Expressing the meaning of quotation

Jan Wiślicki (jan.wislicki@gmail.com)
University of Warsaw

The aim of this talk is twofold. First, it is to argue against the functional approach to quotation, according to which quotation marks play the role of functor, so that the value of quotation function is a quoting expression. Second, it is to present an account that allows to compose the meaning of quotation from quotes and what they flank via standard composition principles.

The problem. According to the widely accepted post-Tarskian view (cf. Richard 1986, Pagin & Westerståhl 2010, Gaskin & Hill 2013), enquotation can be conceived of as a function from expressions of the first-order language (not their names!) to quoting expressions. However, as shown by Read (1997), such an account leads to semantic inconsistencies. To illustrate, let Q stand for a function that maps expressions onto quoting expressions, so that $Q(p) = '\,p\,'$. Consider now p = the second word of the sentence (1). Then if (1) explicates in words the definition of quotation given above, the formula for quotation reads as follows:

(1) Q maps the second word of the sentence (1) onto 'the second word of the sentence (1)'.

This is of course false, since 'the second word of the sentence (1)' does not quote 'maps'. The reason is that the argument *p* was meant to be an expression of the *first*-order language. Thus it is a map the value of which does not allow to calculate the quoting expression. In order to salvage the function *Q* the argument must have denoted a class of material strings. But this is generally meant to be the *value* of quotation function. Therefore one could redefine *Q*, so that $Q('\,p\,') = '\,p\,'$. Then the problem disappears, but *Q* turns out to be the identity function.

This gives rise to the problem of the semantics of quotation. Standardly, the meaning of the expressions of the first-order language is expressed by means of the semantic function defined on their quotational names. Thus for quotation the function $[\![\bullet]\!]$ should be defined on names of quoting expressions. But then the meaning of quotation could not be expressed without another operation of enquotation. The semantics of enquotation remains undefined.

Semantic analysis of quotation. I argue (contra e.g. Gutzmann & Stei 2011) that quotation marks are semantically, not only pragmatically, relevant (cf. Gaskin & Hill 2013). My proposal rests upon two steps. First, I define the input for the semantic analysis of quotation. Second, I discuss how quotation marks can be composed with the quoting expression in order to allow to calculate the meaning of quotation via standard composition principles.

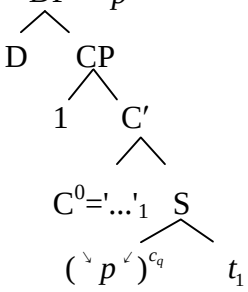
The observation that follows from the discussion above is this. Enquotation cannot be a function defined on expressions of the first-order language or their quotational names. I therefore claim that at least one semantic role of quotation marks lies in determining different input for the semantic analysis. They allow mappings between tokens conceived of as physical objects used in utterances. While a similar observation was the bottom line of Davidson's demonstrative theory, the theory does not specify the semantics of quotation. The idea is to draw on Reichenbach's (1947) arrow quotes. The point is that arrow quotes *do not produce any new name*. $\lfloor\, p \,\rfloor$ is not a semantic map, but a mere physical shape exemplified by this token. By contrast, quotation marks determine the name of a class. $'\lfloor\, p \,\rfloor'$ denotes a class of tokens that are quotable by $\lfloor\, p \,\rfloor$. Note that though $'\lfloor\, p \,\rfloor'$ is a map, the argument $\lfloor\, p \,\rfloor$ allows to calculate its value. Thus $'\lfloor\, p \,\rfloor'$, not its name, can be bound directly by the semantic function.

The problem is which composition principle allows to calculate the meaning of $'\lfloor\, p \,\rfloor'$. The functional application (FA) of the form $[\![' \lfloor p \rfloor ']\!] = [\![' \ldots ']\!]([\![\lfloor p \rfloor]\!])$ will not do. Note that

the functor corresponding to quotes must have been formulated in the metalanguage. But then the definition would be circular. Moreover, since quotation marks are supposed to be semantically relevant, they cannot be treated on a par with $^{\searrow} p^{\swarrow}$, viz. as mere physical shapes.

