On the Semantics of Root Syntax: Challenges and Directions

Chenchen Song

Zhejiang University, Hangzhou 310058, China,
cjs021@zju.edu.cn

Abstract. This paper discusses the challenges of root syntax for formal semantics and presents some possible directions toward a compositional semantics that can handle roots. In particular, two novel, promising directions are developed based on category theory. The paper makes two theoretical contributions. It develops new techniques to bridge a gap between formal syntax and formal semantics on the one hand and demonstrates a new way to apply category theory in linguistics on the other.

Keywords: root syntax, formal semantics, category theory, topos, monad

1 Introduction

In this paper, I discuss the challenges root syntax poses for formal semantics and present some directions toward a compositional semantics that has a suitable place for roots. In this section, I first give a brief introduction to root syntax. In Sec. 2, I go on to list four conceptual and technical challenges of root syntax for formal semantics. Then, in Sec. 3, I comparatively discuss five potential directions to develop a compositional semantics for root syntax, two of which are novel. Next, in Secs. 4–5, I present the two novel directions in more detail, which I deem more promising than the other directions. They both use category theory. The contribution of this paper is twofold. First, it develops new techniques to bridge a conspicuous gap between generative syntax and model-theoretic semantics. Second, it demonstrates a new way to apply category theory in linguistics. While previous applications are mostly embedded in the Lambek school of linguistics, the current application is natively developed for the Chomsky school.

Since Halle and Marantz's (1993) seminal work on distributed morphology, the formal syntactic notion *root* has gained increasing popularity among linguists in the Chomsky school. Syntactic theories that explicitly promote roots can be called root syntax. See Alexiadou et al (2014) and Doron (2014) for some recent studies by researchers in this area. This paper is mainly built on Marantzian root syntax, though the techniques I develop are compatible with any incarnation of root syntax that accepts the following basic definition:

(1) A root is a purely lexical node that is void of categorial information.

The lexical vs. functional division of the lexicon has a long tradition in linguistics. Intuitively, a lexical morpheme (e.g., *dog*) is contentful while a functional

morpheme (e.g., *and*) is grammatical, and a root, being the epitome of lexicality, is deprived of grammatical information, including a category. Such a unit is seldom directly detectable in surface linguistic forms but has nevertheless become an important hypothesis in current generative grammar. A key attraction of root syntax is its pursuit of high granularity in grammatical analysis.

In Marantzian root syntax, a prototypical root-involving structure is that of a bare content word like *dog* or *speak*. Conventionally, these are treated as syntactic atoms. In root syntax, however, they are decomposed into a categorizer and a root, as in (2a). To highlight the point that uncategorized roots do not have concrete content, they are sometimes notated by numerical indices, as in (2b).

(2) a. dog = $[_{\mathrm{N}}\ n\ \sqrt{\textsc{dog}}\ ]$, speak = $[_{\mathrm{V}}\ v\ \sqrt{\textsc{speak}}\ ]$

b. dog = $[_{\mathrm{N}}\ n\ \sqrt{314}\ ]$, speak = $[_{\mathrm{V}}\ v\ \sqrt{528}\ ]$

I will keep using labels like $\sqrt{\textsc{dog}}$, but such labels are merely distinctive and do not imply that roots have concrete meanings like "dog"—or concrete pronunciations like /dɔg/. Familiar sound-meaning pairs like this correspond to the level of categorized roots—namely, the bracketed N/V level in (2).

While root syntax had originally been dedicated to the lexical domain, recently there have been proposals extending it to the functional domain (e.g., Song 2019, Pots 2020). These proposals are motivated by the phenomenon of semifunctionality or semilexicality. Not all vocabulary items in human language are purely lexical or functional; there are also in-betweens, as in (3).[1]

(3) a. *La pasta* ***va*** / ***viene*** *mangiata subito.* [Italian]
the pasta $\textsc{pass}_{\textsc{obl.}}$ $\textsc{pass}_{\textsc{reg.}}$ eaten immediately
"Pasta must be / is eaten immediately."
(adapted from Cardinaletti and Giusti 2001:392)

b. *yī* ***wèi*** / ***míng*** *lǎoshī* [Mandarin]
one $\textsc{clf}_{\textsc{resp.}}$ $\textsc{clf}_{\textsc{prof.}}$ teacher
"a teacher" (adapted from Song 2019:125)

In (3a), the Italian motion verbs *va* 'goes' and *viene* 'comes' are used as passive auxiliaries, but they both carry extra meanings, as is reflected in the English translations "must be eaten" (an obligation) and "is eaten" (a regular event). Similarly, in (3b), the Mandarin classifiers *wèi* and *míng* can both be used for teachers, but they carry different attitudes, with the former being more respectful and the latter being more professional. So, semifunctional items are qualitatively different from purely functional items (e.g., *the*) in that they have conventionalized idiosyncratic content. From the perspective of root syntax, that signals the involvement of a root. Song (2019) proposes a minimal extension of classical root syntax, called *generalized root syntax*, to analyze semifunctional items. On this extension, both content words and semifunctional words are derived by the schema $[_{\mathrm{X}}\ \mathrm{X}\ \sqrt{}\ ]$, where X is any functional category. In natural languages, especially analytic ones, the same root often gets reused in different categories,

[1] Thanks to Michele Sanguanini (p.c.) for clarifying the Italian example to me.

which gives rise to multifunctional situations as in (4a), where *không* is used in three different categories. In generalized root syntax, this corresponds to the derivational distinction in (4b).

(4) a. *tay* ***không***$_{\text{Adj}}$ 'hand **empty**; empty-handed' [Vietnamese]
Tôi ***không***$_{\text{Neg}}$ *hút thuốc.* 'I **not** smoke cigarette; I don't smoke.'
Đúng ***không***$_{\text{Q}}$*?* 'true **Q**; Is that true?'

b. $[_{\text{A}}$ *a* √KHÔNG $]$, $[_{\text{Neg}}$ Neg √KHÔNG $]$, $[_{\text{C}_\text{Q}}$ C_Q √KHÔNG $]$

In sum, the formal syntactic notion root represents the purely lexical aspect of human language. Due to its categoryless nature, a root is not only uninterpretable but also unpronounceable, though it carries the unique information (or identifier) that, when combined with a proper syntactic category, yields a concrete sound-meaning pair. It may be helpful to think of a root as an amorphous nebula of potential states, which relies on categorization to attain a stable state.

