On compositional semantics

Wlodek Zadrozny

IBM Research
T. J. Watson Research Center
Yorktown Heights, NY 10598
WLODZ @ WATSON.IBM.COM

Abstract. We prove a theorem stating that any semantics can be encoded as a compositional semantics, which means that, essentially, the standard definition of compositionality is formally vacuous. We then show that when one requires compositional semantics to be "systematic" (that is the meaning function cannot be arbitrary, but must belong to some class), one can easily distinguish between compositional and non-compositional semantics. We also present an example of a simple grammar for which there is no "systematic" compositional semantics. This implies that it is possible to distinguish "good" and "bad" grammars on the basis of whether they can have compositional semantics. As a result, we believe that the paper clarifies the concept of compositionality and opens a possibility of making systematic comparisons of different systems of grammars and NLU programs.

1.Introduction.

Compositionality is defined as the property that the meaning of a whole is a function of the meaning of its parts (cf. e.g. Keenan and Faltz (1985),pp.24-25).[1] This definition, although intuitively clear, does not work formally. For instance, Hirst (1987) pp.27-43 claims that the semantics of Woods (1967) and (Woods, 1969), is not compositional, because "the interpretation of the word *depart* varies as different prepositional phrases are attached to it":

AA-57 departs from Boston
= > **depart(aa-57, boston).**

AA-57 departs from Boston to Chicago
= > **connect(aa-57, boston, chicago).**

AA-57 departs from Boston on Monday
= > **dday(aa-57, boston, monday).**

AA-57 departs from Boston at 8:00 a.m.
= > **equal(dtime(aa-57, boston), 8:00am).**

AA-57 departs from Boston after 8:00 a.m.
= > **greater(dtime(aa-57,boston),8:00am).**

AA-57 departs from Boston before 8:00 a.m.
= > **greater(8:00am,dtime(aa-57,boston)).**

Although this semantics does look like non-compositional, it is easy to create a function that produces the meanings of all these sentences from the meanings of its parts -- we can simply define such a function by cases: the meaning of *departs|from|* is *connect*, the meaning of *departs|from|on* is *dday*, and so on. Hirst therefore changes the definition of compositionality to "the meaning of a whole is a systematic meaning of the parts" (op. cit. p.27.; the emphasis is ours), but without defining the meaning of the word "systematic."

In this paper we show that, indeed, Hirst was right in assuming that the standard definition of compositionality has to be amended. Namely, we prove a theorem stating that any semantics can be encoded as a compositional semantics, which means that, essentially, the standard definition of compositionality is formally vacuous. We then show that when one requires composi-

[1] An equivalent definition, e.g. Partee et al. (1990), postulates the existence of a homomorphism from syntax to semantics.

tional semantics to be "systematic" (i.e. the meaning function must belong to some class), one can easily distinguish between compositional and non-compositional semantics. We also give an example of a simple grammar for which there is no "systematic" compositional semantics". This result implies that it is possible to distinguish "good" and "bad" grammars on the basis of whether they can have a compositional semantics with a meaning function belonging to a certain class. As a result, we believe that the paper finally clarifies the concept of compositionality and opens a possibility of making systematic comparisons of different systems of grammars and NLU programs.

2.Some compositional meaning function can always be found

Compositional semantics, or CS, is usually defined as a functional dependence of the meaning of an expression on the meanings of its parts. One of the first natural questions we might want to ask is whether a set of NL expressions, i.e. a language, can have some CS. This question has been answered positively by van Benthem (1982). However his result says nothing about what kinds of things should be assigned e.g. to nouns, where, obviously, we would like nouns to be mapped into sets of entities, or something like that. That is, we want semantics to encode some basic intuitions, e.g. that nouns denote sets of entities, and verbs denote relations between entities, and so on.

So what about having a compositional semantics that agrees with intuitions? That is, the questions is whether after deciding what sentences and their parts mean, we can find a function that would compose the meaning of a whole from the meanings of its parts.

The answer to this question is somewhat disturbing. It turns out that whatever we decide that some language expressions should mean, it is always possible to produce a function that would give CS to it (see below for a more precise formulation of this fact). The upshot is that compositionality, as commonly defined, is not a strong constraint on a semantic theory.

