On the semantics of root syntax: Challenges and directions

Chenchen (Julio) Song

School of International Studies
Zhejiang University

Logic and Engineering of Natural Language Semantics 18
Japan (online), Nov 13–15, 2021

Overview

Overview

Classical root syntax

Root syntax: a popular trend in current generative syntax

- Halle & Marantz (1993 et seq.): Distributed Morphology (DM)
- Borer (2005, 2013): Exoskeletal Syntax (XS)
- Chomsky (2019):

“syntax all the way down”

> *If you accept—as I am doing here—the Hagit Borer–Alec Marantz theory of root categorization, which I think is pretty strongly motivated, the roots in the lexicon are independent of category.*

Theory-neutral definition

A root is a purely lexical unit in syntactic derivation that is void of categorial information. It only acquires a syntactic category (and thereby a categorized interpretation) by externally merging with one.

Example (DM):

$$[_{\mathrm{N}}\ n\ \sqrt{\mathrm{DOG}}] \rightarrow \langle /\text{dɔg}/, \text{‘a domestic mammal…’} \rangle$$

$$\sqrt{\mathrm{DOG}} \rightarrow \langle -, - \rangle$$

Classical root syntax

Root syntax: a popular trend in current generative syntax

- Halle & Marantz (1993 et seq.): Distributed Morphology (DM)
- Borer (2005, 2013): Exoskeletal Syntax (XS)
- Chomsky (2019):

"syntax all the way down"

> *If you accept—as I am doing here—the Hagit Borer–Alec Marantz theory of root categorization, which I think is pretty strongly motivated, the roots in the lexicon are independent of category.*

Theory-neutral definition

A root is a purely lexical unit in syntactic derivation that is void of categorial information. It only acquires a syntactic category (and thereby a categorized interpretation) by externally merging with one.

Example (DM):

$$[_{N}\ n\ \sqrt{314}\,] \rightarrow \langle /\text{dɔg}/, \text{'a domestic mammal…'} \rangle$$

$$\sqrt{314} \rightarrow \langle -, - \rangle \text{ (Harley 2014)}$$

Generalized root syntax (GRS)

Core features:

- ultimate lexical decomposition
- complete separation of grammatical and idiosyncratic information
- super fine-grained, "subatomic" analysis

Limit: confined to the lexical domain (basically a morphological tool)

Generalized root syntax (Song 2019)

Extends root syntax into the grammatical domain and thereby makes roots (or root-oriented thinking) into a more general syntactic tool.

- Core motivation: lexical idiosyncrasy in the grammatical domain
- Acedo-Matellán & Real-Puigdollers (2019): different details, same idea

Semigrammaticality

GRS aims to give content and semigrammatical words a unified analysis.

Not all vocabulary items in human language are purely lexical or grammatical. There are also many in-betweens (see Song 2021 for a typology).

(1) a. *La* *pasta* ***va*** / ***viene*** *mangiata* *subito.* [Italian]
the pasta $\text{PASS}_{\text{obligatory}}$ $\text{PASS}_{\text{regular}}$ eaten immediately
"Pasta must be / is eaten immediately." (Cardinaletti & Giusti 2001:392)

b. *yī* ***wèi*** / ***míng*** *lǎoshī* [Mandarin]
one $\text{CLF}_{\text{respectful}}$ $\text{CLF}_{\text{professional}}$ teacher
"a teacher" (Song 2019:125)

c. *Em/Tao* ***không*** / ***đéo*** *cần* *anh/mày* *giúp.* [Vietnamese]
1SG.N/V $\text{NEG}_{\text{neutral}}$ $\text{NEG}_{\text{vulgar}}$ need 2SG.N/V help
"I don't need your help." (Li Nguyen, p.c.)

☞Hallmark: grammatical function + lexical coloration

logical-compositional ⇑ category

conventional-idiosyncratic ⇑ root

➡encyclopedic content, speaker attitude, register conditioning, etc.

Overview

Roots in formal semantics: Status quo

N/A

Roots in formal semantics: Status quo

Mainstream formal semantic studies do not decompose bare words.

- Some theories do pursue lexical decomposition.
  - e.g., neo-Davidsonian event semantics
- But they generally leave stems or morphological roots intact.

Example:

(2) Jones buttered the toast. (Landman 2000:1–2)

$\exists e[\underbrace{\mathrm{BUTTER}(e)}_{\text{bare verb meaning}} \wedge \mathrm{AGENT}(e) = \mathrm{j} \wedge \mathrm{THEME}(e) = \mathrm{t}]$

($\mathrm{BUTTER}: \text{event} \to \mathrm{t}$) ☜ already categorized

What about Frege's Principle?

Syntax :: $[_{\mathrm{V}}\ v\ \sqrt{\text{BUTTER}}\ ] \xrightarrow{\text{mapping}}$? :: Semantics

Challenges of root syntax for formal semantics

1. How to logically represent roots?
2. How to mirror the extreme vagueness of roots in the model?
3. How to compose roots and categories?
4. How to keep up with generalized root syntax?

