Mathematical Model of Language

Matilde Marcolli

Caltech, Winter 2026
Ma191b: Geometry of Neuroscience

- M. Marcolli, N. Chomsky, R.C. Berwick, *Mathematical Structure of Syntactic Merge. An Algebraic Model of Generative Linguistics*, MIT Press, 2025.

Syntax: focus on the "large scale structure" of language, where main structure-building computational process resides

Linguistics input:

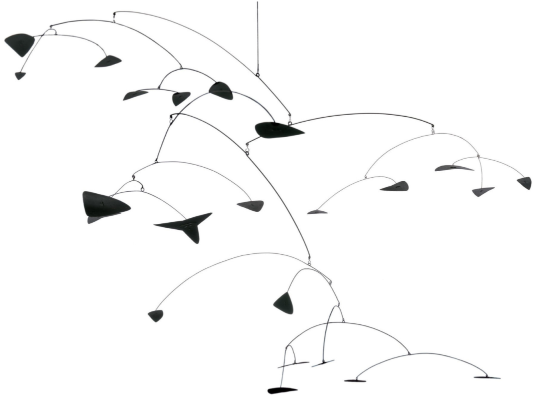
- what are the main structure-building and transformation operations of syntax
- cross-linguistic variation and underlying common universal computational core
- interface between syntax and semantics (structure formation and parsing)
- language acquisition and specialization of core computational mechanism to one particular language

Strings versus Structures: important development in the modeling of language

- what language appears to look like

$$0000200210212111210220020002201210220120002000200002111201002 0\ldots$$

- what language actually looks like

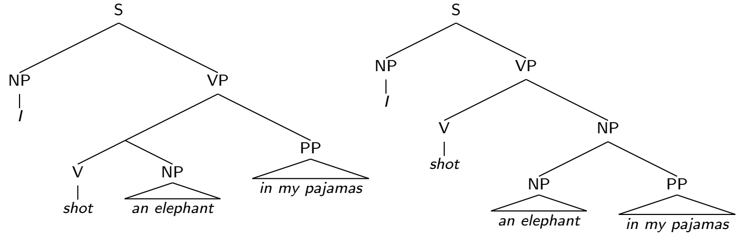


How is language "a structure" beyond the string of words?

Example: look at this sentence

I shot an elephant in my pajamas – what it was doing in my pajamas, I'll never know (Groucho Marx)

... why is it funny? because it conflates two different structures

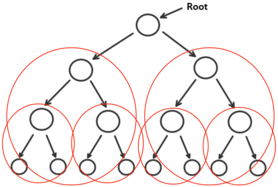


a sentence is **not** just a string of words!

we perceive *structural relations* not *proximity relations* in the ordering of words: structure reflects the different possible way sentences are *generated*

Chomsky's Minimalism Program (1990s onward, reformulated after 2013): identifying core computational process of structure formation in syntax

- *simple* core computational structure: Merge

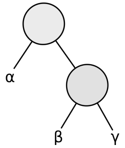


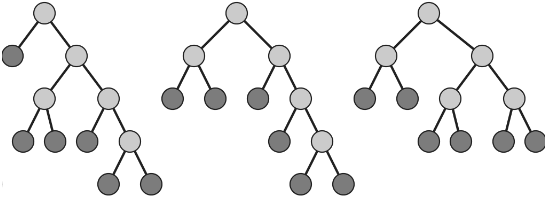
This process generates structures

- called *hierarchical structures* (or *syntactic objects*) in linguistics
- called non-planar (or abstract) binary rooted trees by mathematicians

lexical items (at leaves) combined into a hierarchical structure; the tree is dangling from the root, not lying in the plane, so the lexical items at leaves are not an ordered string of symbols

- a computational process of *structure formation*

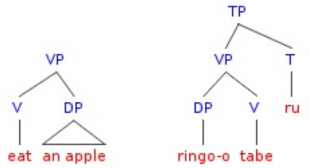




- generative process of *non-planar* full binary rooted trees with labelled leaves
- what about the linear order of sentences?
- we can learn *any* language (widely varying word order structure): so core computational mechanism *common to all languages* should not depend on one particular word order: that is *learned* during language acquisition

Externalization

- follows the generative process of structure formation (where sentences are hierarchical non-planar tree-structures)
- language is externalized through the sensory-motor system (speech, sign) as temporally ordered sequence of words
- equivalently: the trees acquire a *planar structure*
- language-dependent forms of *word order* + other rules
- example: English is head-initial, Japanese is head-final



VP= verb phrase, TP= tense phrase, DP= determiner phrase

- syntactic variability, *syntactic parameters*, *geometry of syntax* (which constraints?)

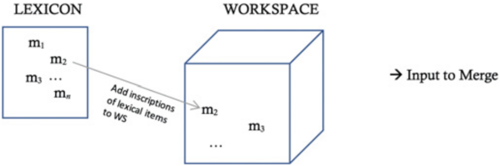
Summarizing the picture so far:

- language starts as hierarchical structures: free floating (non-planar) binary rooted trees encoding only structural relations between words
- the physical constraints of externalization (sound, motion) force a reduction to an ordered sequence
- our brain in language acquisition quickly learns from a small collection of such examples the (non-visible and complex) structure of syntax

- Note: the enormous computational difficulty of solving this *inverse problem*.... yet, our brain does it *easily*

A closer look at structure formation (movement/transformation)

- we not only form sentences: we modify them, transform them, manipulate them according to precise rules (forming questions, changing to passive voice, etc)
- the Merge operation of structure formation has two aspects: Internal Merge and External Merge
- Internal Merge (responsible for "movement") needs to reach inside formed structures (syntactic objects) for constituent parts (accessible terms) available for further calculation
- also structure formation is a "bottom-up operation" that proceeds in steps: intermediate scratchpad for operations (workspaces)



Main aspects of the Merge model of syntax (in its most recent formulation: 2013 onward)

- syntactic objects
- workspaces
- accessible terms
- Merge action on workspaces
- externalization

all these notions have a precise mathematical formulation that shows many aspects of the linguistic model that have empirical grounds in fact follow by constraints from the algebraic structure magma of syntactic objects: free nonassociative commutative magma

- start with a set $\mathcal{SO}_0$ of lexical items and syntactic features
- a binary operation $\mathfrak{M}$ commutative, nonassociative:

$$\mathfrak{M}(\alpha,\beta)=\mathfrak{M}(\beta,\alpha) \quad \text{but} \quad \mathfrak{M}(\gamma,\mathfrak{M}(\alpha,\beta))\neq\mathfrak{M}(\mathfrak{M}(\gamma,\alpha),\beta)$$

- set of syntactic objects $\mathcal{SO}$ is the *free nonassociative commutative magma* generated by $\mathcal{SO}_0$

$$\mathcal{SO}=\mathrm{Magma}_{na,c}(\mathcal{SO}_0,\mathfrak{M})$$

- all elements obtained by repeated application of $\mathfrak{M}$ starting from elements of $\mathcal{SO}_0$

free magma and abstract binary rooted trees

- the free nonassociative commutative magma on a set X is canonically isomorphic to the set $\mathfrak{T}_X$ of abstract binary rooted trees with leaves decorated by elements of the set X

$$\mathrm{Magma}_{na,c}(\mathcal{SO}_0, \mathfrak{M}) = \mathfrak{T}_{\mathcal{SO}_0}$$

- so syntactic objects $T \in \mathcal{SO} = \mathfrak{T}_{\mathcal{SO}_0}$ are *abstract binary rooted trees with leaves decorated by lexical items*
- abstract= no assigned *planar embedding* (also called non-planar)
- leaves do not form an ordered string of elements in $\mathcal{SO}_0$