The intuitions behind this result can be illustrated quite simply: Consider the language of finite strings of digits from 0 to 7. Let's fix a random function (i.e. an intuitively bizarre function) from this language into {0,1}. Let the meaning function be defined as the value of the string as the corresponding number in base 8 if the value of the function is 0, and in base 10, otherwise. Clearly, the meaning of any string is a composition of the meanings of digits (notice that the values of the digits are the same in both bases). But, intuitively, this situation is different from standard cases when we consider only one base and the meaning of a string is given by a simple formula referring only to digits and their positions in the string. The theorem we prove below shows that however complex is the language, and whatever strange meanings we want to assign to its expressions, we can always do it compositionally.

One of the more bizarre consequences of this fact is that we do not have to start building compositional semantics for NL beginning with assigning meanings to words. We can do equally well by assigning meanings to *phonems* or even *LETTERS*, assuring that, for any sentence, the intuitive meaning we associate with it would be a function of the meaning of the letters from which this sentence is composed.

PROVING EXISTENCE OF COMPOSITIONAL SEMANTICS

Let S be any collection of expressions (intuitively, sentences and their parts). Let M be a set s.t. for any $s \in S$, there is $m = m(s)$ which is a member of M s.t. m is the meaning of s. We want to show that there is a compositional semantics for S which agrees with the function associating m with $m(s)$, which will be denoted by $m(x)$.

Since elements of M can be of any type, we do not automatically have (for all elements of S)

$m(s.t) = m(s)\#m(t)$ (where # is some operation on the meanings). To get this kind of homomorphism we have to perform a type raising operation that would map elements of S into functions and then the functions into the required meanings. We begin by trivially extending the language S by adding to it an "end of expression" character \$, which may appear only as the last element of any expression. The purpose of it is to encode the function $m(x)$ in the following way: The meaning function μ that provides compositional semantics for S maps it into a set of functions in such a way that $\mu(s.t) = \mu(s)(\mu(t))$. We want that the original semantics be easily decoded from $\mu(s)$, and therefore we require that, for all s, $\mu(s.\$) = m(s)$. Note that such a type raising operation is quite common both in mathematics (e.g. 1 being a function equal to 1 for all values) and in mathematical linguistics. Secondly, we assume here that there is only one way of composing elements of S -- by concatenation[2] but all our arguments work for languages with many operators as well.

THEOREM. Under the above assumptions. There is a function μ s.t, for all s, $\mu(s.t) = \mu(s)(\mu(t))$, and $\mu(s.\$) = m(s)$.

Proof. See Section 5.1.

3.What do we really want from compositional semantics?

In view of the above theorem, any semantics is equivalent to a compositional semantics, and hence it would be meaningless to keep the definition of compositionality as the existence of a homomorphism from syntax to semantics without imposing some conditions on this homomorphism. Notice that *requiring the computability of the meaning function won't do.*[3]

Proposition. If the original function $m(x)$ is computable, so is the meaning function $\mu(x)$.

Proof. See the proof of the solution lemma in Aczel (1987).

3.1 What do we really want?

We have some intuitions and a bunch of examples associated with the concept of compositionality; e.g. for NP -> Adj N , we can map nouns and adjectives into sets and the concatenation into set intersection, and get an intuitively correct semantics for expressions like "grey carpet", "blue dog", etc.

There seem to be two issues here: (1) Such procedures work for limited domains like "everyday solids" and colors; (2) The function that composes the meanings should be "easily" definable, e.g. in terms of boolean operations on sets. This can be made precise for instance along the lines of Manaster-Ramer and Zadrozny (1990), where we argue that one can compare expressive power of various grammatical formalisms in terms of relations that they allow us to define; the same approach can be applied to semantics, as we show it below.

3.1 A simple grammar without a systematic semantics

If meanings have to be expressed using certain natural, but restricted, set of operations, it may turn out that even simple grammars do not have a compositional semantics.

Consider two grammars of numerals in base 10:

Grammar ND

- N <— N D
- N <— D
- D <— 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

And the second grammar

[2] We do not assume that concatenation is associative, that is (a.(b.c)) = ((a.b).c). Intuitively, this means that we assign semantics to parse trees, not to strings of words. But the method of proof can be modified to handle the case when concatenation is associative.

[3] Also, note that in mathematics (where semantics obviously is compositional) we can talk about noncomputable functions, and it is usually clear what we postulate about them.

