Mathematical Structure of Syntactic Merge
An Algebraic Model for Generative Linguistics

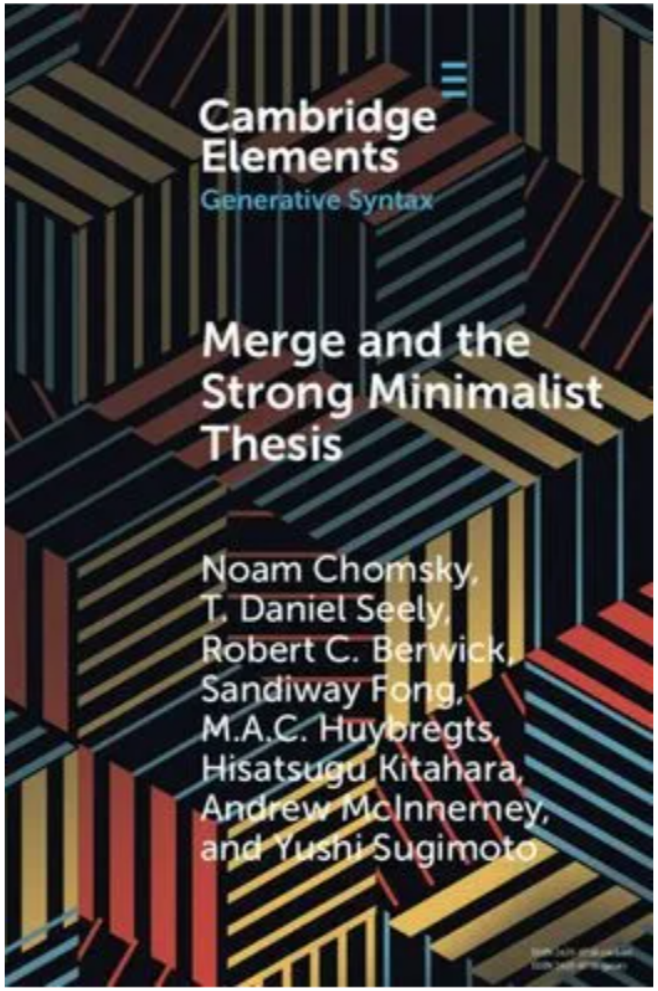
Matilde Marcolli, Noam Chomsky, Robert Berwick

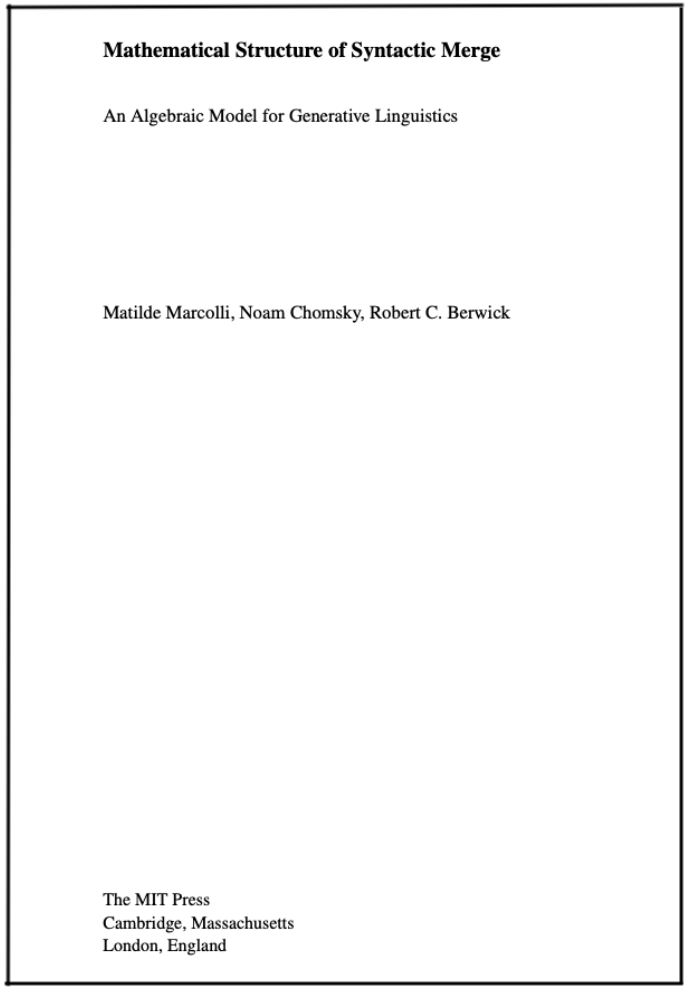
Lectures by Matilde Marcolli (Caltech)
MIT, Linguistics Department, December 2023

This series of lectures is based on the three papers

- Matilde Marcolli, Noam Chomsky, Robert C. Berwick, *Mathematical structure of syntactic Merge*, arXiv:2305.18278
- Matilde Marcolli, Robert C. Berwick, Noam Chomsky, *Old and new Minimalism: a Hopf algebra comparison*, arXiv:2306.10270
- Matilde Marcolli, Robert C. Berwick, Noam Chomsky, *Syntax-semantics interface: an algebraic model*, arXiv:2311.06189

Main Topic: Chomsky's Merge and the Strong Minimalist Thesis





Mathematical Structure of Syntactic Merge

An Algebraic Model for Generative Linguistics

Matilde Marcolli, Noam Chomsky, Robert C. Berwick

The MIT Press
Cambridge, Massachusetts
London, England

Key point: all aspects of this linguistic model have a mathematical formulation and properties of the model fall into place by *structural necessity* of the corresponding algebraic formalism

What does one gain from the use of mathematical formalism in Generative Linguistics?

- mathematics is the study of *structures*
- Generative Linguistics is *not* "becoming more mathematical": it always was
- mathematics is a powerful explanatory tool, because it is both highly constrained and highly flexible
- this is why it is the language of science (or as Galileo had it, the language in which the universe is written)
- it allows you to recognize when similar fundamental structures arise in different contexts: those are a sign of

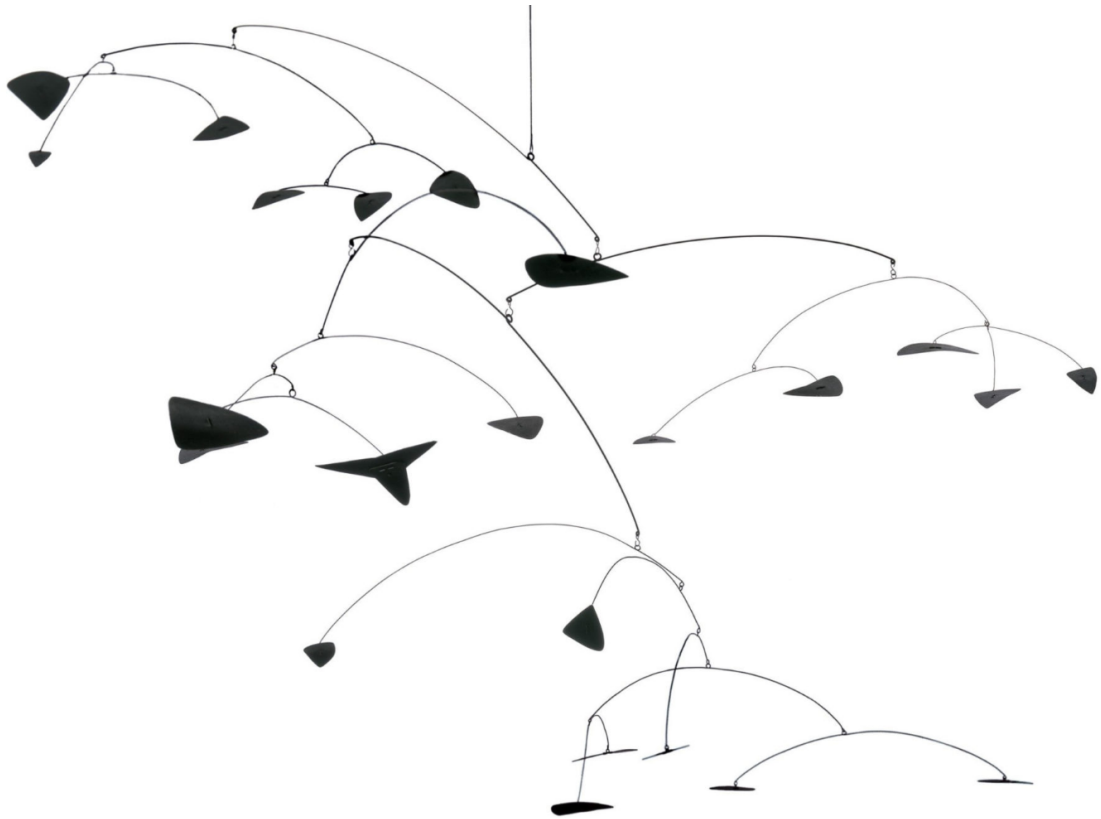
universal laws of nature

heuristic picture about the structure of language

- what language appears to look like

$$0022010021112000121220000200211\ldots$$

- what language actually looks like



syntactic objects

- (single) syntactic "structure building" operation
- "binary set formation"
- example: merging two lexical items like *the* and *apple* yields the (unordered, binary) set, $\{\textit{the}, \textit{apple}\}$
- recursive structure building: $\{\alpha, \beta\}$, $\{\gamma, \{\alpha, \beta\}\}$, $\{\alpha, \{\gamma, \beta\}\}$ etc
- syntactic objects: obtained by repeated applications of this binary set formation operation

N. Chomsky, *Some Puzzling Foundational Issues: The Reading Program*, Catalan Journal of Linguistics Special Issue (2019) 263–285 (and successive refs)

syntactic objects as free magma

this construction of syntactic objects is a well known mathematical structure: 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

- well known mathematical fact: 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

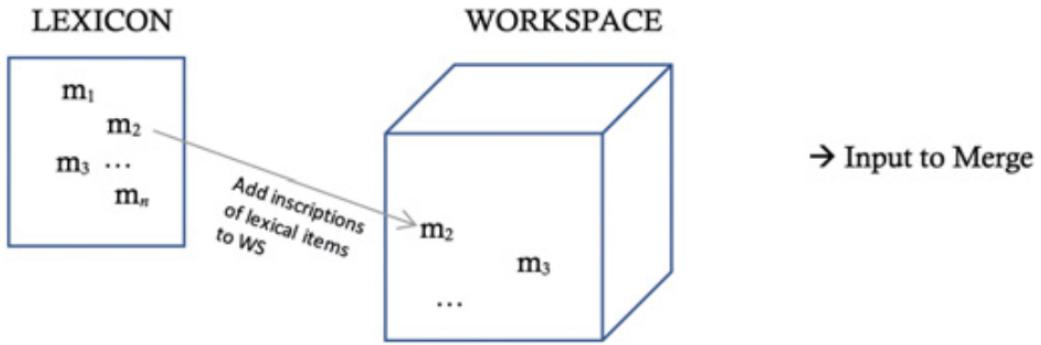
$$\mathrm{Magma}_{na,c}(\mathcal{SO}_0, \mathfrak{M}) = \mathfrak{T}_{\mathcal{SO}_0}$$

- 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* (leaves not ordered!)
- examples

$$\{\alpha, \beta\} = \overset{\alpha \quad \beta}{\frown} = \overset{\beta \quad \alpha}{\frown}$$

$$\{\alpha, \{\beta, \gamma\}\} = \overset{\alpha \quad \beta \quad \gamma}{\frown} = \overset{\gamma \quad \beta \quad \alpha}{\frown} = \overset{\alpha \quad \gamma \quad \beta}{\frown} = \overset{\beta \quad \gamma \quad \alpha}{\frown}$$

workspaces



- sentences built by repeated applications of Merge (this process is called a "derivation")
- starting from an initial set of lexical items, syntactic features
- the operations take place in a kind of "computational scratchpad," called a workspace (WS)
- workspace is the set of available computational resources (a *multiset* of syntactic objects)
- Merge transforms a workspace into a new workspace

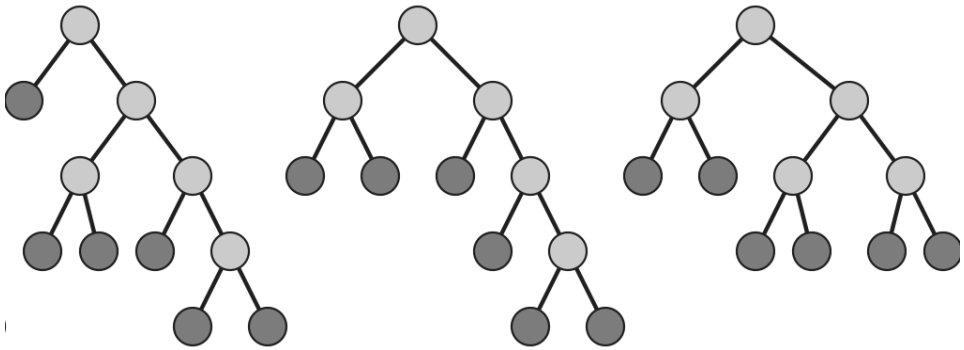
("Merge & SMT" §1)

workspaces as binary forests

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

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

- the trees T_i in F are the *connected components* (or just "components") of F (in linguistics they are called the "members" of the workspace)
- example of a workspace (binary forest) with three components (syntactic objects): order does not matter and trees are not planar



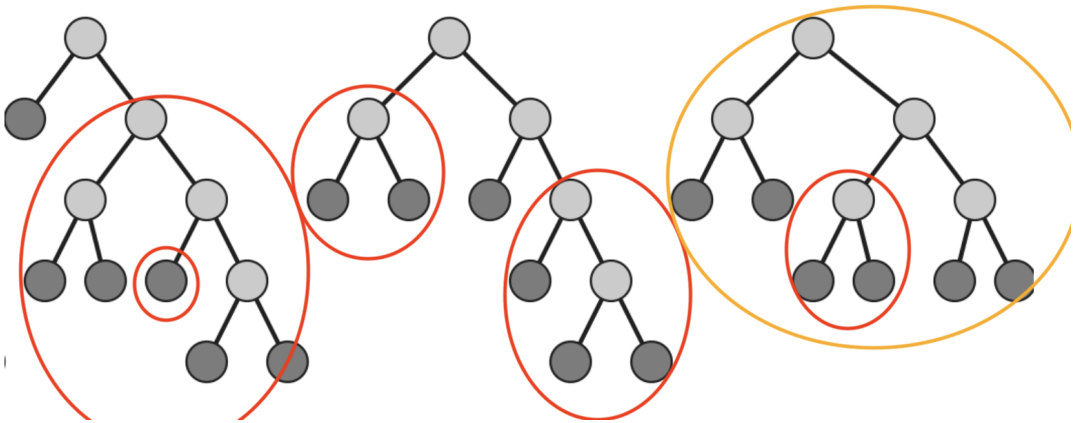
- set of workspaces = set of binary forests $\mathfrak{F}_{\mathcal{SO}_0}$
- Merge operations map the set $\mathfrak{F}_{\mathcal{SO}_0}$ to itself (transform workspaces into new workspaces)

accessible terms

- action of Merge on workspaces not only accounts for structure formation (as for building syntactic objects, External Merge) but also for *movement*, *transformations* (Internal Merge)
- for this type of action Merge needs to *extract substructures* from syntactic objects and copy them into the workspace
- linguistic justification for introducing workspaces
- the substructures required for Merge action are called accessible terms

accessible terms: mathematical definition

- *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:



red circled: examples of accessible terms of syntactic objects (and of workspace); yellow circled: example of accessible term of workspace but not accessible term of the syntactic object (full component)

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

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

- 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 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)

Algebraic structure of workspaces: product and coproduct

- The action of Merge on workspaces is built using a natural underlying algebraic structure
- product operation with two inputs and one output (assembling workspaces from syntactic objects and combining workspaces together)

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

  commutative associative product
- coproduct: disassembles workspaces into constituent parts
- *Note:* usually many *different ways* of decomposing objects (while one way of putting together pieces with $\sqcup$)
- so to define coproduct need to consider *combination of all possibilities*

coproduct

- form a vector space $\mathcal{V} = \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0})$ (over $\mathbb{Q}$ or $\mathbb{R}$): formal device to consider combinations of possibilities as (weighted) sums
- product extends by linearity $\sqcup : \mathcal{V} \otimes \mathcal{V} \to \mathcal{V}$ (two inputs one output)
- coproduct has one input (object to decompose) and two outputs (pieces of the decomposition)

$$\Delta : \mathcal{V} \to \mathcal{V} \otimes \mathcal{V} \qquad \Delta(x) = x \otimes 1 + 1 \otimes x + \sum x' \otimes x''$$

first terms of sum are trivial decompositions, remaining sum is over all different possible nontrivial decompositions $x' \otimes x''$

how to decompose workspaces?

- composition (product) and decomposition (coproduct) need some compatibility (on exchanging order of product/coproduct operations)
- an easy way: composition $\sqcup$ assembles workspaces from syntactic objects; decompose workspaces in the same way
- this would correspond to $\Delta(F) = \sqcup_a \Delta(T_a)$ with $\Delta(T_a) = T_a \otimes 1 + 1 \otimes T_a$ (individual syntactic objects T_a have no further decomposition, "primitive elements")

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

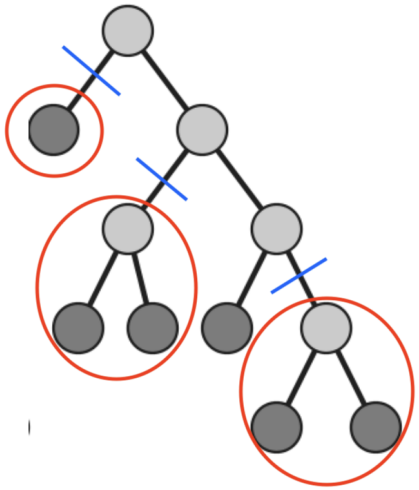
- but this is not a good choice: no accessible terms, no Internal Merge, no movement
- there is a better form of coproduct:
  - still $\Delta(F) = \sqcup_a \Delta(T_a)$
  - but now $\Delta(T_a)$ also has nontrivial decompositions that extract accessible terms

coproduct: admissible cuts

- Given a tree $T \in \mathfrak{T}_{\mathcal{SO}_0}$ consider forests $F_{\underline{v}} \subset T$

$$F_{\underline{v}} = T_{v_1} \sqcup \cdots \sqcup T_{v_n}$$

  with $T_{v_i} \subset T$ accessible terms (disjoint in T)
- such $F_{\underline{v}}$ corresponds to an admissible cut C of T with forest $\pi_C(T) = F_{\underline{v}}$ and remaining tree $\rho_C(T)$ attached to root
- example: subforest of accessible terms and corresponding admissible cut



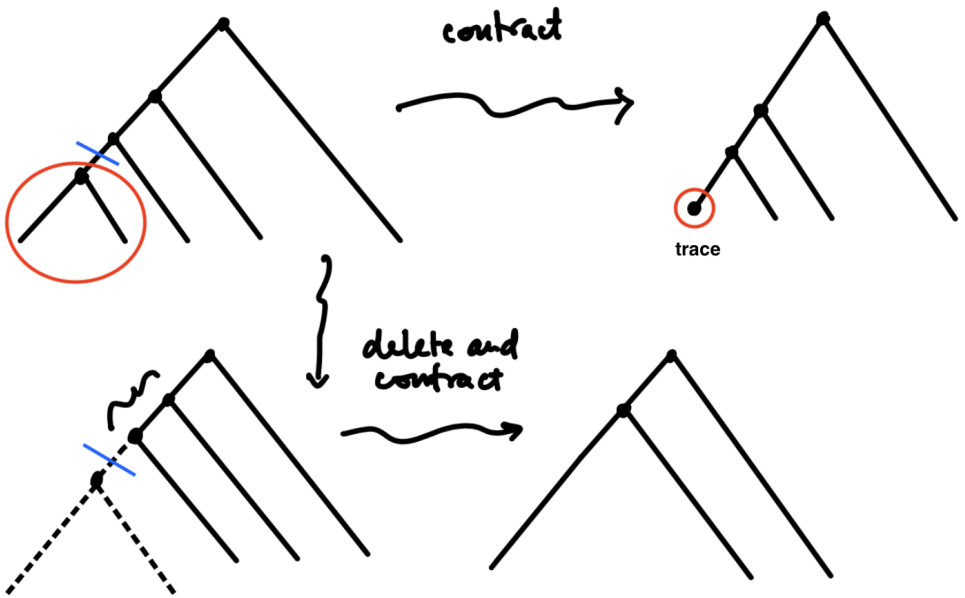
cut is admissible if no more than one cut on each path from root to leaves

(so what is cut off are accessible terms)

- coproduct with admissible cuts

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

- $\Delta : \mathcal{V} \to \mathcal{V} \otimes \mathcal{V}$ (left and right channels of coproduct output)
  - left channel: extracted substructure $F_{\underline{v}}$
  - right channel: what remains when substructure removed; $T/F_{\underline{v}}$ (quotient)
- two ways of thinking about removal of substructure $T/F_{\underline{v}}$
  1. removal and unique binary tree
  2. contraction of substructure to vertex



properties of coproduct

- why extraction of multiple accessible terms $F_{\underline{v}} = T_{v_1} \sqcup \cdots \sqcup T_{v_n}$ not just single one?

$$\Delta_{(2)}(T) = \sum_{v} T_v \otimes T/T_v$$

$$\Delta(T) = \sum_{n \geq 2} \Delta_{(n)}(T)$$

- what kind of compatibility between decompositions (coproduct Δ) and assembling (product $\sqcup$)?
- product is commutative associative (unit is formal empty tree): what are the properties of coproduct?

coproduct and coassociativity

- coproduct is coassociative (good behavior under iterations)

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

- $T \in \mathfrak{T}_{\mathcal{SO}_0}$, subforest $F_{\underline{v}} = T_{v_1} \sqcup \cdots \sqcup T_{v_n} \subset T$ quotient

$$T/F_{\underline{v}} = (\cdots (T/T_{v_1})/T_{v_2} \cdots)/T_{v_n}$$

- coproduct $\Delta(T) = \sum_{\underline{v}} F_{\underline{v}} \otimes (T/F_{\underline{v}})$ and on forests $F = \sqcup_a T_a$ with $\Delta(F) = \sqcup_a \Delta(T_a)$
- need parts of coproduct for coassociativity

$$\Delta(T) = \sum_{n \geq 2} \Delta_{(n)}(T) \quad \text{with} \quad \Delta_{(2)}(T) = \sum_{v} T_v \otimes T/T_v$$

  first terms will be needed for Merge action
- coproduct is not cocommutative (left/right channels of coproduct output have different roles)

- compatibility of product and coproduct

  (exchanging order of prod/coprod)

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

  τ flips middle two terms of $\mathcal{V} \otimes \mathcal{V} \otimes \mathcal{V} \otimes \mathcal{V}$

- grading: binary trees $T \in \mathfrak{T}_{\mathcal{SO}_0} = \mathcal{SO}$ grading by number of leaves (length of sentence)

$$\mathfrak{T}_{\mathcal{SO},n} = \{T \in \mathfrak{T}_{\mathcal{SO}_0} \,|\, \#L(T) = n\}$$

$$\mathfrak{F}_{\mathcal{SO}_0,n} = \{F \in \mathfrak{F}_{\mathcal{SO}_0} \,|\, \#L(F) = \sum_a \#L(T_a) = n\}$$