2 Challenges for formal semantics

The extremely underspecified nature of roots makes them a nontrivial challenge for model-theoretic (aka formal) semantics. Indeed, root syntax has received little attention in formal semantics. Mainstream semantic studies do not decompose bare words. Thus, the bare noun *dog* is usually given the logical form $\lambda x.\, dog(x)$ and interpreted as a named set of entities. And the bare verb *speak* is given either the logical form $\lambda x.\, speak(x)$ or the form $\lambda e.\, speak(e)$, where e is an event variable. The latter logical form, based on neo-Davidsonian event semantics, is already of a relatively high granularity level, for it severs the arguments out of the verb's core meaning. But since neo-Davidsonian semantics takes idiosyncratically defined eventualities like speaking to be theoretical primitives, its granularity level is still lower than that of root syntax. To see how the above state of affairs is reflected in practice, let us look at two concrete examples from the literature.

(5) a. Jones buttered the toast. (adapted from Landman 2000:1)
$\exists e\,[butter(e) \wedge agent(e) = \text{jones} \wedge theme(e) = \text{toast}]$

b. $\text{push}_{\text{Proc}}$ (adapted from Ramchand 2008:61)
$\lambda y \lambda x \lambda e\,[path(y, e) \wedge push(e) \wedge process(e) \wedge subject(x, e)]$

Abstracting away from their theory-internal differences, we can see that both logical forms in (5) involve a predicate that is directly defined by a lexical verb (i.e., *butter* and *push*). Thus, while both authors otherwise pursue lexical decomposition, they apparently assume that bare words need not or cannot be further decomposed in semantic representation. Examples like these show that root-oriented thinking—or the idea of completely separating compositional and idiosyncratic content in meaning representation—has not had much influence on formal semantics. Below I list four challenges of root syntax for formal semantics.

Challenge 1 (C1) How to logically represent roots? As mentioned above, root content is extremely vague—far vaguer than the labels used in conventional log-

ical forms like $dog(x)$ and $butter(e)$. In Acquaviva's (2009:4) words, the meaning of a root is "so radically underspecified [that it] cannot even convey the distinction between argument and predicate." Given this apparently anti-type-theoretic nature, how can roots be incorporated in a type-based semantic theory at all?

Challenge 2 (C2) Given the extreme vagueness of root content, in a model for root syntax there should be no named sets like $\{x\colon x \text{ is a dog}\}$—or at least they cannot be taken for granted. How can we remove such named sets from our semantics and reconstruct them in a "vaguer" (e.g., set-free) way?

Challenge 3 (C3) Root syntax separates lexical and functional information for formal-derivational reasons, but the semantic module must be able to put the syntactically decomposed components back together. What mode of composition is suitable for root categorization and, more generally, for root-containing structures? Is functional application enough?

Challenge 4 (C4) The conventional treatment of bare content words as predicates can hardly be extended to semifunctional items, which may have far more complex types. Suppose generalized root syntax in on the right track, how can content and semifunctional words get a similarly unified treatment in semantics?

3 Directions

A desirable semantics for root syntax should meet the following criteria:

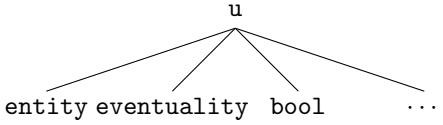
1. It should reflect the categoryless nature of roots. There should not be such primitives as "nominal" or "verbal" roots, for that would be a misunderstanding of root syntax. We need a category-neutral logical form for all roots.
2. It should mirror the syntactic unification of content and semifunctional words. Specifically, any root that can be inserted in a lexical context should be insertable into a functional context as well without causing type errors.

Of course, not all root-categorizer combinations will yield actual sound-meaning pairs, because that depends on what is lexicalized and what is not. I will leave aside the issue of lexicalization and focus on making formal semantics technically compatible with root syntax. P1–3 below are based on Song (2019:59ff.).

Possibility 1 (P1) Roots denote sort-generic predicates, categorizers denote sorting predicates, and the two are composed by $\wedge$. I use the term "sort" for various ground types and save the term "type" for more general (e.g., sort-neutral) contexts. In P1, all the relevant types are sorts, and higher-order types are simply not considered. The idea is that a root denotation has the most generic sort and that the categorizer gives it a more specific sort. See (6) for an illustration.

(6) a. $[\![\sqrt{\text{BOARD}}]\!] = \lambda x\colon \mathtt{u}.\,\text{BOARD}(x), [\![n]\!] = \lambda x\colon \mathtt{u}.\,\mathtt{entity}(x)$
$[\![[_{\text{N}}\ n\ \sqrt{\text{BOARD}}]]\!] = [\![n]\!] \wedge [\![\sqrt{\text{BOARD}}]\!] = \lambda x\colon \mathtt{u}.\,\mathtt{entity}(x) \wedge \text{BOARD}(x)$

b. (a hierarchy of sorts)



Suppose each small capital predicate like BOARD in (6a) gives us a highly generic set in the universe of discourse, such as the set of all individuals that are somehow related to BOARD, then the intersection of this set and the set of entities (which is the extension of n) would be the subset of entities that are somehow related to BOARD. Let us take this to be the lexical semantics of the noun *board*. Again, the "somehow" part is a matter of lexicalization. While P1 is a natural way to interpret the effect of root categorization, it has a major drawback—that is, it requires roots to denote first-order predicates. Thus, P1 is only suitable for classical root syntax but not for generalized root syntax.