Challenge 1 (C1)

How to logically represent roots?

Bare word meanings correspond to (at least) *categorized roots*.

- $[\![\text{dog}]\!] = \lambda x.\, \mathsf{dog}'(x)$, where x is an entity-typed variable
- $[\![\text{speak}]\!] = \lambda e.\, \mathsf{speak}'(e)$, where e is an event-typed variable

Root categorization schema: $[_X\ X\ \surd\ ]$ (X is a category, $\surd$ is a root)

- What do X and $\surd$ respectively denote?

[Root] meaning seems too elusive to be pinned down. This is because ... something so radically underspecified cannot even convey the distinction between argument and predicate. What meaning can a root have that is not yet specified as an entity-, state- or process-referring expression? (Acquaviva 2009:4)

Challenge 2 (C2)

How to mirror the extreme vagueness of roots in the model?

In model-theoretic semantics, bare predicates are modeled by named sets of individuals.

- $[\![\text{dog}]\!] = \{\, x \mid x \text{ is a dog} \,\}$
- $[\![\text{speak}]\!] = \{\, e \mid e \text{ is a speaking event} \,\}$

But in a model of root syntax, such named sets can no longer be taken for granted. They must be somehow reconstructed.

This is reminiscent of the anti-extensionalist view of lexical items:

> *[A]n extensionalist semantic approach, where basic terms of the semantic representation are ultimately defined by what they are true of … cannot possibly shed much light on those aspects of lexical semantic competence based on oppositions in conceptualization rather than in distinct extensions: consider again home vs. house, or broad vs. wide, or use vs. utilize, to say nothing about notorious problematic cases like time, air, or god. (Acquaviva 2014:281)*

Challenge 3 (C3)

How to compose roots and categories?

In other words, what is the logical relationship between the root node and the category node?

1. Function-argument? ⇔ head-complement
2. Coordination? ⇔ head-head
3. Modification? ⇔ head-adjunct

$$[_{X}\ X\ \surd\ ]$$

Both ① and ③ have syntactician followers, while ② is prima facie more natural semantically (e.g., Kelly 2013). It is syntactically less desirable, though, due to the built-in symmetry (and also for reasons like labeling; Chomsky 2013).

☞Desideratum: a neater mapping between syntax and semantics

Challenge 4 (C4)

How to keep up with generalized root syntax?

Suppose GRS is on the right track, whatever compositional semantics we assign content words must work for semigrammatical words too.

As we will see, this calls for a mode of composition that does not hinge on the logical type of the grammatical category X in $[_X \ X \ \sqrt{} \ ]$.

This poses an immediate problem for the coordination approach, as we ideally want the logical type of the root node to be stable.

☞ GRS requires us to think outside the "predicativist" box.

Overview

Directions toward a compositional root semantics

Goals:

- A category-neutral logical form template for all roots
- A type-error-free unification of content and semifunctional words

Possibilities (①–③ from Song 2019, ④–⑤ new):

1. The conjunctivist approach
2. The type variable approach
3. The null denotation approach
4. The categorical logic approach
5. The monadic approach

GRS rules out ①–② and favors ③, of which ④–⑤ are improved versions.

Possibility 1 (P1): The conjunctivist approach

Root:	sort-generic predicate	
Category:	sorting predicate	
Composition:	conjunction	(cf. Kelly 2013)

Example:

the entity-subset of $[\![\sqrt{\text{BOARD}}]\!]$

$\Downarrow$

N $\lambda x{:}\, u.\,\mathtt{entity}(x) \wedge \text{BOARD}(x) \equiv \lambda x{:}\, \mathtt{entity}.\,\text{BOARD}(x)$

n $\sqrt{\text{BOARD}}$

$\lambda x{:}\, u.\,\mathtt{entity}(x)$

$\Uparrow$

set of all entities

$\lambda x{:}\, u.\,\text{BOARD}(x)$

$\Uparrow$

a sort-mixed set defined by $\sqrt{\text{BOARD}}$

u: entity, event, …

Possibility 1 (P1): The conjunctivist approach

Root:	sort-generic predicate
Category:	sorting predicate
Composition:	conjunction

(cf. Kelly 2013)

Example:

the event-subset of $[\![\sqrt{\text{BOARD}}]\!]$

$\Downarrow$

V $\lambda x : u.\, event(x) \wedge \text{BOARD}(x) \equiv \lambda x : event.\, \text{BOARD}(x)$

v — $\lambda x : u.\, event(x)$

$\Uparrow$

set of all events

$\sqrt{\text{BOARD}}$ — $\lambda x : u.\, \text{BOARD}(x)$

$\Uparrow$

a sort-mixed set defined by $\sqrt{\text{BOARD}}$

u: $entity$, $event$, $\ldots$

Possibility 1 (P1): The conjunctivist approach

Root:	sort-generic predicate	
Category:	sorting predicate	
Composition:	conjunction	(cf. Kelly 2013)