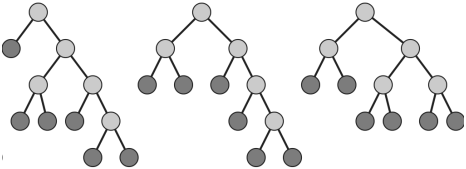
workspaces

- free commutative monoid on the set $\mathcal{SO}$

$$\mathfrak{F}_{\mathcal{SO}_0} = \mathrm{Monoid}_c(\mathcal{SO}, \sqcup) = \mathrm{Magma}_{a,c}(\mathcal{SO}, \sqcup)$$

- binary forests: finite disjoint unions of abstract binary rooted trees

$$F = T_1 \sqcup \cdots \sqcup T_n \quad \text{with} \quad T_i \in \mathfrak{T}_{\mathcal{SO}_0}$$

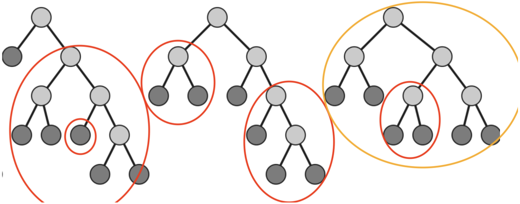


- set of workspaces = set of binary forests $\mathfrak{F}_{\mathcal{SO}_0}$
- Merge operations map the set $\mathfrak{F}_{\mathcal{SO}_0}$ to itself (transform workspaces into new workspaces)

This action should account for two types of operations: structure formation (External Merge) and movement/transformation (Internal Merge), this one requires accessing substructures

accessible terms

- *accessible terms of a syntactic object* T: subtrees T_v, with v a non-root vertex of T and T_v the subtree below v
- *accessible terms of a workspace* $F = \sqcup_a T_a$: accessible terms of each T_a *and* components T_a
- examples of accessible terms:



- tree $T \in \mathfrak{T}_{\mathcal{SO}_0}$ and $v \in V(T)$: subtree T_v rooted at v
- $V^o(T)$ non-root vertices of T
- accessible terms of T

$$Acc(T) = \{ T_v \,|\, v \in V^o(T) \} \text{ and } Acc'(T) = \{ T_v \,|\, v \in V(T) \}$$

- workspace $F \in \mathfrak{F}_{\mathcal{SO}_0}$ with $F = \sqcup_{a \in \mathcal{I}} T_a$

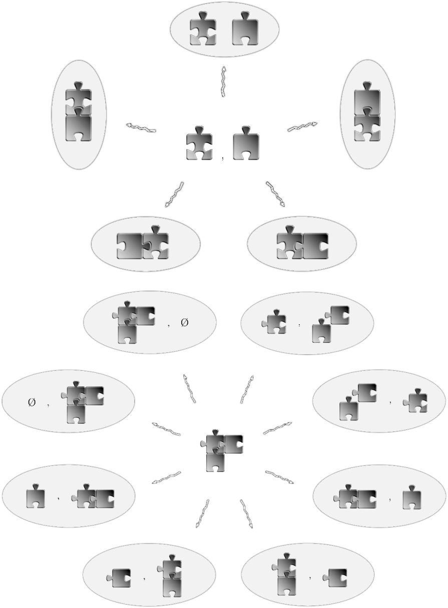
$$Acc'(F) = \bigsqcup_{a \in \mathcal{I}} Acc'(T_a)$$

- What mathematical structure governs workspaces and accessible terms?
- answer: Workspaces form a Hopf algebra

What is a Hopf algebra?

- mathematical method of describing composition–decomposition
- product: an "assemble operation" (two inputs one output) for how to assemble different objects together
- coproduct: a "decomposition operation" (one input two outputs) listing all possible ways of decomposing an objects into parts
- compatibility between these two operations

  (a relation when interchanging order of product/coproduct)



A formal definition of Hopf algebra

- Hopf algebra $\mathcal{H}$ is a vector space over a field $\mathbb{K}$, endowed with
  - multiplication $m : \mathcal{H} \otimes_{\mathbb{K}} \mathcal{H} \to \mathcal{H}$;
  - unit $u : \mathbb{K} \to \mathcal{H}$;
  - comultiplication $\Delta : \mathcal{H} \to \mathcal{H} \otimes_{\mathbb{K}} \mathcal{H}$;
  - counit $\epsilon : \mathcal{H} \to \mathbb{K}$;
  - antipode $S : \mathcal{H} \to \mathcal{H}$
- multiplication is associative
- comultiplication is coassociative
- u is multiplicative unit and ϵ is comultiplicative counit
- comultiplication and counit are homomorphisms of algebras and multiplication and unit are homomorphisms of coalgebras
- S relates m and Δ and u and ϵ
- all this expressed by diagrams
- the formal requirements above are what constitutes a good pair of composition/decomposition operations

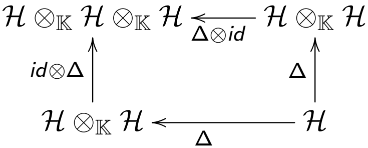
multiplication: associativity and unit

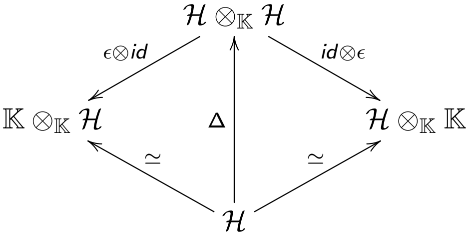
$$\begin{array}{ccc} \mathcal{H}\otimes_{\mathbb{K}}\mathcal{H}\otimes_{\mathbb{K}}\mathcal{H} & \xrightarrow{m\otimes id} & \mathcal{H}\otimes_{\mathbb{K}}\mathcal{H} \\ \Big\downarrow{\scriptstyle id\otimes m} & & \Big\downarrow{\scriptstyle m} \\ \mathcal{H}\otimes_{\mathbb{K}}\mathcal{H} & \xrightarrow{\quad m \quad} & \mathcal{H} \end{array}$$

$$\begin{array}{ccccc} & & \mathcal{H}\otimes_{\mathbb{K}}\mathcal{H} & & \\ & \overset{u\otimes id}{\nearrow} & & \overset{id\otimes u}{\nwarrow} & \\ \mathbb{K}\otimes_{\mathbb{K}}\mathcal{H} & & \Big\downarrow{\scriptstyle m} & & \mathcal{H}\otimes_{\mathbb{K}}\mathbb{K} \\ & \overset{\simeq}{\searrow} & & \overset{\simeq}{\swarrow} & \\ & & \mathcal{H} & & \end{array}$$

commutativity of these diagrams

comultiplication: coassociativity and counit



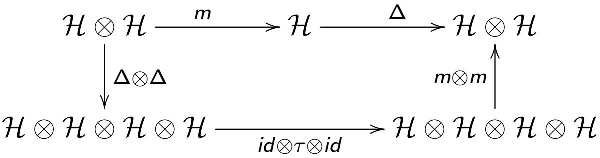


commutativity of these diagrams: coassociativity

$$(\mathrm{id} \otimes \Delta) \circ \Delta = (\Delta \otimes \mathrm{id}) \circ \Delta$$

compatibility of product and coproduct

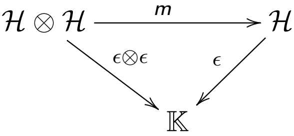
compatibility between product and coproduct:



