On the semantics of root syntax: challenges and directions

1. Introduction Since Marantz's seminal work in the 1990s (e.g., Marantz 1997), the notion *root* has been widely adopted in generative syntax. A typical root-containing structure is seen in lexical word formation, as in $[_{\mathrm{N}}\ n\ \sqrt{\mathrm{DOG}}]$ and $[_{\mathrm{V}}\ v\ \sqrt{\mathrm{EAT}}]$. More recently, there have been proposals extending root syntax to the functional domain (e.g., Acedo-Matellán & Real-Puigdollers 2019, Song 2019, Pots 2020). These proposals are motivated by so-called semilexical or semifunctional items, which are crosslinguiatically observed and especially common in analytical languages. See (1) for some Chinese examples.

(1) a. Numeral classifiers: *duǒ* 'lit. blossom; for flowers, clouds, etc.', *tóu* 'lit. head; for big animals, garlic, etc.' (default: *gè* 'lit. a single bamboo')

b. Nominal conjunctions: *yǔ* 'lit. give, bestow; formal, literary', *gēn* 'lit. follow; colloquial' (default: *hé* 'lit. speak or sing in concert with')

These items are semifunctional in that they carry nontrivial lexical idiosyncrasies beyond their grammatical functions, which give rise to lexical or pragmatic constraints on their usage. In a root-based analysis, they all have a $[_{\mathrm{X}}\ \mathrm{X}\ \sqrt{}]$ structure (X is a functional category). Song (2019) calls the extended root theory *generalized root syntax*.

2. Issue Despite their wide adoption in syntax, roots have received little attention in semantics. Mainstream semantic studies do not decompose bare words. Thus, the bare noun *dog* is usually given the logical form $\lambda x.\mathrm{dog}(x)$ and modeled by a named set of entities. This convention is retained in frameworks that otherwise do use lexical decomposition, as illustrated by the representations in (2), which are respecticaly from neo-Davidsonian event semantics and Ramchand's first-phase syntax. Notice that both butter(e) and push(e) integrate an idiosyncratic predicate label and an event variable.

(2) a. $\exists e[\mathrm{butter}(e) \wedge \mathrm{agent}(e) = \mathrm{johns} \wedge \mathrm{theme}(e) = \mathrm{toast}\ldots]$ (Landman 2000: 12)

b. $\ldots \lambda e[\ldots \mathrm{push}(e) \ldots]$ (Ramchand 2008: 61)

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 in formal semantics. Against this backdrop, in this paper I discuss the challenges that (generalized) root syntax poses for formal semantics and present some possible directions toward a compositional semantics that can handle roots.

3. Challenges Root syntax poses five fundamental challenges for formal semantics.

C1. With the syntactic decomposition of root-containing items, a first question is how the structure $[_{\mathrm{X}}\ \mathrm{X}\ \sqrt{}]$ is semantically represented. What is its logical form?

C2. Root meanings are extremely vague—far vaguer than the labels used in conventional logical forms like $\mathrm{dog}(x)$ and $\mathrm{butter}(e)$. In Acquaviva's (2009: 4) words, the meaning of a root is "too elusive to be pinned down," for "something so radically underspecified cannot even convey the distinction between argument and predicate." In view of this apparently anti-type-theoretic nature, how can roots be incorporated into a type-based semantic theory at all?

C3. Given the extreme vagueness of roots, in a commensurate semantic model there is no way to define characteristic sets like $\{x\colon x \text{ is a dog}\}$ or $\{e\colon e \text{ is an eating event}\}$. How to reconstruct these vital components of formal semantics in the new model?

C4. Root syntax separates functional and lexical information out of formal-derivational (i.e., proof-theoretic) motivation, as this decompositional strategy helps syntacticians achieve rational grammatical analysis down to the subatomic level. Neverthe-

less, the semantic module must be able to assemble the syntactically decomposed components back together in the final interpretation of a sentence. While this integration issue does not arise in current formal semantics (where bare lexical items are intact), how it happens is a nonnegligible question in the face of root syntax.

C5. The conventional treatment of bare lexical words as predicate labels can hardly be extended to semifunctional items, which are not first-order predicates and may have very complex types. Suppose generalized root syntax in on the right track, how can lexical and semifunctional items get a similarly unified treatment in semantics?