Possibility 2 (P2) Roots denote type-general predicates, categorizers denote types, and the two are composed by type-level application. Unlike in P1, here the root's denotation is sort- and type-open, without even the generic `u`. Accordingly, it has a type variable to be filled by the categorizer. To abstract over types, we need second-order typed λ-calculus. See (7) for an illustration, where α is a type variable, x is a term variable, and $*$ is the type of all types (aka a "kind").

(7) $[\![\sqrt{\text{BOARD}}]\!] = \lambda\alpha\colon *\,.\,\lambda x\colon \alpha.\,\text{BOARD}(x), [\![n]\!] = \texttt{entity}\colon *$
$[\![[_{\text{N}}\ n\ \sqrt{\text{BOARD}}]]\!] = [\![\sqrt{\text{BOARD}}]\!]([\![n]\!]) = \lambda x\colon \texttt{entity}.\,\text{BOARD}(x)$

Here we get essentially the same typed predicate as in (6), but unlike in P1, where categorizers denote first-order predicates just like roots do, in P2 they simply denote types, which is made possible by a more complex root denotation. This complication makes P2 compatible with higher-order types, but does that make it compatible with generalized root syntax too? Unfortunately no. Consider (8):

(8) $[\![\sqrt{\text{KHÔNG}}]\!] = \lambda\alpha\colon *\,.\,\lambda x\colon \alpha.\,\text{KHÔNG}(x)$ [Vietnamese]
$[\![\text{Neg}]\!] = \texttt{bool} \to \texttt{bool}\colon *$
$[\![[_{\text{Neg}}\ \text{Neg}\ \sqrt{\text{KHÔNG}}]]\!] = [\![\sqrt{\text{KHÔNG}}]\!]([\![\text{Neg}]\!]) = \lambda x\colon \texttt{bool} \to \texttt{bool}.\,\text{KHÔNG}(x)$

Applying the same pattern from (7) to the semifunctional *không* does not give us the desired logical form. A semifunctional word is still a function word and should perform its grammatical function as a purely functional word does. This means that insofar as computation is concerned, $[\![\text{Neg}]\!]$ and $[\![[_{\text{Neg}}\ \text{Neg}\ \sqrt{\text{KHÔNG}}]]\!]$ should have the same type. But that is not the case in (8), where the semifunctional negator ends up denoting a sorting predicate that sorts a particular function into the KHÔNG class. Thus, P2 cannot handle generalized root syntax either.

Possibility 3 (P3) Roots lack model-theoretic denotations, and there is no composition between roots and categorizers. This is also the approach in Acquaviva (2019), where the author denies the root node any type and treats it as "an unanalyzable name, a label maximally underdetermined except for the fact of being formally distinct from other names" (p.45). This brings us back to a basic idea in distributed morphology. As Marantz (1995:4) puts it: "[I]t is not clear that the computational system of language ... must know whether a node contains 'dog' or 'cat.' ... [T]his difference ... is a matter of Encyclopedic knowledge ... [and] such knowledge is used in semantic interpretation of LF, but not in grammatical computations over LF or involving LF." The key message here is a complete separation of compositional and noncompositional meanings,

whereby roots are invisible not only to syntactic derivation but also to formal semantic composition (i.e., computations involving LF), and their idiosyncratic semantic effects are only manifested when compositional and noncompositional meanings are integrated (i.e., in the semantic interpretation of LF), which is presumably not a totally model-theoretic process. See (9) for an illustration.

(9) $[\![\sqrt{\text{BOARD}}]\!]$ is undefined, $[\![n]\!] = \lambda x\colon \mathtt{u}.\,\mathtt{entity}(x)$
$[\![[_{\text{N}}\ n\ \sqrt{\text{BOARD}}]]\!] = [\![n]\!] = \lambda x\colon \mathtt{u}.\,\mathtt{entity}(x)$
(an entity that is idiosyncratically characterized by $\sqrt{\text{BOARD}}$)

What exactly the idiosyncratic characterization is is a matter of lexicalization. For instance, the lexicalization of $[_{\text{N}}\ n\ \sqrt{\text{BOARD}}\ ]$ involves three essential meanings and some 30 detailed meanings according to Merriam-Webster, not to mention that $\sqrt{\text{BOARD}}$ can also be categorized by v. How the numerous lexicalized meanings of a word are organized in the mental lexicon and how they are disambiguated in LF interpretation are both intriguing questions, and there are many previous studies on both.[2] However, those questions do not belong to the root level but only arise at the categorized root (or stem) level. Hence, they are orthogonal to the main concern of this paper. In relation to root syntax, they correspond to the inner content of the Marantzian "encyclopedia."

Since P3 ignores the root node, it can be easily extended to generalized root syntax, as in (10). Notice how the purely functional Neg and the semifunctional *không*/*đéo* are type-identical this time.

(10) a. $[\![\sqrt{\text{KHÔNG}}]\!]$ is undefined, $[\![\text{Neg}]\!] = \lambda P\colon \mathtt{bool}.\,\neg P\colon \mathtt{bool}$ [Vietnamese]
$[\![[_{\text{Neg}}\ \text{Neg}\ \sqrt{\text{KHÔNG}}]]\!] = [\![\text{Neg}]\!] = \lambda P\colon \mathtt{bool}.\,\neg P\colon \mathtt{bool}$
(a negating function that is idiosyncratically characterized by $\sqrt{\text{KHÔNG}}$)

b. $[\![\sqrt{\text{ĐÉO}}]\!]$ is undefined, $[\![\text{Neg}]\!] = \lambda P\colon \mathtt{bool}.\,\neg P\colon \mathtt{bool}$
$[\![[_{\text{Neg}}\ \text{Neg}\ \sqrt{\text{ĐÉO}}]]\!] = [\![\text{Neg}]\!] = \lambda P\colon \mathtt{bool}.\,\neg P\colon \mathtt{bool}$
(a negating function that is idiosyncratically characterized by $\sqrt{\text{ĐÉO}}$)

This state of affairs is intuitively correct, since the difference between the two negators *không* and *đéo*—with the former being neutral and the latter being vulgar—is indeed irrelevant to grammatical computation.