- in coproduct

$$\Delta(T) = T \otimes 1 + 1 \otimes T + \sum \pi_C(T) \otimes \rho_C(T)$$

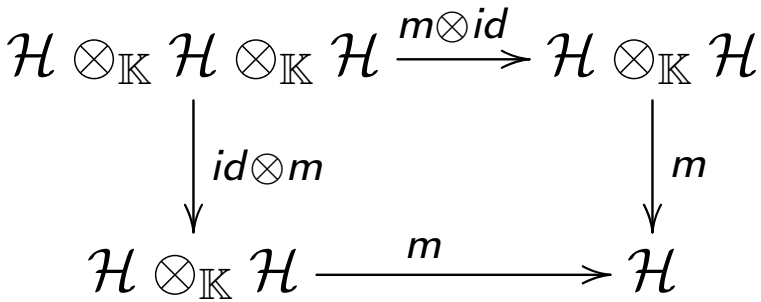
  all terms with nontrivial cut have lower degree than T

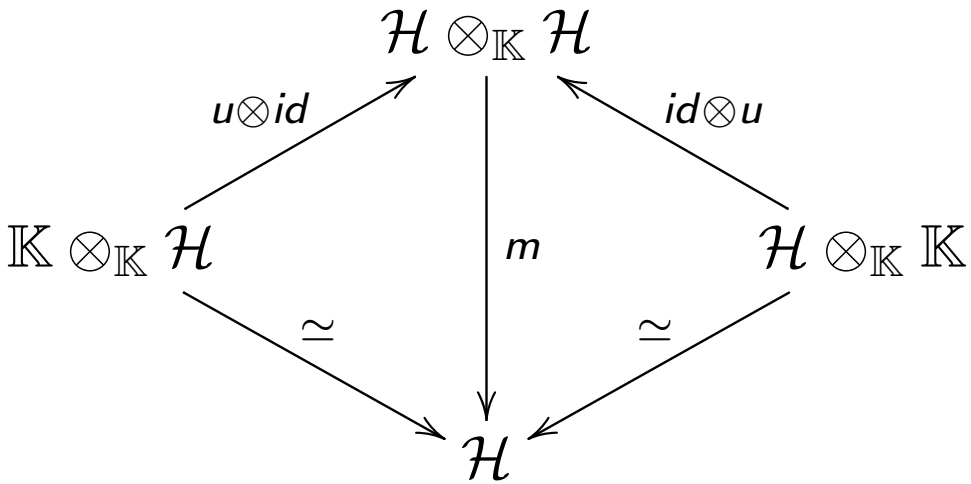
- $\mathcal{V} = \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}) = \oplus_{n \geq 0} \mathcal{V}_n$ with $\mathcal{V}_n = \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0,n})$ $n \geq 1$ and $\mathcal{V}_0 = \mathbb{Q}$

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
- 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

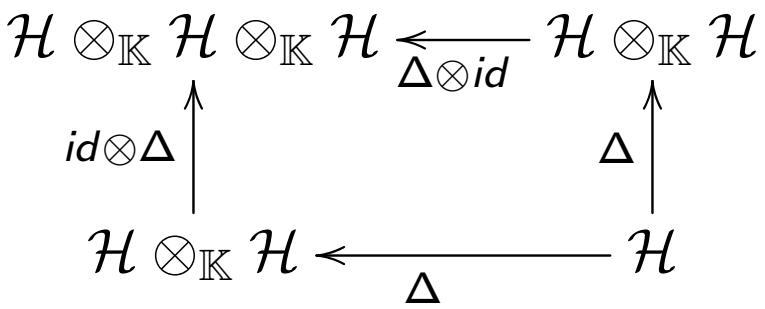
multiplication: associativity and unit

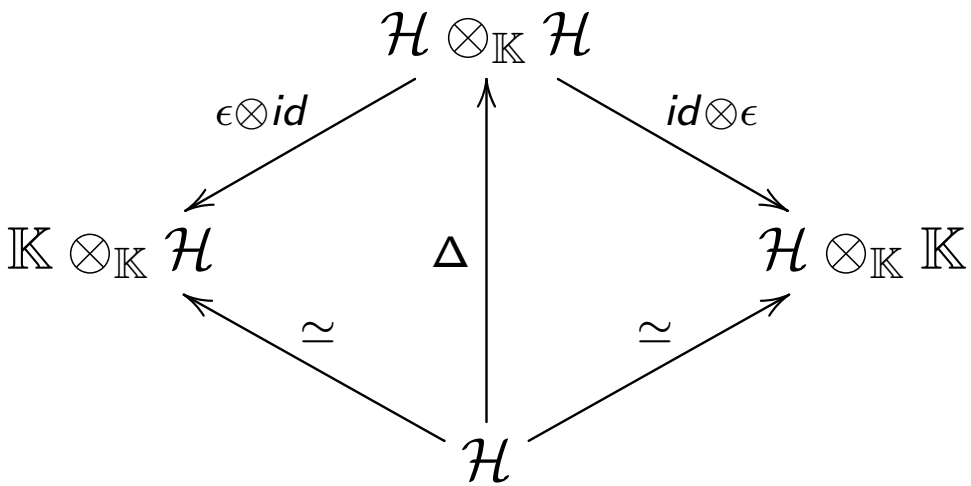




commutativity of these diagrams

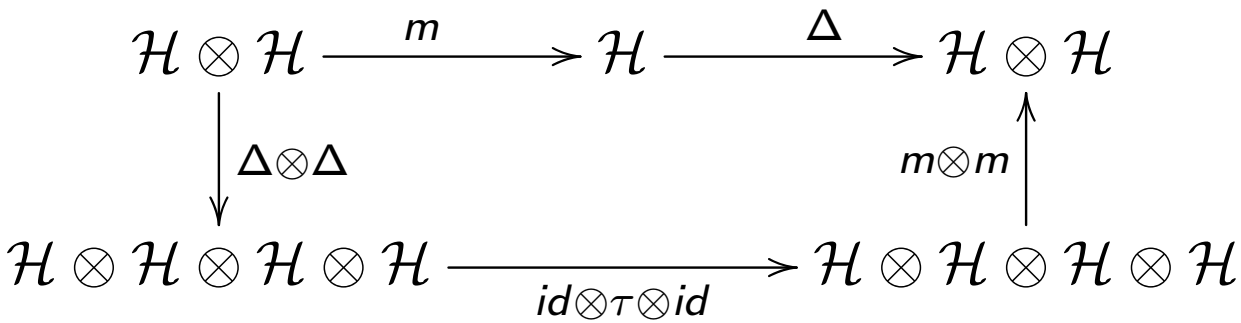
comultiplication: coassociativity and counit



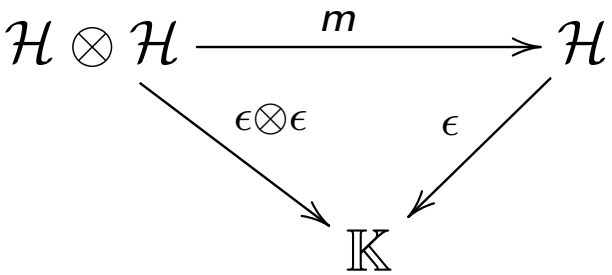


commutativity of these diagrams

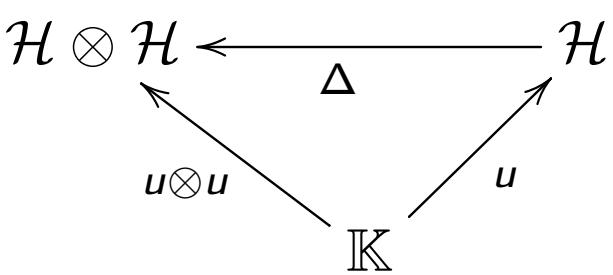
compatibility of product and coproduct
compatibility between product and coproduct:



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

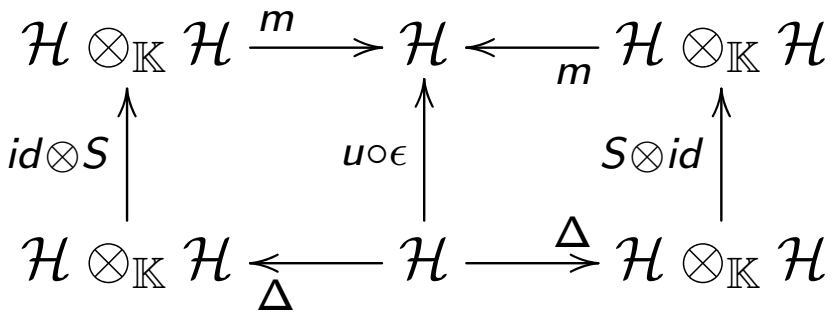


and



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

antipode: further compatibility: commutativity of diagram



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

Conclusion: workspaces form a Hopf algebra

action of Merge on workspaces

- Merge acts as operators $\mathfrak{M}_{S,S'}$ for pairs of syntactic objects $S, S' \in \mathcal{SO}$
- Given a workspace $F = T_1 \sqcup \cdots \sqcup T_n$ the operator $\mathfrak{M}_{S,S'}$ searches among the accessible terms of F for matching pairs to S, S'
- when a matching pair is located $S \simeq T_v$ and $S' \simeq T_w$ these two terms are merged into

$$\mathfrak{M}(T_v, T_w) = \overbrace{\quad T_v \qquad T_w \quad}$$

- this new syntactic object is added to the new workspace
- components T_i T_j of the old workspace that contained the extracted terms T_v and T_w are replaced by cancellation of (the deeper copies of) T_v and T_w
- all other components $\hat{F}_{i,j} = \sqcup_{a \neq i,j} T_a$ are left unchanged in the new workspace

- summarizing:

$$\mathfrak{M}_{S,S'} : F \mapsto F' = \mathfrak{M}(T_v, T_w) \sqcup T_i/T_v \sqcup T_j/T_w \sqcup \hat{F}_{i,j}$$

where $S \simeq T_v \subset T_i$ and $S' \simeq T_w \subset T_j$

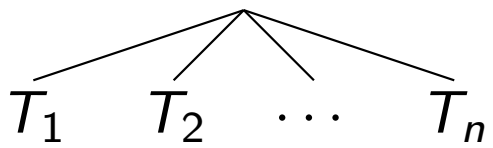
(written as $\mathrm{WS}' = \mathrm{Merge}(S, S', \mathrm{WS})$ in the notation of "Merge & SMT")

- Note: this action of Merge also contains some "unwanted" forms of Merge (sideward, countercyclic)

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

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

Cases of Merge (too many forms of Merge?)
The modified part of work space looks like

$$\mathfrak{M}(T_v, T_w) \sqcup (T_a/T_v) \sqcup (T_b/T_w)$$

Various cases $\mathfrak{M}(\alpha, \beta)$

1. $\alpha = T_i$ and $\beta = T_j$ with $T_i, T_j \in F$ and $i \neq j$;
2. $\alpha = T_i \in F$ and $\beta \in Acc(T_j)$ for some $T_j \in F$, with two sub-cases:
   a) $i = j$
   b) $i \neq j$
3. $\alpha \in Acc(T_i)$ and $\beta \in Acc(T_j)$ for some $T_i, T_j \in F$, with two sub-cases:
   a) $i = j$
   b) $i \neq j$

(1) External Merge; (2a) Internal Merge; (2b) and (3b) Sideward Merge; (3a) Countercyclic Merge

External Merge: an example (from "Merge & SMT")

- workspace $WS = [\text{eaten}, \{\text{the}, \text{apple}\}]$
- in our notation $F = T_1 \sqcup T_2$

$$T_2 = \overbrace{\beta \quad \gamma} \quad \text{with} \quad \beta = \text{the} \;\; \gamma = \text{apple}$$

T_1 the tree with a single vertex labeled by the lexical item α =eat(en)

- perform Merge with $\mathfrak{M}_{S,S'}$ with $S = \alpha \simeq T_1$ and $S' \simeq T_2$
- coproduct lists forests of accessible terms

$$\Delta(F) = F \otimes 1 + 1 \otimes F + \alpha \otimes T_2 + T_2 \otimes \alpha$$

$$+\alpha \sqcup \beta \otimes \gamma + \alpha \sqcup \gamma \otimes \beta + \beta \sqcup \gamma \otimes \alpha + \alpha \sqcup \beta \sqcup \gamma \otimes 1$$

$$\Delta(\text{eaten} \sqcup \overset{\text{the}}{\frown}\overset{\text{apple}}{}) = \text{eaten} \sqcup \overset{\text{the}}{\frown}\overset{\text{apple}}{} \otimes 1 + 1 \otimes \text{eaten} \sqcup \overset{\text{the}}{\frown}\overset{\text{apple}}{}$$

$$+ \text{eaten} \otimes \overset{\text{the}}{\frown}\overset{\text{apple}}{} + \overset{\text{the}}{\frown}\overset{\text{apple}}{} \otimes \text{eaten}$$

$$+ \text{ eaten } \sqcup \text{ the } \otimes \text{ apple } + \text{ eaten } \sqcup \text{ apple } \otimes \text{ the } + \text{ the } \sqcup \text{ apple } \otimes \text{ eaten}$$

$$\text{eaten } \sqcup \text{ the } \sqcup \text{ apple } \otimes 1$$

(this presents the complete list of all the possible extractions of accessible terms each accompanied by the corresponding residual term)

- $\delta_{S,S'}$ selects term with matching

$$\delta_{S,S'}(\alpha \sqcup T_2 \otimes 1) = \alpha \sqcup T_2 \otimes 1 = \text{eaten} \sqcup \begin{array}{c} \\ \text{the} \quad \text{apple} \end{array} \otimes 1$$

- grafting

$$(\mathcal{B} \otimes \mathrm{id})(\alpha \sqcup T_2 \otimes 1) = \begin{array}{c} \alpha \quad T_2 \end{array} \otimes 1$$

- 1 is unit of product

$$\begin{array}{c} \alpha \quad T_2 \end{array} \sqcup 1 = \begin{array}{c} \alpha \quad T_2 \end{array}$$

- so applying $\sqcup$ reassembles workspace to single syntactic object

$$\begin{array}{c} \alpha \quad T_2 \end{array} \sqcup 1 = \begin{array}{c} \alpha \quad T_2 \end{array} = \begin{array}{ccc} & & \\ \text{eaten} & & \\ & \text{the} \quad \text{apple} & \end{array}$$

Comment on Internal Merge

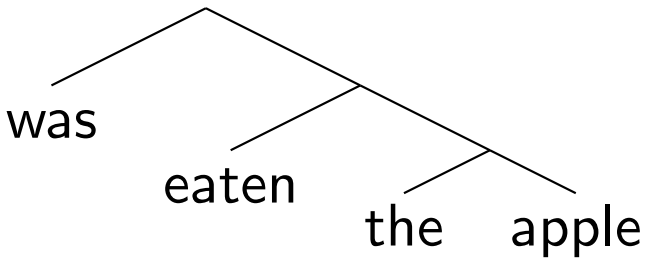
- We have a formal empty tree $\mathbf{1}$ that satisfies

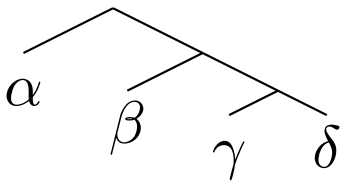
$$\mathfrak{M}(T, \mathbf{1}) = \mathfrak{M}(\mathbf{1}, T) = T$$

- Note: language does not incorporate arithmetic (but almost)
  - a unary Merge is needed in the first step of the successor function of Peano arithmetic $\emptyset \mapsto \{\emptyset\}$
  - then von Neumann description of the nonnegative integers is just (Internal) Merge $\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}, \{\emptyset, \{\emptyset, \{\emptyset\}\}\}, \ldots$
  - but in syntactic objects (as binary rooted trees) merging with the empty tree just leaves T unchanged
- If $\beta \in Acc(T_a)$ for a component T_a of workspace F the Merge $\mathfrak{M}_{\beta,\mathbf{1}}$ produces a new workspace with the component T_a replaced by $\beta \sqcup T_a/\beta$; External Merge on this workspace then gives $\mathfrak{M}(\beta, T_a/\beta)$ giving Internal Merge
- Note: we'll see that $\mathfrak{M}_{\beta,\mathbf{1}}$ does not "exist in isolation" only in composition as Internal Merge

Internal Merge: an example (from "Merge & SMT")

- check that Internal Merge described this way is *same* as usual linguistics description
- start with workspace $WS = [\{\text{was}, \{\text{eaten}, \{\text{the}, \text{apple}\}\}\}]$
- in our notation

$F = T =$ 

$=$ 

- perform Merge with $\mathfrak{M}_{S,S'}$ with $S = T$ and $S' = \overbrace{\gamma \quad \delta} = \overbrace{\text{the} \quad \text{apple}}$
- according to our description first act with $\mathfrak{M}_{S',1}$ and then with $\mathfrak{M}_{S/S',S'}$

- use notation

$$T_1 = \overbrace{\alpha \quad \beta} \quad \text{and} \quad T_2 = \overbrace{\gamma \quad \delta}$$

- $\delta_{\mathcal{S}',1}$ finds a match in the coproduct $\Delta(T)$: term

$$\overbrace{\gamma \quad \delta} \otimes \overbrace{\alpha \quad \beta}$$

$$T_1 = \overbrace{\alpha \quad \beta} = T/\overbrace{\gamma \quad \delta} = T/T_2$$

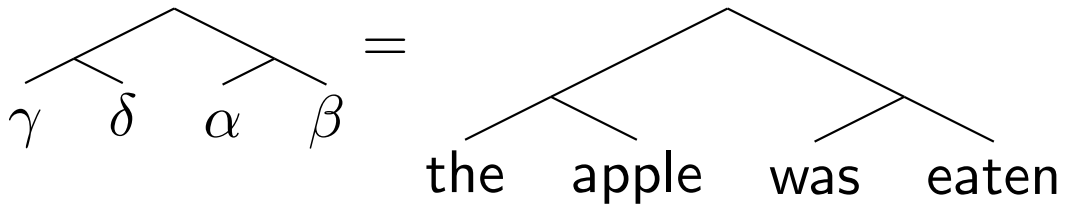
- read this coproduct term as

$$\overbrace{\gamma \quad \delta} \otimes \overbrace{\alpha \quad \beta} = \overbrace{\gamma \quad \delta} \sqcup 1 \otimes \overbrace{\alpha \quad \beta}$$

- then have $\mathcal{B}(T \sqcup 1) = T$
- so $\mathfrak{M}_{\mathcal{S}',1}$ produces an output

$$\overbrace{\gamma \quad \delta} \sqcup \overbrace{\alpha \quad \beta}$$

- then $\mathfrak{M}_{S/S',S'}$ produces from this the new workspace



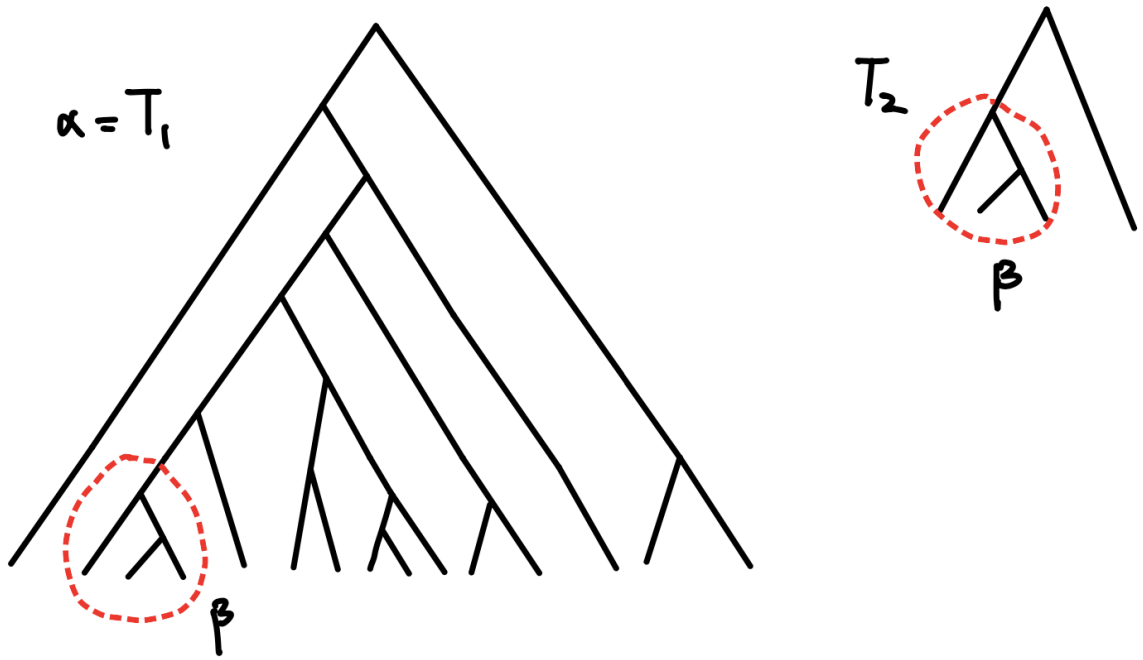
- this is the new worspace

$$WS' = [\{\{\text{the}, \text{apple}\}, \{\text{was}, \{\text{eaten}, \{\sout{\text{the, apple}}\}\}\}\}]$$

- so our description of Internal Merge matches what should be

Minimal Search

- Problem: on some accounts Sideward and Countercyclic Merge have undesirable linguistic properties
- in Minimalism these undesirable forms of Merge are eliminated on a principle of Minimal Search
- key idea: efficient search for matching terms (our $\delta_{S,S'}$) would first look for matching components of workspace (External Merge) then for a single component and an accessible term of the same (Internal Merge): everything else is a less efficient search
- How to formalize Minimal Search in our Hopf algebra setting?
- Minimal with respect to what? Cost function? Leading order term in an expansion?



