xi'], \xi) = [\xi' \wedge \xi]$$

But this means that (32) entails LC-Subst( $\mu, C$ ), or if you wish the corresponding principle for  $(S, \Psi)$ . Note here that LC-Subst( $\mu, C$ ) is about simultaneous substitution of terms, but in view of (29), if  $t \equiv^{\xi' \wedge \xi} u$ , substitution of  $u$  for  $t$  can never lead from a meaningful to a meaningless term. Therefore, the substitutions in LC-Subst( $\mu, C$ ) can be performed one by one,<sup>30</sup> and (32) says that meaning is preserved at each step. Thus, LC-Funct( $S, \Psi$ ) holds as well (cf. Proposition 4 and Fact 5.)  $\square$

*Proof of Fact 7:* We prove the case  $n = 1$  (i.e. GEN) and indicate how the idea extends to other  $n$ . Consider a grammar with three atomic terms  $a, b, c$  and one one-place operator  $\alpha$ , and let  $\mu$  be a total semantics such that

$$a \equiv_{\mu} b \equiv_{\mu} c, \quad \bar{\alpha}(a) \equiv_{\mu} \bar{\alpha}(b) \not\equiv_{\mu} \bar{\alpha}(c), \text{ and } \bar{\alpha}(\bar{\alpha}(a)) \not\equiv_{\mu} \bar{\alpha}(\bar{\alpha}(b)).$$

We claim that  $\mu$  is not compositionally generalizable. Assume, for *reductio*, that there is a pair  $(S, \Psi)$  generalizing  $\mu$  that satisfies GEN. Observe first that since in the context  $\xi = \langle (\alpha, 1) \rangle$  we have

$$\bar{\alpha}(b) = \bar{\alpha}(b)[(b, \xi)] \not\equiv_{\mu} \bar{\alpha}(b)[(c, \xi)] = \bar{\alpha}(c),$$

it follows by LC-Funct( $S, \Psi$ ) (or the corresponding substitution condition) that  $b \not\equiv_{\mu[\xi]} c$ , where  $\mu[\xi] = \Psi(\alpha, 1, \mu)$ . Since  $b \equiv_{\mu} c$ , the pair  $(\alpha, 1)$  is not a keeper, whence it must be a switcher, which means that:

$$(33) \quad \text{For any function } \nu \text{ in } S, \Psi(\alpha, 1, \nu) = \mu^{\xi}.$$

Next, by assumption, for any term  $s$  containing  $\xi$  we have

<sup>30</sup>Hodges [13] notes that a sufficient condition for this is the *Husserl property*, i.e. that synonymous terms are meaningful in the same linguistic contexts. The semantics given in (29) clearly has the Husserl property.

$$s[(a, \xi)] \equiv_{\mu} s[(b, \xi)]$$

It then follows by MAX that

$$(34) \quad a \equiv_{\mu^{[\xi]}} b$$

Let  $\xi' = \xi \wedge \langle (\alpha, 1) \rangle$ . Now, since

$$\bar{\alpha}(\bar{\alpha}(a)) = \bar{\alpha}(\bar{\alpha}(a))[(a, \xi')] \not\equiv_{\mu} \bar{\alpha}(\bar{\alpha}(a))[(b, \xi')] = \bar{\alpha}(\bar{\alpha}(b))$$

we must, as above, have  $a \not\equiv_{\mu^{[\xi']}} b$ , where  $\mu^{[\xi']} = \Psi(\alpha, 1, \mu^{[\xi]})$ . But from (33) it follows that  $\mu^{[\xi']} = \mu^{[\xi]}$ , which contradicts (34). Thus,  $\mu$  is not first-grade compositionally generalizable.

We can easily extend this substitution failure format to yield a function  $\mu$  that is not generalizable into a *second-grade* g-compositional semantics: just start with four ( $\mu$ -)synonymous atoms, of which three are synonymous under  $\bar{\alpha}$ , of which two are synonymous under  $\bar{\alpha}(\bar{\alpha})$ , which in turn are not synonymous under  $\bar{\alpha}(\bar{\alpha}(\bar{\alpha}))$ . And so on.  $\square$

The construction in this proof gives a natural generalization of the substitution failure format, where standard substitution failure is the first element of this sequence, with two terms that are atomically synonymous but not under  $\bar{\alpha}$ . This also speaks in favor of the naturalness of the generalization requirements in GEN. We may note that Frege's infinite hierarchy of indirect reference can provide substitution failure for generalization of any finite grade.