In sum, neither P1 nor P2 is desirable, as they are both confined to classical root syntax. For this reason, Song (2019) prefers P3, which is in line with Chomsky's view that lexical items do not refer (Acquaviva 2014:281 has a similar view on roots). I have two additional remarks. First, between P1 and P2, P1 is even less desirable, as the conjunctivist approach strictly requires type match between its conjuncts, and so there is no hope to extend it to generalized root syntax. In P2, by contrast, since α ranges over all ordinary types, there might still be some way to make the approach compatible with generalized root syntax. Second, while P3 can uniformly apply to both classical and generalized root syntax, it leaves C3 unaddressed, since the root node is simply ignored. Thus, it can certainly be further improved. P4–5 can be viewed as improvements of P3.

[2] See Asher (2011) for a category-theoretic study of lexical meanings, which addresses stem meaning flexibility among other issues. Thanks to a reviewer for this reference.

Possibility 4 (P4) C2 suggests a higher-level abstraction in semantics, where the universe of discourse does not inherently distinguish idiosyncratic sorts like `dog` and `speak` but only distinguishes built-in ones like `entity` and `eventuality`. Since there is no way to further divide these general sorts in a model of root syntax, they can be viewed as atomic concepts. This is reminiscent of the definition of *object* in category theory, which are opaque by design. In fact, it is common practice in categorical models of logic to interpret types as objects. Thus, it may be interesting to lift the set-theoretic model for natural language semantics to a category-theoretic one and see what that does to root syntax.

Possibility 5 (P5) Our desired compositional vs. noncompositional separation in meaning representation is reminiscent of the two-dimensional view of semantics in Potts (2005, 2007) and Asudeh and Giorgolo (2020), which involves an "at-issue" (truth-conditional) and a "side-issue" (non-truth-conditional) dimension. Root-based meanings are similar to side-issue meanings. Asudeh and Giorgolo tackle side-issue phenomena with the category-theoretic tool of monad. It may be interesting to explore whether that tool can be applied to root syntax.

P4–5 highlight the usefulness of category theory in linguistics. That is not surprising, for there is a long tradition of categorial grammar with many recent developments (mostly based on Lambek's seminal work in the 1980s). That said, in this paper I will stay away from tools commonly used in that area (e.g., tensor products), since the syntax I work with (minimalism) has a fundamentally different design, which those tools simply do not fit. In a sense, the linguistic application of category theory in this paper is "native" to the Chomsky school.

4 A categorical model

In this section, I present P4 in more detail. My goal is to lift the usual set-theoretic model of natural language semantics to a category-theoretic one. In conventional formal semantics, λ-calculus serves as the intermediate language between natural language syntax and its set-theoretic model, so all we need is a categorical model for λ-calculus. This model already exists (Crole 1993), so we can use that as a point of departure. Due to space limitations, I cannot provide definitions for many of the category-theoretic notions I use. Therefore, I must assume some basic familiarity with category theory in my presentation. I recommend the textbooks Crole (1993), Goldblatt (2006), and Awodey (2010).

The category I use to model root syntax is **Set**, the category of sets and functions. Mundane as it is, **Set** has several desirable settings for our purpose. First, it has the closest categorical structure to the usual set-theoretic structure in formal semantics. Second, it is cartesian closed and automatically supports product and function types. Third, it is a topos, which makes it furthermore support subtyping. Below I define global element and subobject classifier, which are directly used in our categorical modeling of root syntax.

Definition 1. A *terminal object* 1 of a category $\mathcal{C}$ has the property that for any object C in $\mathcal{C}$, there is a unique morphism $!_C \colon C \to 1$. A *global element* of an object A in $\mathcal{C}$ is any morphism $1 \to A$.

Remark 1. In **Set**, the terminal object is any singleton set, and global elements allow us to refer to set elements without actually "seeing" them.

Definition 2. In a category $\mathcal{C}$ with a terminal object 1, a *subobject classifier* is an object Ω with a morphism $true\colon 1 \to \Omega$ such that for each monic $f\colon A \to D$ there is a unique $\chi_f\colon D \to \Omega$ making the following diagram a pullback.

$$\begin{array}{ccc} A & \xrightarrow{f} & D \\ {\scriptstyle !}\downarrow & & \downarrow{\scriptstyle \chi_f} \\ 1 & \xrightarrow{true} & \Omega \end{array}$$

Remark 2. In **Set**, Ω is just the set of truth values, and *true* picks out True. Being monic in **Set** makes f an injection, so A is essentially a subset of D, whose characteristic function is χ_f. The commutativity of the diagram simply confirms this subset relation, and Ω classifies all subsets of D in this way.

The logical signature for root syntax is almost identical to that of λ-calculus, with only a tiny difference regarding the definition of ground types. The ground types we need include `u` and its subsorts from (6b) and a type `r` for roots. Our categorical model is also only minimally different from that of λ-calculus.

Definition 3. A model **M** of root syntax in **Set** is specified by giving
- every ground type γ a **Set**-object $[\![\gamma]\!]$, specifically an object for each subsort of `u` and an object for `r`,
- every constant $k\colon \alpha$ a global element $[\![k]\!]\colon 1 \to [\![\alpha]\!]$, and
- every function $f\colon \alpha_1 \dots \alpha_n \to \beta$ a morphism $[\![f]\!]\colon [\![\alpha_1]\!] \times \cdots \times [\![\alpha_n]\!] \to [\![\beta]\!]$,

where we define $[\![\alpha]\!]$ for an arbitrary type α via induction, setting $[\![\texttt{unit}]\!] \coloneqq 1$, $[\![\texttt{bool}]\!] \coloneqq \Omega$, $[\![\alpha \times \beta]\!] \coloneqq [\![\alpha]\!] \times [\![\beta]\!]$, and $[\![\alpha \to \beta]\!] \coloneqq [\![\alpha]\!] \to [\![\beta]\!]$.