Need a cost function / order degree justifying why finding the copy of β deep inside T_1 is "more efficient" than finding the copy of β high up into another component (regardless of number of components of F and of depth of T_1)

Minimal Search as Leading Order Extraction

- introduce degrees in the coproduct

$$\Delta^{(\epsilon,\eta)}(T) = \sum_{\underline{v}} \epsilon^{d_{\underline{v}}} F_{\underline{v}} \otimes \eta^{d_{\underline{v}}}(T/F_{\underline{v}})$$

  $\underline{v} = \{v_1, \ldots, v_n\}$ take $d_{\underline{v}} = d_{v_1} + \cdots + d_{v_n}$ with d_v the distance of a vertex v to the root of T

- Merge action correspondingly weighted

$$\mathfrak{M}^{\epsilon}_{S,S'} = \sqcup \circ (\mathfrak{M}^{\epsilon} \otimes \mathrm{id}) \circ \delta_{S,S'} \circ \Delta^{(\epsilon,\epsilon^{-1})}$$

$$\mathfrak{M}^{\epsilon}(\epsilon^{d}\alpha, \epsilon^{\ell}\beta) = \epsilon^{|d+\ell|}\, \mathfrak{M}(\alpha, \beta)$$

- Take the leading term for $\epsilon \to 0$
- only Merge derivations that survive in the limit are compositions of just External and Internal Merge

non-expansion of workspaces

"Resource Restriction and Minimal Yield" ("Third Factor principles")
Good operations should not increase the number of components of workspace (derivation converges) and not decrease number of accessible terms (no syntactic information gets destroyed)

- External and Internal Merge

	b_0	$\#Acc$	σ	$\hat{\sigma}$
External	-1	$+2$	$+1$	0
Internal	0	0	0	0

External: components decrease by one in $\mathfrak{M}(T, T')$ and two roots become new accessible terms
Internal: same components and because of how quotient defined (deletion)

$$\#Acc(T_v) + \#Acc(T/T_v) + 2 = \#Acc(T)$$

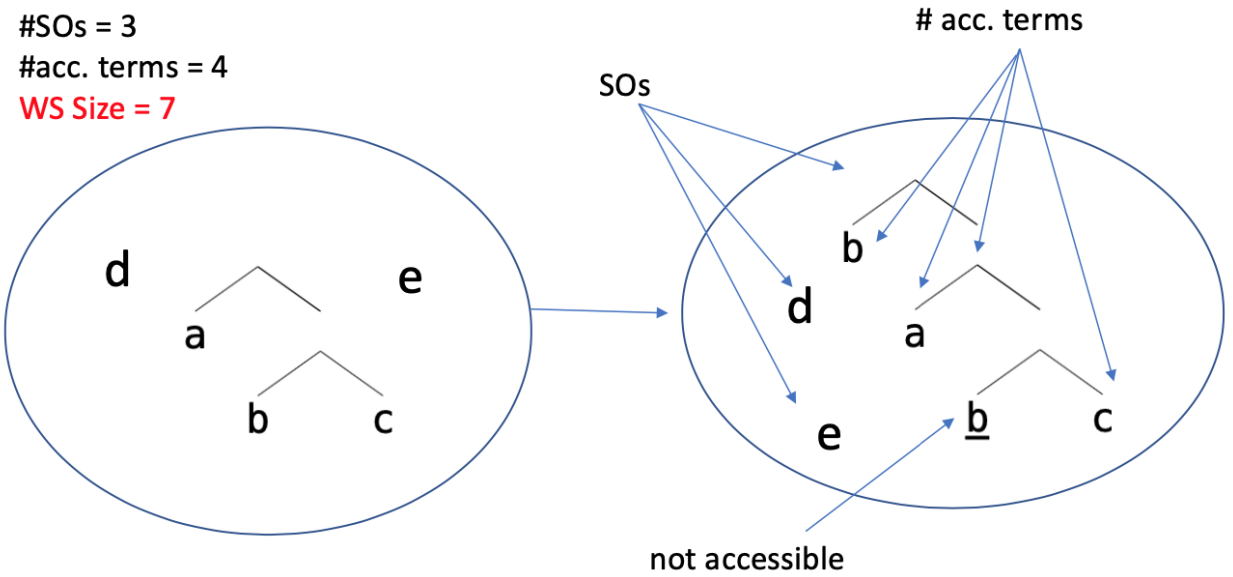
in T/T_v all the vertices of T_v and one additional vertex of T removed

- $\sigma = \#V = b_0 + \#\mathrm{Acc}$ and $\hat{\sigma} = b_0 + \#V$

This counting of size agrees with the counting for Internal Merge in S. Fong, R. Berwick, J. Ginsburg, *The combinatorics of merge and workspace right-sizing*, Evolinguistics Workshop, 2019

- Internal Merge of b and {a, {b, c}}

Minimal Search: exclude lower copies from set of accessible terms
- Copy of b is indicated as b



i.e. same way of taking "quotient tree"

- Merge $\mathfrak{M}_{\beta,\mathbf{1}}$ ruled out (except in compositions that form Internal Merge)

	b_0	$\#Acc$	σ	$\hat{\sigma}$
$\mathfrak{M}_{S,1}$	$+1$	-2	-1	0

- Sideward and Countercyclic Merge

	b_0	$\#Acc$	σ	$\hat{\sigma}$
Sideward (3b)	$+1$	0	$+1$	$+2$
Sideward (2b)	0	$+1$	$+1$	$+1$
Countercyclic (3a)	$+1$	$\#Acc(T_{a,w_a})$	$\sigma(T_{a,w_a})$	$\sigma(T_{a,w_a})+1$
Countercyclic (3a)	$+1$	-2	-1	0

Possible constraints on workspace size functions

- number of accessible terms non-decreasing ($\Delta\alpha \geq 0$)
- number of syntactic objects non-increasing ($\Delta b_0 \leq 0$)
- size of the workspace not decreasing and not increasing more than one ($0 \leq \Delta\sigma \leq 1$)
- $\hat{\sigma}$ is a conserved quantity ($\Delta\hat{\sigma} = 0$)

Internal and External Merge satisfy all of these; other cases:

	$\Delta b_0 \leq 0$	$\Delta\alpha \geq 0$	$0 \leq \Delta\sigma \leq 1$	$\Delta\hat{\sigma} = 0$
Sideward (3b)	N	Y	Y	N
Sideward (2b)	Y	Y	Y	N
Countercyclic (3a)	N	Y	N	N
Countercyclic (3a)	N	N	N	Y

Observation (by Riny Huijbregts): quotient by contraction also selects only Internal/External Merge by effect on accessible terms

Merge must be binary

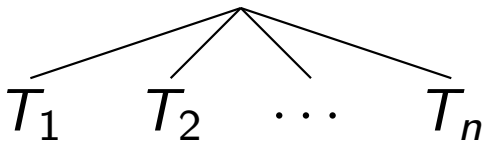
- Also an optimization: a Merge with any other $n \geq 3$ arity would both undergenerate and overgenerate with respect to binary Merge (observed by Riny Huijbregts)
- syntactic objects of a hypothetical n-ary Merge

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

- rooted n-ary trees (without planar structure)

$$\mathcal{SO}^{(n)} \simeq \mathfrak{T}^{(n)}_{\mathcal{SO}_0}$$

$$(T_1, \ldots, T_n) \mapsto \mathfrak{M}(T_1, \ldots, T_n) =$$

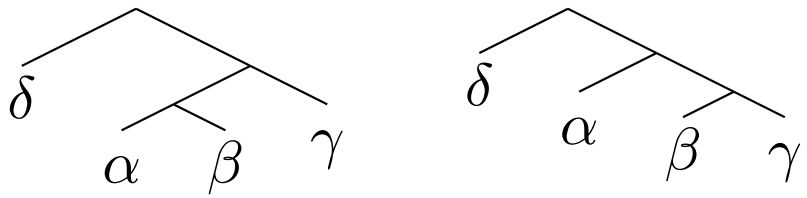


- by number of leaves

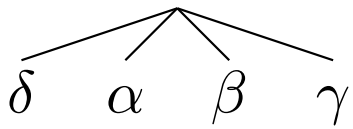
$$\mathcal{SO}^{(n)} = \bigsqcup_{k \geq 1} \mathcal{SO}^{(n)}_{k(n-1)+1}$$

two forms of undergeneration

1. achievable lengths only $\ell = k(n-1)+1$ for $k \geq 1$ (excludes examples like *it rains*)
2. ambiguities are not detected: example



(ambiguity of *I saw someone with a telescope*) become undetectable:



- undergeneration depends on syntactic objects, overgeneration depends on action on workspaces

action on workspaces of n-ary Merge and overgeneration

- workspaces are n-ary forests $F \in \mathfrak{F}^{(n)}_{\mathcal{SO}_0}$, same form of product and coproduct
- but for n-ary trees need to take quotients as contraction (so problem with labels reappears)
- Merge operations depending on an n-tuple of n-ary syntactic objects (with n-ary $\mathfrak{B}$)

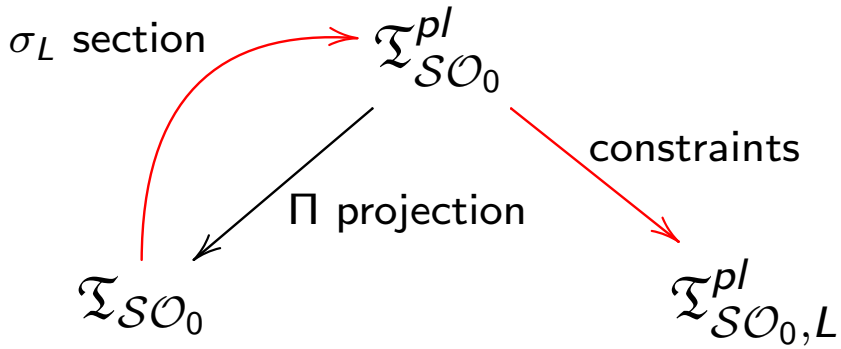
$$\mathfrak{M}_{S_1,\ldots,S_n} = \sqcup \circ (\mathfrak{B} \otimes \mathrm{id}) \circ \delta_{S_1,\ldots,S_n} \circ \Delta$$

- overgeneration: example (by Riny Huijbregts) with $n = 3$ and $F = \{\alpha, \beta, \gamma\} \sqcup \delta \sqcup \eta$ $S = (S_1, S_2, S_3)$ given by $S_1 = \alpha$, $S_2 = \beta$, and $S_3 = \{\alpha, \beta, \gamma\}$ gives new workspace $\{\alpha, \beta, \{\alpha, \beta, \gamma\}\} \sqcup \delta \sqcup \eta$ and further application with $S_1 = \delta$, $S_2 = \eta$, and $S_3 = \{\alpha, \beta, \gamma\}$ gives $\{\delta, \eta, \{\alpha, \beta, \gamma\}\}$ (responsible for examples like **peanuts monkeys children will throw*)

  (this excludes ternary Merge, unlike post-Externalization patterns like SVO)
- can count explicit amount of undergeneration and of overgeneration as a function of size of trees (number of leaves)

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 $\mathfrak{T}^{pl}_{\mathcal{SO}_0,L}$

Observation: morphisms of magmas

- free symmetric Merge: free commutative non-associative magma of syntactic objects

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

- also have a free non-commutative, non-associative magma on the same set $\mathcal{SO}_0$

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

- it generates the planar binary rooted trees with leaves labelled by $\mathcal{SO}_0$

$$\mathcal{SO}^{nc} \simeq \mathfrak{T}^{pl}_{\mathcal{SO}_0}$$

  write these as T^π (with T for abstract tree, π for choice of planar embedding)
- there is a *morphism of magmas* $\Pi : \mathcal{SO}^{nc} \to \mathcal{SO}$ that forgets the planar embedding (*canonical projection*) $T^\pi \mapsto T$

- Problem: the map $\Pi : \mathcal{SO}^{nc} \to \mathcal{SO}$ runs in the *opposite direction* to Externalization
- and... there is no morphism of magmas going the other way from $\mathcal{SO}$ to $\mathcal{SO}^{nc}$
- because since $(\mathcal{SO}, \mathfrak{M})$ is commutative it should map to a commutative sub-magma of $(\mathcal{SO}^{nc}, \mathfrak{M}^{nc})$
- but $(\mathcal{SO}^{nc}, \mathfrak{M}^{nc})$ does not have nontrivial commutative sub-magmas: if a nonempty planar tree T^{π} is in a commutative sub-magmas then $\mathfrak{M}^{nc}(T^{\pi}, T^{\pi})$ also is but this contradicts commutativity since

$$\mathfrak{M}^{nc}(T^{\pi}, \mathfrak{M}^{nc}(T^{\pi}, T^{\pi})) \neq \mathfrak{M}^{nc}(\mathfrak{M}^{nc}(T^{\pi}, T^{\pi}), T^{\pi})$$

so what is to be done?

Externalization first step: section σ_L of the projection Π

- can construct (non-canonically: dependent on choices) a non-unique *section*

$$\mathfrak{T}^{pl}_{\mathcal{SO}_0} \underset{\Pi\,\mathrm{proj}}{\overset{\sigma_{\mathcal{L}}}{\rightleftarrows}} \mathfrak{T}_{\mathcal{SO}_0}$$

- a *choice* of a point in each fiber $\Pi^{-1}(T)$ of the projection
- taking the one-way street Π in the opposite direction comes at a cost (loss of some good properties of the map):
  1. $\sigma_L : \mathfrak{T}_{\mathcal{SO}_0} \to \mathfrak{T}^{pl}_{\mathcal{SO}_0}$ is not a morphism of magmas
  2. $\sigma_L : \mathfrak{T}_{\mathcal{SO}_0} \to \mathfrak{T}^{pl}_{\mathcal{SO}_0}$ is not unique and depends on choices
- linguistic consequences:
  1. Merge can act either *before* Externalization (New Minimalism SMT) or *after* (on planar trees as in Old Minimalism) but *not both ways consistently*
  2. Externalization is *necessarily* language-dependent and not uniquely defined

summary so far: 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 $\sigma_{\mathcal{L}}$ is compatibility with word-order parameters of given language $\mathcal{L}$
- obtain in this way a planar tree $T^{\pi_{\mathcal{L}}} = \sigma_{\mathcal{L}}(T)$ for every syntactic object $T \in \mathcal{SO}$ with no further restriction (for instance no restriction on assignment of labels in $\mathcal{SO}_0$ at the leaves)

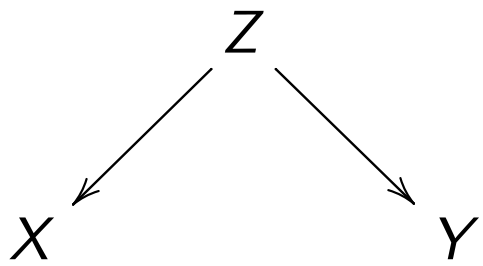
Externalization second step: other constraints

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

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

Externalization as correspondence

- two-step externalization: section of a projection followed by another projection ... correspondence
- in mathematics the simplest way of describing transformation is through functions $f : X \to Y$
  (single valued assignments $x \mapsto f(x)$)
- but sometimes functions are not the best way of going from X to Y and a better notion is correspondences



climbing one arrow "the wrong way" then going down the other one (includes the case of multivalued functions)

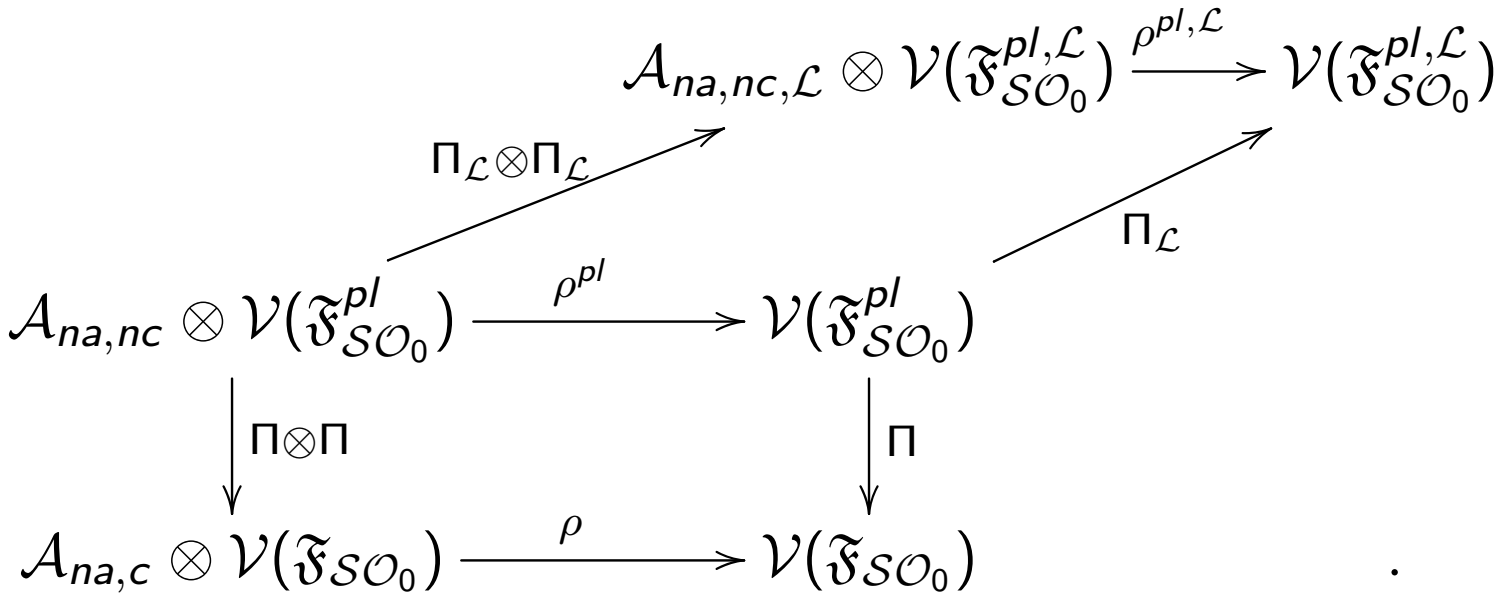
What happens to action on workspaces in Externalization?

- since all Merge operations happen with symmetric Merge before externalization it seems one cannot see at all this action after externalization (because magma structure not preserved by planarization $\sigma_{\mathcal{L}}$)
- but one can still see part of it
- $\mathcal{A}_{na,c} = (\mathcal{V}(\mathfrak{T}_{\mathcal{SO}_0}), \mathfrak{M})$ non-associative commutative algebra
- representations for a non-associative algebras $\mathcal{A}$ are just *linear maps* (not algebra homomorphisms) $\rho : \mathcal{A} \to \mathrm{End}(\mathcal{V})$, endomorphisms of vector space $\mathcal{V}$
- fix an argument of Merge: $\mathfrak{M}^T(T') := \mathfrak{M}(T, T')$
- then representation (in the above sense) from action on workspaces $F = \sqcup_a T_a$

$$\rho(T)(F) = \sqcup \circ (\mathfrak{M}^T \otimes 1) \circ \Delta(F) = \sqcup_a (\mathfrak{M}(T, T_{a,v}) \sqcup T_a / T_{a,v})$$

suffices to determine full action if known for all T

- the projection part is compatible with action of asymmetric Merge
- but section $\sigma_{\mathcal{L}}$ is not a magma morphism so only projection in the other direction is compatible with Merge action



- but climbing up the projection Π with the section $\sigma_{\mathcal{L}}$ leads to only a partially defined Merge action on the image
- indeed in old Minimalism, where Merge is after planarization, Merge is partially defined with specific conditions on domains

Comparison with Old Minimalism: Stabler's computational minimalism

- constructing I/E Merge directly on planar trees
- including labeling and domains for applicability based on labels
- Hopf algebras of planar binary rooted trees (Loday–Ronco Hopf algebra)
  1. no workspaces: only work with trees (not forests) makes compatible product and coproduct structure more difficult
  2. partially defined Merge operations (feature checking) introduces further layers of algebraic structure
- role of I/E Merge in terms of Loday–Ronco Hopf algebra
- different structures for Internal and External Merge (not coming from same operation)
  - Internal Merge determines system of right-ideal coideals (weak notion of quotient)
  - External Merge determines partially defined operated algebra

planar binary rooted trees

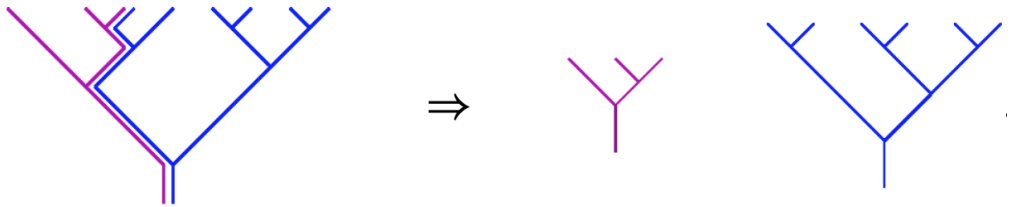
- $\mathcal{V} = \oplus_{k \geq 0} \mathcal{V}_k$ vector space spanned by planar binary rooted tree, $k =$ number of non-leaf vertices ($k+1$ leaves)
- now will have labeling of internal vertices also D_V set of labels
- for $d \in D_V$ grafting operator $\wedge_d$

$$\wedge_d : \mathcal{V} \otimes \mathcal{V} \to \mathcal{V}, \quad T_1 \otimes T_2 \mapsto T = T_1 \wedge_d T_2 = \quad \begin{array}{c} \mathrm{d} \\ T_1 \quad T_2 \end{array}$$

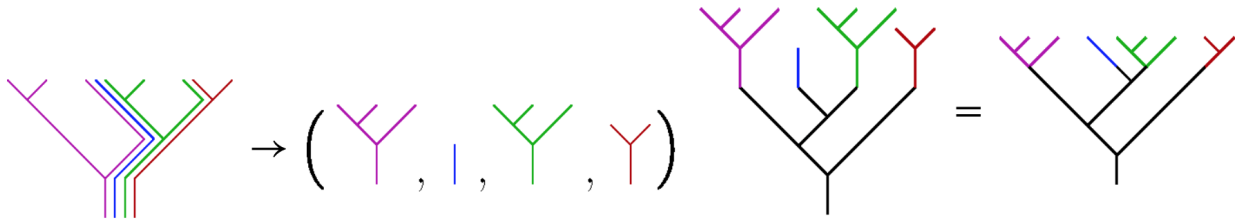
- $S \backslash T$ (S under T) grafting root of T to rightmost leaf of S
- T / S (S over T) grafting the root of T to leftmost leaf of S
- $T_1 \wedge_d T_2 = T_1 / S \backslash T_2$ with S planar binary tree with single non-leaf vertex decorated by $d \in D_V$
- each planar rooted tree is $T = T_\ell \wedge_d T_r$ (left and right subtrees below root)

Loday–Ronco Hopf algebra of planar binary rooted trees $\mathcal{H}_{LR}$

- product and coproduct defined inductively by degrees
- can also see graphically
  - coproduct sum $\Delta(T) = \sum T' \otimes T''$ over all decompositions of tree along paths from one of leaves to root



  - product $T \star T' = \sum_{(T_0,\ldots,T_n)} \gamma(T_0,\ldots,T_n;T')$ using same decompositions of first tree into as many pieces as leaves of second tree then grafting to leaves

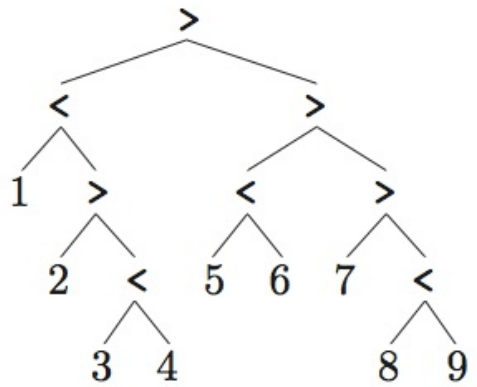


  - antipode inductively constructed by degrees

Stabler's Computational Minimalism

- example of old formulation of Minimalism

  (Stabler's formulation also known for relation to formal languages)
- planar binary rooted trees with labels:
  - leaves labelled by lexical items and syntactic features $X \in \{N, V, A, P, C, T, D, \ldots\}$
  - also "selector" features σX for *head* selecting a phrase XP
  - can also have labels that are strings (ordered finite sets) $\alpha = X_0 X_1 \cdots X_r$ of syntactic features
  - labels "licensor" ω and "licensee" $\bar{\omega}$
  - internal vertices labelled by $\{>, <\}$ following *head* of subtree



External and Internal Merge: combinatorial structure

- External Merge

$$\mathcal{E}(T_1 \otimes T_2) = \begin{cases} \bullet \wedge T_2 & T_1 = \bullet \\ T_2 \wedge T_1 & \text{otherwise,} \end{cases}$$