Example:

the event-subset of $[\![\sqrt{\text{BOARD}}]\!]$

$\Downarrow$

V $\lambda x : u.\, event(x) \wedge \text{BOARD}(x) \equiv \lambda x : event.\, \text{BOARD}(x)$

v — $\lambda x : u.\, event(x)$

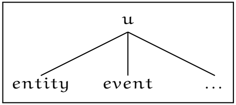
$\Uparrow$

set of all events

$\sqrt{\text{BOARD}}$ — $\lambda x : u.\, \text{BOARD}(x)$

$\Uparrow$

a sort-mixed set defined by $\sqrt{\text{BOARD}}$



☞Requirement: X-√ type match (first-order predicate)

⇒Tailor-made for classical root syntax

Possibility 2 (P2): The type variable approach

Root:	type-open predicate	
Category:	type	
Composition:	type-level application	(Song 2019)

Example:

N $\lambda x\colon \mathtt{entity}.\, \text{BOARD}(x)$

- n — $\mathtt{entity}$
- $\sqrt{\text{BOARD}}$ — $\lambda\alpha\colon *.\, \lambda x\colon \alpha.\, \text{BOARD}(x)$

$\Uparrow$ type variable
($*$ is the "kind" of all types)

V $\lambda x\colon \mathtt{event}.\, \text{BOARD}(x)$

- v — $\mathtt{event}$
- $\sqrt{\text{BOARD}}$ — $\lambda\alpha\colon *.\, \lambda x\colon \alpha.\, \text{BOARD}(x)$

☞Requirement: second-order λ-calculus

⇛same result as P1, different means
(doesn't require type match)

Possibility 2 (P2): The type variable approach

Root:	type-open predicate
Category:	type
Composition:	type-level application (Song 2019)

Example (semigrammatical word):

Neg $\lambda x: t{\to}t.\,\textsc{đéo}(x)$

Neg $t{\to}t$ √ĐÉO $\lambda\alpha: *.\,\lambda x: \alpha.\,\textsc{đéo}(x)$

Cl $\lambda x: (en{\to}t){\to}en{\to}t.\,\textsc{wèi}(x)$

Cl $(en{\to}t){\to}en{\to}t$ √WÈI $\lambda\alpha: *.\,\lambda x: \alpha.\,\textsc{wèi}(x)$

⇑

the classifier type
(simplified from Li 2013)

Not a real solution to GRS!
(function words lose their functions)

Possibility 2 (P2): The type variable approach

Root:	type-open predicate
Category:	type
Composition:	type-level application (Song 2019)

Example (semigrammatical word):

Neg $\lambda x : t \to t.\ \textsc{đéo}(x)$

Neg $t \to t$

$\sqrt{\text{ĐÉO}}$ $\lambda \alpha : *.\ \lambda x : \alpha.\ \textsc{đéo}(x)$

Cl $\lambda x : (en \to t) \to en \to t.\ \textsc{wèi}(x)$

Cl $(en \to t) \to en \to t$

$\Uparrow$

the classifier type (simplified from Li 2013)

$\sqrt{\text{WÈI}}$ $\lambda \alpha : *.\ \lambda x : \alpha.\ \textsc{wèi}(x)$

Not a real solution to GRS! (function words lose their functions)

"Root-supported" or not, function words should keep their normal functionality.

Possibility 3 (P3): The null denotation approach

Root:	no denotation	
Category:	normal denotation	
Composition:	none	(cf. Acquaviva 2019)

This brings us back to a basic idea in DM:

> *[I]t is not clear that the computational system of language … must know whether a node contains "dog" or "cat." …[T]his different … 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. (Marantz 1995:4)*

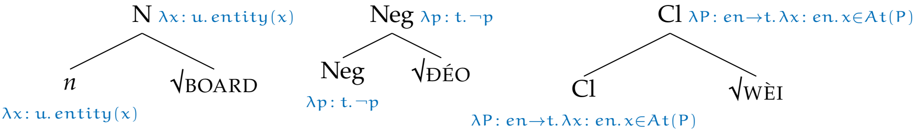
> *[The root is] an unanalyzable name, a label maximally underdetermined except for the fact of being formally distinct from other names. (Acquaviva 2019:45)*

Possibility 3 (P3): The null denotation approach

Root:	no denotation
Category:	normal denotation
Composition:	none

(cf. Acquaviva 2019)

Example:

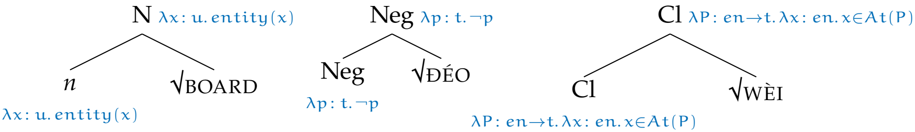


Possibility 3 (P3): The null denotation approach

Root:	no denotation
Category:	normal denotation
Composition:	none

(cf. Acquaviva 2019)

Example:



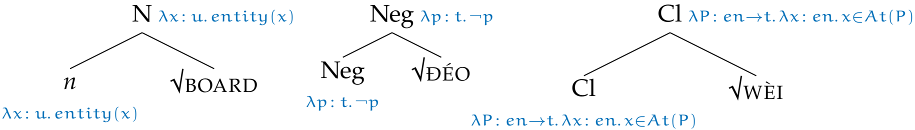
The status of the root in this approach resembles that of a *modifier*.