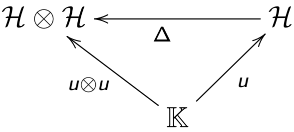
where $\tau : \mathcal{H}\otimes\mathcal{H}\otimes\mathcal{H}\otimes\mathcal{H} \to \mathcal{H}\otimes\mathcal{H}\otimes\mathcal{H}\otimes\mathcal{H}$ permutes the two middle factors:

$$\Delta \circ \sqcup = (\sqcup \otimes \sqcup) \circ \tau \circ (\Delta \otimes \Delta)$$

behavior of unit and counit with respect to coproduct and product:



and



using the identification $\mathbb{K}\otimes\mathbb{K} = \mathbb{K}$.

antipode: further compatibility, commutativity of diagram

$$\begin{array}{ccccc} \mathcal{H}\otimes_{\mathbb{K}}\mathcal{H} & \xrightarrow{m} & \mathcal{H} & \xleftarrow[m]{} & \mathcal{H}\otimes_{\mathbb{K}}\mathcal{H} \\ {\scriptstyle id\otimes S}\uparrow & & {\scriptstyle u\circ\epsilon}\uparrow & & {\scriptstyle S\otimes id}\uparrow \\ \mathcal{H}\otimes_{\mathbb{K}}\mathcal{H} & \xleftarrow[\Delta]{} & \mathcal{H} & \xrightarrow{\Delta} & \mathcal{H}\otimes_{\mathbb{K}}\mathcal{H} \end{array}$$

Note: if the Hopf algebra is graded $\mathcal{H} = \oplus_{\ell\geq 0}\mathcal{H}_\ell$ with $\mathcal{H}_0 = \mathbb{K}$ and m, Δ compatible with grading, antipode comes for free

(ie S is determined by the rest of the structure: is not an additional constraint)

$$S(x) = -x - \sum S(x')x''$$

inductively for $\Delta(x) = x\otimes 1 + 1\otimes x + \sum x'\otimes x''$ with x', x'' terms of lower degree (grading by number of leaves)

Workspaces form a Hopf algebra

- vector space $\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0})$ spanned by forests
- product: assemble workspaces (forests) by disjoint union of syntactic objects (trees)

$$(T_1, T_2) \mapsto F = T_1 \sqcup T_2 \quad \text{and} \quad (F_1, F_2) \mapsto F = F_1 \sqcup F_2$$

- coproduct: disassembles workspaces into constituent parts
- if use coproduct where trees are primitive elements $\Delta(T_a) = T_a \otimes 1 + 1 \otimes T_a$ get for $\Delta(F) = \sqcup_a \Delta(T_a)$

$$\Delta(F) = \sum_{\mathcal{I}=\mathcal{I}'\sqcup\mathcal{I}''} (\sqcup_{a\in\mathcal{I}'} T_a) \otimes (\sqcup_{a\in\mathcal{I}''} T_a) \qquad \text{for } F = \sqcup_{a\in\mathcal{I}} T_a$$

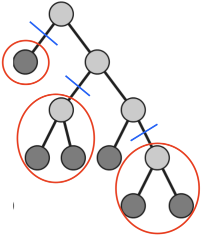
- would get only *partitions of the workspace*, no access to substructures (accessible terms)
- would get External Merge but no movement (Internal Merge)

coproduct: admissible cuts

- in a tree $T \in \mathfrak{T}_{\mathcal{SO}_0}$ consider forests $F_{\underline{v}} \subset T$
  $F_{\underline{v}} = T_{v_1} \sqcup \cdots \sqcup T_{v_n}$ of accessible terms
- coproduct

$$\Delta(T) = T \otimes 1 + 1 \otimes T + \sum_{\underline{v}} F_{\underline{v}} \otimes T/F_{\underline{v}}$$

- such $F_{\underline{v}}$ corresponds to an admissible cut C of T with forest $\pi_C(T) = F_{\underline{v}}$ and remaining tree $\rho_C(T) = T/F_{\underline{v}}$
- admissible cut: at most one cut on any path from root to leaves



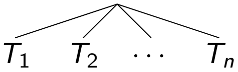
- *Warning*: some care in defining $T/F_{\underline{v}}$ for coassociativity

Merge action

$$\mathfrak{M}_{S,S'} = \sqcup \circ (\mathfrak{B} \otimes 1) \circ \delta_{S,S'} \circ \Delta$$

1. coproduct Δ extracts and displays all accessible terms
2. $\delta_{S,S'}$ locates matching pairs of accessible terms
3. grafting operator

$$\mathfrak{B} : T_1 \sqcup \cdots \sqcup T_n =$$



4. product $\sqcup$ recomposes the new workspace

note that because of $\delta_{S,S'}$ only binary $\mathfrak{B}$ applies

focus on $\Delta = \Delta^d$ coproduct: $(\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}), \sqcup, \Delta)$ Hopf algebra of workspaces

types of Merge operations

- External Merge (EM): S, S' components of workspace
- Sideward Merge (EM): at least one of S, S' an accessible term
- Internal Merge (IM): $\mathfrak{M}_{T_v, T/T_v} \circ \mathfrak{M}_{T_v,1}$ (extraction of accessible term, merged with its own complement)
- assembled Merge operation (all possible Merge transformations on a workspace)

$$\mathcal{K}_2 = \sum_{S,S'} \mathfrak{M}_{S,S'} = \sqcup \circ (\mathfrak{B} \otimes \mathrm{id}) \circ \Pi^{(2)} \circ \Delta$$

  and $\mathcal{K}_1 = \sqcup \circ \Pi^{(1)} \circ \Delta = \sum_S \mathfrak{M}_{S,1}$
- $\Pi^{(k)}$ projection on terms of coproduct with k components $T_{v_1} \sqcup \cdots \sqcup T_{v_k}$ in l.h.s.

Merge Hopf algebra Markov chain

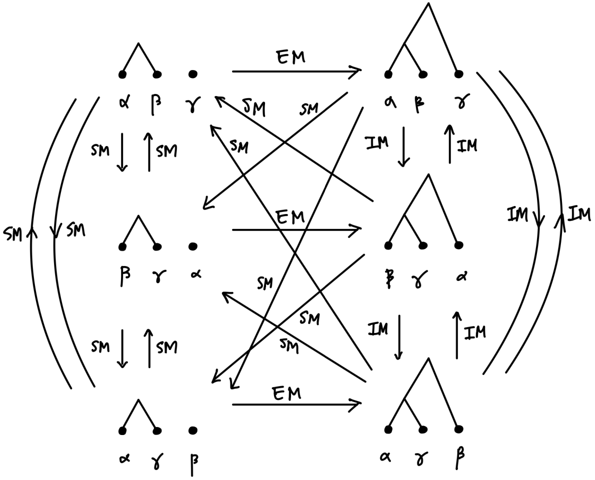
- here we consider the Merge operations $\mathfrak{M}_{S,S'}$ as forming a HAMC and we study its dynamical properties
- syntax formation and transformation/movement as a dynamical system
- Main Results:
  1. directed graph of Merge HAMC breaks down into finite graphs $\mathcal{G}_{n,A}$ on number of leaves, *sparse graphs* for large n
  2. each $\mathcal{G}_{n,A}$ strongly connected aperiodic: convergence to stationary distribution
  3. IM arrows give subgraphs $\mathcal{G}^{\rm IM}_{n,A}$ where each component has uniform stationary distribution
  4. full dynamics reduces to much smaller graph with vertices the partitions $\wp \in \mathcal{P}(n)$ (instead of forests)
  5. can weight dynamics by cost functions on edges
  6. full dynamics of Merge HAMC maximizes entropy (if weighted by cost function optimizes free energy)

Merge graph

- directed graph $\mathcal{G}$ with vertices the forests $F \in \mathfrak{F}_{\mathcal{SO}_0}$ with nonempty set of edges, $E(F) \neq \emptyset$
- an edge from vertex F to vertex F' if $\exists$ Merge operation (EM, minimal SM, or IM) with $F' = \mathfrak{M}_{S,S'}(F)$ or $F' = \mathfrak{M}_{S,T/S}(\mathfrak{M}_{S,1}(F))$