4. Directions With the above challenges in mind, next I present some possible directions toward a model-theoretic semantics with root-syntax-handling capacity. I will first go through all those directions and filter out some unpromising ones (this section) and then provide more details on two directions that I deem more promising (§5–6). To begin with, Song (2019: 59–66) discusses three possibilities of root denotation.

P1. Roots denote sort-generic predicates, categorizers denote sorting predicates, and the two are composed by conjunction. This is also the approach in Kelly (2013). E.g.,
$[\![\sqrt{\text{BOARD}}]\!] = \lambda x\colon u.\, \text{BOARD}(x), [\![n]\!] = \lambda x\colon u.\, \text{entity}(x)$
$[\![[_{\text{N}}\ n\ \sqrt{\text{BOARD}}]]\!] = [\![n]\!] \wedge [\![\text{BOARD}]\!] = \lambda x\colon u.\, \text{entity}(x) \wedge \text{BOARD}(x)$
(where u is an individual-type variable of the most generic sort)

P2. Roots denote type-open predicates, categorizers denote types, and the two are composed by type-level application (in second-order typed lambda calculus). E.g.,
$[\![\sqrt{\text{BOARD}}]\!] = \lambda\alpha\colon *\,.\, \lambda x\colon \alpha.\, \text{BOARD}(x), [\![n]\!] = \text{entity}\colon *$
$[\![[_{\text{N}}\ n\ \sqrt{\text{BOARD}}]]\!] = [\![\sqrt{\text{BOARD}}]\!]([\![n]\!]) = \lambda x\colon \text{entity}.\, \text{BOARD}(x)$
(where α is a type variable, x is term variable, and $*$ is the type of all types)

P3. Roots lack model-theoretic denotations, and categorizers denote sorting predicates. There is no composition between the two. E.g.,
$[\![\sqrt{\text{BOARD}}]\!]$ is undefined, $[\![n]\!] = \lambda x\colon u.\, \text{entity}(x)$
$[\![[_{\text{N}}\ n\ \sqrt{\text{BOARD}}]]\!] = [\![n]\!] = \lambda x\colon u.\, \text{entity}(x)$
(where the root does not participate in compositional semantics)

P1–2 are not desirable (at least not w/o significant modification) as they can't be extended to semifunctional items, whose X in the $[_{\text{X}}\ \text{X}\ \sqrt{}]$ schema is not a first-order predicate. For this reason, Song (2019) prefers P3, which is in line with Chomsky's view that lexical items do not refer (Acquaviva 2014 has a similar view on roots). I have two additional remarks on the three possibilities. First, between P1 and P2, P1 is even less desirable, as the conjunctivist approach strictly requires type match. In P2, by comparison, since α ranges over all ordinary types, the composition in principle also works for semifunctional items, though it is unclear what the function label BOARD would mean in that case. In fact, it is hard to decipher the real contribution of the root in both P1 and P2 due to C2–3. Second, while P3 avoids the vague label problem and can easily apply to semifunctional items as well, it leave C3–4 unaddressed. Thus, it can certainly be further improved. The next two possible directions can be viewed as improvements of P3.

P4. C3 suggests a higher-level abstraction in semantics, where the individual domain does not inherently distinguish idiosyncratic sorts like dog and board. It could presumably still distinguish broad-brush sorts like entity and eventuality, though, which may simply be a universal part of our grammatical ontology. Since there is no way to further divide these general sorts, they can be treated like monolithic objects in the model. This is reminiscent of the definition of *objects* in mathematical category theory. In fact, it is common practice in categorical models of logic to

interpret types as objects (see, e.g., Crole 1993). So, it might be interesting to lift the conventional set-theoretic model for natural language semantics to a category-theoretic one and see what does to root syntax.

P5. The compositional vs. idiosyncratic division in C4 is reminiscent of the two-dimensional view of semantic content in Potts (2005, 2007) and Asudeh & Giorgolo (2020), which consists of an *at-issue* dimension (truth-conditional) and a *side-issue* dimension (non-truth-conditional). Root-based meanings are quite similar in nature to Asudeh & Giorgolo's side-issue meanings (e.g., conventional implicature, perspectival inference). Asudeh & Giorgolo tackle side-issue phenomena with the category-theoretic tool of *monad*. It might be interesting to try and see if the same tool can be applied to modeling root syntax.