- …though not a logical one
- This “modification” only takes effect when the LF itself is interpreted.

Possibility 3 (P3): The null denotation approach

Root:	no denotation	
Category:	normal denotation	
Composition:	none	(cf. Acquaviva 2019)

Example:



The status of the root in this approach resembles that of a *modifier*.

- …though not a logical one
- This “modification” only takes effect when the LF itself is interpreted.

P3 truly unifies classical and generalized root syntax, but it totally ignores the root and nullifies our goal of a compositional semantics for root syntax.

New contribution: Two improved versions of P3

P4: a categorical model

Suppose the universe of discourse still contain dogs, cats, eating events, etc., that we cannot readily reference them in a suitable model of root syntax (C2) makes their supersort(s) "opaque." And our task is exactly to reconstruct sortal predicates in such an opaque setting. This is reminiscent of how things are done in category theory, where *objects* are opaque by definition.

P5: composition via monad

Our concern as a whole is reminiscent of the "at-issue" (truth-conditional) vs. "side-issue" (non-truth-conditional) distinction in Asudeh & Giorgolo (2020), where composition of meanings in these two dimensions is implemented via the category-theoretic notion *monad*.

Overall, P4–5 highlight the usefulness of category theory in linguistics.

Overview

Possibility 4 (P4): The categorical logic approach

Idea

We can try to lift the usual set-theoretic model in formal semantics to a category-theoretic one and see what it does to root syntax.

natural language syntax ➙ logical syntax ➙ set structure
(λ-calculus)

Possibility 4 (P4): The categorical logic approach

Idea

We can try to lift the usual set-theoretic model in formal semantics to a category-theoretic one and see what it does to root syntax.

natural language syntax → logical syntax (λ-calculus) → ~~set structure~~ ↘ categorical structure

Possibility 4 (P4): The categorical logic approach

Idea

We can try to lift the usual set-theoretic model in formal semantics to a category-theoretic one and see what it does to root syntax.

natural language syntax → logical syntax → ~~set structure~~
(λ-calculus) ↘ categorical structure

There already exists a categorical semantics for λ-calculus (Crole 1993, Pitts 2000), so we can build on that.

- Types are interpreted as objects.
- Terms are interpreted as morphisms.

In the category **Set**, the objects are opaque sets and the morphisms are total functions—just what we need.

The categorical setting of **Set**

In category theory, a category $\mathcal{C}$ is defined by

- a collection of objects $A, B, C, \ldots$ (which are opaque)
- a collection of morphisms $f, g, h, \ldots$ between objects (including an identity morphism 1_A for each object A), and
- morphism composition (obeying associativity and unit law)

$$\begin{tikzcd} A \arrow[r, "f"] & B \arrow[r, "g"] \arrow[loop above, "1_B"] & C \arrow[r, "h"] & D \end{tikzcd}$$

$$h \circ (g \circ f) = (h \circ g) \circ f$$

$$1_B \circ f = f, g \circ 1_B = g$$

In **Set**, objects are sets, morphisms are functions, and morphism composition is function composition.

The categorical setting of **Set**

In addition, **Set** has categorified versions of

1. cartesian products ➙ product objects $A \times B, A \times B \times C, \ldots$
2. singleton sets ➙ a terminal object 1 such that $\forall C \exists C \xrightarrow{!_C} 1$
3. set elements ➙ via "global elements" of the form $1 \xrightarrow{a} A$ for any $a \in A$
4. function spaces ➙ exponential objects $B^A, C^{B^A}, \ldots$
5. the subset relation ➙ via a subobject classifier $1 \xrightarrow{\mathtt{true}} \Omega = \{\mathtt{true}, \mathtt{false}\}$

①②④ make **Set** a *cartesian closed category*, and ①②④⑤ make it a *topos*.

☞**Set** has abundant good features for our "root syntax experiment."

Possibility 4 (P4): The categorical logic approach

To translate root syntax into logical syntax, we only need a tiny bit of modification to the usual λ-calculus signature (Crole 1993):

- Our ground types include the generic sort $\mathtt{u}$ and its subsorts (e.g., $\mathtt{entity}$, $\mathtt{event}$), the type $\mathtt{t}$ of truth values, and a type $\mathtt{r}$ for roots.