- Internal Merge

$$\mathcal{I}(T) = \pi_C(T) \wedge \rho_C(T)$$

C elementary admissible cut of T with $\rho_C(T)$ pruned tree containing root of T and $\pi_C(T)$ part severed by cut (elementary cut: tree not forest)

Note: admissible cuts are *not* the Loday-Ronco Hopf algebra coproduct now: there is a relation to the Loday-Ronco coproduct but is more involved

External Merge: domain

- $T[\alpha]$ for tree where head label starts with α
- domain of External Merge

$$\mathrm{Dom}(\mathcal{E}) = \mathrm{span}_{\mathbb{Q}}\{(T_1[\beta], T_2[\alpha]) \,|\, \beta = \sigma\alpha\}$$

- for $\alpha = X_0X_1\cdots X_r$ or $\alpha = \sigma X_0X_1\cdots X_r$ take $\hat{\alpha} = X_1\cdots X_r$
- External Merge

$$\mathcal{E}(T_1[\sigma\alpha], T_2[\alpha]) = \begin{cases} T_1[\widehat{\sigma\alpha}] \wedge_< T_2[\hat{\alpha}] & |T_1| = 1 \\ T_2[\hat{\alpha}] \wedge_> T_1[\widehat{\sigma\alpha}] & |T_1| > 1 \end{cases}$$

Internal Merge: domain

- tree $T[\alpha]$ where $\alpha = X_0 \cdots X_r$ or $\alpha = \sigma X_0 \cdots X_r$ or $\alpha = \omega X_0 \cdots X_r$ or $\alpha = \bar{\omega} X_0 \cdots X_r$
- domain of Internal Merge

$$\mathrm{Dom}(\mathcal{I}) = \mathrm{span}_{\mathbb{Q}} \left\{ T[\alpha] \,|\, \exists\, T_1[\beta] \subset T[\alpha], \text{with} \begin{array}{l} \beta = \bar{\omega} X_0 \hat{\beta}, \\ \alpha = \omega X_0 \hat{\alpha} \end{array} \right\}$$

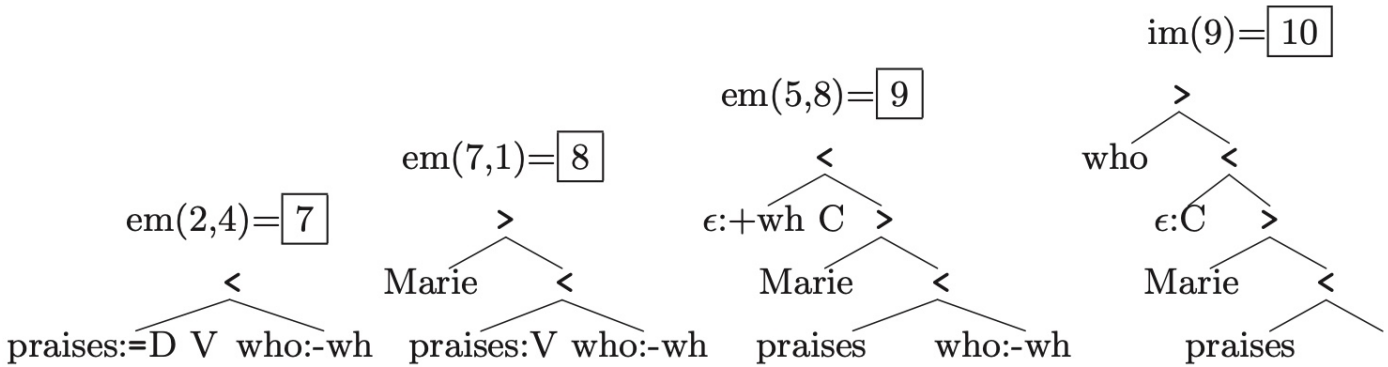
- Internal Merge (Stabler's notation and admissible cuts notation)

$$\mathcal{I}(T[\alpha]) = T_1^M[\hat{\beta}] \wedge_> T\{T_1[\beta]^M \to \emptyset\} = \pi_C(T) \wedge_> \rho_C(T)$$

Note: there are issues with EM and IM in this form (unlabelable exocentric constructions) producing $\{XP, YP\}$ results (observation by Riny Huijbregts)

Examples from Stabler

0	Pierre::D	who::D -wh	4
1	Marie::D	ϵ::=V +wh C	5
2	praises::=D =D V	knows::=C =D V	6
3	ϵ::=V C		



domains under iteration (compounding problem)

- iterations of the internal merge, $\mathcal{D}_{N+1} \subset \mathcal{D}_N$ with $\mathcal{D}_N := \mathrm{Dom}(\mathcal{I}^N)$
  1. N (complete) subtrees $T_1, \ldots, T_N$ in T
  2. $T_1^M, \ldots, T_N^M$ maximal projections of subtrees (also complete subtrees)
  3. subtrees T_i^M are disjoint.

$$\mathrm{Dom}(\mathcal{I}^N) = \Big\{ T[\alpha] \,\Big|\, \exists T_1[\beta^{(1)}], \ldots, T_N[\beta^{(N)}] \text{ with } \begin{array}{l} (1), (2), (3) \text{ are satisfied} \\ \beta_0^{(1)} = \bar{\omega} X_0, \ldots, \beta_0^{(N)} = \bar{\omega} X_{N-1} \\ \alpha = \omega X_0 \omega X_1 \cdots \omega X_{N-1} \cdots \end{array} \Big\}$$

$$\mathcal{I}^{\#C}(T[X]) = \bigwedge^{1+\#C} \Big(\pi_C(T)[\hat{\mathbb{Y}}] \; \rho_C(T)[\hat{X}^N] \Big)$$

$$\pi_C(T)[\hat{\mathbb{Y}}] = T_N^M[\hat{\beta}^{(N)}] \cdots T_1^M[\hat{\beta}^{(1)}]$$

label $[\hat{\alpha}^N]$ of tree $\rho_C(T)$: what remains of original label X after removing initial terms $\omega X_0 \omega X_1 \cdots \omega X_{N-1}$

feature checking complexity

- vertex labeling: strings $\alpha = X_0 X_1 \dots X_r$ of syntactic features
- set Σ_ℓ strings of length ℓ with $\#\Sigma_\ell = \mathfrak{s}^\ell$ with $\mathfrak{s}$ total number of syntactic features
- $\Sigma_\ell(a_0 \dots a_r)$ sequences starting with $a_0, \dots, a_r$
- counting formula for planar binary rooted trees with k internal vertices labeled by Σ_ℓ

$$d_{k,\ell} = (\#\Sigma_\ell)^k \frac{(2k)!}{k!(k+1)!} = \mathfrak{s}^{k\,\ell} \frac{(2k)!}{k!(k+1)!}$$

- trees with given label $\alpha \in \Sigma_\ell$ at root vertex and arbitrary labels elsewhere

$$d_{k,\ell}(\alpha) = (\#\Sigma_\ell)^{k-1} \frac{(2k)!}{k!(k+1)!} = \mathfrak{s}^{(k-1)\,\ell} \frac{(2k)!}{k!(k+1)!}$$

feature checking: one application of Internal Merge

- necessary condition defining domain

$$\mathcal{R}_{\mathcal{I}} := \{(\alpha, \beta) \in \Sigma_\ell \times \Sigma_\ell \,|\, \alpha = \omega X_0 \hat{\alpha}\,, \quad \beta = \bar{\omega} X_0 \hat{\beta}\}$$

- dimension of $\mathcal{D}_{1,k,\ell}(\alpha) = \mathcal{D}_1 \cap \mathcal{V}_{ling,k,\ell}(\alpha)$ for $\mathcal{D}_1 = \mathrm{Dom}(\mathcal{I})$

$$d_{\mathcal{I},k,\ell}(\alpha) = (\mathfrak{s}^{(k-1)\,\ell} - \mathfrak{s}^{(k-1)(\ell-2)}(\mathfrak{s}^2 - 1)^{k-1})\frac{(2k)!}{k!(k+1)!}$$

- because with root label $a_0 a_1 \ldots$ fixed counting all possible ways of having (at least) one of the vertices labeled by $\Sigma_\ell(a_0 a_1)$
- same as all the assignments of labels in Σ_ℓ not all of them in the complement $\Sigma_\ell \setminus \Sigma_\ell(a_0 a_1)$ and

$$\mathfrak{s}^\ell - \mathfrak{s}^{\ell-2} = \mathfrak{s}^{\ell-2}(\mathfrak{s}^2 - 1) = \#(\Sigma_\ell \setminus \Sigma_\ell(a_0 a_1))$$

feature checking: repeated applications of Internal Merge

- necessary condition defining domain

$$\mathcal{R}_{\mathcal{I}^N} = \left\{ (\alpha, \beta_1, \ldots, \beta_N) \;\middle|\; \begin{array}{l} \alpha = \alpha = \omega X_0 \omega X_1 \cdots \omega X_{N-1} \cdots \\ \beta_1 = \bar{\omega} X_0, \ldots \\ \cdots \\ \beta_N = \bar{\omega} X_{N-1} \end{array} \right\}$$

- some counting functions

$$S_N(a,b) := \binom{k-1}{N} b^N (a-b)^{k-1-N}$$

$$S_{N,k}(a,b) := S_N(a,b) + S_{N+1}(a,b) + \cdots + S_{k-1}(a,b) \leq a^{k-1}$$

- $S_N(a,b)$ counts number of label assignments to a set of $k-1$ points where N of them have labels in a set $B \subset A$ with $b = \#B$, $a = \#A$ and the remaining $k-1-N$ have labels in the complement $A \setminus B$

- dimension of $\mathcal{D}_{N,k,\ell}(\alpha) = \mathcal{D}_N \cap \mathcal{V}_{ling,k,\ell}(\alpha)$, with $D_N = \mathrm{Dom}(\mathcal{I}^N)$ is

$$d_{\mathcal{I},k,\ell,N}(\alpha) = S_{N,k}(\mathfrak{s}^\ell, \mathfrak{s}^{\ell-2N}(\mathfrak{s}^{2N} - 1))\frac{(2k)!}{k!(k+1)!}$$

- again use

$$\mathfrak{s}^\ell - \mathfrak{s}^{\ell-2N} = \mathfrak{s}^{\ell-2N}(\mathfrak{s}^{2N} - 1) = \#(\Sigma_\ell \setminus \Sigma_\ell(a_0 \dots a_{2N-1}))$$

- counting all the possible ways in which among the $k-1$ labels assigned to root vertices at least N are not in the complement of $\Sigma_\ell(a_0 \dots a_{2N-1})$

(analysis of complexity of computational implementations of Minimalism, see also Indurkya 2020, 2021; also Berwick succintness result compared to formal language description)

Internal Merge and coproduct and product in $\mathcal{H}_{LR}$

- coproduct $\Delta(T) = \sum T' \otimes T''$ decompositions: in each one side contains *head* of tree T (both if on boundary line of cut)
- if *head* of T and *head* of $\pi_C(T)$ same side then that side is in $\mathrm{Dom}(\mathcal{I})$
- so pieces of the coproduct are in $\mathrm{Dom}(\mathcal{I}) \otimes \mathcal{H}_{ling} + \mathcal{H}_{ling} \otimes \mathrm{Dom}(\mathcal{I})$
- other terms (different sides) are in $\mathcal{H}_{ling} \otimes \mathcal{H}_{ling}$
- T in $\mathrm{Dom}(\mathcal{I})$ and C elementary admissible determined by label condition; set of partitions

$$\mathcal{P}_{\mathcal{I}}(T) = \{ T = (T', T'') \mid \begin{array}{l} (h(T) \in T' \text{ and } h(\pi_C(T)) \in T') \text{ or} \\ (h(T) \in T'' \text{ and } h(\pi_C(T)) \in T'') \end{array} \}$$

- modify coproduct

$$\Delta_{\mathcal{I}}(T) := \sum_{(T',T'') \in \mathcal{P}_{\mathcal{I}}(T)} T' \otimes T''$$

and remains same outside of $\mathrm{Dom}(\mathcal{I})$

- with this modified coproduct $\mathrm{Dom}(\mathcal{I})$ is a *coideal* of the coalgebra $\mathcal{H}_{ling}$

$$\Delta_{\mathcal{I}}(\mathrm{Dom}(\mathcal{I})) \subset \mathrm{Dom}(\mathcal{I}) \otimes \mathcal{H}_{ling} + \mathcal{H}_{ling} \otimes \mathrm{Dom}(\mathcal{I})$$

- modified product $\star_{\mathcal{I}}$ on $\mathcal{H}_{ling}$: for trees T, T' where T' has $n+1$ leaves: decompositions where *head* of T in component grafted to *head* of T'

$$\mathcal{P}_{\mathcal{I}}(T, T') := \{(T_0, \ldots, T_n) \,|\, h(T) \text{ and } h(\pi_C(T)) \in T_{h(T')}\}$$

$$T \star_{\mathcal{I}} T' = \sum_{(T_0,\ldots,T_n)\in\mathcal{P}_{\mathcal{I}}(T,T')} \gamma(T_0, \ldots, T_n; T')$$

- $h(T \star_{\mathcal{I}} T') = h(T)$ as *head* of each $\gamma(T_0, \ldots, T_n; T')$ same as the *head* of T
- component $T_{h(T')}$ is in $\mathrm{Dom}(\mathcal{I})$ when $T \in \mathrm{Dom}(\mathcal{I})$ so $T \star_{\mathcal{I}} T'$ also in $\mathrm{Dom}(\mathcal{I})$
- $\mathrm{Dom}(\mathcal{I})$ right-ideal of algebra $(\mathcal{H}_{ling}, \star_{\mathcal{I}})$

$$\mathrm{Dom}(\mathcal{I}) \star_{\mathcal{I}} \mathcal{H}_{ling} \subset \mathrm{Dom}(\mathcal{I})$$

- not left-ideal

[planar trees so noncommutative product, and left and right ideals differ]

- form of Internal Merge $\mathcal{I}(T \star_{\mathcal{I}} T') =$

$$\sum_{(T_0,\ldots,T_n)\in\mathcal{P}_{\mathcal{I}}(T,T')} \pi_C(T_{h(T')}) \wedge_> \gamma(T_0,\ldots,\rho_C(T_{h(T')}),\ldots,T_n; T')$$

- internal merge $\mathcal{I}$ defines a right $(\mathcal{H}_{ling}, \star_{\mathcal{I}})$-module given by the cosets

$$\mathcal{M}_{\mathcal{I}} := \mathrm{Dom}(\mathcal{I})\backslash\mathcal{H}_{ling}$$

- combined with iteration of domains: $\mathcal{D}_{N+1}\backslash\mathcal{D}_N$ determines a coideal in the coalgebra $\mathcal{D}_{N+1}\backslash\mathcal{H}_{ling}$
- this gives a projective system of right-module coalgebras

$$\mathcal{M}_{\mathcal{I}^N} := \mathrm{Dom}(\mathcal{I}^N)\backslash\mathcal{H}_{ling}$$

- quotient right-module coalgebras or "generalized quotients" of Hopf algebras: suitable notion of quotients in the case of noncommutative Hopf algebras

Old External Merge and "operated algebras"

- context: Rota's "operated algebras" program (algebras together with linear operators satisfying polynomial constraints (eg Rota-Baxter ops, Leibniz rule, etc) ... version for binary operations
- $\wedge_\Omega$-algebra: algebra $(\mathcal{A}, \star)$ with binary operations $\wedge_\alpha$, $\alpha \in \Omega$

$$a \star b = a_1 \wedge_\alpha (a_2 \star b) + (a \star b_1) \wedge_\alpha b_2$$

where $a = a_1 \wedge_\alpha a_2$ and $b = b_1 \wedge_\alpha b_2$

- if also Hopf algebra: cocycle condition

$$\Delta(a \wedge_\alpha b) = (a \wedge_\alpha b) \otimes 1 + (\star \otimes \wedge_\alpha) \circ \tau(\Delta(a) \otimes \Delta(b))$$

- External Merge in Stabler's Minimalism is a cocycle $\wedge_\Omega$-algebra structure on the LR Hopf algebra
- Conclusion: the implementation of Merge at the level of planar trees introduces significant complications in algebraic structure compared to free symmetric Merge followed by Externalization (confirmed also by analysis of computational implementations, Indurkya, Berwick...)

head-driven planarization proposals

e.g. Kayne's Linear Correspondence Axiom (LCA)

- vertex v in T c-commands another vertex w if neither dominates the other and the lowest vertex that dominates v also dominates w; asymmetrically c-commands if c-commands and not sister vertices
- LCA procedure: assign ordering: v precedes v' iff
  1. v asymmetrically c-commands v' or
  2. a maximal projection w dominating v (T_w not in any larger tree with same *head*) c-commands v'
- difficulty: cannot have σ^{LCA} defined on all of $\mathcal{SO}$ (even if just a section that is not a morphism of magmas as in externalization)
- to see this: we can abstract the formal properties of the syntactic *head*

- *head function* $h_T : V^{int}(T) \to L(T)$ from non-leaf vertices to leaves
- if $T_v \subseteq T_w$ and $h_T(w) \in L(T_v) \subseteq L(T_w)$, then $h_T(w) = h_T(v)$
- write $h(T)$ for value of h_T at root of T
- for a pair (T, h_T) and $(T', h_{T'})$, there are two possible $h_{\mathfrak{M}(T,T')}$: marking one or the other of the two edges attached to new root
- i.e. choices of $h(\mathfrak{M}(T, T')) = h(T)$ or $h(\mathfrak{M}(T, T')) = h(T')$
- so total of $2^{\#V^{int}(T)}$ possible *head functions* on a tree T
- *head* of a subtree $T_v \subset T$ is leaf $h_T(v)$ reached by following path of only marked edges (that determine h_T) from v

Note: this notion of *head function* is equivalent to the properties of *head* defined in Chomsky's "Bare phrase structure", 1995

Inductive properties characterizing *head* in Chomsky's "Bare phrase structure"

1. For $T = \mathfrak{M}(\alpha, \beta)$, with $\alpha, \beta \in \mathcal{SO}_0$, the *head* $h(T)$ should be one or the other of the two items α, β. The item that becomes the *head* $h(T)$ is said to *project*.
2. In further projections the *head* is obtained as the "head from which they ultimately project, restricting the term head to terminal elements".
3. Under Merge operations $T = \mathfrak{M}(T_1, T_2)$ one of the two syntactic objects $T_1, T_2 \in \mathcal{SO}$ projects and its *head* becomes the *head* $h(T)$. The label of the structure T formed by Merge is the *head* of the constituent that projects.

these three properties are equivalent to our definition of *head function*

head functions and planar embeddings

- a *head function* $h_T : V^{int}(T) \rightarrow L(T)$ determines a planar embedding $T^{\pi^{h_T}}$ of T: put the marked edge below each vertex to the left of the other
- question of a special (canonical) choice of planarization σ^{LCA} becomes a canonical choice of a *head function*

$$\mathfrak{T}_{\mathcal{SO}_0} \ni T \overset{h^{LCA}}{\mapsto} h_T$$

- this mapping h^{LCA} should be determined by the labels $\lambda(\ell) \in \mathcal{SO}_0$
- note that in $\mathfrak{T}_{\mathcal{SO}_0}$ leaf-labels are arbitrary (only later, in the quotient map $\Pi_{\mathcal{L}}$ step of externalization some are ruled out)
- so to have σ^{LCA} defined on all $\mathcal{SO}$ should be able to choose one of the two $h_{\mathfrak{M}(T,T')}$ based on $\lambda(h_T(T))$ and $\lambda(h_{T'}(T'))$
- Problem: if $\lambda(h_T(T)) = \lambda(h_{T'}(T'))$ cannot distinguish two possible $h_{\mathfrak{M}(T,T')}$ even if $\mathcal{SO}_0$ were totally ordered

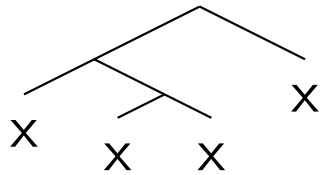
Core computational structure of Merge

- free non-associative commutative magma $\mathfrak{T}$
- elements are balanced bracketed expressions in a single variable x, with the binary operation (binary set formation)

$$(\alpha, \beta) \mapsto \mathfrak{M}(\alpha, \beta) = \{\alpha, \beta\}$$

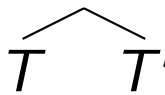
  where α, β are two such balanced bracketed expressions
- equivalent description: elements are finite binary rooted trees (no assignment of planar structure)

$$\{\{x\{xx\}\}x\} \longleftrightarrow$$



- operation on trees

$$\mathfrak{M}(T, T') =$$



Generative process of $\mathfrak{T}$

- same formal trick: take *vector space* $\mathcal{V}(\mathfrak{T})$ (say over $\mathbb{Q}$) spanned by elements of $\mathfrak{T}$ (convenient for writing a list of possibilities as a sum)
- mathematical note: the magma operation $\mathfrak{M}$ on $\mathfrak{T}$ identifies $\mathcal{V}(\mathfrak{T})$ with the free commutative non-associative algebra generated by a single variable x (free algebra over the quadratic operad freely generated by the single commutative binary operation)
- assign a grading (a weight, measuring size) to the binary rooted trees by the number of leaves, $\ell = \#L(T)$, so the vector space decomposes $\mathcal{V}(\mathfrak{T}) = \oplus_\ell \mathcal{V}(\mathfrak{T})_\ell$
- in a formal infinite sum $X = \sum_\ell X_\ell$ of variables X_ℓ in $\mathcal{V}(\mathfrak{T})_\ell$

$$X = \mathfrak{M}(X, X)$$

fixed point equation

- the equation $X = \mathfrak{M}(X, X)$ can be solved recursively by degrees

$$X_n = \mathfrak{M}(X, X)_n = \sum_{j=1}^{n-1} \mathfrak{M}(X_j, X_{n-j})$$

- solution $X_1 = x$, $X_2 = \{xx\}$,
  $X_3 = \{x\{xx\}\} + \{\{xx\}x\} = 2\{x\{xx\}\}$,
  $X_4 = 2\{x\{x\{xx\}\}\} + \{\{xx\}\{xx\}\}$, and so on
- coefficients: $\{x\{xx\}\}$ and $\{\{xx\}x\}$ same abstract tree (while two different planar embeddings)
- recursive solution describes the generative process of $\mathfrak{T}$ through the Merge operation $\mathfrak{M}$

Recursive fixed point equation: Dyson–Schwinger

- case above $X = \mathfrak{M}(X, X)$ is special fundamental case of combinatorial Dyson–Schwinger equations