Grammar DN

- N <— D N
- N <— D
- D <— 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

For the grammar ND, the meaning of any numeral can be expressed in the model $(Nat, +, \times, 10)$ as

$$\mu(N\ D) = 10 \times \mu(N) + \mu(D)$$

that is a polynomial in two variables with coefficients in natural numbers.

For the grammar DN, we can prove that no such a polynomial exists, that is

Theorem. There is no polynomial p in the variables $\mu(D)$, $\mu(N)$ such that

$$\mu(D\ N) = p(\mu(D), \mu(N))$$

and such that the value of $\mu(D\ N)$ is the number expressed by the string $D\ N$ in base 10.

Proof. See Section 5.2.

But notice that there is a compositional semantics for the grammar DN that does not agree with intuitions: $\mu(D\ N) = 10 \times \mu(N) + \mu(D)$, which corresponds to reading the number backwards. And there are many other semantics corresponding to all possible polynomials in $\mu(D)$ and $\mu(N)$.

Also observe that (a) if we specify enough values of the meaning function we can exclude any particular polynomial; (b) if we do not restrict the degree of the polynomial, we can write one that would give any values we want on a finite number of words in the grammar.

The moral is that not only it is natural to restrict meaning functions to, say, polynomials, but to further restrict them, e.g. to polynomials of degree 1. Then by specifying only three values of the meaning function we can (a) have a unique compositional semantics for the first grammar; (b) show that there is no compositional semantics for the second grammar (directly from the proof of the above theorem).

4. Conclusions

4.1. Relevance for theories of grammar

4.1.1. On reduction of syntax to lexical meanings

T. Wasow on pp.204-205 of Sells (1985) writes:

> It is interesting that contemporary syntactic theories seem to be converging on the idea that sentence structure is generally predictable from word meanings [...]. [...] The surprising thing (to linguist) has been how little needs to be stipulated beyond lexical meaning. [...]

The reader should notice that the meaning function m in our main theorem is arbitrary. In particular we can take $m(s)$ to be the preferred *syntactic* analysis of the string s. The theorem then confirms the above observation: indeed, the syntax can be reduced to lexical meanings. At the same time, it both trivializes it and calls out for a deeper explanation. It trivializes the reduction to lexical meanings, since it also says that with no restriction on the types of meanings permitted, syntax can be reduced to the meaning of phonems or letters. The benefits of the reduction to lexical meanings would have to be explained, especially if such meanings refer to abstract properties such as binding features, BAR, or different kinds of subcategorization.

It is the view of this author (cf. Zadrozny and Manaster-Ramer (199?)) and, implicitly, of Fillmore et al. (1988), that such a reductionist approach is inappropriate. But we have no room to elaborate it here.

4.1.2. On good and bad grammars

By introducing restrictions on semantic functions, i.e. demanding the *systematicity* of semantics, we can for the first time formalize the intuitions that linguists have had for a long time about "good" and "bad" grammars (cf. Manaster-Ramer and Zadrozny (1992)). This allows us

to begin dealing in a rigorous way with the problem (posed by Marsh and Partee) of constraining the power of the semantic as well as the syntactic components of a grammar.

We can show for instance (*ibid.*) that some restrictions have the effect of making it in principle impossible to assign correct meanings to arbitrary sets of matched singulars and plurals if the underlying grammar does not have a unitary rule of reduplication. Thus, grammars such as wrapping grammars (Bach (1979)), TAGs (e.g., Joshi (1987)), head grammars, LFG (e.g., Sells (1985)) and queue grammars (Manaster-Ramer (1986)), all of which can generate such a language, all fail to have systematic semantics for it. On the other hand, we can exhibit other grammars (including old-fashioned transformational grammars) which do have systematic semantics for such a language.

4.2. What kind of semantics for NL?

In view of the above results we should perhaps discuss some of the options we have in semantics of NL, especially in context of NLU by computers. To focus our attention, let's consider the options we have to deal with the semantics of *depart* as described in Section 1.