Possibility 4 (P4): The categorical logic approach

To translate root syntax into logical syntax, we only need a tiny bit of modification to the usual λ-calculus signature (Crole 1993):

- Our ground types include the generic sort u and its subsorts (e.g., entity, event), the type t of truth values, and a type r for roots.

The model in **Set** is also quite straightforward.

- We give every ground type γ a **Set**-object $[\![\gamma]\!]$, specifically an object for each sort, an object Ω for t, and an object R for r.
- Unit, product, and function types are modeled in their usual ways (as terminal, product, and exponential objects).

Possibility 4 (P4): The categorical logic approach

To translate root syntax into logical syntax, we only need a tiny bit of modification to the usual λ-calculus signature (Crole 1993):

- Our ground types include the generic sort u and its subsorts (e.g., entity, event), the type t of truth values, and a type r for roots.

The model in **Set** is also quite straightforward.

- We give every ground type γ a **Set**-object $[\![\gamma]\!]$, specifically an object for each sort, an object Ω for t, and an object R for r.
- Unit, product, and function types are modeled in their usual ways (as terminal, product, and exponential objects).

We can let each root denote a constant of type r and translate $[_{\mathsf{X}}\ \mathsf{X}\ \sqrt{}\ ]$ as $\langle [\![\mathsf{X}]\!], [\![\sqrt{}]\!] \rangle$. It has the type $\alpha \times \mathsf{r}$, where α is the semantic type of X. That is, we view root categorization simply as a pairing procedure, with the root serving as a "tag name" for the grammatical category.

Possibility 4 (P4): The categorical logic approach

How does P4 resolve the four challenges?

1. *How to logically represent roots?*
   As constants of type $\mathtt{r}$.
2. *How to mirror the extreme vagueness of roots in the model?*
   By not giving them set-theoretic extensions. Instead, each root qua a constant is modeled by a global element $1 \to \mathsf{R}$ in **Set**. Similarly, each idiosyncratic sorting predicate is modeled by a morphism from its sort-object to Ω (e.g., $[\![\mathtt{dog}]\!] = [\![\mathtt{entity}]\!] \xrightarrow{\mathtt{dog}} \Omega$).
3. *How to compose roots and categories?*
   By product formation in **Set**, which is a categorified version of the conjunctivist approach but free from the type match constraint.
4. *How to keep up with generalized root syntax?*
   Since there is no type match constraint, the product-based composition is uniformly applicable to content and semigrammatical words.

Possibility 4 (P4): The categorical logic approach

Example (suppose little x categories denote sorts):

$$\mathrm{N}\langle \mathtt{entity}, \mathrm{BOARD}\rangle \Rightarrow \langle \mathtt{id}_{[\![\mathtt{entity}]\!]}, \mathrm{BOARD}\rangle$$

n — $\mathtt{entity}$ — $[\![\mathtt{entity}]\!]$

$\sqrt{\mathrm{BOARD}}$ — BOARD — $1 \xrightarrow{\mathrm{BOARD}} [\![\mathrm{r}]\!]$

via root syntax

$\boxed{\text{two ways to model the bare noun } \textit{board}}$

not via root syntax

$$[\![\mathrm{board}]\!] = [\![\mathtt{entity}]\!] \xrightarrow{\mathrm{board}} \Omega$$

Possibility 4 (P4): The categorical logic approach

Example (suppose little x categories denote sorts):

$\mathrm{N}\langle \mathtt{entity}, \mathrm{BOARD}\rangle \Rrightarrow \langle \mathrm{id}_{[\![\mathtt{entity}]\!]}, \mathrm{BOARD}\rangle$

n $\quad$ $\sqrt{}\mathrm{BOARD}$

$\mathtt{entity}$ $\quad$ BOARD

$[\![\mathtt{entity}]\!]$ $\quad$ $1 \xrightarrow{\mathrm{BOARD}} [\![\mathrm{r}]\!]$

$[\![\mathrm{board}]\!] = [\![\mathtt{entity}]\!] \xrightarrow{\mathrm{board}} \Omega$

via root syntax

$\boxed{\text{two ways to model the bare noun } \textit{board}}$

not via root syntax

There is a one-one correspondence between morphisms $[\![\mathtt{entity}]\!] \to \Omega$ and pairs $\langle \mathrm{id}_{[\![\mathtt{entity}]\!]}, [\![\sqrt{}]\!]\rangle$ for each viable $\sqrt{}$ (similarly for other types).

Possibility 4 (P4): The categorical logic approach

More generally, the root categorization schema $[_X \; X \sqrt{\;}\;]$ gives rise to an isomorphism between two sets of morphisms:

$$\{\mathrm{id}_{[\![\alpha]\!]}\} \times \mathrm{Hom}(1, R) \cong \mathrm{Hom}([\![\alpha]\!], \Omega)$$

which “naturally” holds in our model however $[\![\alpha]\!]$ (i.e., X) varies. (The *hom-set* $\mathrm{Hom}(A, B)$ is the set of all morphisms $A \to B$.)