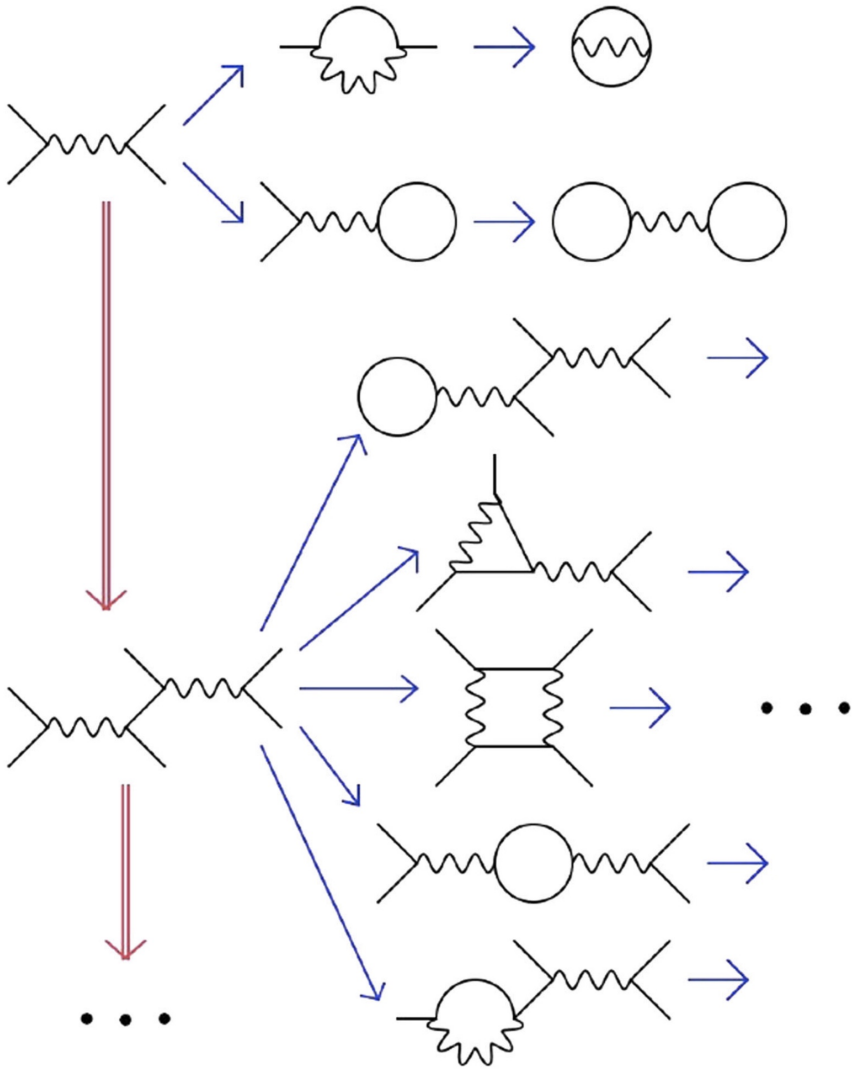
$$X = \mathfrak{B}(P(X))$$

with $X = \sum_\ell X_\ell$ by degrees, $P(X)$ a polynomial function (here a single quadratic term) and $\mathfrak{B}$ a type of (possibly n-ary) Merge operation

- Dyson–Schwinger equations and recursive construction of solutions of equations of motion in quantum field theory

Linguistic Merge versus Physical DS equations: a useful parallel

- in quantum field theory we have a generative process involving graphs (Feynman graphs)
- can be described in terms of formal languages (using graph grammars)
- however not the best way to think of Feynman graphs
- Hopf algebra structure: product $\sqcup$, coproduct $\Delta(\Gamma) = \sum \gamma \otimes \Gamma/\gamma$ subgraphs and quotient graphs (Connes-Kreimer)
- better for factorization problems (extraction of meaningful physical values = renormalization) with consistency across subgraphs
- better for recursive solutions of equations of motion $X = \mathfrak{B}(P(X))$ Dyson–Schwinger equation
- known in QFT that solutions of DS are the quantum implementation of the "least action principle" for classical solutions: optimization

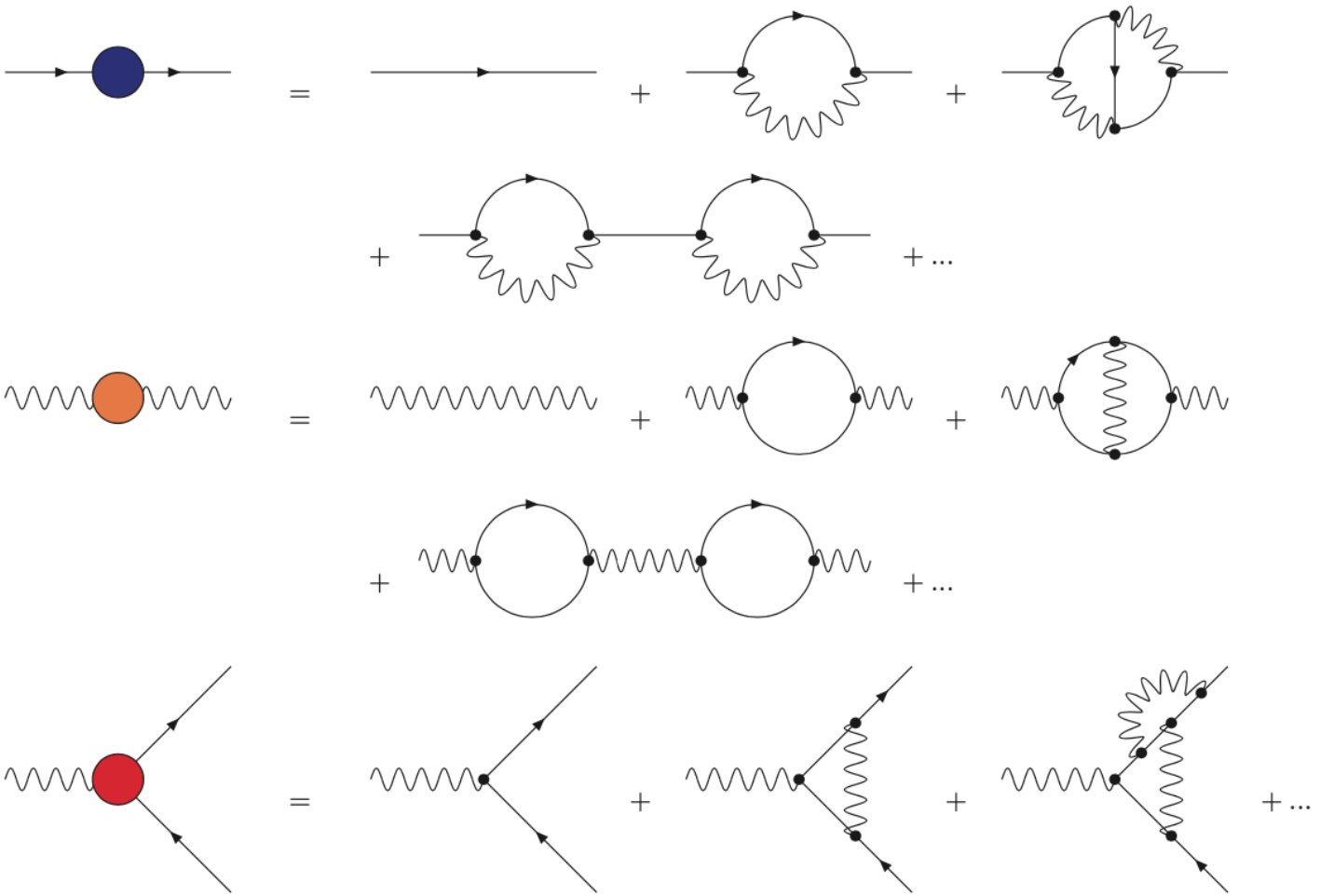


Example: formal languages approach – the generative grammar for the Feynman graphs of the $\phi^2 A$ physical theory (graph grammars: usually context sensitive)

$$\Delta\left(\right) = \otimes \mathbb{I} + \mathbb{I} \otimes + 2 \otimes$$

$$S\left(\right) = - +$$

Example: the generative structure of Feynman graphs encoded in the coproduct and the antipode of a Hopf algebra



Examples: recursive solutions of Dyson–Schwinger equations in quantum electrodynamics

- the formalism of Hopf algebras and extraction of finite parts was adapted to the theory of computation (Manin, 2009) as extraction of computable parts from undecidable problems
- "extraction of meaning" (finite values from divergent integrals in physics; computable parts of non-computable functions in theory of computation) via the formalism of renormalization (factorization of maps from Hopf algebras to Rota–Baxter algebras)
- suggests a possible strategy to extend the computational model of syntax to a computational model of the syntactic-semantic interface

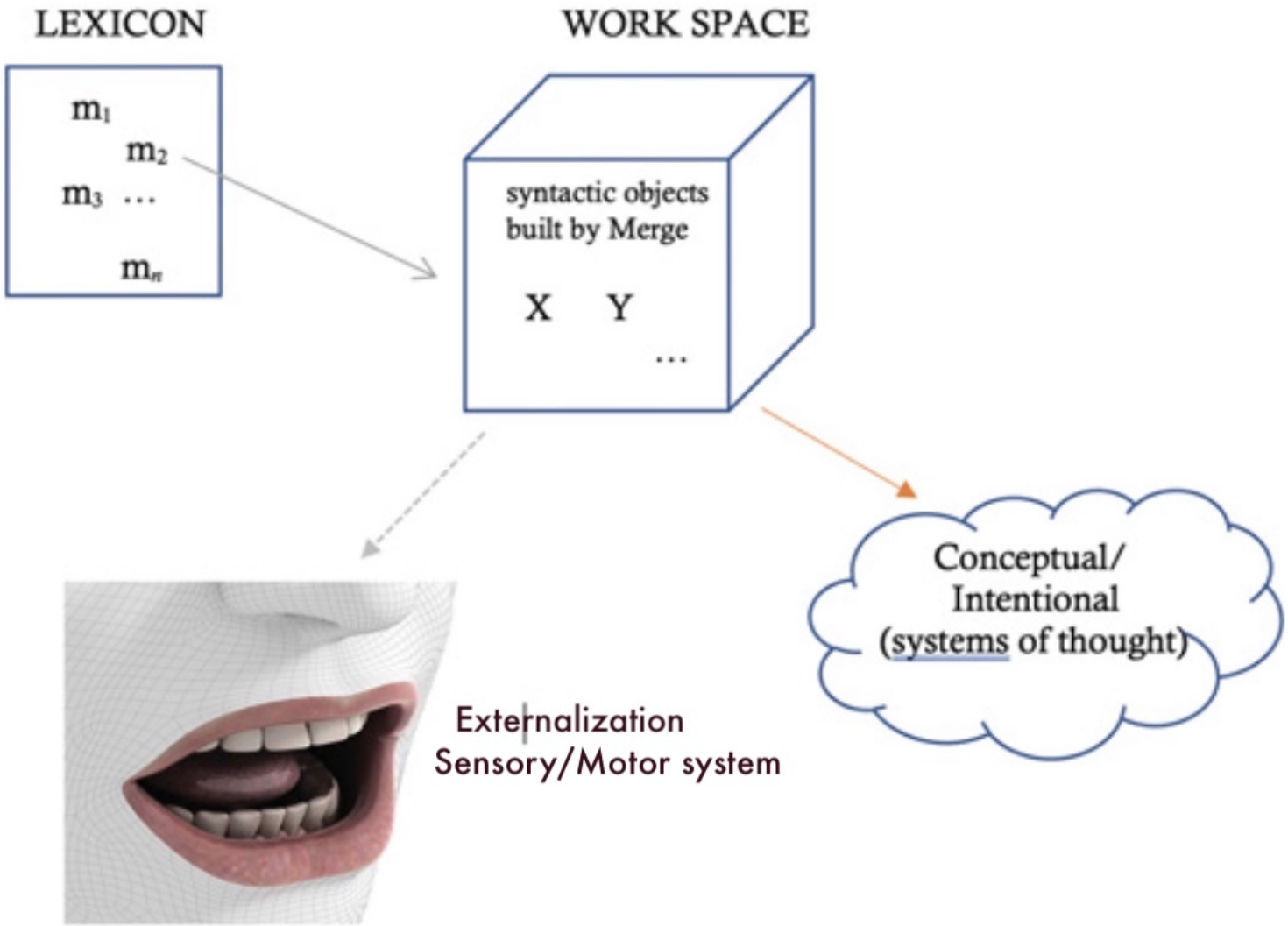
...this comparison is the base for our construction of a syntax-semantics interface model

Syntax-Semantics interface: conceptual requirements

(not all would agree, but we take these as our background assumptions)

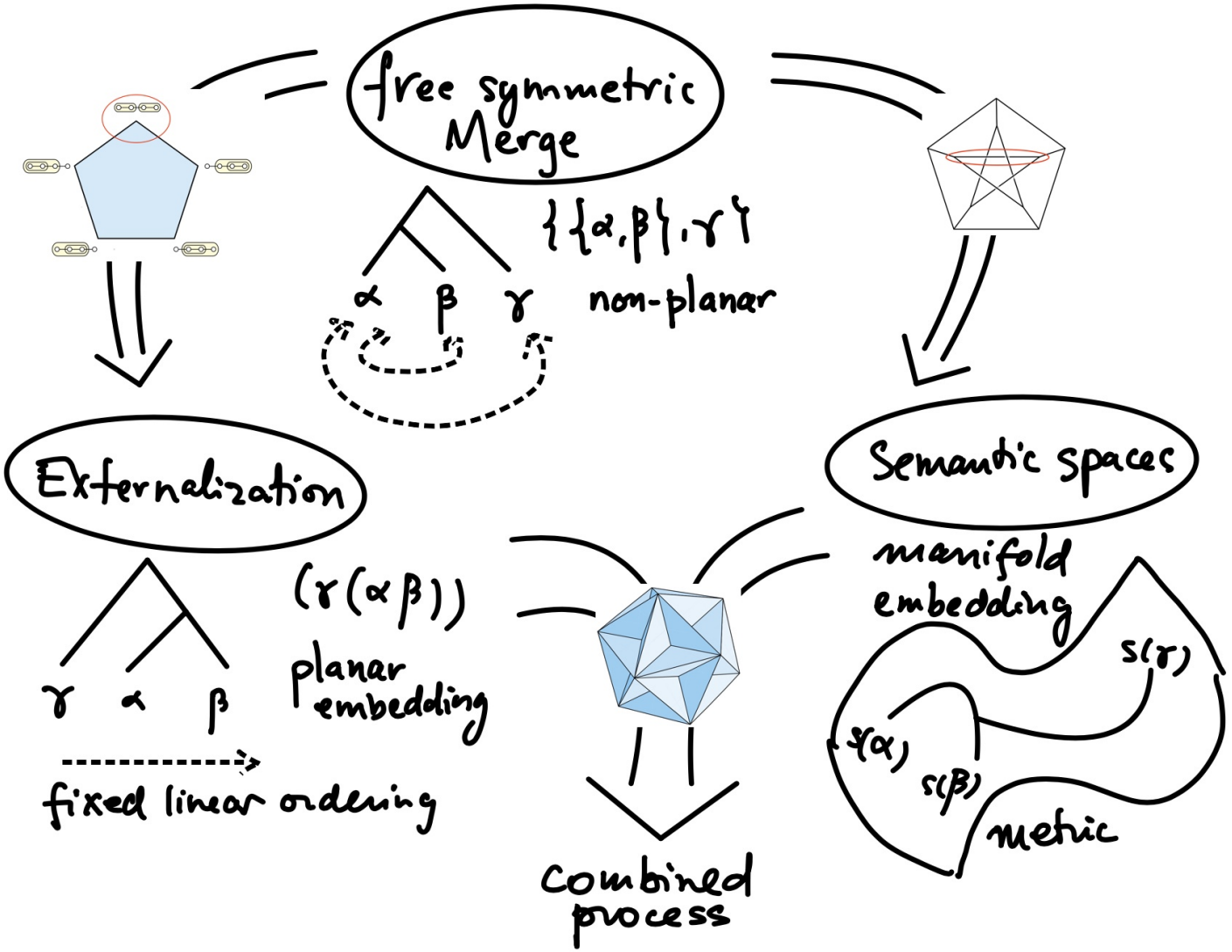
1. Autonomy of syntax
2. Syntax supports semantic interpretation
3. Semantic interpretation is, to a large extent, independent of externalization
4. Compositionality

- 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



(from "Merge & SMT")

Preview of what this same picture looks like in our setting:



Basic properties

our view of distinction between the roles of syntax and semantics:

1. Syntax is a computational process.
2. Semantics is *not* a computational process and is in essence grounded on a notion of topological proximity.

- first statement is clear in the context of generative linguistics,
- second claim means possible additional structures on the side of semantics (metric, linear, semiring, etc) instantiate or quantify proximity relations, don't play a computational generative role like syntax

Birkhoff factorization: key idea

- 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

(conceptually similar to e.g. truth-values assignments using trees in context free setting, but recursive checking is directly built into the value function, and context-free hypothesis is not needed)

- 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

Birkhoff factorization: formal costruction

- 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 $(\mathcal{R}, R)$

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

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

- 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 $\mathcal{H}$

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

- Birkhoff factorizations are constructed inductively

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

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

where $\Delta(x) = 1 \otimes x + x \otimes 1 + \sum x' \otimes x''$ with x', x'' of 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

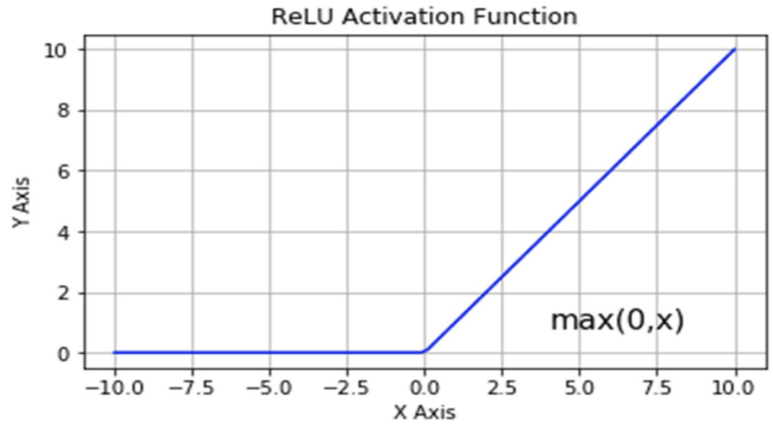
- semiring case

$$\begin{aligned}
\phi_-(x) = &\quad R(\tilde{\phi}(x)) &&= R(\phi(x) \boxdot \phi_-(x') \odot \phi(x'')) \\
\phi_+(x) = &\quad (\phi_- \star \phi)(x) &&= \phi(x) \boxdot \phi_-(x) \boxdot \phi_-(x') \odot \phi(x'') \\
= &\quad \phi_- \boxdot \tilde{\phi}
\end{aligned}$$

multiplicative with respect to the semiring product
$\phi_\pm(xy) = \phi_\pm(x) \odot \phi_\pm(y)$

toy model example in vector space semantics

- 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	$x + y$

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

- very minimal hypothesis on semantic space model: $\mathcal{S}$ 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

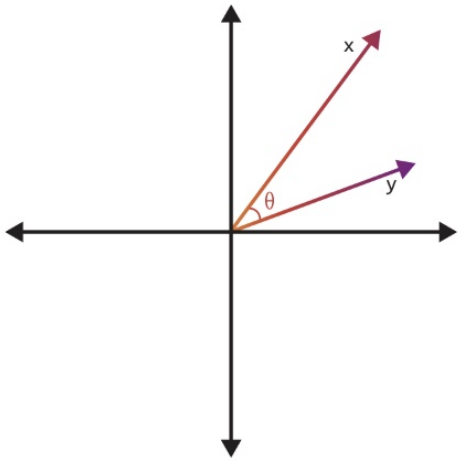
$$\Upsilon(s) = \langle s, v_\Upsilon \rangle$$

- given a semantic space $\mathcal{S}$, a probe $\Upsilon : \mathcal{S} \to \mathbb{R}$, a map $s : \mathcal{SO}_0 \to \mathcal{S}$ assigning semantic values to lexical items and a *head function* h on a domain $\mathrm{Dom}(h) \subset \mathfrak{T}_{\mathcal{SO}_0}$
- build from this a test of semantic agreement/disagreement

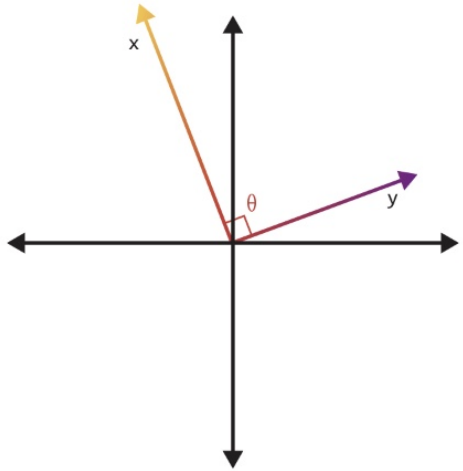
$$\Upsilon_{s,h} : T \mapsto \begin{cases} \Upsilon(s(h(T)) & T \in \mathrm{Dom}(h) \\ -\infty & T \notin \mathrm{Dom}(h)\,. \end{cases}$$

$$\phi_{\Upsilon,s,h} : \mathfrak{F}_{\mathcal{SO}_0} \to \mathbb{R} \cup \{-\infty\}$$

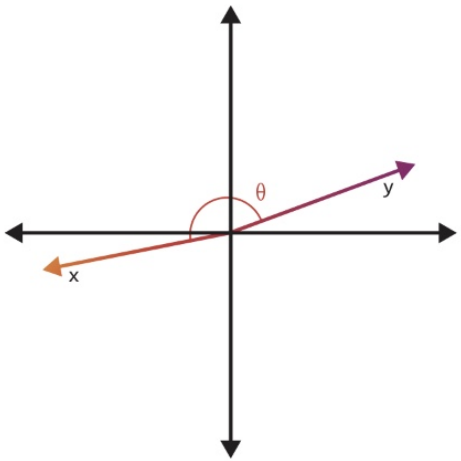
$$\phi_{\Upsilon,s,h}(F) = \sum_a \Upsilon_{s,h}(T_a)\,, \quad \text{for } F = \sqcup_a T_a$$



Angle θ close to 0°
Cos(θ) close to 1 → **Similar vectors**



Angle θ close to 90°
Cos(θ) close to 0 → **Orthogonal vectors**



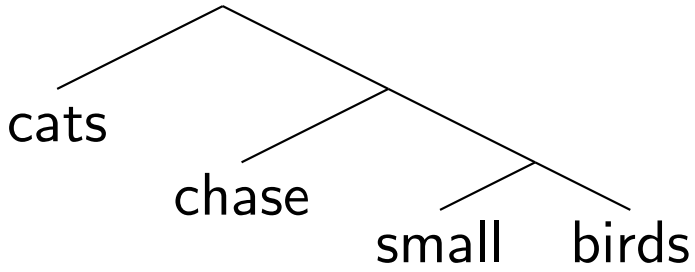
Angle θ close to 180°
Cos(θ) close to -1 → **Opposite vectors**

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

- Birkhoff factorization of $\phi\Upsilon_{,s,h}$ with respect to ReLU Rota–Baxter operator
- $\phi\Upsilon_{,s,h,-}(T)$ 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 where maximum consistent agreement with the chosen probe is achieved
- problem: this only uses the *head* $h(T)$ for location in semantic space (not right: values of lexical items only)
- ...will correct for this using a *geodesic convexity* assumption on semantic space
- but first let's see the issue better in some examples