- minimal SM: $\mathfrak{M}_{S,S'}$ where both S, S' consist of a single leaf (at least one of them in a tree with more than one leaf)
- two possible choices (affect properties of Δ^d): allow cutting of both edges below same vertex or not, only relevant to SM case
- algebraic properties of Δ^d better if don't, dynamical system properties better if do (so we allow)
- condition $E(F) \neq \emptyset$ because every Merge operation creates some edge, so any F with $E(F) = \emptyset$ *transient* (does not contribute to long term dynamical behavior)

Example:



The Merge graph for $n = 3$ (excluding cutting of both edges below same vertex)

Decomposing the Merge graph

- Merge operations preserve the number of leaves of forests (using Δ^d)
- infinite graph $\mathcal{G}$ breaks into disjoint union of finite graphs

$$\mathcal{G} = \bigsqcup_{n \geq 2} \mathcal{G}_n$$

- case $\mathcal{G}_{n=1}$ trivial (no dynamics) so exclude
- Merge operations do not alter the labeling of the leaves so further decompose

$$\mathcal{G}_n = \bigsqcup_{A \in \mathrm{Sym}^n(\mathcal{SO}_0)} \mathcal{G}_{n,A}$$

set of labels $\{\alpha_\ell\}_{\ell \in L(F)} = A$, finite set of graph vertices $V(\mathcal{G}_{n,A}) = \mathfrak{F}_{A,n} \subset \mathfrak{F}_{\mathcal{SO}_0}$

counting vertices = counting forests (nonplanar, labelled leaves)

- count forests $F \in \mathfrak{F}_{A,n}$
- forest $F = T_1 \sqcup \cdots \sqcup T_r$, set of leaves

$$L(F) = L(T_1) \sqcup \cdots \sqcup L(T_r)$$

- set A of leaves labels partitioned among the $L(T_i)$
- $\mathcal{P}(n)$ set of partitions of n as a sum $n = k_1 + \cdots + k_r = \sum_i k_i$ of integers with $k_i \geq 1$
- $\wp = \{k_1, \ldots, k_r\}$ (with possible repetitions among the k_i)
- or with multiplicities a_i written explicitly (and no repetitions among the k_i)

$$\wp = \{\underbrace{k_1, \ldots, k_1}_{a_1}, \ldots, \underbrace{k_r, \ldots, k_r}_{a_r}\}$$

with $n = \underbrace{k_1 + \cdots + k_1}_{a_1\text{-times}} + \cdots + \underbrace{k_r + \cdots + k_r}_{a_r\text{-times}} = \sum_i a_i k_i$

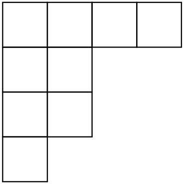
- partitions with at least one of the $k_i \geq 2$

$$\mathcal{P}'(n) := \mathcal{P}(n) \smallsetminus \{\wp_{1^n}\}$$

Partitions and Young diagrams

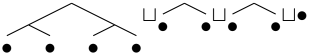
- partitions $\wp = \{k_1, \ldots, k_r\} \in \mathcal{P}(n)$ can be represented as Young diagrams: n boxes arranged in r rows of lengths k_i (non-decreasing order)

$$\wp = \{4, 2, 2, 1\} \in \mathcal{P}(9) \Leftrightarrow D(\wp) =$$

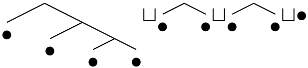


- projection workspaces to partitions: $\wp = \{4, 2, 2, 1\} = p(F)$

$F =$



or $F =$



some counting functions

- partition

$$\wp = \{\underbrace{k_1, \ldots, k_1}_{a_1}, \ldots, \underbrace{k_r, \ldots, k_r}_{a_r}\} \quad \text{with} \quad n = \underbrace{k_1 + \cdots + k_1}_{a_1\text{-times}} + \cdots + \underbrace{k_r + \cdots + k_r}_{a_r\text{-times}}$$

- multinomial coefficient

$$\mu_{\wp,n} := \binom{n}{\underbrace{k_1, \ldots, k_1}_{a_1\text{-times}} \ldots, \underbrace{k_r, \ldots, k_r}_{a_r\text{-times}}} = \frac{n!}{(k_1!)^{a_1} \cdots (k_r!)^{a_r}}$$

- generalized multinomial coefficient

$$\Upsilon_{\wp,n} := \frac{1}{a_1! \cdots a_r!} \binom{n}{\underbrace{k_1, \ldots, k_1}_{a_1\text{-times}} \ldots, \underbrace{k_r, \ldots, k_r}_{a_r\text{-times}}} = \frac{1}{a_1! \cdots a_r!} \frac{n!}{(k_1!)^{a_1} \cdots (k_r!)^{a_r}}$$

- counting function

$$\Lambda_{\wp,n} := \frac{1}{a_1! \cdots a_r!} \binom{n}{\underbrace{k_1, \ldots, k_1}_{a_1\text{-times}} \ldots, \underbrace{k_r, \ldots, k_r}_{a_r\text{-times}}} \prod_{i=1}^{r} ((2k_i - 3)!!)^{a_i}$$

what do they count?

- tree topologies: number of non-planar full binary rooted trees with n *labelled* leaves

$$\#\mathfrak{T}_{A,n} = (2n-3)!!$$

- number of forests $F \in \mathfrak{F}_{A,n}$ with fixed partition $p(F) = \wp$

$$\#\{F \in \mathfrak{F}_{A,n} \,|\, p(F) = \wp\} = \Upsilon_{\wp,n} \prod_{i=1}^{r} ((2k_i - 3)!!)^{a_i} = \Lambda_{\wp,n}$$

- multinomial coefficient counts labels splitting between components
- normalization by $a_1! \cdots a_r!$ symmetry of same-length rows of partition
- number of vertices in Merge graphs $V(\mathcal{G}_{n,A}) = \mathfrak{F}_{A,n}$

$$\Lambda_n := \#\mathfrak{F}_{A,n} = \sum_{\wp=\{k_1,\ldots,k_r\}\in\mathcal{P}'(n)} \Lambda_{\wp,n}$$

connectivity of Merge graphs

- number of outgoing External Merge (EM) arrows at vertex F

$$N^{\rm out}_{\rm EM}(F) = \begin{cases} \binom{r}{2} & r = b_0(F) \geq 2 \\ 0 & r = 1\,. \end{cases}$$

- number of incoming External Merge (EM) arrows at vertex F

$$N^{\rm in}_{\rm EM}(F) = c(F)$$

with $c(F) := \#\pi_{0,E}(F)$, for $\pi_{0,E}(F)$ set of connected components T of F with nontrivial set of edges

- number of outgoing Internal Merge (IM) arrows at vertex F

$$N_{\rm IM}^{\rm out}(F) = \sum_{i=1}^{c(F)} (2k_i - 4)$$

with $c(F) := \#\pi_{0,E}(F)$, number of components with edges, and $k_i = \#L(T_i)$ leaves in those components (nontrivial IM only for $k_i \geq 3$)

- number of incoming Internal Merge (IM) arrows at vertex F is same

$$N_{\rm IM}^{\rm in}(F) = \sum_{i=1}^{c(F)} (2k_i - 4) = N_{\rm IM}^{\rm out}(F)$$