I assign $[_{\mathrm{X}}\ \mathrm{X}\ \surd\ ]$ the type $\alpha \times \texttt{r}$, where α is the semantic type of X. Accordingly, I assign it the logical form $\langle \mathrm{X}, \surd \rangle$, or $\mathrm{X}_{\surd}$ for short, and regard this as a root-tagged functional category. In this way, roots both get a place in compositional semantics and do not really participate in the composition (but just tag along). The idiosyncratic sorts qua sets mentioned in C2 (e.g., $\{x\colon x \text{ is a dog}\}$) are reconstructed as morphisms of the shape $[\![\sigma]\!] \to \Omega$, where σ is an ontological sort like `entity`. Since there are many such morphisms, we can view each of them as a conventional sortal predicate. See (11) for an illustration.[3]

$$\begin{aligned} (11)\quad & [\![\surd\mathrm{BOARD}]\!] = 1 \xrightarrow{\mathrm{BOARD}} [\![\texttt{r}]\!], [\![n]\!] = [\![\texttt{entity}]\!] \\ & [\![[_{\mathrm{N}}\ n\ \surd\mathrm{BOARD}]]\!] = [\![n]\!] \times [\![\surd\mathrm{BOARD}]\!] = \langle id_{[\![\texttt{entity}]\!]}, \mathrm{BOARD}\rangle \\ & [\![\textit{board}]\!] = [\![\texttt{entity}]\!] \xrightarrow{board} \Omega \end{aligned}$$

Since each noun like *board* is yielded by a structure like $[_{\mathrm{N}}\ n\ \surd\mathrm{BOARD}\ ]$, there is a bijection between morphisms $f\colon [\![\texttt{entity}]\!] \to \Omega$ and pairs $\langle id_{[\![\texttt{entity}]\!]}, \mathrm{R}\rangle$. But since each f is a characteristic function, and each idiosyncratic sort of entity is

[3] Here I identify $[\![\texttt{entity}]\!]$ with $id_{[\![\texttt{entity}]\!]}$. This is a convenient change of perspective that is common in the literature, including the category theory canon Mac Lane (1998:8).

a subsort of entity, we further get the following pullback for each viable root $\sqrt{\text{R}}$, with the morphism *true* being "pulled back" by the noun yielded by $[_{\text{N}}\ n\ \sqrt{\text{R}}\ ]$.

$$\begin{array}{ccc} \langle id_{[\![\texttt{entity}]\!]}, \text{R}\rangle & \overset{r}{\rightarrowtail} & [\![\texttt{entity}]\!] \\ {\scriptstyle !}\downarrow & & \downarrow{\scriptstyle \chi_{\langle id_{[\![\texttt{entity}]\!]}, \text{R}\rangle}} \\ 1 & \xrightarrow{\quad true \quad} & \Omega \end{array}$$

The same modeling perfectly extends to generalized root syntax, as in (12).

(12) $[\![\sqrt{\text{KHÔNG}}]\!] = 1 \xrightarrow{\text{KHÔNG}} [\![\texttt{r}]\!], [\![\text{Neg}]\!] = [\![\texttt{bool} \to \texttt{bool}]\!]$
$[\![[_{\text{Neg}}\ \text{Neg}\ \sqrt{\text{KHÔNG}}]]\!] = [\![\text{Neg}]\!] \times [\![\sqrt{\text{KHÔNG}}]\!] = \langle id_{[\![\texttt{bool}\to\texttt{bool}]\!]}, \text{KHÔNG}\rangle$
$[\![\textit{không}]\!] = [\![\texttt{bool} \to \texttt{bool}]\!] \xrightarrow{\textit{không}} \Omega$

But here we must understand the morphism named *không* in a different way—not as a classification of individuals as in (11) but as one of negation forces. Accordingly, r in the following pullback says that the root-tagged version of the type $\texttt{bool} \to \texttt{bool}$ is a subtype of its non-root-tagged version.

$$\begin{array}{ccc} \langle id_{[\![\texttt{bool}\to\texttt{bool}]\!]}, \text{R}\rangle & \overset{r}{\rightarrowtail} & [\![\texttt{bool} \to \texttt{bool}]\!] \\ {\scriptstyle !}\downarrow & & \downarrow{\scriptstyle \chi_{\langle id_{[\![\texttt{bool}\to\texttt{bool}]\!]}, \text{R}\rangle}} \\ 1 & \xrightarrow{\quad true \quad} & \Omega \end{array}$$

P4 is clearly a solution to all of C1–4. It inherits the null denotation solution from P3 for C1, redefines idiosyncratic sortal predicates as subobjects of built-in sorts for C2, reassembles roots and categorizers via products for C3, and uses this method for both content and semifunctional words for C4.

5 Monad

Next I present P5 in more detail. It is a direct application of the categorical tool developed in Asudeh and Giorgolo (2020). The relevant phenomenon from Asudeh and Giorgolo's work is conventional implicature—namely, the conventional, non-truth-conditional meanings of certain words. See (13) for example.

(13) a. Donald is a **Yank**.
b. This **cur** bit me. (Asudeh and Giorgolo 2020:13)

In (13), *Yank* and *cur* basically mean "American" and "dog," but they also both carry a negative attitude from the speaker. This is highly similar to the situation with semifunctional items that we have seen in Sec. 1. In fact, we can view semifunctionality as a manifestation of conventional implicature in the functional domain, whereas Asudeh and Giorgolo's slur word example is a manifestation of it in the lexical domain. To understand Asudeh and Giorgolo's theory, we need a few more background notions, including natural transformation, monad, and monoid. Due to space limitations, I only define monad here.