## References

- [1] Herman Cappelen and Ernest LePore. *Language Turned on Itself. The Semantics and Pragmatics of Metalinguistic Discourse*. Oxford University Press, Oxford, 2007.
- [2] Herman Cappelen and Ernest LePore. Quotation. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Winter 2008.
- [3] Herbert Clark and Richard Gerrig. Quotations as demonstrations. *Language*, 66(4):764–805, 1990.
- [4] Donald Davidson. Quotation. *Theory and Decision*, 11:27–40, 1979. Reprinted in [5], 79–92.
- [5] Donald Davidson. *Inquiries into Truth and Interpretation*. Clarendon Press, Oxford, 1984.
- [6] Gottlob Frege. Über Sinn und Bedeutung. In Peter Geach and Max Black, editors, *Translations from the Philosophical Writings of Gottlob Frege*, 1952, pages 56–78. Blackwell, Oxford, 1892.
- [7] Gottlob Frege. Letter to Russell 28.12 1902. In Gottfried Gabriel, editor, *Wissenschaftlicher Briefwechsel*, pages 234–7. Felix Meiner Verlag, Hamburg, 1976.
- [8] P. Geach. *Mental Acts*. Routledge Kegan Paul, London, 1957.

- [9] Kathrin Glüer and Peter Pagin. Proper names and relational modality. *Linguistics & Philosophy*, pages 507–35, 2006.
- [10] Kathrin Glüer and Peter Pagin. Relational modality. *Journal of Logic, Language and Information*, 17:307–22, 2008.
- [11] Kathrin Glüer and Peter Pagin. General terms and relational modality. *Nous*, forthcoming, 2010.
- [12] Wolfram Hinzen, Edouard Machery, and Markus Werning, editors. *The Oxford Handbook of Compositionality*. Oxford University Press, Oxford, to appear.
- [13] Wilfrid Hodges. Formal features of compositionality. *Journal of Logic, Language and Information*, 10:7–28, 2001.
- [14] David Kaplan. Demonstratives: An essay on the semantics, logic, metaphysics, and epistemology of demonstratives and other indexicals. In Joseph Almog, John Perry, and Howard Wettstein, editors, *Themes from Kaplan*, pages 481–566. Oxford University Press, Oxford, 1989.
- [15] Marcus Kracht. The emergence of syntactic structure. *Linguistics & Philosophy*, 30:47–95, 2007.
- [16] David Lewis. Index, context, and content. In Stig Kanger and Sven Öhman, editors, *Philosophy and Grammar*. D. Reidel, Dordrecht, 1980.
- [17] Peter Pagin. Compositionality and context. In Gerhard Preyer and Georg Peter, editors, *Contextualism in Philosophy*, pages 303–48. Oxford University Press, Oxford, 2005.
- [18] Peter Pagin and Dag Westerståhl. Compositionality I: Definitions and variants. *Philosophy Compass*, xx:xx, 2010.
- [19] Peter Pagin and Dag Westerståhl. Compositionality II: Arguments and problems. *Philosophy Compass*, xx:xx, 2010.
- [20] Terence Parsons. What do quotation marks name? Frege’s theory of quotations and that clauses. *Philosophical Studies*, 42:3:315–328, 1982.
- [21] Christopher Potts. The dimensions of quotation. In Chris Barker and Pauline Jacobson, editors, *Direct Compositionality*, pages 405–31. Oxford University Press, Oxford, 2007.
- [22] Stefano Predelli. The demonstrative theory of quotation. *Linguistics & Philosophy*, 31:5:555–572, 2008.
- [23] W.V.O. Quine. *Mathematical Logic*. Harvard University Press, Boston, Mass., 1940.
- [24] W.V.O. Quine. *Word and Object*. MIT Press, Cambridge, Mass., 1960.
- [25] François Recanati. Open quotation. *Mind*, 110:637–687, 2001.
- [26] François Recanati. Open quotation revisited. *Philosophical Perspectives*, 22:443–471, 2008.
- [27] François Recanati. Compositionality, flexibility and context-dependence. In Hinzen et al. [12].
- [28] P. Saka. Quotation and the use-mention distinction. *Mind*, 107:113–136, 1998.
- [29] John Searle. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, Cambridge, 1969.

- [30] A. Tarski. The concept of truth in formalized languages. In *Logic, Semantics, Metamathematics, 2nd ed.*, pages 152–278. Hackett, 1983, Indianapolis, 1933.
- [31] A. Tarski. The semantic conception of truth. In L. Linsky, editor, *Semantics and the Philosophy of Language*, pages 13–47. University of Illinois Press, Urbana, 1952.
- [32] C. Washington. The identity theory of quotation. *The Journal of Philosophy*, 89:582–605, 1992.
- [33] Markus Werning. Right and wrong reasons for compositionality. In Edouard Machery, Markus Werning, and Gerhard Schurz, editors, *The Compositionality of Meaning and Content: Foundational Issues*, volume I, pages 285–309. Ontos, Frankfurt/Lancaster, 2005.
- [34] Dag Westerståhl. On mathematical proofs of the vacuity of compositionality. *Linguistics & Philosophy*, 21:635–643, 1998.
- [35] Dag Westerståhl. On the compositionality of idioms; an abstract approach. In D. Barker-Plummer et al., editor, *Words, Proofs, and Diagrams*, pages 241–271. CSLI Publications, Stanford, 2002.
- [36] Dag Westerståhl. On the compositional extension problem. *Journal of Philosophical Logic*, 33:549–582, 2004.
- [37] Dag Westerståhl. Compositionality in Kaplan style semantics. In Hinzen et al. [12].