Incidentally, P4–5 bring to the fore the potential usefulness of category theory in linguistics. That is not surprising, for there is a long tradition of categorial grammar in linguistics with many recent developments (mainly based on Lambek's seminal work in the 1980s). That said, in this paper I will largely stay away from the tools from that area (e.g., tensor products, left/right duals), for the syntax I work with (i.e., minimalism) has a fundamentally different design (and purpose), which those tools simply do not fit.

5. A categorical model In this section I present P4 in more detail. My goal is to lift the usual set-theoretic model for natural language semantics to a category-theoretic one. This is in fact straightforward. 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. As mentioned above, this model already exists and is fairly well established, so we can use that as a point of departure.

The model (Crole 1993, Pitts 2018) The model $\mathcal{M} \coloneqq \langle \mathbb{C}, \mathbf{I} \rangle$ is defined by a cartesian closed category $\mathbb{C}$ and an interpretation function $\mathbf{I}$. We can simply take $\mathbb{C}$ to be **Set**, which is not only cartesian closed but furthermore a topos (Lambek 1988 similarly used a topos). The model can be extended to accommodate intensionality, but here I abstract away from that complication and focus on extensional semantics. $\mathbb{C}$-objects correspond to types in λ-calculus, and $\mathbb{C}$-morphisms correspond to terms. Being cartesian closed, $\mathbb{C}$ has a terminal object $\mathbf{1}$, finite products $X \times Y$, and exponentials X^Y. Morphisms of the shape $\mathbf{1} \to X$ pick out X-elements, so $\mathbf{1} \to X \times Y$ picks out an ordered pair $\langle x, y \rangle$, which can be suitably taken to be a y-tagged x.

Root syntax As revealed by the undesirability of P1–2 and the desirability of P3, neither conjunction nor functional application serves as an appropriate composition rule for the root categorization structure $[_{\mathrm{X}}\ \mathrm{X}\ \sqrt{}]$. Under the above basic setting, we can interpret it by the product $X \times \sqrt{}$, which is equivalent to a root-tagged type $X_{\sqrt{}}$ (Song 2019 also uses this notation in syntax and calls it a root-supported head). In this way, roots both get a place in compositional semantics and need not really participate in logical computation (but can just tag along). This in turn requires us to reserve a $\mathbb{C}$-object $\sqrt{}$ for the root type (in **Set** this is just the set of all roots in the grammar being modeled). The characteristic sets mentioned in C3 (e.g., $\{x\colon x \text{ is a dog}\}$) can be reinterpreted as morphisms of the shape $T \to \Omega$, where T ranges over the $\mathbb{C}$-objects for the generic sorts entity, eventuality, etc. and Ω is the usual truth value object in a topos. Since there can be multiple such morphisms, we can view each of them as a conventional sortal predicate.

As we can see, P4 is a simple solution for C1, C2, C3, and C5. As for C4, we would have to assume that the semantic module involves not only a compositional submodule

but also an integration submodule, which serves to eventually integrate compositional and noncompositional information in a suitable manner. Although P4 does not directly address how or where this integration takes place, it does prepares the ground for it by keeping the root-tagged type and the tagging root "separately together" (as a pair).

6. Monad In this section I present P5 in more detail. Having no space to introduce the background category theory, I can only roughly show how the writer monad in Asudeh & Giorgolo (2020) also works for root syntax (see paper version for full detail). In P6 the root does participate in computation, so P6 might be better-equipped to address C4, though as I argue in the paper, its ambient model setting is not as clear as that of P5, nor is its interpretation rule for root categorization as simple.

Writer monad Asudeh & Giorgolo (2020: 51) define a monad $\langle \Diamond, \eta, \star \rangle$, where $\Diamond$ is the monad functor, η is the unit natural transformation, and $\star$ is the *bind* operation borrowed from functional programming (defined by another natural transformation μ). Simply put, this monad recursively writes extra (e.g., noncompositional) information into a "log" slot (a set) of the computation, hence its name. In particular, η wraps an ordinary type in a trivial monadic type (with no real extra information), while $\star$ feeds a monadic type as input into a function (thus chaining "programs"). Via η and $\star$, logical computation proceeds as usual, while the extra, nonlogical information piles up in the dedicated slot (via set union).