- Do nothing. That is, to assume that the semantics is given by sets of procedures associated with particular patterns; e.g. "X departs from Y" gets translated into "depart(X,Y)".
- Give it semantics a la Montague, for instance, along the lines of Dowty (1979) (see esp. Chapter 7). Such a semantics is quite complicated, not very readable, and it is not clear what would be accomplished by doing this. However note that this doesn't mean that it would not be computational -- Hobbs and Rosenschein (1977) show how to translate Montagovian semantics into Lisp functions.
- Restrict the meaning of compositionality requiring for example that the meaning of a verb is a relation with the number of arguments equal to the number of arguments of the verb. If the PP following the verb is treated as one argument, there is no compositional semantics that would agree with the intended meanings of the example sentences. This would formally justify the arguments of Hirst.
- Recognize that depending on the PPs the meaning of "X departs PP" varies, and describe this dependence via a set of meaning postulates (Bernth and Lappin (1991) show how to do it in a computational context). In such a case the semantics is not given directly as a homomorphism from syntax into some algebra of meanings, but indirectly, by restricting the class of such algebras by the meaning postulates.
- Admit that the separation of syntax and semantics does not work in practice, and work with representations in which form and meaning are not separated, that is, there cannot be a separate syntax, except in restricted domains or for small fragments of language. This view of language has been advocated by Fillmore et al. (1988), and shown in Zadrozny and Manaster-Ramer (199?), Zadrozny and Manaster-Ramer (199?) to be computationally feasible.
- Hope that the meaning will emerge from non-symbolic representations, as advocated by the "connectionists."

5.The proofs

5.1. The Existence of compositional meaning functions

Let S be any collection of expressions (intuitively, sentences and their parts). Let M be a set s.t. for any s which is a member of S, there is $m = m(s)$ which is a member of M s.t. m is the meaning of s. We want to show that there is a compositional semantics for S which agrees with the function associating m with $m(s)$, which will be denoted by $m(x)$. To get the ho-

momorphism from syntax to semantics we have to perform a type raising operation that would map elements of S into functions and then the functions into the required meanings.

As we have described it in Section 2, we extend S by adding to it the "end of expression" character $\$$, which may appear only as the last element of any expression. Under these assumptions we prove:

THEOREM. There is a function μ s.t, for all s,

$\mu(s.t) = \mu(s)(\mu(t))$, and

$\mu(s.\$) = m(s)$.

Proof. Let $t(0), t(1), \ldots, t(\alpha)$ enumerate S. We can create a big table specifying meaning values for all strings and their combinations. Then the conditions above can be written as a set of equations shown in the figure below

$$\mu(t(0)) = \{ <\$, m(t(0))>, <\mu(t(0)), \mu(t(0).t(0))>, \ldots, <\mu(t(\alpha)), \mu(t(0).t(\alpha))>, \ldots \}$$
$$\mu(t(1)) = \{ <\$, m(t(1))>, <\mu(t(0)), \mu(t(1).t(0))>, \ldots, <\mu(t(\alpha)), \mu(t(1).t(\alpha))>, \ldots \}$$
$$\ldots$$
$$\mu(t(\alpha)) = \{ <\$, m(t(\alpha))>, <\mu(t(0)), \mu(t(\alpha).t(0))>, \ldots, <\mu(t(\alpha)), \mu(t(\alpha).t(\alpha))>, \ldots \}$$
$$\ldots$$

Continuing the proof: By the solution lemma (Aczel (1987) and Barwise and Etchemendy (1987)) this set of equations has a solution (unique), which is a function.

To finish the proof we have to make sure that the equation $\mu(\$) = \$$ holds. Formally, this requires adding the pair $<\$, \$>$ into the graph of μ that was obtained from the solution lemma.

□

We have directly specified the function as a set of pairs with appropriate values. Note that that there is place for recursion in syntactic categories. Also, if a certain string does not belong to the language, we assume that the corresponding value in this table is undefined; thus μ is not necessarily defined for all possible concatenations of strings of S.

Note: The above theorem has been proved in set theory with the anti-foundation axiom, ZFA. This set theory is equiconsistent with the standard system of ZFC, thus the theorem does not assume anything more than what is needed for "standard mathematical practice". Furthermore, ZFA is better suited as foundations for semantics of NL than ZFC (Barwise and Etchemendy (1987)).