Definition 4. A *monad* on a category $\mathcal{C}$ is a triple $\langle T, \eta, \mu \rangle$ consisting of an endofunctor T and two natural transformations $\eta\colon 1_{\mathcal{C}} \to T$ (called *unit*) and $\mu\colon T^2 \to T$ (called *multiplication*), which satisfy the associativity and unit laws (details omitted). We also define an operator $\gg\!=$,[4] called *bind*, via μ. Given two objects A, B and a morphism $f\colon A \to TB$, we can feed TA to f by

$$T(A) \gg\!= f \coloneqq \mu_B(Tf(TA)), \text{ diagrammatically}$$

$$TA \xrightarrow{Tf} T^2B \xrightarrow{\mu_B} TB \xleftarrow{f} A$$

We can examine Asudeh and Giorgolo's categorical semantics now. They do not go all the way to a categorical model but mostly keep their discussion at the logical form level, for which they build on Moggi's (1989) computational λ-calculus (which has monads), Dalrymple et al's (1993) glue semantics (to relate the Moggian calculus to natural language syntax), and Moortgat's (1997) version of categorial grammar (to represent natural language syntax). I assume none of these in my monadic modeling of root syntax and solely build on Chomskyan syntax and conventional formal semantics. I keep my discussion at the logical form level too and leave a full-fledged categorical model to future work.

The particular monad Asudeh and Giorgolo use to model conventional implicature is the writer monad from functional programming. It is defined as follows. I have adjusted their notation to match my style.

Definition 5. The writer monad for conventional implicature is a monad $\langle T, \eta, \gg\!= \rangle$ in a type-logical category $\mathcal{C}$ where the objects are types. It makes use of the power set $\mathcal{P}(\mathbf{P})$ of the set $\mathbf{P}$ of all propositions, which is a monoid under set union.

$$\begin{aligned}
T &\coloneqq - \times \mathcal{P}(\mathbf{P}), \text{ where } - \text{ is a placeholder for } \mathcal{C}\text{-objects} \\
TA &\coloneqq \langle A, \{p\} \rangle, \text{ where } A \text{ is a } \mathcal{C}\text{-object and } p \in \mathbf{P} \\
Tf &\coloneqq \lambda \langle x, Q \rangle . \langle fx, Q \rangle, \text{ where } f \text{ is a } \mathcal{C}\text{-morphism and } Q \in \mathcal{P}(\mathbf{P}) \\
\eta_A x &\coloneqq \langle x, \emptyset \rangle, \text{ where } x \text{ is an element of type } A \\
\mu_A(\langle \langle x, P \rangle, Q \rangle) &\coloneqq \langle x, P \cup Q \rangle, \text{ where } P, Q \in \mathcal{P}(\mathbf{P}) \\
TA \gg\!= f &\coloneqq \mu_B(Tf(TA)), \text{ where } f\colon A \to TB \\
\texttt{write}(p) &\coloneqq \langle 1, \{p\} \rangle, \text{ where 1 is of terminal type and } p \in \mathbf{P}
\end{aligned}$$

Intuitively, this monad recursively writes extra information (e.g., "The speaker does not like Americans.") into a "log" slot of the computation (i.e., the second component of a monadic type TA). In particular, η wraps an ordinary type in a trivial monadic type, with no real extra information, while $\gg\!=$ feeds a monadic type as input to a suitable function. Via η and $\gg\!=$, logical computation proceeds as usual, while the extra information piles up in the log slot via $\cup$. In addition to the usual ingredients of a monad, Asudeh and Giorgolo also define an auxiliary function `write`, which directly writes some extra information into the log slot of a dummy monadic object, whose first component is eventually discarded due to the special nature of the function $\texttt{write}(p)$ is typically chained with.

[4] This is the usual symbol for bind. Asudeh and Giorgolo use a different symbol $\star$.

I use Asudeh and Giorgolo's (2020:55ff.) analysis of *Donald is a Yank* to show how the monad works. They let *Yank* denote $\texttt{write}(neg(^*ame))\gg\!=\!\lambda y.\,\eta(ame)$, where $neg(^*ame)$ is a technical way of saying "The speaker has a negative attitude toward Americans." Since *Yank* is the only monadic-type word in this example, to write its conventional implicature into the log slot we need to feed $[\![\mathit{Yank}]\!]$ to some function via $\gg\!=$. Asudeh and Giorgolo give the sentence the logical form $[\![\mathit{Yank}]\!]\gg\!=\!\lambda x.\,\eta([\![a]\!]([\![is]\!](x))([\![\mathit{Donald}]\!]))$, where the function to the right of $\gg\!=$ trivially "monadicizes" $[\![a]\!]([\![is]\!](x))([\![\mathit{Donald}]\!])$. The purpose of this type-lifting is to create an empty log slot to hold the implicature from *Yank*. Even though the authors do not specify this, due to the definition of $\gg\!=$ the variable x has to be of type $e \to t$ (i.e., it should type-match with the truth-conditional part of $[\![\mathit{Yank}]\!]$). The rest of the example is just routine calculation, which ultimately yields the pair $\langle ame(don), \{neg(^*ame)\}\rangle$. This says exactly that Donald is an American and that the speaker has a negative attitude toward Americans.

We can directly apply the writer monad to root syntax, with only a small modification. In our case, it is not enough to just let the root information pile up, but we need to record which root tags which functional head too.

Definition 6. The writer monad for root syntax is $\langle T, \eta, \gg\!=\rangle$. It uses the power set $\mathcal{P}(\mathbf{H}\times\mathbf{R})$ of the cartesian product of the set $\mathbf{H}$ of all functional heads and the set $\mathbf{R}$ of all roots in a language. The power set is a monoid under set union.