Example

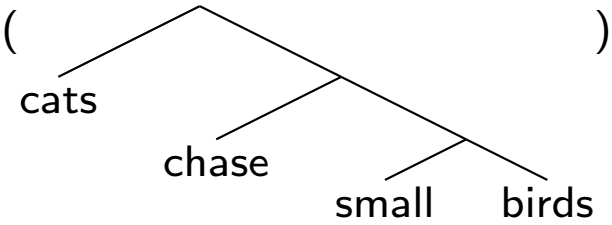
- sentence like



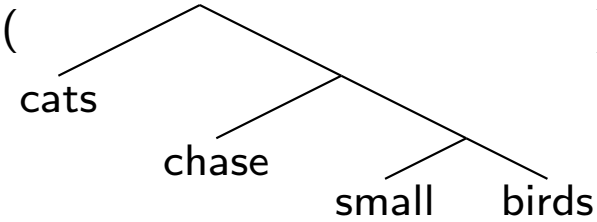
- lexical items as vectors in a vector space model of semantics
- probe v_Υ, for example the vector associated to "predation"
- vectors v for "cats", "chase", "birds" positively correlate (as predator, prey, hunting action) to the semantic probe
- extracting coproduct terms and building recursive factorization

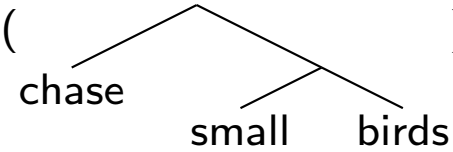
$$\phi_{\Upsilon,s,h,-}(T) = (\max\{\phi_{\Upsilon,s,h}(T), \phi_{\Upsilon,s,h}(F_{\underline{v}})^{+}\phi_{\Upsilon,s,h}(T/F_{\underline{v}}),$$

$$\ldots, \phi_{\Upsilon,s,h}(F_{\underline{v}_N})^{+}\phi(F_{\underline{v}_{N-1}}/F_{\underline{v}_N})\cdots\phi(T/F_{\underline{v}_1})\})^{+}$$

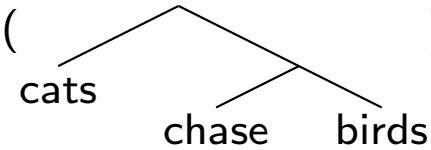
$\phi\Upsilon,s,h,-($ 

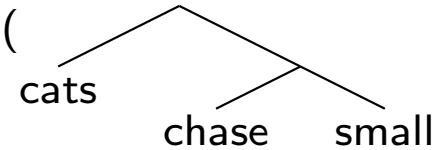
$) =$

$(\max\{\phi\Upsilon,s,h($ 

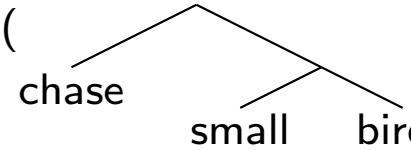
$), \phi\Upsilon,s,h(\text{cats})^{+}\phi\Upsilon,s,h($ 

$),$

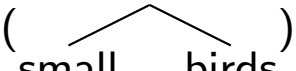
$\phi\Upsilon,s,h(\text{small})^{+}\phi\Upsilon,s,h($ 

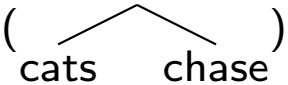
$), \phi\Upsilon,s,h(\text{birds})^{+}\phi\Upsilon,s,h($ 

$),$

$\phi\Upsilon,s,h($ 

$)^{+}\phi\Upsilon,s,h(\text{cats}),$

$\phi\Upsilon,s,h($ 

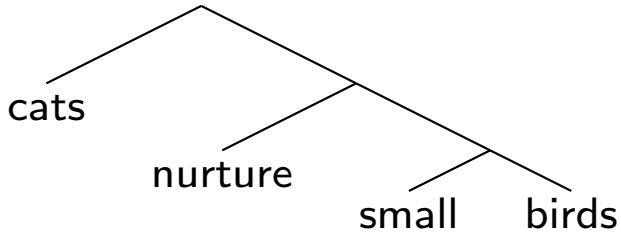
$)^{+}\phi\Upsilon,s,h($ 

$), \ldots\})^{+}$

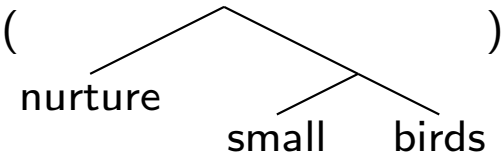
where ... stands for terms of the coproduct involving a forest

here *head* is either verb "chase" or noun "bird" on the subtrees and if R is ReLU (or some threshold) get agreement with probe on substructures

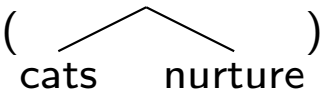
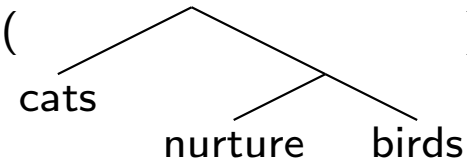
- take same sentence but replace "chase" with "nurture"



- with the same probe v_Υ semantic vector of "predation"
- now some substructures are filtered out by ReLU because disagreement with probe

$\phi_{\Upsilon,s,h}($  $)^+ = 0$

- also final application of $R = ReLU$ to max in $\phi_{\Upsilon,s,h,-} = R(\tilde{\phi}_{\Upsilon,s,h})$ also cuts off all structures with *head* the verb "nurture" because of disagreement with probe

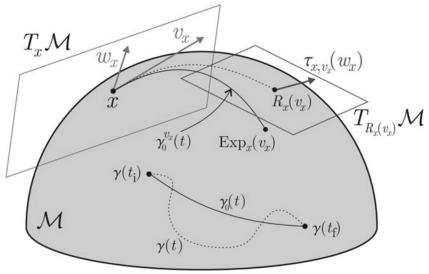
$\phi_{\Upsilon,s,h}($  $)^+ = 0,$ $\phi_{\Upsilon,s,h}($  $)^+ = 0,$ etc

but ... another example

- consider again example of "France is a hexagonal republic"
- in a vector space model of semantics all the lexical items "France", "hexagon(al)", "republic" have corresponding vectors, say $v_{Fr}, v_{hex}, v_{rep} \in \mathcal{S}$
- choose a probe v_Υ, for example the vector associated to "government"
- would have $\langle v_\Upsilon, v_{Fr} \rangle > 0$, $\langle v_\Upsilon, v_{rep} \rangle > 0$, but $\langle v_\Upsilon, v_{hex} \rangle \leq 0$
- same tree structure... *but* now see that just looking at the *head* leaf of the trees does *not* help detect where problem
- what is needed: one can see in this example that what one wants to have is a point $s(T) \in \mathcal{S}$ that displaces the position of $s(h(T))$ in the direction of the *complement* of the head, just slightly if large agreement, a lot if disagreement so that $\phi_{\Upsilon,s,h}(T)$ can differ significantly from $\phi_{\Upsilon,s,h}(h(T))$ if there is disagreement inside the structure T

first improvement

- use some notion of distance/proximity on $\mathcal{S}$ and convexity (both ok in vector spaces)
- semantic space $\mathcal{S}$ geodesically convex region in Riemannian manifold (M, g) (min length geodesic arcs between points)



- $\{\lambda s + (1-\lambda)s' \mid \lambda \in [0,1]\}$ segment between s, s' in $\mathcal{S}$
- also suppose have some comparison functions

$$\mathbb{P} : \mathcal{S} \times \mathcal{S} \to [0,1]$$

$$\mathfrak{C} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$$

likelihood of relatedness $\mathbb{P}$ or degree of agreement/disagreement $\mathfrak{C}$ (also ok in vector spaces for instance $\mathfrak{C}(s, s') = \frac{\langle s, s' \rangle}{\|s\| \, \|s'\|}$)

- discuss construction with $\mathbb{P}$ (case of $\mathfrak{C}$ similar)

- 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
- extend $s : \mathcal{SO}_0 \to \mathcal{S}$ to a map $s : \mathrm{Dom}(h) \to \mathcal{S}$ inductively
  - for $\alpha, \beta \in \mathcal{SO}_0$

$$T = \mathfrak{M}(\alpha, \beta) = \overbrace{\alpha \quad \beta} \quad \Rightarrow \quad s(T) = p\, s(\alpha) + (1-p)\, s(\beta)$$

$$p = \begin{cases} p_{\alpha,\beta} & \alpha = h(T) \\ 1 - p_{\alpha,\beta} & \beta = h(T)\,. \end{cases} \qquad \text{with} \quad p_{\alpha,\beta} := \mathbb{P}(s(\alpha), s(\beta))$$

  - then inductively if $T = \mathfrak{M}(\alpha, \beta) \in \mathrm{Dom}(h) \subset \mathcal{SO}$ take

$$s(T) = p\, s(T_1) + (1-p)\, s(T_2) \quad \text{and} \quad p_{s(T_1),s(T_2)} = \mathbb{P}(s(T_1), s(T_2))$$

$$p = \begin{cases} p_{s(T_1),s(T_2)} & h(T) = h(T_1) \\ 1 - p_{s(T_1),s(T_2)} & h(T) = h(T_2) \end{cases}$$

- Viterbi valued characters $\phi_{s,\mathbb{P},h} : \mathfrak{FSO}_0 \to \mathcal{P}$
  (used for probabilistic assignments)

$$\phi_{s,\mathbb{P},h}(T) = p_{s(T_1),s(T_2)}$$

  and $\phi_{s,\mathbb{P},h}(T) = 0$ outside $\mathrm{Dom}(h)$
- Birkhoff factorization of $\phi_{s,\mathbb{P},h}$ with respect to threshold Rota–Baxter operators identifies substructures with large semantic agreement between constituent parts

$$c_\lambda(\max\{\phi_{s,\mathbb{P},h}(T), c_\lambda(\phi_{s,\mathbb{P},h}(T_v)) \cdot \phi_{s,\mathbb{P},h}(T/T_v)\}) =$$

$$c_\lambda(\max\{p_{s(T_1),s(T_2)}, c_\lambda(p_{s(T_{v,1})s(T_v,2)}) \cdot p_{s(T_1),s(T_2)}\}) = c_\lambda(p_{s(T_1),s(T_2)})$$

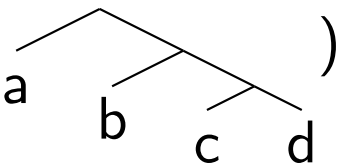
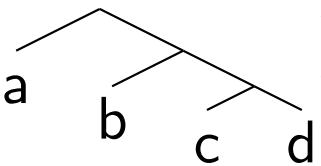
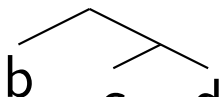
  max value when all $p_{s(T_{v,1})s(T_v,2)} \geq \lambda$ and $p_{s(T_1),s(T_2)} \geq \lambda$
- maximizers are accessible terms that carry large semantic agreement between their constituent parts
- similar example with $\mathfrak{C}$ and ReLU Rota-Baxter operator

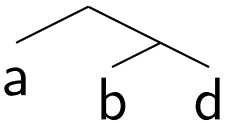
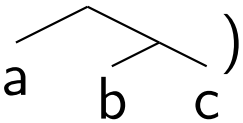
Boolean example

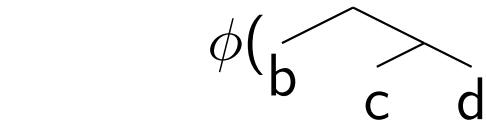
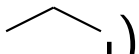
- Boolean semiring $\mathcal{B} = (\{0,1\}, \vee, \wedge) = (\{0,1\}, \max, \cdot)$
- map $\phi : \mathfrak{T}_{\mathcal{SO}_0} \to \mathcal{B}$ is an assignment of truth values extended to $\phi : \mathfrak{F}_{\mathcal{SO}_0} \to \mathcal{B}$ by $\phi(F) = \prod_i \phi(T_i)$ for $F = \sqcup_i T_i$
- identity as Rota–Baxter operator of weight -1
- Bogolyubov preparation

$$\tilde{\phi}(T) = \max\{\phi(T), \phi(F_{\underline{v}})\phi(T/F_{\underline{v}}), \ldots, \phi(F_{\underline{v}_N})\phi(F_{\underline{v}_{N-1}}/F_{\underline{v}_N}) \cdots \phi(T/F_{\underline{v}_1})\}$$

- again example of "France is a hexagonal republic"

$\tilde{\phi}($  $) = \max\{\phi($  $),\ \phi(a)\phi($  $),$

$\phi(c)\phi($  $),\ \phi(d)\phi($  $),$

$\phi($  $)\phi(a),\ \phi($  $)\phi($  $), \ldots\}$

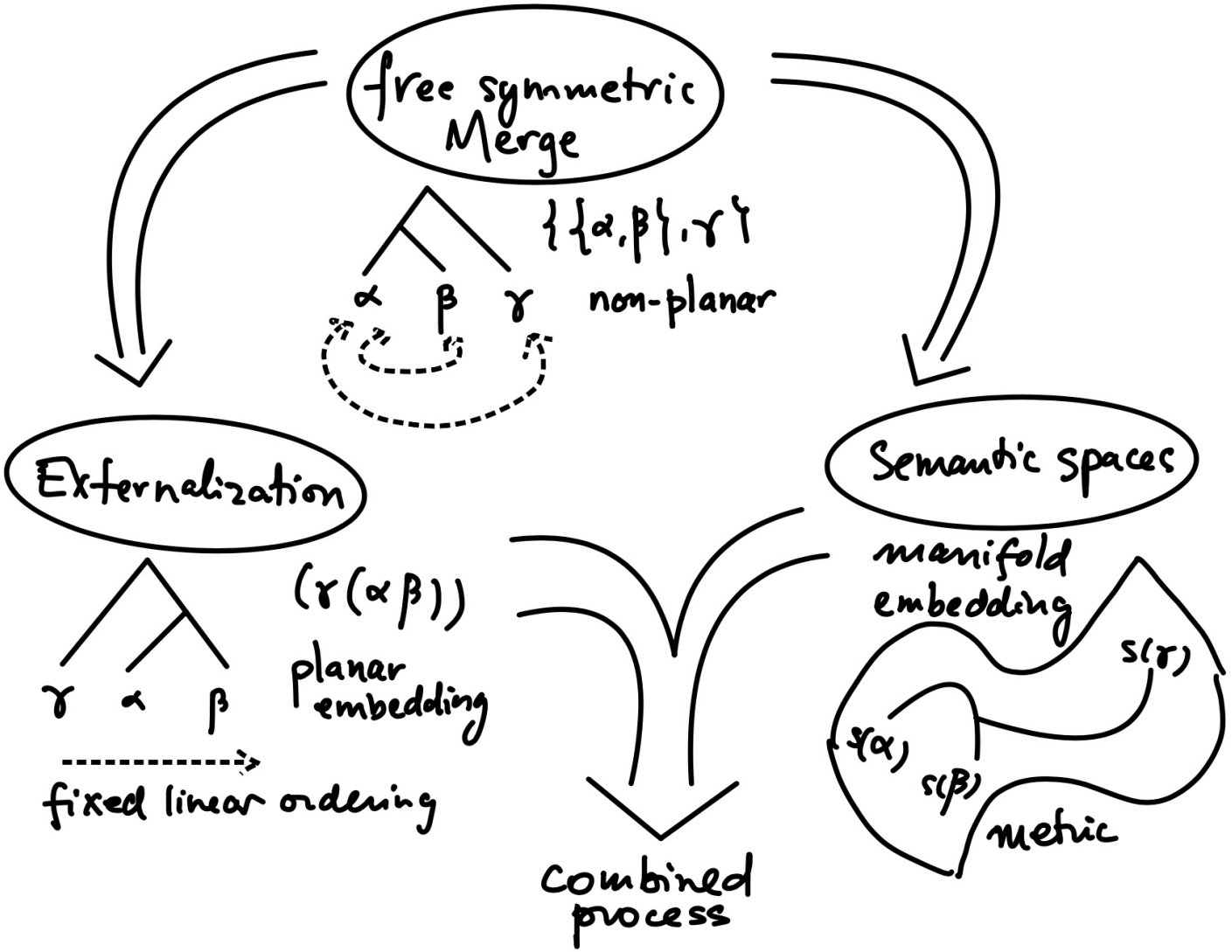
where ... stands for terms of the coproduct involving a forest

- $0/1$ = consistent/inconsistent identifies the only consistent substructures "France is hexagonal" and "France is a republic"

image of syntax inside semantics

- semantic space $\mathcal{S}$ geodesically convex Riemannian manifold
- if for $s \neq s'$ one has $\mathbb{P}(s, s') \in (0, 1)$
- obtain embeddings of trees $T \in \mathrm{Dom}(h) \subset \mathfrak{T}_{\mathcal{SO}_0}$ *inside* $\mathcal{S}$
- edges geodesic arcs, vertices points
  $s(T) = p\, s(T_1) + (1 - p)\, s(T_2)$ on geodesics arcs
- the image $\mathcal{I}(T_v)$ of a subtree T_v as the union of the images $\mathcal{I}(T_{v,1})$ and $\mathcal{I}(T_{v,2})$, where $T_v = \mathfrak{M}(T_{v,1}, T_{v,2})$, and the geodesic arc between $s(T_{v,1})$ and $s(T_{v,2})$ with root vertex at $s(T_v)$

Externalization and syntax-semantics interface

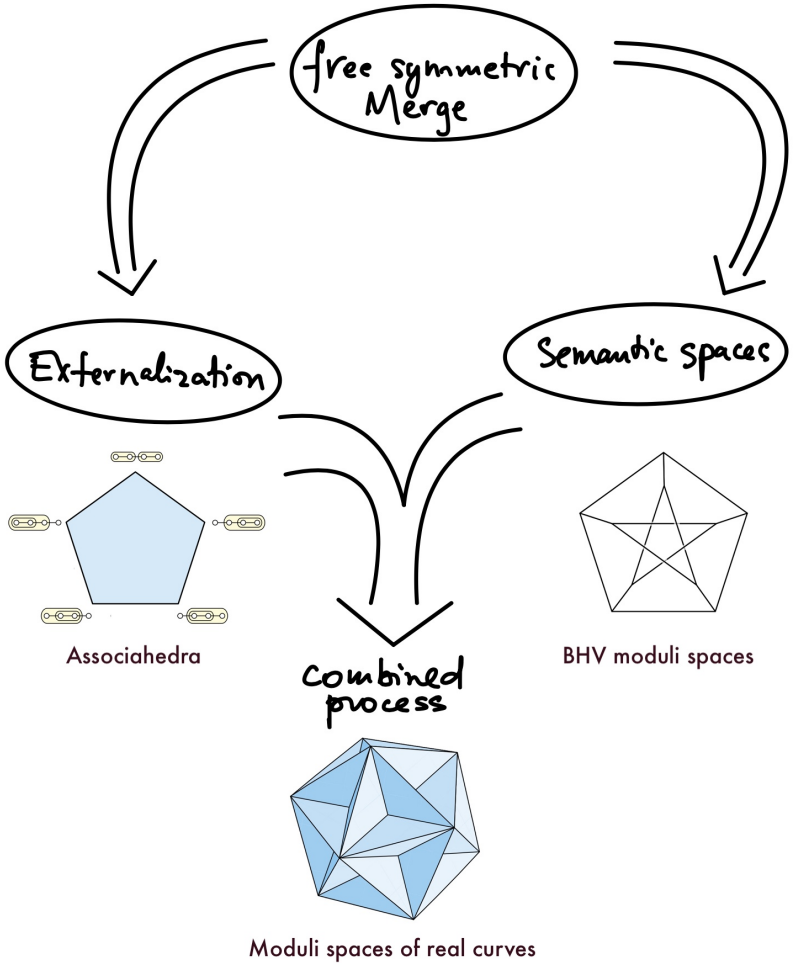


goal: giving a geometric model for the Conceptual/Intentional and Sensory/Motor channels and their interaction

main ideas

- planarization of Externalization is language dependent, we need a space where different planarization can be compared $\Rightarrow$ associahedron.
- hierarchical structures produced by free symmetric Merge acquire a metric structure through mapping to semantic spaces
- metric structure keeps track of semantic relatedness across substructures
- assignment of metric data on (non-planar) binary rooted trees described by BHV moduli space
- so previous picture becomes geometric relation between associahedra and BHV moduli spaces of metric trees (relation realized by moduli space of real curves of genus zero with marked points)

Externalization and syntax-semantics interface



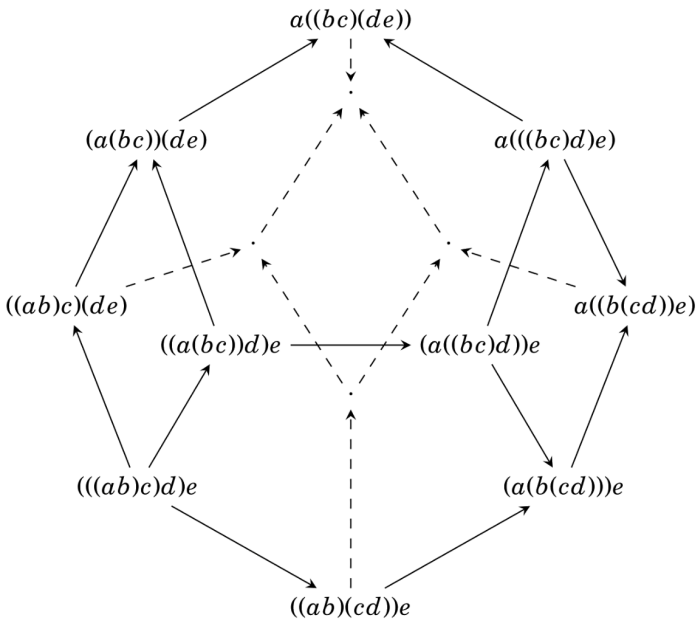
the resulting moduli space parameterizes simultaneous compatible choices of a metric embedding of the non-planar syntactic object and of a planarization

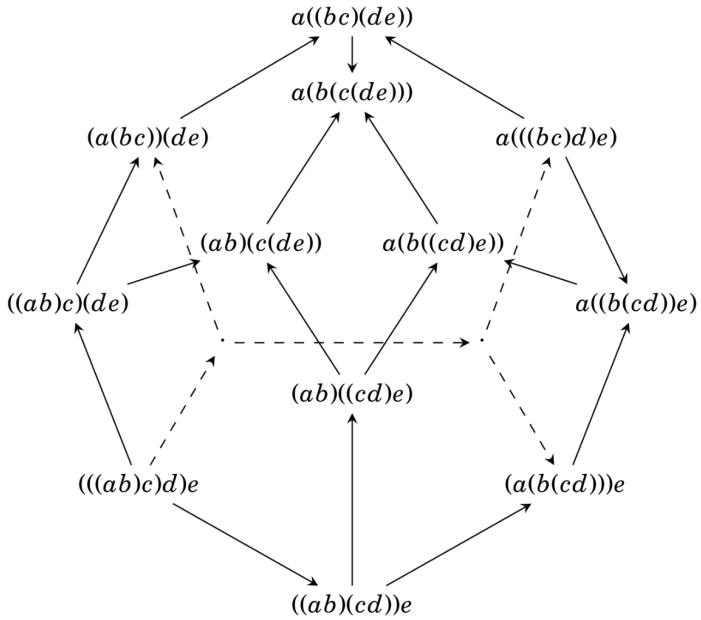
associahedra (Stasheff)

- 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:



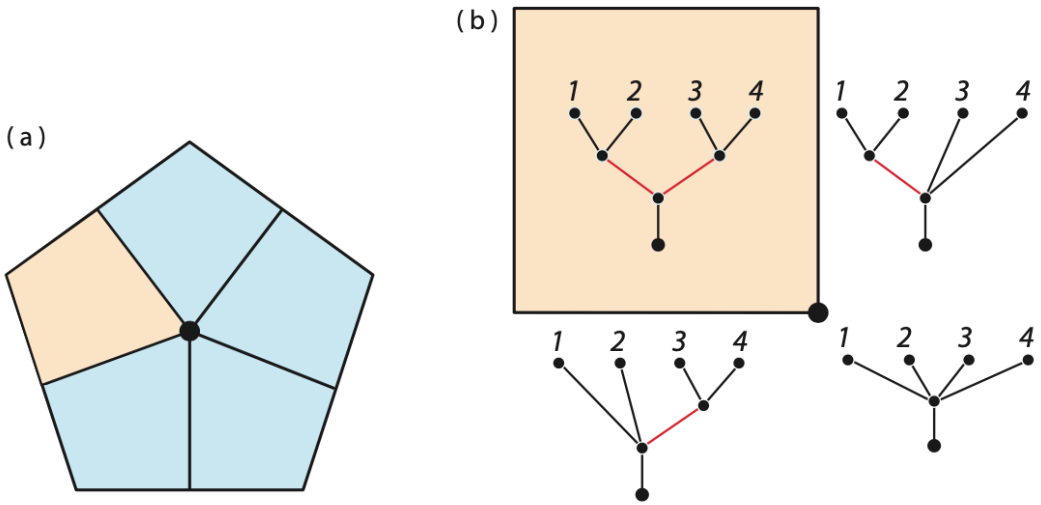


associahedra with metric data

- associahedron K_n decomposed into C_{n-1} cubes of dimension $n-2$ with Catalan number

$$C_{n-1} = \frac{1}{n}\binom{2n-2}{n-1}$$

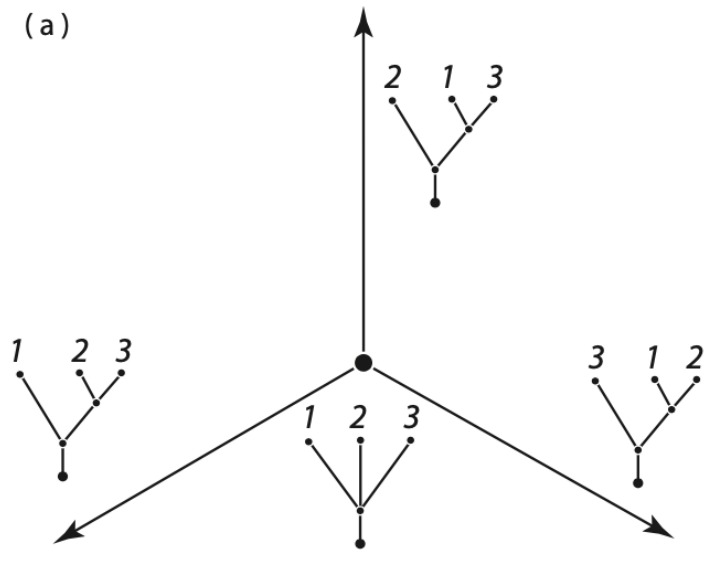
- cubical decomposition of K_4

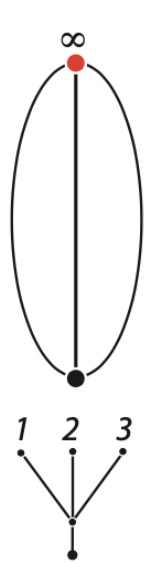


- polytope points as metric structures on planar binary rooted trees: weights in $\mathbb{R}_{\geq}0$ to the internal edges of the tree

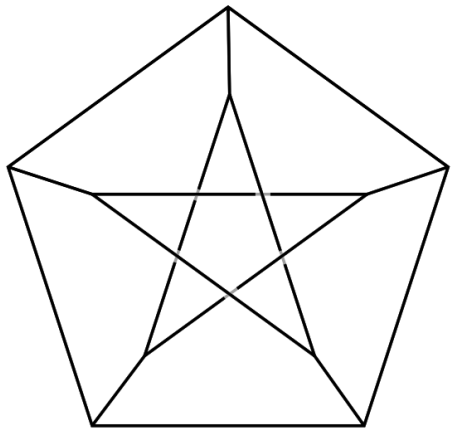
BHV moduli spaces of metric trees (Billera, Holmes, Vogtmann)

- moduli space BHV_n of abstract binary rooted trees with n leaves (with no assigned planar structure) along with weighted internal edges
- one-point compactification BHV^+
- consider all the $(2n-3)!!$ abstract binary rooted trees with n labeled leaves: they all have $n-2$ internal edges
- for each tree an orthant $\mathbb{R}^{n-2}_{\geq 0}$: all the possible choices of weight (length) of internal edges
- moduli space BHV_3 and one-point compactification BHV_3^+



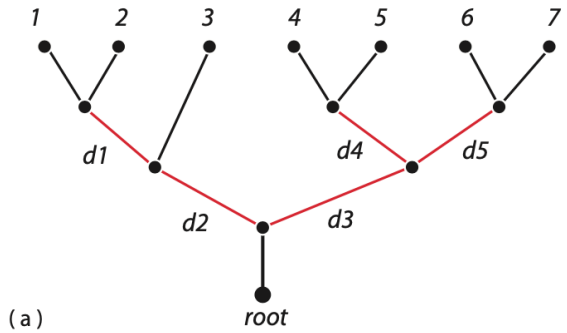


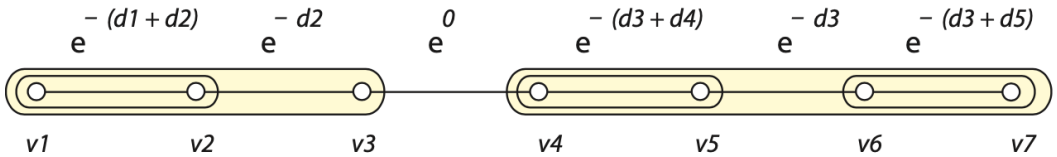
- link $\mathcal{L}_n$ of the origin in BHV_n is an $(n-3)$-dimensional simplicial complex
- case $n = 3$ just three points
- case $n = 4$ Peterson graph



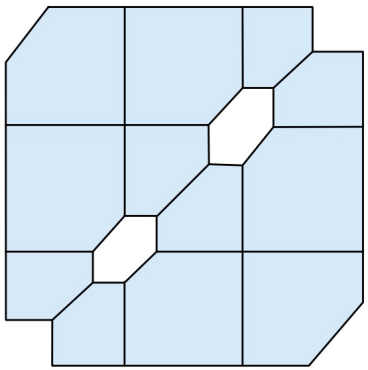
Relation between associahedra and BHVs: moduli spaces of curves

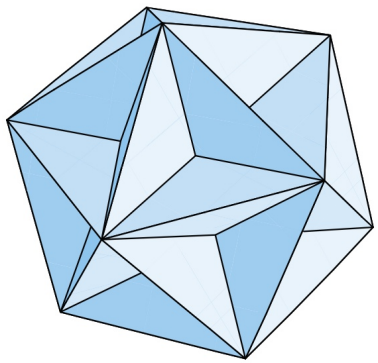
- in relating associahedra and BHV need to account for:
  - different planar structures (in associahedron not in BHV)
  - different permutations of the labels at the leaves (in BHV not in associahedron)
- one geometric object that accounts for both: moduli space of real curves $\bar{M}_{0,n+1}(\mathbb{R})$, orientation double cover $\bar{M}^{or}_{0,n+1}(\mathbb{R})$
- think of ordered leaves of tree as ordered set of points in the real line $\mathbb{R}$
- take coordinates of points as functions of weights of edges





- $\bar{M}_{0,n+1}(\mathbb{R})$: configurations of $n+1$ points on $\mathbb{P}^1(\mathbb{R})$ compactified according to collisions (can always fix 0, 1, ∞ by a choice of coordinate in $\mathbb{P}^1(\mathbb{R})$ so $n-2$ remaining variables
- associahedra K_n have dim $n-2$
- known (Devadoss-Morava): $\bar{M}^{or}_{0,n+1}(\mathbb{R})$ decomposes into a union of $n!$ associahedra K_n glued together ($n!$ permutations of labelled leaves of trees)





Twelve associahedra K_4 assemble into the space $\bar{M}_{0,5}(\mathbb{R})$; orientation double cover gives 24 associahedra assembled into $\bar{M}^{or}_{0,5}(\mathbb{R})$ identified with the great dodecahedron

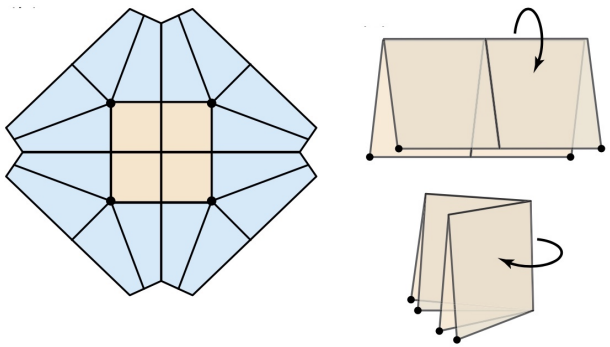
- also known (Devadoss-Morava): projection map

$$\Pi_n : \overline{M}^{or}_{0,n+1}(\mathbb{R}) \twoheadrightarrow \mathrm{BHV}^+_n$$

- how these things fit together:

$$n! \cdot C_{n-1} = 2^{n-1} \cdot (2n-3)!!$$

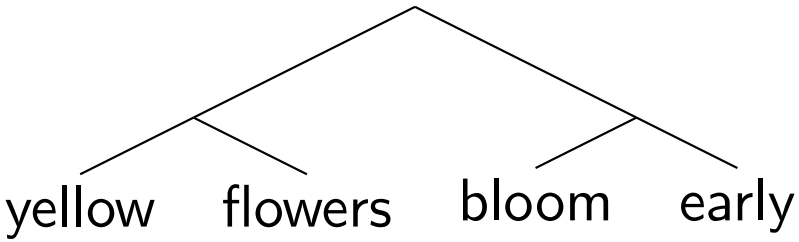
  - on left: C_{n-1} cubes of $n!$ associahedra in $\overline{M}^{or}_{0,n+1}(\mathbb{R})$
  - on right: $(2n-3)!!$ simplexes dim $(n-3)$ of $\mathcal{L}_n$
  - 2^{n-1} is the (general) multiplicity of the projection map



Four squares in adjacent associahedra K_4 are folded together (origami folding) in the projection: in double cover 8 squares identified in $\Pi_4 : \overline{M}^{or}_{0,5}(\mathbb{R}) \twoheadrightarrow \mathrm{BHV}^+_4$

Example

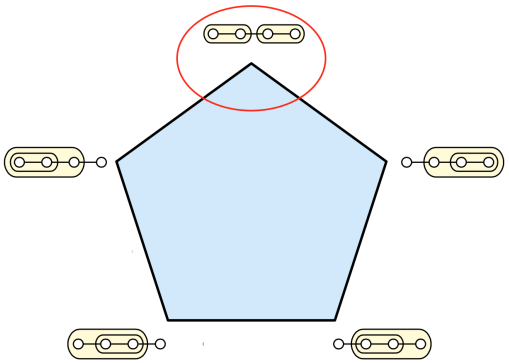
- a English sentence like



$$= \quad = ((\alpha\ \beta)\,(\gamma\ \delta))$$

$\alpha \quad \beta \quad \gamma \quad \delta$

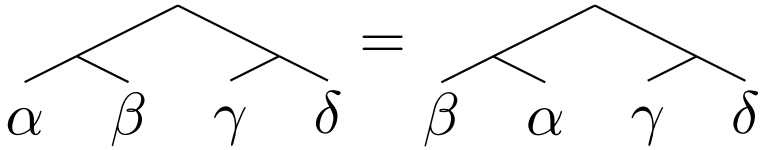
- planar structure selects a vertex of the associahedron K_4



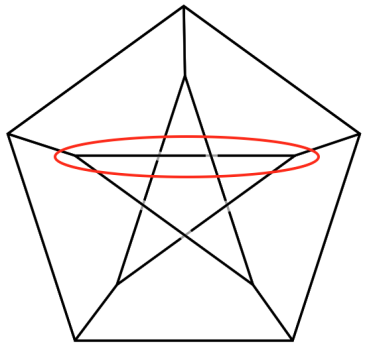
- this associahedron is one of the $4! = 24$ associahedra that correspond to the 4! permutations of the leaves labels
- this choice of one vertex on one of the 24 associahedra is the Externalization map

- underlying syntactic object

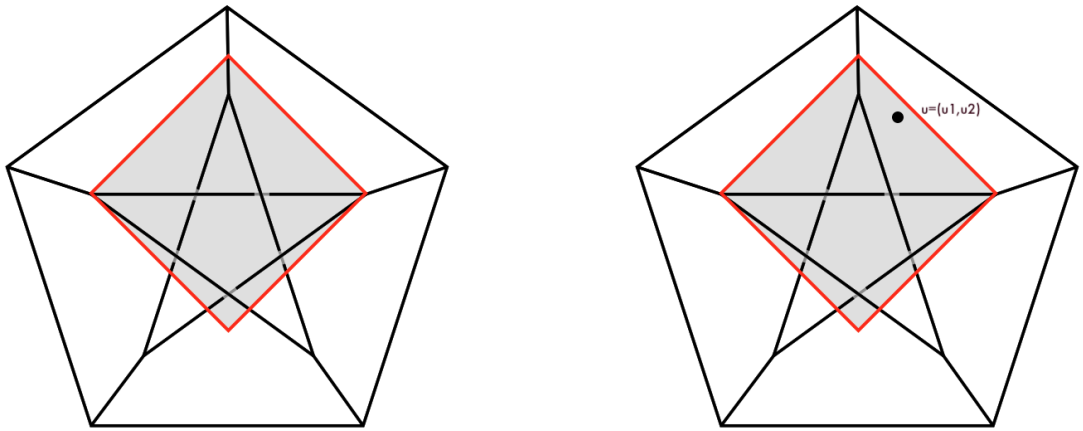
$$\{\{\alpha,\beta\},\{\gamma,\delta\}\} = \quad = \quad = \cdots$$



- this abstract tree is one of the $15 = (2n-3)!!$, for $n = 4$, possible abstract binary rooted trees on four labeled leaves
- these 15 possibilities are the 15 edges of the link $\mathcal{L}_4$ of the origin in the moduli space BHV_4, so the syntactic object selects an edge



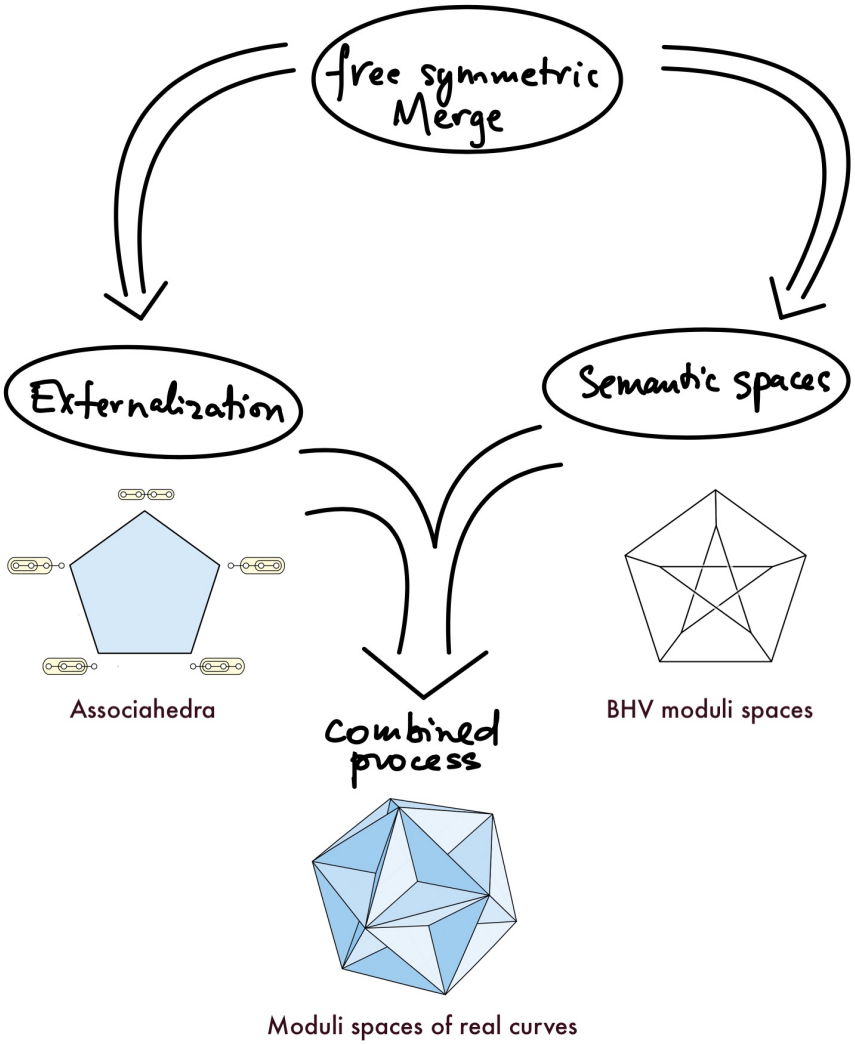
- given a semantic space $\mathcal{S}$ and relatedness measures $u_1 = \mathbb{P}(s(\alpha), s(\beta))$ (relating "yellow" and "flower") and $u_2 = \mathbb{P}(s(\gamma), s(\delta))$ (relating "blooming" and "early") in $\mathcal{S}$
- get two coordinates $(u_1, u_2) \in [0, 1]^2$
- the selected edge of $\mathcal{L}_4$ is the line $u_1 + u_2 = 1$ (normalized link of origin)
- so obtain a point in (a square tile of) BHV_4^+



- this is the map of the syntax-semantics interface

syntax-semantic interface and Externalization: geometric model

the compatibility of these two maps (Externalization in K_4 and syntax-semantics interface in BHV_4) is through compatibility of associahedra and BHV moduli spaces via $\overline{M}^{or}_{0,5}(\mathbb{R})$, given by the map $\overline{M}^{or}_{0,5}(\mathbb{R}) \twoheadrightarrow \mathrm{BHV}^{+}_{4}$



geometric view of syntactic parameters

- revisit Externalization as language-dependent section of a projection

$$\overline{M}^{or}_{0,n+1}(\mathbb{R}) \underset{\Pi_n}{\overset{\sigma_{L,n}}{\rightleftarrows}} \mathrm{BHV}^{+}_{n}$$

- dependence on syntactic parameters: covering transformations $\sigma_{L,n} = \gamma_{L,L',n} \circ \sigma_{L',n}$
- compare with other algorithms like LCA (on domain of *head*) $\sigma_{L,n} = \gamma_{L,n} \circ \sigma_{LCA,n}$
- possibility of studying syntactic parameters in terms of properties of symmetric groups (orders etc)

relevant for Geometry of Syntax question

- the region of "possible languages" among all configurations of syntactic parameters (A. Moro)
- estimations of dimension and geometric structure of locus of possible languages (topology/geometry), e.g.
  - A.Ortegaray, R.C.Berwick, M.Marcolli, "Heat kernel analysis of syntactic structures". Math. Comput. Sci. 15 (2021), no. 4, 643–660.
  - A.Port, T.Karidi, M.Marcolli, "Topological analysis of syntactic structures". Math. Comput. Sci. 16 (2022), no. 1, Paper No. 2, 68 pp.
- add further geometric information coming from comparison of sections $\sigma_{L,n}$ for different languages L (detects differences in parameters that affect word order)

Question of the geometry of syntactic parameters is a main open problem suitable for mathematical treatment

semiring parsing

- relation between grammars and semirings first developed by Chomsky–Schützenberger (1963)
- then commonly used framework of *semiring parsing* for *context free* grammars (or mildly context sensitive like TAGs)
- main setting: deduction rules of the form

$$\frac{A_1 \dots A_k}{B} C_1 \dots C_\ell$$

  - terms A_i (main conditions) are rules R of the grammar or input nonterminals
  - C_i are (non-probabilistic) Boolean side conditions
  - fraction notation means that if the numerator terms hold then the denominator term also does
- to main conditions one assigns semiring values, combine with semiring operations, obtain value for deduced output
- Question: what type of algebraic structure replaces this form of semiring parsing in Minimalism based on free symmetric Merge action on workspaces?