This could potentially be extended to a natural isomorphism:

$$\begin{tikzcd} \mathbf{Set} \arrow[r, bend left, "\mathrm{id}_{(-)} \times \mathrm{Hom}(1, R)"] \arrow[r, bend right, "\mathrm{Hom}(-, \Omega)"'] \arrow[r, phantom, "\Updownarrow\theta"] & \mathbf{Set} \end{tikzcd}$$

But I will not further discuss that here.

Another way to bear out the bijection is via a topos pullback

$$\begin{tikzcd} \langle \mathrm{id}_{[\![\mathtt{entity}]\!]}, [\![\sqrt{\text{BOARD}}]\!]\rangle \arrow[r, tail, "u"] \arrow[d, "!"'] & [\![\mathtt{entity}]\!] \arrow[d, "\chi_{\langle \mathrm{id}_{[\![\mathtt{entity}]\!]}, [\![\sqrt{\text{BOARD}}]\!]\rangle}"] \\ 1 \arrow[r, "\mathtt{true}"] & \Omega \end{tikzcd}$$

which basically classifies $\langle \mathrm{id}_{[\![\mathtt{entity}]\!]}, [\![\sqrt{\text{BOARD}}]\!]\rangle$ as a subtype of entity. (I use the same notation $\langle \mathrm{id}_{[\![\mathtt{entity}]\!]}, [\![\sqrt{\text{BOARD}}]\!]\rangle$ to represent the idiosyncratic set the categorized root corresponds to.)

So, we actually have a three-way correspondence in the model:

morphisms like $[\![\mathtt{entity}]\!] \xrightarrow{\text{board}} \Omega$ (characteristic functions) $\leftrightarrow$

morphism pairs like $\langle \mathrm{id}_{[\![\mathtt{entity}]\!]}, [\![\sqrt{\text{BOARD}}]\!]\rangle$ (root categorization) $\leftrightarrow$

independent objects like Board (lifted extension sets)

Possibility 4 (P4): The categorical logic approach

This pullback can be extended to semigrammatical words too.

$$\begin{tikzcd}
\langle \mathrm{id}_{[\![t \to t]\!]}, [\![\sqrt{\text{ĐÉO}}]\!] \rangle \arrow[r, tail, "u"] \arrow[d, "!"'] & [\![t \to t]\!] \arrow[d, "\chi_{\langle \mathrm{id}_{[\![t \to t]\!]}, [\![\sqrt{\text{ĐÉO}}]\!] \rangle}"] \\
1 \arrow[r, "\mathrm{true}"] & \Omega
\end{tikzcd}$$

which classifies $\langle \mathrm{id}_{[\![t \to t]\!]}, [\![\sqrt{\text{ĐÉO}}]\!] \rangle$ as a subtype of $[\![t \to t]\!]$.
(Here we must understand the characteristic function in a different way—not as a characterization of individuals but as one of negation forces.)

Overall, this pullback is not as intuitively natural as the previous one, as the negator *đéo* that $\langle \mathrm{id}_{[\![t \to t]\!]}, [\![\sqrt{\text{ĐÉO}}]\!] \rangle$ corresponds to does not have a ground-level extension. (Maybe we need a more general category here?)

Overview

Possibility 5 (P5): The monadic approach

Asudeh & Giorgolo (2020) use monads to compose Potts's (2005, 2007) "at-issue" (truth-conditional) and "side-issue" (non-truth-conditional) meanings. P5 is based on their treatment of *conventional implicature*.

(3) a. Donald is a **Yank**.

b. This **cur** bit me. (Asudeh & Giorgolo 2020:13)

Words like *Yank* and *cur* carry speaker attitudes besides their basic meanings. A&G view these as conventional, non-truth-conditional.

☞This is highly similar to what we see in semigrammatical items.

To further generalize the idea, the purely idiosyncratic meanings of content words are also conventionalized (i.e., lexicalized). Their roots, too, are *modifiers* of the grammatical/logical structure.

Possibility 5 (P5): The monadic approach

Idea

Let logical computation proceed as usual and store idiosyncratic, conventional meanings in a "log" area of the computation. The monad tool keeps track of the two semantic dimensions in one big function.

Monad is a highly general concept in category theory, but the particular monad A&G use, the *writer monad*, is from functional programming.

> *[T]he writer monad [is] used for logging or tracing the execution of functions. It's also an example of a more general mechanism for embedding [side] effects in pure computations. (Milewski 2019:49)*

`type Writer a = (a, String)` ⇒ creates a "log" area for a type and "writes" a string into it

⇑ any type

This extends to an endofunctor on the category of types.