$$
\begin{aligned}
T &\coloneqq - \times \mathcal{P}(\mathbf{H}\times\mathbf{R}), \text{ where } - \text{ is a placeholder for } \mathcal{C}\text{-objects}\\
TA &\coloneqq \langle A, \{\mathrm{X}_{\surd}\rangle\}, \text{ where } A \text{ is a } \mathcal{C}\text{-object and } \mathrm{X}_{\surd} \in \mathbf{H}\times\mathbf{R}\\
Tf &\coloneqq \lambda\langle x, Q\rangle.\,\langle fx, Q\rangle, \text{ where } f \text{ is a } \mathcal{C}\text{-morphism and } Q \in \mathcal{P}(\mathbf{H}\times\mathbf{R})\\
\eta_A x &\coloneqq \langle x, \emptyset\rangle, \text{ where } x \text{ is an element of type } A\\
\mu_A(\langle\langle x, P\rangle, Q\rangle) &\coloneqq \langle x, P\cup Q\rangle, \text{ where } P, Q \in \mathcal{P}(\mathbf{H}\times\mathbf{R})\\
TA\gg\!=\!f &\coloneqq \mu_B(Tf(TA)), \text{ where } f\colon A \to TB\\
\texttt{write}(\mathrm{X}_{\surd}) &\coloneqq \langle 1, \{\mathrm{X}_{\surd}\}\rangle, \text{ where 1 is of terminal type and } \mathrm{X}_{\surd} \in \mathbf{H}\times\mathbf{R}
\end{aligned}
$$

I assign $[_{\mathrm{X}}\ \mathrm{X}\ \surd\ ]$ the logical form $\texttt{write}(\mathrm{X}_{\surd})\gg\!=\!\lambda y.\eta([\![\mathrm{X}]\!])$, which means that the extra information "X is tagged by $\surd$" is written into the log slot of a vacuous wrapper lifted from $[\![\mathrm{X}]\!]$. In $[_{\mathrm{X}}\ \mathrm{X}\ \surd\ ]$, X keeps its own denotation, while the root does not denote anything, for the logical function to the right of $\gg\!=$ does not reference it. And since the $\mathrm{X}_{\surd}$ in the argument position of `write` is nonlogical, it is not involved in logical computation either but merely gets registered in the log. Below I show how the monad works with four examples: a noun, a semifunctional item, an ordinary phrase, and a semifunctional-item-containing phrase.

(14) $[\![[_{\mathrm{N}}\ n\ \surd\textsc{board}\ ]]\!] = \texttt{write}(n_{\surd\textsc{board}})\gg\!=\!\lambda y.\,\eta[\![n]\!] = \langle[\![n]\!], \{n_{\surd\textsc{board}}\}\rangle$
(an entity that is idiosyncratically characterized by $\surd$BOARD)

(15) $[\![[_{\mathrm{Neg}}\ \mathrm{Neg}\ \surd\textsc{déo}\ ]]\!] = \texttt{write}(\mathrm{Neg}_{\surd\textsc{déo}})\gg\!=\!\lambda y.\,\eta[\![\mathrm{Neg}]\!] = \langle[\![\mathrm{Neg}]\!], \{\mathrm{Neg}_{\surd\textsc{déo}}\}\rangle$
(a negator that is idiosyncratically characterized by $\surd$DÉO)

(16) *Mary walks.* (to save space I only consider the syntax up to VoiceP)

$[\![[_{\text{VoiceP}}\ \text{Mary}\ [_{\text{VoiceP}}\ \text{Voice}\ [_{\text{V}}\ v\ \sqrt{\text{WALK}}]]]]\!]$ (based on the syntax in Bowers 2010)

$$= ([\![\text{V}]\!] \gg\!= \underline{\lambda y.\, \eta([\![\text{Voice}]\!]y)}) \gg\!= \lambda z.\, \eta(z([\![\text{Mary}]\!]))$$

$$= ((\texttt{write}(v_{\sqrt{\text{WALK}}}) \gg\!= \lambda x.\, \eta[\![v]\!]) \gg\!= \lambda y.\, \eta([\![\text{Voice}]\!]y)) \gg\!= \lambda z.\, \eta(z([\![\text{Mary}]\!]))$$

$$= (\langle [\![v]\!], \{v_{\sqrt{\text{WALK}}}\}\rangle \gg\!= \lambda y.\, \eta([\![\text{Voice}]\!]y)) \gg\!= \lambda z.\, \eta(z([\![\text{Mary}]\!]))$$

$$= \langle [\![\text{Voice}]\!]([\![v]\!]), \{v_{\sqrt{\text{WALK}}}\}\rangle \gg\!= \lambda z.\, \eta(z([\![\text{Mary}]\!])) = \langle ([\![\text{Voice}]\!]([\![v]\!]))[\![\text{Mary}]\!], \{v_{\sqrt{\text{WALK}}}\}\rangle$$

$$= \langle \lambda x\, [\texttt{eventuality}(x) \wedge Ag(x, mary)], \{v_{\sqrt{\text{WALK}}}\}\rangle$$

(an event with agent Mary; it is idiosyncratically characterized by $\sqrt{\text{WALK}}$)

(17) *wǔ duǒ huā* 'five $\text{CLF}_{\text{flowerlike}}$ flower; five flowers' [Chinese]

$[\![[_{\text{NumP}}\ \text{Num}_5\ [_{\text{ClP}}\ [_{\text{Cl}_{\sqrt{}}}\ \text{Cl}\ \sqrt{\text{DUO}}\ ]\ [_{\text{N}}\ n\ \sqrt{\text{HUA}}]]]]\!]$ (based on the syntax in Li 2013)

$$= ([\![\text{N}]\!] \gg\!= \underline{\lambda z.\, \eta\langle(\pi_1[\![\text{Cl}_{\sqrt{}}]\!])z, \pi_2[\![\text{Cl}_{\sqrt{}}]\!]\rangle}) \gg\!= \lambda w.\, \eta([\![\text{Num}_5]\!]w)$$

$$= (\langle [\![n]\!], \{n_{\sqrt{\text{HUA}}}\}\rangle \gg\!= \lambda z.\, \eta\langle [\![\text{Cl}]\!]z, \{\text{Cl}_{\sqrt{\text{DUO}}}\}\rangle) \gg\!= \lambda w.\, \eta([\![\text{Num}_5]\!]w)$$

$$= (\langle [\![\text{Cl}]\!]([\![n]\!]), \{\{\text{Cl}_{\sqrt{\text{DUO}}}\}, \{n_{\sqrt{\text{HUA}}}\}\}\rangle) \gg\!= \lambda w.\, \eta([\![\text{Num}_5]\!]w)$$

$$= \langle [\![\text{Num}_5]\!]([\![\text{Cl}]\!]([\![n]\!])), \{\{\text{Cl}_{\sqrt{\text{DUO}}}\}, \{n_{\sqrt{\text{HUA}}}\}\}\rangle$$

$$= \langle \lambda y.\, [\text{Pl}(\texttt{entity})](y) \wedge |y| = 5, \{\{\text{Cl}_{\sqrt{\text{DUO}}}\}, \{n_{\sqrt{\text{HUA}}}\}\}\rangle$$