Root syntax I directly adopt Asudeh & Giorgolo's theory, except that in our case it is not enough to just let the roots pile up, but we need to record which root tags which X as well. As in P5, I use an ordered pair for this purpose. In this monadic setting, $[\![[_{\mathrm{X}}\ \mathrm{X}\ \surd]]\!] = \texttt{write}(\langle [\![\mathrm{X}]\!], \surd \rangle) \star \lambda y.\eta([\![\mathrm{X}]\!])$ (`write` is an ancillary function), which roughly means that the extra information "$[\![\mathrm{X}]\!]$ is tagged by $\surd$" is written into the log slot of a vacuous wrapper lifted from $[\![\mathrm{X}]\!]$. This $\star$-term can be obtained by letting the root denote a function that takes X as input (i.e., $\lambda [\![\mathrm{X}]\!].[\![\surd]\!]([\![\mathrm{X}]\!]) : \star$).

Examples I give one of A&G's examples plus two root syntax examples, one from neo-Davidsonian event semantics (based on the syntax in Bowers 2010) and the other from the grammatical domain of Chinese classifiers (based on the syntax in Li 2013).

- $[\![\text{Donald is a Yank}]\!] = \cdots = [\![\text{Yank}]\!] \star \lambda x.\eta(x(\text{don})) = \texttt{write}(neg(^*\text{Ame})) \star \lambda y.\eta(\text{Ame}) \star \lambda x.\eta(x(\text{don})) = \langle \text{Ame(don)}, \{\text{neg}(^*\text{Ame})\} \rangle$ (A&G 2020: 57)
  (Donald is an American; the speaker has a negative attitude toward Americans)
- $[\![\text{Mary walks}]\!] = [\![[_{\text{VoiceP}}\ \text{Mary}\ [_{\text{VoiceP}}\ \text{Voice}\ [_{\text{V}}\ v\ \sqrt{\text{WALK}}]]]]\!] = (([\![\sqrt{\text{WALK}}]\!]([\![v]\!])) \star \lambda y.\eta([\![\text{Ag}]\!]y)) \star \lambda z.\eta(z([\![\text{Mary}]\!])) = \cdots = \langle \lambda x[\text{Ev}(x) \wedge \text{Ag}(x, \text{mary})], \{\langle [\![v]\!], \text{WALK} \rangle\} \rangle$
  (an event whose Ag is M.; it is idiosyncratically characterized by $\sqrt{\text{WALK}}$)
- $[\![\textit{wǔ-duǒ-hūa}$ '5-blossom-flower'$]\!] = [\![[_{\text{NumP}}\ \text{Num}_5\ [_{\text{ClP}}\ [_{\text{Cl}_\surd}\ \text{Cl}\ \sqrt{\text{DUO}}\ ]\ [_{\text{N}}\ n\ \sqrt{\text{HUA}}]]]]\!]$
  $= (([\![\sqrt{\text{HUA}}]\!]([\![n]\!])) \star \lambda z.(([\![\sqrt{\text{DUO}}]\!]([\![\text{Cl}]\!]))z)) \star \lambda w.\eta([\![\text{Num}_5]\!]w) = \ldots$
  $= \langle \lambda y.[\text{Pl(Ent)}](y) \wedge |y| = 5, \{\langle [\![n]\!], \text{HUA} \rangle, \langle [\![\text{Cl}]\!], \sqrt{\text{DUO}} \rangle\} \rangle$
  (a pl. entity of card. 5; it is i.ch.-ed by $\sqrt{\text{HUA}}$, its atomic unit is i.ch.-ed by $\sqrt{\text{DUO}}$)

P6 is a more complex solution to C1, C2, and C5, and it also solves C4. Insofar as the above examples are concerned, C3 is solved via the log slot. But since a monad functor is a special endofunctor on a category, we can presumably give P6 a categorical model as well, in which case C3 could also have an "at-issue" solution. A future research direction is to see if P5–6 can be combined in the same model.

Selected references Acquaviva, P. (2009). Roots and lexicality in distributed morphology. *YPL2 special issue*. Asudeh, A. & G. Giorgolo (2020). *Enriched meanings*. Pitts, A. (2018). Category Theory lecture notes. UCam. Song, C. (2019). On the formal flexibility of syntactic categories. UCam dissertation.