- source of IM arrows, one for each accessible term $T_v \subset T_i$ so $2k_i - 2$ non-root vertices ($2k_i - 4$ excluding the two below the root that give id)
- target of IM arrow has one component $T_i = \mathfrak{M}(T_{i,1}, T_{i,2})$ coming from a $T_{i,2} \lhd_e T_{I,1}$ or $T_{i,1} \lhd_{e'} T_{I,2}$ with edge insertion $\lhd_e$ so choice of $(2k_{i,1} - 2) + (2k_{i,2} - 2) = 2k_i - 4$ edges

- number of outgoing minimal Sideward Merge (SM) arrows at vertex F

$$N^{\rm out}_{\rm SM,\, min}(F) - d''(F)$$

with $d(F) = \#\mathcal{C}(F)$ for $\mathcal{C}(F) \subset L(F) \times L(F)$ set of pairs (ℓ, ℓ') of leaves that form a cherry, $d'(F) = \#\mathcal{C}'(F)$ for $\mathcal{C}'(F) = \mathcal{C}(F) \cap \pi_0(F)$ set of cherries that are connected components and $d''(F) = d(F) - d'(F)$ (cherries contained in larger trees)

- number of incoming minimal Sideward Merge (SM) arrows at vertex F

$$N^{\rm in}_{\rm SM,\, min}(F) = 6d'(F)(n - c(F)) +$$

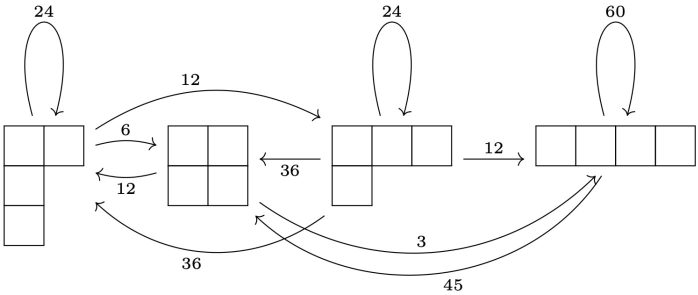
$$2 \sum_{i=1}^{d'(F)} \sum_{a \neq b \;\, a,b \neq i \;\, a,b=1}^{c(F)} (2k_a - 2)(2k_b - 2) +$$

$$2 \sum_{i=1}^{d'(F)} \sum_{j \neq i, \;\, j=1}^{c(F)} (2k_j - 2)(2k_j + 1 + n - c(F))$$

- seems complicated but... key observation: all these $N^{\rm in/out}$ depend only on $\wp = p(F)$ not on F

Use this fact to reduce the graph by projection: Example structure of the projection $\mathcal{G}_{A,4} \to \mathcal{G}_{\mathcal{P}'(4)}$

- edge multiplicities



- vertex multiplicities: $6, 3, 12, 15$
- $\mathcal{G}_{\mathcal{P}'(4)}$ is a small and manageable graph
- $\mathcal{G}_{A,4}$ has 270 edges and 36 vertices already difficult for PF computations

Sparsity

- unlike what the case $n = 3$ suggests, the Merge graphs $\mathcal{G}_{n,A}$ become rapidly increasingly sparse as n grows
- density of a graph (ratio of number edges to max number)

$$D(G) = \frac{\#E(G)}{\#V(G) \cdot (\#V(G) - 1)}$$

- for Merge graphs

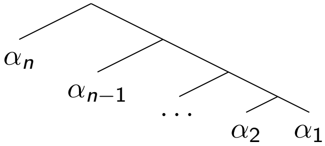
$$D(\mathcal{G}_{n,A}) \leq \frac{P(n)}{\Lambda_n} \longrightarrow_{n\to\infty} 0$$

with $P(n)$ polynomial of degree ≤ 3 (from connectivity counting of $N^{\rm in/out}$) while Λ_n of forest count grows at least exponentially

strong connectedness and aperiodicity

- directed graph G is strongly connected if there is a a *directed* path of edges between any ordered pair (v, w) of vertices in G
- graph G is periodic if all the cycle-lengths in the graph have a common gcd > 1 and is aperiodic if gcd $= 1$
- shown in [MCB] that Merge graphs are strongly connected (*need* at least minimal SM)
- also aperiodic: direct inspection for $n = 3$ (have cycles length 2 and 3); for $n \geq 4$ take vertex $F = T_n$ single *comb tree*

$T_n =$



- IM transformation that extracts α_1, repeat: cycle length $n - 1$
- also have $F = T_k \sqcup \alpha_{k+1} \sqcup \cdots \sqcup \alpha_n$ with comb T_k so also all cycle lengths $k - 1$ (so gcd 1)
- strong connectedness and aperiodicity will have important dynamical systems properties...

From the graph to the dynamical system

- two different approaches:
  1. random walk on the graph
  2. maximal entropy random walk (here same as the Hopf algebra Markov chain)
- random walk: take transition probabilites to be normalization

$$\mathcal{RW}_{F,F'} := \deg^{out}(F)^{-1}\, \mathcal{K}_{F,F'}$$

  with $\deg^{out}(F) = N^{\rm out}_{\rm EM}(F) + N^{\rm out}_{\rm IM}(F) + N^{\rm out}_{\rm SM\, min}(F)$
- drawback: meaning of rescaling? Merge action is Markovian is what the linguistics theory predicts, so it should be $\mathcal{K}$ itself the Markov chain, not $\mathcal{RW}$
- maximal entropy random walk (and HAMCs in general): find the basis in which $\mathcal{K}$ *itself* is a stochastic matrix (in that basis)

Perron-Frobenius theorem

- $\mathcal{K}^{(A,n)}_{F,F'}$ matrix with real non-negative entries
- associated graph $\mathcal{G}_{A,n}$ is strongly connected
- then $\exists$ Perron-Frobenius eigenvalue $\lambda = \lambda_{A,n} > 0$ (=spectral radius) simple with
- right-Perron-Frobenius eigenvector $\eta_F > 0$

$$\sum_{F'} \mathcal{K}^{(A,n)}_{F,F'} \, \eta(F') = \lambda \, \eta(F)$$

- also a left-Perron-Frobenius eigenvector $\psi_F > 0$ with

$$\sum_{F} \psi(F) \, \mathcal{K}^{(A,n)}_{F,F'} = \lambda \, \psi(F)$$

new basis from PF-eigenvector

- $\{\eta(F)^{-1}F\}$ (same combinatorial generators F, rescaled by PF)
- same linear map $\mathcal{K}^{(A,n)}$ written in this new basis gives matrix

$$\hat{\mathcal{K}}^{(A,n)}_{F,F'} = \lambda^{-1}\frac{\eta(F')}{\eta(F)}\mathcal{K}^{(A,n)}_{F,F'}$$

- this matrix is stochastic

$$\sum_{F'}\hat{\mathcal{K}}^{(A,n)}_{F,F'} = \lambda^{-1}\sum_{F'}\frac{\eta(F')}{\eta(F)}\mathcal{K}^{(A,n)}_{F,F'} = \lambda^{-1}\frac{\lambda\eta(F)}{\eta(F)} = 1$$

- so $\hat{\mathcal{K}}^{(A,n)}_{F,F'}$ is a Markov chain ("Merge is Markovian") stationary distribution

- first important information about the properties of a Markov chain as a dynamical system is the form of the stationary distribution (rules long term behavior of the dynamics)
- stationary distribution: probability distribution $\pi(F)$ on workspaces with

$$\sum_F \pi(F)\,\hat{\mathcal{K}}_{F,F'} = \pi(F')$$

general fact on stationary distribution:

- Markov chain $\hat{\mathcal{K}}_{x,y}$ obtained via PF rescaled basis from adjacency matrix $\mathcal{K}$ of a strongly connected graph
- probability distribution $\sum_x \pi(x)\,\hat{\mathcal{K}}_{x,y} = \pi(y)$ is given by

$$\pi(x) = \psi(x)\eta(x)$$

with η and ψ the right/left PF eigenvectors

$$\sum_x \psi(x)\eta(x)\lambda^{-1}\frac{\eta(y)}{\eta(x)}\mathcal{K}_{x,y} = \sum_x \psi(x)\mathcal{K}_{x,y}\lambda^{-1}\eta(y) = \psi(y)\eta(y)$$

Ergodicity for Markov chains

- another important property of dynamical systems is ergodicity (the long-term time average of a single trajectory equals the ensemble average across many trajectories)
- ergodic Markov chain $\mathcal{K}$: if
  1. stationary distribution is unique
  2. any initial distribution (prob on set of vertices of graph) converges under iterations of $\mathcal{K}$ to stationary distribution

ergodicity of Markov chain follows from strong connectedness and aperiodicity of graph

the Markov chain $\hat{\mathcal{K}}^{(A,n)}$ Merge action on workspaces is ergodic

Entropy of Markov chains

- directed graph $\mathcal{G}$ with adjacency matrix $\mathcal{K}_{x,y}$ and Markov chain $\hat{\mathcal{K}}_{x,y}$ in PF-rescaled basis
- consider all $\mathcal{S}$ arbitrary Markov chains with same underlying graph $\mathcal{G}$ (eg random walk or any stochastic matrix with nonzero entries where adjacency $\mathcal{K}_{x,y} = 1$)
- Entropy of a Markov chain (also called *entropy rate*):

$$\mathrm{Sh}(\mathcal{S}) = -\sum_x \pi(x) \sum_y \mathcal{S}(x,y) \log \mathcal{S}(x,y)$$

- maximum over all Markov chains $\mathcal{S}$ with same graph $\mathcal{G}$

$$S_{\mathrm{top}}(\mathcal{G}) := \mathrm{argmax}_{\mathcal{S}} Sh(\mathcal{S}) = \log \lambda_{\mathcal{G}}$$

$\lambda_{\mathcal{G}}$ Perron-Frobenius eigenvalue of adjacency matrix $\mathcal{K}$ of $\mathcal{G}$

Maximal Entropy Random Walk

- probability distribution on paths in $\mathcal{G}$ with Markov chain $\mathcal{S}$

$$\mathbb{P}_\ell(\gamma) = \pi(x_0)\,\mathcal{S}(x_0, x_1) \cdots \mathcal{S}(x_{\ell-1}, x_\ell)$$

- related to entropy by

$$\mathrm{Sh}(\mathcal{S}) = \lim_{\ell\to\infty} \frac{\mathrm{Sh}(\mathbb{P}_\ell)}{\ell}$$

- $\exists$ a special $\mathcal{S}_{\mathrm{MERW}}$ among Markov chain with same $\mathcal{G}$:
  1. entropy is maximal: $\mathrm{Sh}(\mathcal{S}_{\mathrm{MERW}}) = S_{\mathrm{top}}(\mathcal{G})$
  2. the path probability distribution induced by $\mathcal{S}_{\mathrm{MERW}}$ is uniform on all paths with the same length and endpoints
  3. $\mathcal{S}_{\mathrm{MERW}}$ is the PF-rescaled adjacency matrix of $\mathcal{G}$

$$\mathcal{S}_{\mathrm{MERW}}(x, y) = \hat{\mathcal{K}}(x, y) = \frac{1}{\lambda}\frac{\eta(y)}{\eta(x)}\mathcal{K}(x, y)$$

Weighted adjacency matrices: from Entropy to Free Energy

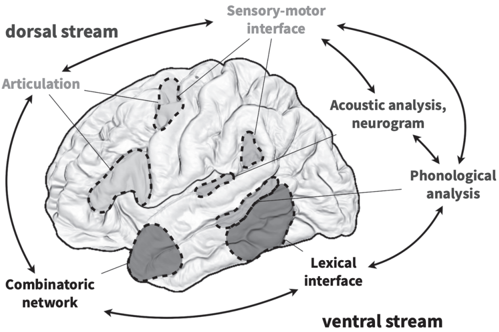
- different Merge operations (EM, IM, SM) have associated cost functions
- these cost functions give a weighted adjacency matrix of the Merge graph $\mathcal{G}_{n,A}$
- to be able to incorporate cost functions need to replace MERW result for entropy with another optimization
- Ruelle: free energy optimization

This Markov chain represents the *core computational process of syntax formation and transformation*, prior to embodiment in the grammar of any one particular language (the core "universal grammar")

Main question: possible neurocomputational realizations? (realizations of Markov chains in Hopfield networks, realizations through cross frequency phase synchronization on wavelets)

Interfaces: Externalization/Sensory-Motor; Sematics/Conceptual

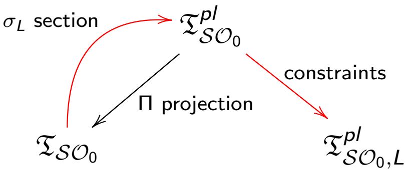
- in addition to the core computational mechanism: interfaces
- model motivated by dual stream (what/how)



- Externalization interface also incorporates (for theoretical linguistics) the specialization of the core computational mechanism to one particular language through positive examples during language acquisition (conditioning of the Markov chain + planarization of trees via word order rules)

Externalization:

key idea in a nutshell



- $\mathfrak{T}_{\mathcal{SO}_0} = \mathcal{SO}$ syntactic objects from free symmetric Merge
- $\mathfrak{T}^{pl}_{\mathcal{SO}_0}$ planar binary rooted trees (ordered leaves)
- $\Pi : \mathfrak{T}^{pl}_{\mathcal{SO}_0} \to \mathfrak{T}_{\mathcal{SO}_0}$ canonical projection (morphism of magmas)
- $\sigma_L : \mathfrak{T}_{\mathcal{SO}_0} \to \mathfrak{T}^{pl}_{\mathcal{SO}_0}$ non-canonical (language dependent) section of Π (not a morphism of magmas)
- constraints (syntactic parameters, theta-theory) projection to language-dependent $\mathfrak{T}^{pl}_{\mathcal{SO}_0,L}$

first step $\sigma_L : \mathfrak{T}_{\mathcal{SO}_0} \to \mathfrak{T}^{pl}_{\mathcal{SO}_0}$

- planarization σ_L via a language-dependent non-unique section of the projection
- only requirement on σ_L is compatibility with word-order parameters of given language L
- obtain in this way a planar tree $T^{\pi_L} = \sigma_L(T)$ for every syntactic object $T \in \mathcal{SO}$ no further restriction

Externalization second step: other constraints

- need further elimination of those objects $T^{\pi_L} \in \mathcal{SO}^{nc}$ that violate linguistic constraints (more syntactic parameters) of a particular language L (not word order related)
- other language dependent conditions: *theta-theory*, obligatory control, etc (eliminate trees that fail these)

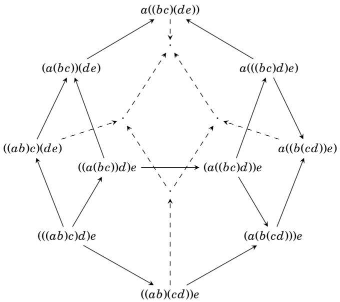
quotient map $\Pi_L : \mathfrak{T}^{pl}_{\mathcal{SO}_0} \to \mathfrak{T}^{pl,L}_{\mathcal{SO}_0}$ projection that eliminates what does not satisfy these further constraints