(a plural entity with a cardinality of 5; it is idiosyncratically characterized by $\sqrt{\text{HUA}}$; its atomic unit is idiosyncratically characterized by $\sqrt{\text{DUO}}$)

Notice a key difference between (17) and (16). The lexical word is merged with a purely functional head Voice in (16) but with a semifunctional head $\text{Cl}_{\sqrt{}}$ in (17). The latter requires a more complicated $\gg\!=$ pattern than the former (compare the underscored parts). In general, the pattern in (17) is needed whenever an already root-tagged structure merges with another root-tagged head.

P5 is clearly a solution to C1, C3, and C4. For C1 it inherits the solution from P3, and for C3–4 it consistently uses the operation $\gg\!=$. Also, the log slot in P5 makes it possible for root effects to percolate upward in the course of meaning computation. P5 may well be able to resolve C2 as well, though the details will only become clear when we have a full-fledged categorical model in hand.

6 Conclusion

In this paper, I discussed four challenges of root syntax for formal semantics: (C1) How to logically represent roots? (C2) How to redefine idiosyncratic sorts in a root-based model? (C3) How to compose roots and categorizers? and (C4) How to semantically model purely lexical and semifunctional items in a unified way? To address these challenges, I presented five possible directions: (P1) the conjunctivist approach, (P2) the type variable approach, (P3) the null denotation approach, (P4) the categorical logic approach, and (P5) the monadic approach. Among the five directions, P1–2 are undesirable as they are confined to classical root syntax. P3 is conceptually more appealing, and its two category-theoretic improvements P4–5 are promising both conceptually and technically. A future research direction is to see if P4–5 can be combined in the same model.

Bibliography

Acquaviva P (2009) Roots and lexicality in distributed morphology. In: Galani A, Redinger D, Yeo N (eds) YPL2 Special Issue: Fifth York-Essex Morphology Meeting, Department of Language and Linguistic Science, University of York

Acquaviva P (2014) Distributing roots: Listemes across components in distributed morphology. Theoretical Linguistics 40(3/4):277–286

Acquaviva P (2019) Categorization as noun construction: Gender, number, and entity types. In: Mathieu É, Dali M, Zareikar G (eds) Gender and Noun Classification, Oxford University Press, Oxford, pp 41–63

Alexiadou A, Borer H, Schäffer F (eds) (2014) The roots of syntax and the syntax of roots. Oxford University Press, Oxford

Asher N (2011) Lexical meaning in context: A web of words. Cambridge University Press, Cambridge

Asudeh A, Giorgolo G (2020) Enriched Meanings: Natural Language Semantics with Category Theory. Oxford University Press, Oxford

Awodey S (2010) Category Theory, 2nd edn. Oxford University Press, Oxford

Bowers J (2010) Arguments as Relations. MIT Press, Cambridge, MA

Cardinaletti A, Giusti G (2001) Semi-lexical motion verbs in romance and germanic. In: Corver N, van Riemsdijk H (eds) Semi-lexical categories: On the function of content words and content of function words, Mouton de Gruyter, Berlin, pp 371–414

Crole R (1993) Categories for Types. Cambridge University Press, Cambridge

Dalrymple M, Lamping J, Saraswat V (1993) LFG semantics via constraints. In: Krauwer S, Moortgat M, des Tombe L (eds) Proceedings of the Sixth Conference of the European ACL, Association for Computational Linguistics, Utrecht, pp 97–105

Doron E (ed) (2014) Theoretical Linguistics 40(3/4): On the identity of roots. Walter de Gruyter GmbH, Berlin

Goldblatt R (2006) Topoi: The Categorial Analysis of Logic, revised edn. Dover, Mineola

Halle M, Marantz A (1993) Distributed morphology and the pieces of inflection. In: Hale K, Keyser SJ (eds) Essays in Linguistics in Honor of Sylvain Bromberger, no. 20 in The View from Building, MIT Press, Cambridge MA, pp 111–176

Landman F (2000) Events and plurality: The Jerusalem lectures. Springer, Dortrecht

Li X (2013) Numeral classifiers in Chinese. De Gruyter Mouton, Berlin

Mac Lane S (1998) Categories for the Working Mathematician, 2nd edn. Springer, New York

Marantz A (1995) *Cat* as a phrasal idiom: Consequences of late insertion in distributed morphology, manuscript, Massachusetts Institute of Technology

Moggi E (1989) Computational lambda-calculus and monads. In: Proceedings of the Fourth Annual Symposium on Logic in Computer Science, IEEE Press, New York, pp 14–23

Moortgat M (1997) Categorial type logics. In: van Benthem J, ter Meulen A (eds) Handbook of Logic and Language, 1st edn, North-Holland, Amsterdam, pp 93–177

Pots C (2020) Roots in progress: Semi-lexicality in the Dutch and Afrikaans verbal domain. PhD thesis, KU Leuven

Potts C (2005) The Logic of Conventional Implicatures. Oxford University Press, Oxford

Potts C (2007) The expressive dimension. Theoretical Linguistics 33(2):165–197

Ramchand G (2008) Verb Meaning and the Lexicon: A First Phase Syntax. Cambridge University Press, Cambridge

Song C (2019) On the formal flexibility of syntactic categories. PhD thesis, University of Cambridge