I propose that the role of quotation marks is twofold. First, they work like *such that* clauses with complementizer-like node in that they signal predicate abstraction (PA). Second, they impose the particular type of linguistic context that specifies the denotation. In order to encode both functions of quotation marks, I use Sharvit's (2011) extended account of PA, viz. $[\![\sim]\!]^{c,g} = {}^{a} x \in D.[\![¥]\!]^{c,g_{[k\to x]}}$, where the context index is added. However, the rule for quotation is slightly different. The idea is that the whole mechanism is triggered by *two types of context and the function g*. I let then a context $c = \{c_l, c_s\}$ consist of the linguistic context and the situational context. The crucial step comes from Pagin & Westerståhl (2010) who show how c_l may impose the *quotational context*. The LF of a quoting expression looks then:

(2)



such that: $^{\searrow} p^{\swarrow}, t_1 \in D_e$ & $'^{\searrow} p^{\swarrow}' \in D_{<et,t>}$ & $(^{\searrow} p^{\swarrow})^{c_q} \in D_{<e,t^{c_q}>}$, where $<e,t^{c_q}>$ is a predicate that maps onto truth value *provided the token* $^{\searrow} p^{\swarrow}$ *is used in the linguistic quotational context* c_q

The tree above shows the following semantic structure. A quoting expression, say 'dog', denotes a *class* of tokens *such that* for every element of this class *it is true that the quoting token* $^{\searrow}$dog$^{\swarrow}$ *quotes this element*. Quotes can be thus conceived of as an element of the linguistic context that provides a description and specifies the predicate; the function g and the quoting token determine the arguments. The semantic rule for quotation reads then:

$[\![' {}^{\searrow} p^{\swarrow} ']\!]^{c,g_{[k\to x]}} = {}^{a} x \in D.[\![{}^{\searrow} p^{\swarrow}$ quotes $t_1]\!]^{c,g_{[1\to x]}} = {}^{a} x \in D.[\![{}^{\searrow} p^{\swarrow}$ quotes$]\!]^{c_l}([\![t_1]\!]^{g_{[1\to x]}})$ in $c_s =$

$= {}^{a} x \in D.{}^{\searrow} p^{\swarrow}$ quotes x in $c_s = 1$, iff $^{\searrow} p^{\swarrow}$ quotes x in $c_s : c_l, c_s \in c$ & $k \in c_l$ & $k = 1$

Quotation thus determines a set of tokens that are quotable by the given token in c.

Conclusion. The gist of the foregoing discussion is this. Quotation cannot, on pains of circularity or semantic inconsistency, be conceived of as a functor. A compositional semantic definition can be thus given via Sharvit-like PA, but not FA. Moreover, the extended account of PA allows specifying two facts. First, which node of the LF (viz. which part of c_l) specifies the crucial predicate. The predicate determines the quotational character of the expression. Second, that c_s may leave some room for non-verbatim modes of quotation, e.g. in direct discourse. The quoting expression and the quoted expression do not have to be identical; they must possess only those features that are salient in a given c. This makes Davidson's theory of quotation more explicit and introduces its formal semantic interpretation.

References

Gaskin Richard, Hill Daniel J. (2013), 'Reach's puzzle and Mention', *dialectica* (67) 201-222

Gutzmann Daniel, Stei Erik (2011), 'How quotation marks what people do with words', *Journal of Pragmatics* (43): 2650-2663

Pagin Peter, Westerståhl Dag (2010), 'Pure quotation and general compositionality', *Linguistics and Philosophy* (33): 381-415

Read Stephen (1997), 'Quotation and Reach's Puzzle', *Acta Analytica* (19): 9-20

Reichenbach Hans (1947), *Elements of Symbolic Logic*, New York: Free Press

Richard Mark (1986), 'Quotation, grammar, and opacity', *Linguistics and Philosophy* (9): 383-403

Sharvit Yael (2011), 'Covaluation and Unexpected BT Effects', *Journal of Semantics* (28): 55-106