5.2. A grammar without compositional semantics

For the grammar DN, we can prove that no such a polynomial exists, that is

Theorem. There is no polynomial p in the variables $\mu(D)$, $\mu(N)$ such that

$$\mu(D\ N) = p(\mu(D), \mu(N))$$

and such that the value of $\mu(D\ N)$ is the number expressed by the string $D\ N$ in base 10.

Proof. We are looking for

$$\mu(D\ N) = p(\mu(N), \mu(D))$$
$$= \mu(D) \times (10^{length(N)}) + \mu(N)$$

where the function p must be a polynomial in these two variables. But such a polynomial does not exist, since it would have to be equal to $\mu(N)$ for $\mu(N)$ in the interval 0..9, and to $\mu(D) \times 10 + \mu(N)$ for $\mu(N)$ in 10..99, and to $\mu(D) \times 100 + \mu(N)$ for $\mu(N)$ in 100..999, and so on. And if the degree of this polynomial was less than 10^n , for some n, it would have to be equal identically to $\mu(D) \times 10^n + \mu(N)$, since it would share with it all the values in $10^n..10^n - 1$, and therefore could not give correct values on the other intervals.

Acknowledgements. Alexis Manaster Ramer brought to my attention the need to clarify the meaning of compositionality, and commented on various aspects of this work. I also benefited greatly from a discussion with John Nerbonne and exchanges of e-mail with Fernando Pereira and Nelson Correa. (Needless to say, all the remaining faults of the paper are mine).

Parts of this paper were written at the University of Kaiserslautern; I'd like to thank Alexander von Humboldt Stiftung for supporting my visit there.

6. References

P. Aczel (1987). *Lectures on Non-Well-Founded Sets.* Stanford, CA: CSLI Lecture Notes.

E. Bach (1979). Control in Montague Grammar. *Linguistic Inquiry*, 515-531.

J. Barwise & J. Etchemendy (1987). *The Liar.* New York, NY: Oxford University Press.

A. Bernth & S. Lappin (1991). *A meaning postulate based inference system for natural language.* RC 16947, Yorktown Heights, NY: IBM T.J. Watson Research Center.

D. R. Dowty (1979). *Word Meaning and Montague Grammar.* Dordrecht, Holland: D. Reidel.

C. J. Fillmore, P. Kay, & M. C. O'Connor (1988). Regularity and idiomaticity in grammatical constructions. *Language, 3*, 501-538.

G. Hirst (1987). *Semantic interpretation and the resolution of ambiguity.* Cambridge, Great Britain: Cambridge University Press.

J. Hobbs & S. Rosenschein (1977). Making computational sense of Montague's intentional logic. *Artificial Intelligence, 3*, 287-306.

A. K. Joshi (1987). An Introduction to Tree Adjoining Grammars: In A. Manaster-Ramer (Ed.), *Mathematics of Language* (pp. 87-114). Amsterdam/Philadelphia: John Benjamins.

E. L. Keenan & L. M. Faltz (1985). *Boolean Semantics for Natural Language.* Dordrecht, Holland: D Reidel.

A. Manaster-Ramer (1986). Copying in Natural Languages, Context-Freeness, and Queue Grammars. *Proc. ACL'86*, 85-89.

A. Manaster-Ramer & W. Zadrozny (1990). *Expressive Power of Grammatical Formalisms (Proceedings of Coling-90).* Helsinki, Finnland: Universitas Helsingiensis.

A. Manaster-Ramer & W. Zadrozny (1992). Systematic Semantics. *in preparation.*

B. H. Partee, A. t. Meulen, & R. E. Wall (1990). *Mathematical Methods in Lingusitics.* Dordrecht, The Netherlands: Kluwer.

P. Sells (1985). *Lectures on Contemporary Syntactic Theories.* Stanford, CA: CSLI Lecture Notes (3).

J. van Benthem (1982). The Logic of Semantics: In Fred Lanman & Frank Veltman (Eds.), *Varieties of Formal Semantics.* Dordrecht, Holland: Foris.

W. A. Woods (1967). *Semantics for a question answering system.* Harvard University. PhDThesis

W. A. Woods (1969). Procedural semantics for a question answering machine. *AFIPS conference proceedings, 33*, 457-471.

W. Zadrozny & A. Manaster-Ramer (199?a). The Significance of Constructions. *submitted.*

W. Zadrozny & A. Manaster-Ramer (199?b). Assembling Constructions. *submitted.*