— Objects: types | Morphisms: a –> Writer b

— Composition: (>=>) :: (a –> Writer b) –> (b –> Writer c) –> (a –> Writer c)

Possibility 5 (P5): The monadic approach

Basically, a monad is such an endofunctor together with two natural transformations, respectively called *unit* (η) and *multiplication* (μ), which satisfy the associativity and unit laws. For the writer monad, we need a further operator *bind* ($>>=$), which can be defined by μ.

In the case of the writer monad:

- $\eta(x) = \langle x, e \rangle : a \to \text{Writer } a$ *embeds a value in a trivial wrapper (e is the empty string)*
- $\mu\langle\langle x, s_1\rangle, s_2\rangle = \langle x, s_1 ++ s_2\rangle : \text{Writer } (\text{Writer } a) \to \text{Writer } a$
  combines log entries by concatenation (s_1 and s_2 are strings) ⇛so the log slot relies on a monoid
- $\langle x, s_1\rangle >>= \lambda u.\langle f(u), s_2\rangle = \mu((\lambda\langle u, s\rangle.\langle\langle f(u), s_2\rangle, s\rangle)\langle x, s_1\rangle)$
  $= \mu\langle\langle f(x), s_2\rangle, s_1\rangle = \langle f(x), s_2 ++ s_1\rangle$
  $: \text{Writer } a \to (a \to \text{Writer } b) \to \text{Writer } b$ *pure function and logging proceed in parallel*

☞Equivalent definitions: $\langle \text{Writer}, \eta, \mu\rangle \equiv \langle \text{Writer}, \eta, >>=\rangle$
(Remember that Writer is a *functor*.)

Possibility 5 (P5): The monadic approach

Example: *Donald is a Yank.* (Asudeh & Giorgolo 2020:55ff.)

- At-issue: Donald is an American.
- Side-issue: The speaker has a negative attitude toward Americans. call this p

The negative attitude is encoded in *Yank*, so

- *Yank* should have a *monad-type* denotation, and
- it should be composed with other ingredients *monadically*.

Specifically, this helper function wraps p in a dummy monadic term $\langle 1, \{p\} \rangle$

- $[\![\text{Yank}]\!] = \mathtt{write}(p) \mathbin{>\!\!>\!\!=} \lambda y.\, \eta(\mathtt{American}) = \langle \mathtt{American}, \{p\} \rangle$
- $[\![\text{Yank}]\!] \mathbin{>\!\!>\!\!=} \lambda x.\, \eta([\![\text{a}]\!]([\![\text{is}]\!](x))([\![\text{Donald}]\!])) = [\![\text{Yank}]\!] \mathbin{>\!\!>\!\!=} \lambda x.\, \eta(x(\mathtt{Donald}))$
  $= \langle \mathtt{American}, \{p\} \rangle \mathbin{>\!\!>\!\!=} \lambda x.\, \langle x(\mathtt{Donald}), \emptyset \rangle = \langle \mathtt{American}(\mathtt{Donald}), \emptyset \cup \{p\} \rangle$

here μ works by set union

(the monoid is the power set of the set of all propositions)

Possibility 5 (P5): The monadic approach

We can directly apply A&G's writer monad to root syntax, with one small modification—it is not enough to simply let root information pile up in the log area; we need to record which root tags which category.

Let's begin with the schema $[_X \text{ X} \surd\ ]$. I assign it the logical form

$$write(X_{\surd}) >>= \lambda y.\eta([\![X]\!])$$

which writes $\{X_{\surd}\}$ into the log slot of a vacuous monadic term.

Example:

- Content word: $[\![[_N\ n\ \surd\text{BOARD}\ ]]\!] = write(n_{\surd\text{BOARD}}) >>= \lambda y.\eta[\![n]\!] = \langle [\![n]\!], \{n_{\surd\text{BOARD}}\}\rangle$ (an entity that is idiosyncratically characterized by √BOARD)
- Semigrammatical word: $[\![[_{\text{Neg}}\ \text{Neg}\ \surd\text{ĐÉO}\ ]]\!] = write(\text{Neg}_{\surd\text{ĐÉO}}) >>= \lambda y.\eta[\![\text{Neg}]\!] = \langle [\![\text{Neg}]\!], \{\text{Neg}_{\surd\text{ĐÉO}}\}\rangle$ (a negator that is idiosyncratically characterized by √ĐÉO)

Possibility 5 (P5): The monadic approach

How does P5 resolve the four challenges?