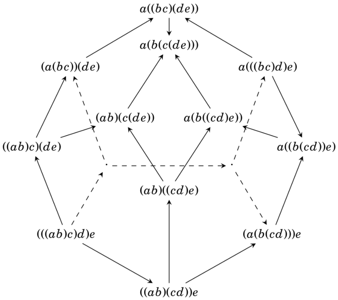
Additional aspects: associahedra (Stasheff)

- associahedron K_n convex polytope of dimension $n-2$, vertices are all the balanced parentheses insertions on an ordered string of n symbols (all planar binary rooted trees on n leaves)
- 1-dimensional associahedron K_3

$$((ab)c) \longleftrightarrow (a(bc))$$

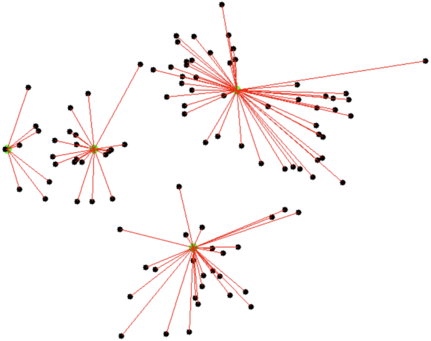
- 2-dimensional K_4 a pentagon
- 3-dimensional K_5:





Syntax-semantics interface overall picture:

- in a similar way, formulation of the generative process of syntax in terms of coproduct/grafting/product operation allows for consistent mapping of syntactic structures and substructures to semantic spaces
- this is called the syntax-semantics interface
- many different models of semantics but simplest is keeping track of proximity relations, agreement, similarity, co-occurrence (eg *vector space models*)



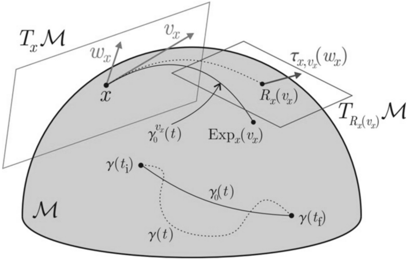
Syntax-Semantics interface: conceptual requirements

1. Autonomy of syntax
2. Syntax supports semantic interpretation
3. Semantic interpretation is, to a large extent, independent of externalization
4. Compositionality

- autonomy of syntax: Merge computational generative process of syntax independent of semantics
- syntax-first view: syntax-semantic interface proceeds *from* syntax *to* semantics
- two channels: from core Merge mechanism to Conceptual-Intentional system (syntax-semantics interface) and to Sensory-Motor system (externalization)
- compositionality: consistency across syntactic sub-structures

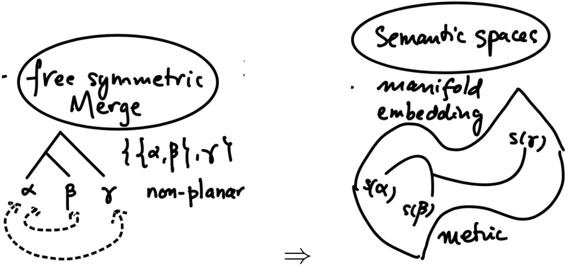
Geometry of semantic spaces

- language is a recent evolutionary step: requires a *small* modification: Merge as single evolutionary step (Berwick-Chomsky)
- everything else already evolved: semantic/conceptual spaces not specific to language (syntax specific to language)... e.g. conceptual manifolds in vision
- basic structures for a semantic space
  1. proximity measurement (topology/metric)
  2. interpolation (convexity)
  3. agreement/disagreement measurements

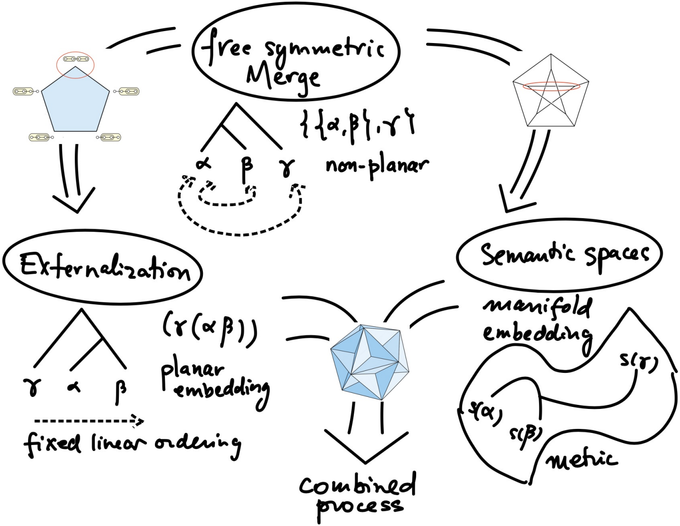


Semantic parsing of syntactic structures

- lexical items map to semantic space $s : \mathcal{SO}_0 \to \mathcal{S}$ (words in context)
- syntactic objects (trees) T also map to $\mathcal{S}$ using proximity and interpolation property of semantic space together with additional aspects (syntactic head, phases, labeling) on syntactic objects



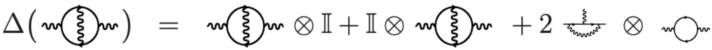
Externalization and syntax-semantics interface



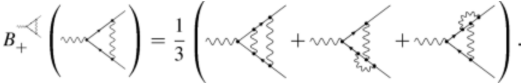
goal: giving a geometric model for the Conceptual/Intentional and Sensory/Motor channels and their interaction

algebraic approach: a lesson from theoretical physics

- same algebraic structure with Feynman diagrams instead of syntactic objects:
  - coproduct decomposition allows for extraction of meaningful physical values of Feynman integrals (consistently over substructures):



  - a similar coproduct/grafting/product operation describes the recursive solution of the quantum equations of motion (Dyson–Schwinger equations)



- *renormalization problem of quantum field theory* key idea ("Birkhoff factorization")

- start with an assignment of semantic values to lexical items $s : \mathcal{SO}_0 \to \mathcal{S}$
- want to extend this to an assignment for syntactic objects that is *consistent over substructures*
- *key idea*: incorporate consistency checking in an inductively defined function
- a simple mapping $\phi : \mathcal{SO} \to \mathcal{S}$ (which will include inconsistent cases) modified in a recursive way
- this recursive construction of consistency checking is known in physics as "Bogolyubov preparation"

- recursive construction of consistency checking needs two main ingredients:
  1. a way of extracting substructures T_v of a given structure T: checking separately T_v and T/T_v
  2. a way of separating out, in the target space $\mathcal{S}$, agreement and disagreement (or to filter by levels of agreement and disagreement)
- Note: the first is on the side of syntax, the second on the side of semantics
- the first is provided by the *coproduct* (Hopf algebra structure); the second is some form of projection (known as Rota–Baxter structure)

consistency over substructures (an example)

- consider the sentences "France is a republic", "France is hexagonal" and "France is a hexagonal republic"
- first two have clear unproblematic semantic parsing, the third seems awkward
- syntactic object of the form $T = \{a, \{b, \{c, d\}\}\}$, with a, b, c, d the lexical items France, is, hexagonal, republic
- Hopf algebra coproduct produces terms of the form $T_v \otimes T/T_v$ including

$$c \otimes \{a, \{b, d\}\} \quad \text{and} \quad d \otimes \{a, \{b, c\}\}$$

  where quotient structures $\{a, \{b, d\}\}$ and $\{a, \{b, c\}\}$ are the two sentences "France is a republic" and "France is hexagonal" (uncontroversial semantic parsing)
- coproduct also also contains terms like

$$\{c, d\} \otimes \{a, b\}$$

  that track the precise location (the extracted term $\{c, d\}$ "hexagonal republic") where the assignment of semantic values runs into problems