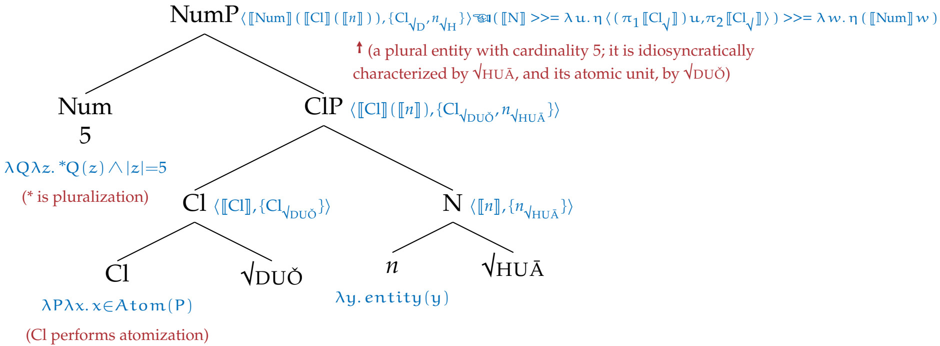
1. *How to logically represent roots?*
   Roots need no logical denotations (as in P3), since they do not participate in "at-issue" computation.
2. *How to mirror the extreme vagueness of roots in the model?*
   Not clear yet, as we have only focused on the LF level (so do A&G).
3. *How to compose roots and categories?*
   Via the monadic >>=.
4. *How to keep up with generalized root syntax?*
   As in P4, the composition mode here is uniformly applicable to content and semigrammatical words.

Possibility 5 (P5): The monadic approach

The logging does not interfere with "at-issue" computation.

- Recall that in P2 the root-categorizer composition messes up the grammatical category's normal functionality.
- There's no such trouble in P5 thanks to the definition of >>=.

Example: *wǔ duǒ huā* 'five CLF flower; five flowers' (simplified from Li 2013)



Possibility 5 (P5): The monadic approach

The logging does not interfere with "at-issue" computation.

- Recall that in P2 the root-categorizer composition messes up the grammatical category's normal functionality.
- There's no such trouble in P5 thanks to the definition of >>=.

Example: *Mary walks* (based on Bowers 2010, via Lohndal 2019)

VoiceP $\langle ([\![\text{Voice}]\!]([\![v]\!]))[\![\text{Mary}]\!], \{v_{\sqrt{\text{WALK}}}\}\rangle$ ☜ $([\![\text{V}]\!] \mathrel{>\!\!>\!=} \lambda z.\eta([\![\text{Voice}]\!]z)) \mathrel{>\!\!>\!=} \lambda w.\eta(w([\![\text{Mary}]\!]))$

↑ (an event whose Agent is Mary; it is idiosyncratically characterized by √WALK)

DP Mary

VoiceP $\langle [\![\text{Voice}]\!]([\![v]\!]), \{v_{\sqrt{\text{WALK}}}\}\rangle$

Due to the upward continuation of monadic computation, P5 has global ramifications for semantic composition (as in A&G's work).

Voice $\lambda P \lambda y \lambda x.[P(x) \wedge Agent(x,y)]$

(Voice introduces Agent)

V $\langle [\![v]\!], \{v_{\sqrt{\text{WALK}}}\}\rangle$

v $\lambda x.event(x)$

√WALK

Overview

Summary

Four challenges of (generalized) root syntax for formal semantics:

- C1 How to logically represent roots?
- C2 How to mirror the extreme vagueness of roots in the model?
- C3 How to compose roots and categories?
- C4 How to keep up with generalized root syntax?

Five possible directions:

- P1 The conjunctivist approach ✗
- P2 The type variable approach ✗
- P3 The null denotation approach ✔
- P4 The categorical logic approach ✔
- P5 The monadic approach ✔

Next step: (i) a model for P5; (ii) potential combination of P4–5.

Thank you!

Selected references I

Acedo-Matellán, V. & C. Real-Puigdollers
Roots into functional nodes: Exploring locality and semi-lexicality
The Linguistic Review 36(3), 411–436, 2019

Acquaviva, P.
Roots and lexicality in distributed morphology
YPL2 special issue, 2009

Acquaviva, P.
Categorization as noun construction
Gender and noun classification
OUP, 2019

Asudeh, A. & G. Giorgolo
Enriched meanings
OUP, 2020

Selected references II

Borer, H.
Structuring sense: In name only (Vol. 1)
OUP, 2005

Chomsky, N.
UCLA lectures (https://linguistics.ucla.edu/noam-chomsky/)
Apr 29–May 2, 2019

Crole, R.
Categories for types
CUP, 1993

Halle, M. & A. Marantz
Some key features of distributed morphology
MIT working papers in linguistics 21, 275–288, 1993

Selected references III

Kelly, J.
The syntax-semantics interface in distributed morphology
Georgetown University dissertation, 2013

Milewski, B.
Categories theory for programmers
Blurb, 2019

Pitts, A.
Categorical logic
Handbook of logic in computer science (Vol. 5)
OUP, 2000

Potts, C.
The logic of conventional implicatures
OUP, 2005

Selected references IV

Song, C.
On the formal flexibility of syntactic categories
University of Cambridge dissertation, 2019

Song, C.
A typology of semilexicality and the locus of grammatical variation (https://youtu.be/2X0-LODELfc)
Talk at ICFL9, 2021