Basic model of semantic parsing

- main constructions:
  1. semantic space $\mathcal{S}$ (topological/metric, interpolation/convexity): geodesically convex Riemannian manifold; mapping of syntactic objects (using head/phases)
  2. evaluations (of distances or agreement/disagreement): algebra or semiring $\mathcal{R}$ with some "threshold operation" R (Rota-Baxter structure)
  3. semantic probes: extract evaluations (comparative) from image of syntactic objects in semantic space

evaluations and thresholds (Rota–Baxter structures)

- Rota-Baxter algebra $(\mathcal{R}, R)$ of weight -1: commutative associative algebra $\mathcal{R}$, linear operator $R : \mathcal{R} \to \mathcal{R}$ with

$$R(a)R(b) = R(aR(b)) + R(R(a)b) - R(ab)$$

- effect: R and $1 - R$ spit $\mathcal{R}$ into *subalgebras* $\mathcal{R}_{\mp}$
- Rota-Baxter semiring $(\mathcal{R}, R)$

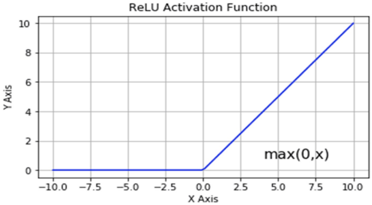
$$R(a) \odot R(b) = R(a \odot R(b)) \boxdot R(R(a) \odot b) \boxdot R(a \odot b) \quad \text{(weight } +1)$$

$$R(a) \odot R(b) \boxdot R(a \odot b) = R(a \odot R(b)) \boxdot R(R(a) \odot b) \quad \text{(weight } -1)$$

Meaning of this Rota–Baxter identity? it's the behavior of a "threshold operator"

Example: Max-plus (tropical) semiring and ReLU

- max-plus semiring (tropical semiring) $\mathcal{R} = (\mathbb{R} \cup \{-\infty\}, \max, +)$



- ReLU operator $R : x \mapsto x^+ = \max\{x, 0\}$ is a Rota–Baxter operator of weight $+1$ on $\mathcal{R}$

	$x \leq 0,\ y \leq 0$	$x \geq 0,\ y \leq 0$	$x \leq 0,\ y \geq 0$	$x \geq 0,\ y \geq 0$
$x^+ + y^+$	0	x	y	$x+y$
$(x^+ + y)^+$	0	$\begin{cases} x+y & x+y \geq 0 \\ 0 & x+y \leq 0 \end{cases}$	y	$x+y$
$(x + y^+)^+$	0	x	$\begin{cases} x+y & x+y \geq 0 \\ 0 & x+y \leq 0 \end{cases}$	$x+y$
$(x+y)^+$	0	$\begin{cases} x+y & x+y \geq 0 \\ 0 & x+y \leq 0 \end{cases}$	$\begin{cases} x+y & x+y \geq 0 \\ 0 & x+y \leq 0 \end{cases}$	$x+y$
max	0	x	y	$x+y$

$$x^+ + y^+ = \max\{(x^+ + y)^+, (x + y^+)^+, (x+y)^+\}$$

Example: Viterbi (probabilistic) semiring and threshold

- Viterbi semiring $\mathcal{P} = ([0,1], \max, \cdot, 0, 1)$
- threshold operators c_λ with $0 \leq \lambda \leq 1$

$$c_\lambda(x) = \begin{cases} x & x < \lambda \\ 1 & x \geq \lambda \end{cases}$$

- Rota–Baxter operators of weight -1

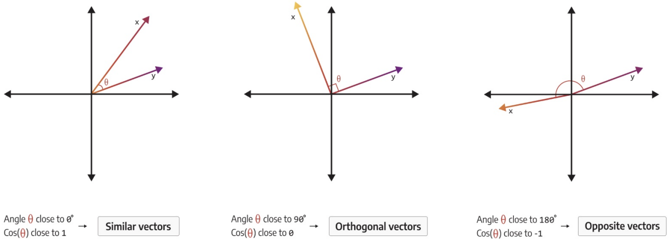
Example: Boolean semiring

- Boolean semiring $\mathcal{B} = (\{0,1\}, \vee, \wedge) = (\{0,1\}, \max, \cdot)$
- identity as Rota–Baxter operator of weight -1 (no room for more subtle thresholds in this case)

semantic probes

- semantic space $\mathcal{S}$ has *probes* by functions $\Upsilon : \mathcal{S} \to \mathbb{R}$ checking degree of agreement or disagreement with a particular semantic hypothesis
- eg a chosen vector v_Υ in vector space models

$$\Upsilon(s) = \langle s, v_\Upsilon \rangle$$



inner product $\langle s, v_\Upsilon \rangle$ with a fixed probe vector v_Υ detects agreement/disagreement (can normalize to cosine similarity)

consistency over substructures

- start with $\phi : \mathcal{H} \to \mathcal{R}$ where $\mathcal{H}$ Hopf and $\mathcal{R}$ Rota-Baxter... but ϕ only morphism of commutative algebras (multiplicative: independent structures go to independent values)
- these maps are called *characters* of the Hopf algebra
- Note: target $\mathcal{R}$ does not have same kind of "computational" structure as source $\mathcal{H}$ (no coproduct operation), but has RB projection R
- Birkhoff factorization of ϕ into $\phi_\pm : \mathcal{H} \to \mathcal{R}_\pm$ (alg homom)

$$\phi = (\phi_- \circ S) \star \phi_+ \qquad \text{or} \qquad \phi_+ = \phi_- \star \phi$$

- product $\star$ is determined by coproduct of $\mathcal{H}$

$$(\phi_1 \star \phi_2)(x) = (\phi_1 \otimes \phi_2)\,\Delta(x)$$

ϕ_- part is detecting *where right/wrong agreement*: substructures where consistent/inconsistent parsing is located (for a given probe and evaluation)

- Birkhoff factorizations are constructed inductively

$$\phi_-(x) = -R(\phi(x) + \sum \phi_-(x')\phi(x''))$$

$$\phi_+(x) = (1 - R)(\phi(x) + \sum \phi_-(x')\phi(x''))$$

$\Delta(x) = 1 \otimes x + x \otimes 1 + \sum x' \otimes x''$ with x', x'' lower degree

- Bogolyubov preparation is the expression

$$\tilde{\phi}(x) := \phi(x) + \sum \phi_-(x')\phi(x'')$$

that inductively incorporates the search for possible inconsistencies in substructures

- semiring case

$$\phi_-(x) = R(\tilde{\phi}(x)) = R(\phi(x) \boxdot \phi_-(x') \odot \phi(x''))$$

$\phi_+(x) = \phi_- \boxdot \tilde{\phi}$ multiplicative for semiring product
$\phi_\pm(xy) = \phi_\pm(x) \odot \phi_\pm(y)$

tropical semiring and ReLU example

- vector space model of semantics $\mathcal{S}$ with probe vectors v_Υ
- character

$$\phi_{\Upsilon,s,h}(T) = \begin{cases} \Upsilon(s(T)) = \langle s(T), v_\Upsilon \rangle & T \in \mathrm{Dom}(h) \\ -\infty & \text{otherwise.} \end{cases}$$

- $\phi_{\Upsilon,s,h,-}(T)$ identifies maximum value
  $\phi_{\Upsilon,s,h}(F_{\underline{v}_N}) + \phi_{\Upsilon,s,h}(F_{\underline{v}_{N-1}}) + \cdots + \phi_{\Upsilon,s,h}(F_{\underline{v}_1}) + \phi_{\Upsilon,s,h}(T)$
  over all nested sequences with all $\phi_{\Upsilon,s,h}(F_{\underline{v}_i}) > 0$
- identifies where are *chains of substructures realizing maximum consistent agreement with the chosen probe*