Roadmap: there's a bit of math to unpack in this part of the story

- first step (warmup): revisiting Minimal Search as an example of Birkhoff factorization of a character of the Hopf algebra of workspaces with *target* a ring of (Laurent series of) Merge derivations
- second step: need to incorporate Merge derivations as *source* of parsing, this requires passing from Hopf algebras to Hopf algebroids (composition on matching source/target of Merge action)
- third step: *target* of parsing correspondingly needs to adapt from algebras/semirings to (a suitable notion of) algebroids/semiringoids with a suitable notion of Rota–Baxter structure
- fourth step: then "semiring parsing" becomes Birkhoff factorization again, but in this "-iod" setting

First warmup step: revisiting Minimal Search

Recall from earlier: different forms of Merge and action on workspaces

- **EM:** $F = T \sqcup T' \sqcup \hat{F} \mapsto F' = \mathfrak{M}(T, T') \sqcup \hat{F}$
- **IM:** $F = T \sqcup \hat{F} \mapsto F' = \mathfrak{M}(T_v, T/T_v) \sqcup \hat{F}$
- **SM(i):** $F = T \sqcup T' \sqcup \hat{F} \mapsto F' = \mathfrak{M}(T_v, T'_w) \sqcup T/T_v \sqcup T'/T'_w \sqcup \hat{F}$
- **SM(ii):** $F = T \sqcup T' \sqcup \hat{F} \mapsto F' = \mathfrak{M}(T, T'_w) \sqcup T'/T'_w \sqcup \hat{F}$
- **CM(i):** $F = T \sqcup \hat{F} \mapsto F' = \mathfrak{M}(T_v, T_w) \sqcup T/T_v \sqcup \hat{F}$
- **CM(ii):** $F = T \sqcup \hat{F} \mapsto F' = \mathfrak{M}(T_v, T_w) \sqcup T/T_w \sqcup \hat{F}$
- **CM(iii):** $F = T \sqcup \hat{F} \mapsto F' = \mathfrak{M}(T_v, T_w) \sqcup T/(T_v \sqcup T_w) \sqcup \hat{F}$

$\hat{F}$ denotes part of the workspace that remains unaffected

All but EM and IM are eliminated by Minimal Search and effect on size of workspaces (Resource Restriction and Minimal Yield): these are not two different mechanisms but the same

size counting again

- $b_0(F)$ number of connected components, $\alpha(F)$ number of accessible terms, $\sigma(F) = b_0(F) + \alpha(F) = \#V(F)$ and $\hat{\sigma}(F) = b_0(F) + \sigma(F)$
- introduce combined size variable

$$\delta = -\Delta(3b_0 + \alpha) = -\Delta(2b_0 + \#V)$$

	Δb_0	$\Delta\alpha$	$\Delta\sigma$	$\Delta\hat{\sigma}$	δ
EM	-1	$+2$	$+1$	0	1
IM	0	0	0	0	0
SM(i)	$+1$	0	$+1$	$+2$	-3
SM(ii)	0	$+1$	$+1$	$+1$	-1
CM(i)	$+1$	$\#Acc(T_{a,w_a})$	$\sigma(T_{a,w_a})$	$\sigma(T_{a,w_a}) + 1$	≤ -2
CM(ii)	$+1$	$\#Acc(T_{a,v_a})$	$\sigma(T_{a,v_a})$	$\sigma(T_{a,v_a}) + 1$	≤ -2
CM(iii)	$+1$	-2	-1	0	≤ -1

- extracting cases with $\delta \geq 0$ eliminates all unwanted forms of Merge (keeps only EM and IM)

algebra of Merge derivations

- $\mathcal{DM}$ is the commutative associative $\mathbb{Q}$-algebra with the underlying vector space spanned by

$$\varphi_A = (F \overset{\mathfrak{M}_A}{\to} F')$$

$A \subset \mathcal{SO} \times \mathcal{SO}$ set of pairs (S, S') of syntactic objects

$$F \overset{\mathfrak{M}_{S_1,S_1'}}{\longrightarrow} F_1 \to \cdots F_{N-1} \overset{\mathfrak{M}_{S_N,S_N'}}{\longrightarrow} F'$$

all possible chains of Merge operations with $(S_i, S_i') \in A$

- algebra multiplication, for $\varphi_A = (F \overset{\mathfrak{M}_A}{\to} F')$ and $\varphi_B = (\tilde{F} \overset{\mathfrak{M}_B}{\to} \tilde{F}')$

$$\varphi_A \sqcup \varphi_B = (F \sqcup \tilde{F} \overset{\mathfrak{M}_{A \sqcup B}}{\to} F' \sqcup \tilde{F}')$$

- meaning of product: perform in parallel different Merge operations that affect different parts of a workspace
- unit empty forest mapped to itself

Laurent series and Birkhoff factorization

- commutative associative algebra $\mathcal{A}$ and the algebra of Laurent series $\mathcal{A}[t^{-1}][[t]]$
- linear operator $R : \mathcal{A}[t^{-1}][[t]] \to \mathcal{A}[t^{-1}][[t]]$ that projects onto the polar part

$$R(\sum_{i=-N}^{\infty} a_i t^i) = \sum_{i=-N}^{-1} a_i t^i$$

- makes $(\mathcal{A}[t^{-1}][[t]], R)$ a Rota–Baxter algebra of weight -1
- Note: this is the way to "subtract divergences" in physics
- here consider $\mathcal{DM}[t^{-1}][[t]]$ Laurent series with coefficients in the algebra of Merge derivations $\mathcal{DM}$

- character: morphism of commutative algebras
  $\phi_t : \mathcal{H} \to \mathcal{DM}[t^{-1}][[t]]$

$$\phi_t(F) = (L(F) \overset{\mathfrak{M}_{A(L(F),F)}}{\longrightarrow} F)\ t^{\delta(\mathfrak{M}_{A(L(F),F)})}$$

- set $A(L(F), F)$ of all Merge derivations from the (multi)set of individual lexical items (leaves $L(F)$) to forest F
- Note: this character has
  $\delta(T) = (2b_0 + \#V)(L) - (2b_0 + \#V)(T) =$
  $3\ell - 2 - (2\ell - 1) = \ell - 1 \geq 0$ so always in
  $\mathcal{DM}[[t]] = (1 - R)\,\mathcal{DM}[t^{-1}][[t]]$ (no polar part)
- modify with

$$\psi_t(T) = \sum_{(F,F') \in \mathcal{F}_T} (F \overset{\mathfrak{M}_{A(F,F')}}{\longrightarrow} F')\ t^{\delta(\mathfrak{M}_{A(F,F')})}$$

considers all the intermediate derivations from $L(T)$ to T, weighted with corresponding δ

observation

- failure of the Rota–Baxter operator to be an algebra homomorphism

$$R((\sum_{i=-N}^{\infty} a_i t^i)(\sum_{j=-M}^{\infty} b_j t^j)) = R(\sum_{n=-(N+M)}^{\infty} \sum_{i+j=n} a_i b_j \ t^n)$$

$$= \sum_{n=-(N+M)}^{-1} \sum_{i+j=n} a_i b_j \ t^n$$

$$\neq R(\sum_{i=-N}^{\infty} a_i t^i) R(\sum_{j=-M}^{\infty} b_j t^j) = \sum_{n=-(N+M)}^{-1} \sum_{i+j=n, i<0, j<0} a_i b_j \ t^n$$

- in a product of series can end up in the polar (respectively, non-polar) part of the product without being in the polar (respectively, non-polar) part of the individual factor, because of t^{i+j}

- so just projecting with $(1 - R)$ on $\phi(F)$ not sufficient to get rid of unwanted forms of Merge
- *but...* Birkhoff factorization achieves that result
- $\psi_{t,+}(T) = (1 - R)\tilde{\psi}_t(T)$ alg homom $\psi_{t,+} : \mathcal{H} \to \mathcal{DM}[[t]]$

$$\tilde{\psi}_t(T) = \psi_t(T) + \sum \psi_{t,-}(F_{\underline{v}})\psi_t(T/F_{\underline{v}})$$

- more details: if there is a term in $\psi_t(T)$ of the form $(F \to F')t^{\delta}$ where the derivation is a Sideward or Countercyclic Merge, the forest F' will occur as a collection of accessible terms $F' = F_{\underline{v}}$ in T
- so in $\tilde{\psi}_t(T)$ the term $\psi_{t,-}(F_{\underline{v}})\psi_t(T/F_{\underline{v}})$ will contain a term $R(\psi_t(F'))\psi_t(T/F_{\underline{v}})$ which will contain a summand equal to $-(F \to F')t^{\delta}$
- has the effect of removing the unwanted derivation, while any term $(F \to F')t^{\delta}$ in $\psi_t(T)$ that only contains derivations using Internal/External Merge is not cancelled by anything coming from the terms $\psi_{t,-}(F_{\underline{v}})\psi_t(T/F_{\underline{v}})$

Next step: Birkhoff factorization in algebroids

- refine the previous construction: in $\mathcal{DM}$ commutative product only accounts for "independent" derivations that affect different parts of workspace (hence commute)
- want to incorporate all derivations in the algebraic structure
- something that generalizes "derivation forest semirings" of context-free semiring parsing
- key idea: composing Merge transformations on workspaces is like composing arrows (morphisms of a category), composition only defined when target of first arrow is source of second one
- difference between a *group* (composition always defined) and a *groupoid* (composition defined with matching target/source)
- commutative Hopf algebras are "dual to groups" (group schemes)... the notion dual to groupoids is *Hopf algebroids*

commutative Hopf algebroid (dually groupoid scheme)

- pair of commutative algebras $\mathcal{A}^{(0)}$ and $\mathcal{H}^{(1)}$
- for any other commutative algebra $\mathcal{R}$, sets $\mathcal{G}^{(0)}(\mathcal{R}) = \mathrm{Hom}(\mathcal{A}^{(0)}, \mathcal{R})$ and $\mathcal{G}^{(1)}(\mathcal{R}) = \mathrm{Hom}(\mathcal{H}^{(1)}, \mathcal{R})$ are the objects and morphisms of a groupoid $\mathcal{G}$
- unpack this:
  - pair of commutative algebras $(\mathcal{A}^{(0)}, \mathcal{H}^{(1)})$ (functions on objects and arrows)
  - homomorphisms $\eta_s, \eta_t : \mathcal{A}^{(0)} \to \mathcal{H}^{(1)}$ give $\mathcal{H}^{(1)}$ the structure of a $\mathcal{A}^{(0)}$-bimodule (dual to source and target)
  - coproduct given by morphism of $\mathcal{A}^{(0}$-bimodules

$$\Delta : \mathcal{H}^{(1)} \to \mathcal{H}^{(1)} \otimes_{\mathcal{A}^{(0)}} \mathcal{H}^{(1)}$$

    (dual to composition of arrows in groupoid)
  - conjugation $S : \mathcal{H}^{(1)} \to \mathcal{H}^{(1)}$ (dual to inverse of morphisms in groupoid)
- bialgebroid: same without S, dual to a "semigroupoid" (small category) instead of groupoid

further properties

- counit $\epsilon : \mathcal{H}^{(1)} \to \mathcal{A}^{(0)}$ morphism of $\mathcal{A}^{(0}$-bimodules (dual to identity morphisms)
- $\epsilon\eta_s = \epsilon\eta_t = 1$ (identity morphisms have same source and target)
- $(1 \otimes \epsilon)\Delta = (\epsilon \otimes 1)\Delta = 1$ (composition with the identity morphism)
- $(1 \otimes \Delta)\Delta = (\Delta \otimes 1)\Delta$ (associativity of composition of morphisms)
- $S^2 = 1$ and $S\eta_s = \eta_t$ (inversion is an involution and exchanges source and target of morphisms)
- composition of a morphism with its inverse is identity morphism $\eta_t\epsilon = \mu(S \otimes 1)\Delta$ and $\eta_s\epsilon = \mu(1 \otimes S)\Delta$ $\mu : \mathcal{H}^{(1)} \otimes_{\mathcal{A}^{(0)}} \mathcal{H}^{(1)} \to \mathcal{H}^{(1)}$ extending the algebra multiplication $\mu : \mathcal{H}^{(1)} \otimes_{\mathbb{Q}} \mathcal{H}^{(1)} \to \mathcal{H}^{(1)}$
- $\Delta\eta_s = 1 \otimes \eta_s$, $\Delta\eta_t = \eta_t \otimes 1$ (source of composition of arrows is source of the first and target of composition is target of second)
- morphism $f : (\mathcal{A}_1^{(0)}, \mathcal{H}_1^{(1)}) \to (\mathcal{A}_2^{(0)}, \mathcal{H}_2^{(1)})$: algebra homomorphisms $f^{(0)} : \mathcal{A}_1^{(0)} \to \mathcal{A}_2^{(0)}$ and $f^{(1)} : \mathcal{H}_1^{(1)} \to \mathcal{H}_2^{(1)}$ with $f^{(0)} \circ \epsilon_1 = \epsilon_2 \circ f^{(1)}$, $f^{(1)} \circ \eta_{s,1} = \eta_{s,2} \circ f^{(0)}$, $f^{(1)} \circ \eta_{t,1} = \eta_{t,2} \circ f^{(0)}$, $f^{(1)} \circ S_1 = S_2 \circ f^{(1)}$, $\Delta_2 \circ f^{(1)} = (f^{(1)} \otimes f^{(1)}) \circ \Delta_1$

bialgebroid of Merge derivations (replaces "derivation forests")

- data $\mathcal{A}^{(0)} = (\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}), \sqcup)$ and $\mathcal{H}^{(1)} = (\mathcal{DM}, \sqcup)$, define a bialgebroid
  - left and right $\mathcal{A}^{(0)}$-module structures (source and target)

$$\eta_s(F)\varphi_A = \begin{cases} \varphi_A & s(\varphi_A) = F \\ 0 & \text{otherwise} \end{cases} \qquad \eta_t(F)\varphi_A = \begin{cases} \varphi_A & t(\varphi_A) = F \\ 0 & \text{otherwise} \end{cases}$$

  - coproduct

$$\Delta(\varphi_A) = \varphi_A \otimes 1 + 1 \otimes \varphi_A + \sum_{\varphi_A = \varphi_{A_1} \circ \varphi_{A_2}} \varphi_{A_1} \otimes \varphi_{A_2}$$

where $\varphi_{A_2} = (F \overset{\mathfrak{M}_{A_2}}{\to} F')$ and $\varphi_{A_1} = (F' \overset{\mathfrak{M}_{A_1}}{\to} F'')$ with composition

$$\varphi_{A_1} \circ \varphi_{A_2} = (F \overset{\mathfrak{M}_{A_1 \circ A_2}}{\to} F'')$$

$\mathfrak{M}_{A_1 \circ A_2} = \mathfrak{M}_{A_1} \circ \mathfrak{M}_{A_2}$ set of all compositions of a chain of Merge derivations in set A_2 followed by one in A_1

- Note: coproduct of $\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0})$ (Hopf algebra of workspaces) now built into the arrows φ_A as part of Merge action

Rota-Baxter algebroids

- Merge bialgebroid as replacement of context-free derivation forests
- want then replacement of semiring parsing using again the Birkhoff factorization idea
- instead of $\phi : \mathcal{H} \to \mathcal{R}$ from Hopf algebra to Rota-Baxter algebra (or semiring) need analog from Hopf *algebroids* (or bialgebroid) to a suitable generalization of a Rota-Baxter algebra (or semiring) ... *algebroid* (*semiringoid*)
- Warning: different notions of *algebroid*, *semiringoid* are used in math, ours is motivated by compatibility with the notion of Hopf algebroid and bialgebroid
- so here algebroid like part of bialgebroid that does not involve coproduct
  - pair of commutative algebras $(\mathcal{A}, \mathcal{E})$
  - two morphisms $\eta_s, \eta_t : \mathcal{A} \to \mathcal{E}$ that make $\mathcal{E}$ bimodule over $\mathcal{A}$
  - morphism of $\mathcal{A}$-bimodules $\epsilon : \mathcal{E} \to \mathcal{A}$ with $\epsilon\eta_s = \epsilon\eta_t = 1_{\mathcal{A}}$

- if Hopf algebroids are like functions on a groupoid, bialgebroids on a small category (semigroupoid), what about algebroids? ... functions on a *directed graph*
- $(\mathcal{A}, \mathcal{E})$ commutative algebroid: for any commutative algebra $\mathcal{R}$ the sets $V(\mathcal{R}) = \mathrm{Hom}(\mathcal{A}, \mathcal{R})$ and $E(\mathcal{R}) = \mathrm{Hom}(\mathcal{E}, \mathcal{R})$ are sets of vertices and edges of a directed graph $G(\mathcal{R})$ with source and target maps $s, t : E(\mathcal{R}) \to V(\mathcal{R})$ determined by the morphisms $\eta_s, \eta_t : \mathcal{A} \to \mathcal{E}$ *(directed graph scheme)*
- also each vertex $v \in V(\mathcal{R})$ has a looping edge $e_v \in E(\mathcal{R})$ with $s(e_v) = t(e_v) = v$
- bialgebroid = case where the directed graph satisfies reflexivity and transitivity (small category); Hopf algebroid = case where reflexivity, transitivity, symmetry (groupoid)

Rota-Baxter structure on algebroids

- commutative algebroid $(\mathcal{A}, \mathcal{E})$ with pair of maps $R = (R_V, R_E)$ with $R_V \in \mathrm{End}(\mathcal{A})$ algebra hom and $R_E : \mathcal{E} \to \mathcal{E}$ linear

  $$R_E(\eta_s(a)\cdot\xi) = \eta_s(R_V(a))\cdot R_E(\xi) \quad R_E(\eta_t(a)\cdot\xi) = \eta_t(R_V(a))\cdot R_E(\xi)$$

  and $\epsilon \circ R_E = R_E \circ \epsilon$ with Rota–Baxter identity (weight -1)

  $$R_E(\xi) \cdot R_E(\zeta) = R_E(R_E(\xi) \cdot \zeta) + R_E(\xi \cdot R_E(\zeta)) - R_E(\xi \cdot \zeta)$$

  normalization $R_E(1_{\mathcal{E}}) = 0$ or $R_E(1_{\mathcal{E}}) = 1_{\mathcal{E}}$, for $1_{\mathcal{E}}$ the unit of the algebra $\mathcal{E}$
- Main example: functions on edges of a directed graph, with values in a Rota-Baxter algebra with Rota-Baxter operator acting only on coefficients of functions
  - G directed graph, $(\mathcal{R}, R)$ Rota–Baxter algebra
  - $\mathcal{A} = \mathbb{Q}[V_G]$ and $\mathcal{E} = \mathbb{Q}[E_G] \otimes_{\mathbb{Q}} \mathcal{R}$
  - morphisms $\eta_s, \eta_t : \mathcal{A} \to \mathcal{E}$ precomposition with $s, t : E_G \to V_G$
  - $R_V = \mathrm{id}$ and $R_E = 1 \otimes R$

semiringoids

- $(\mathcal{A}, \mathcal{E})$ two commutative semirings
- semiring homomorphisms $\eta_s, \eta_t : \mathcal{A} \to \mathcal{E}$ that make $\mathcal{E}$ bi-semimodule over $\mathcal{A}$
- bi-semimodule homomorphism $\epsilon : \mathcal{E} \to \mathcal{A}$ with $\epsilon\eta_s = \epsilon\eta_t = 1_{\mathcal{A}}$

Rota-Baxter semiringoid (weight $+1$)

- semiringoid $(\mathcal{A}, \mathcal{E})$ with semiring endomorphism $R_V : \mathcal{A} \to \mathcal{A}$ and an $R_E : \mathcal{E} \to \mathcal{E}$ morphism of $\mathbb{Z}_{\geq 0}$-semimodules with

$$R_E(\eta_s(a) \odot \xi) = \eta_s(R_V(a)) \odot R_E(\xi)$$

$$R_E(\eta_t(a) \odot \xi) = \eta_t(R_V(a)) \odot R_E(\xi)$$

and $\epsilon \circ R_E = R_E \circ \epsilon$, with Rota–Baxter relation of weight $+1$

$$R_E(\xi) \odot R_E(\zeta) = R_E(R_E(\xi) \odot \zeta) \boxdot R_E(\xi \odot R_E(\zeta))$$

Birkhoff factorization in algebroids/semiringoids

- $(\mathcal{A}^0), \mathcal{H}^{(1)})$ Hopf algebroid and $(\mathcal{A}, \mathcal{E})$ algebroid with Rota–Baxter structure (R_V, R_E) weight -1
- morphism $\Phi : (\mathcal{A}^{(0)}, \mathcal{H}^{(1)}) \to (\mathcal{A}, \mathcal{E})$ of algebroids
- have inductive construction of factorization

$$\Phi_{+,E}(f) = (\Phi_{-,E} \star \Phi_E)(f) = (\Phi_{-,E} \otimes \Phi_E)(\Delta f)$$

$$\Phi_{-,E}(f) = -R_E(\tilde{\Phi}_E(f))$$

$$\tilde{\Phi}_E(f) = \Phi_E(f) + \sum \Phi_{-,E}(f')\Phi_E(f'')$$

for $\Delta(f) = f \otimes 1 + 1 \otimes f + \sum f' \otimes f''$, and with $\Phi_{+,E}(f) = (1 - R_E)(\tilde{\Phi}_E(f))$

Similar for case of semiringoids

what does this mean?

- algebroid $(\mathcal{A}, \mathcal{E}) \Leftrightarrow$ directed graph G
- $\Phi : (\mathcal{A}^{(0)}, \mathcal{H}^{(1)}) \to (\mathcal{A}, \mathcal{E})$ map of graphs $\alpha : G \to \mathcal{G}$ (where $\mathcal{G}$ also a category)
- take $f = \delta_\gamma$ for γ an arrow in $\mathcal{G}$
- if $R = \mathrm{id}$ (trivial RB structure weight -1)

$$\tilde{\Phi}_E(\delta_\gamma) = \sum_{e \in E_G \,:\, \alpha(e) = \gamma} \delta_e + \cdots + \sum_{e_1, \ldots, e_n \in E_G \,:\, \gamma = \alpha(e_1) \circ \cdots \circ \alpha(e_n)} \delta_{e_1} \cdots \delta_{e_n}$$

lists all the possible ways of obtaining γ as compositions of images of arrows in G

- for a weighted combination $\sum_i \lambda_i e_i$ in diagram G (eg probability) with R_E RB

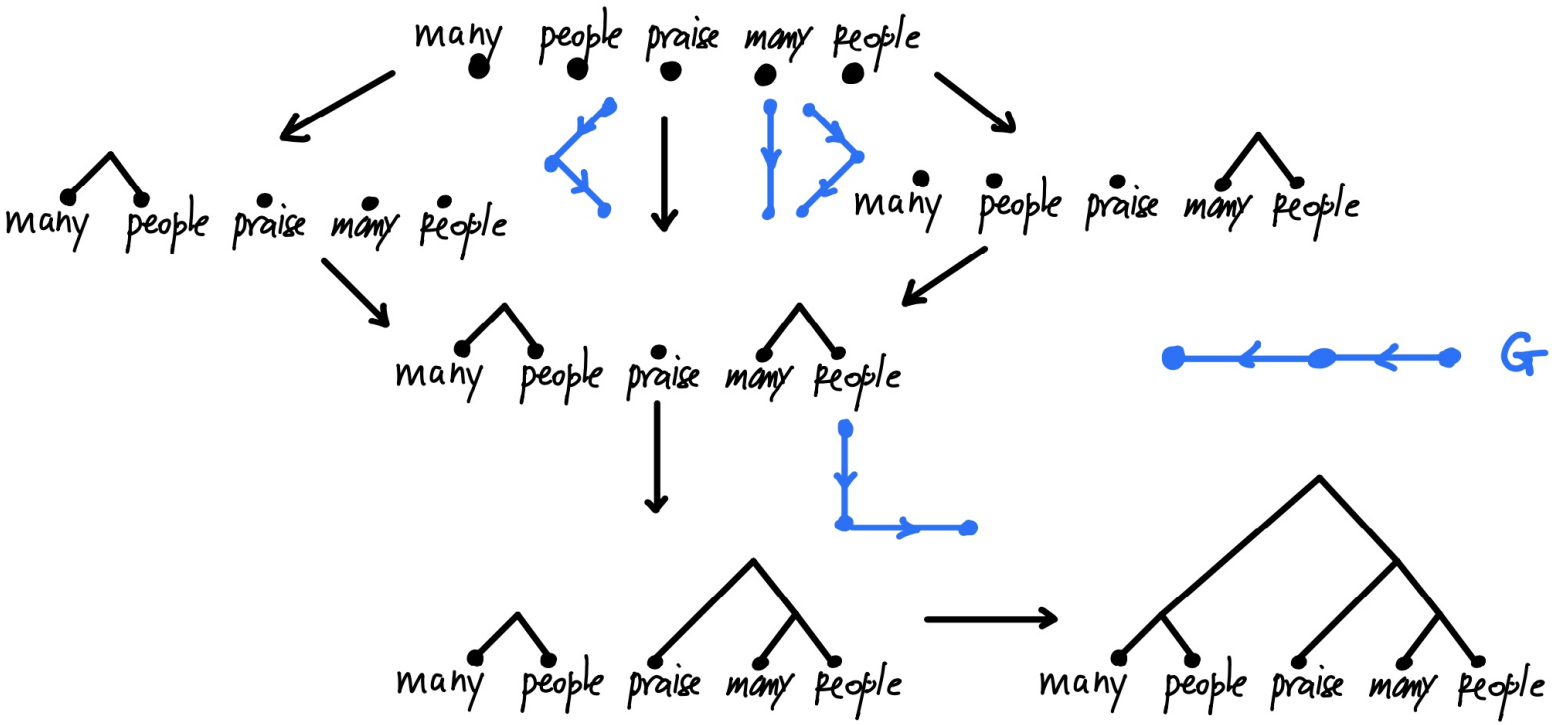
$$\Phi_{E,-}(\delta_\gamma)(\sum_i \lambda_i e_i) = -(\sum_{\alpha(e) = \gamma} R_E(\lambda_e) + \sum_{\alpha(e_1) \circ \alpha(e_2) = \gamma} R_E(R_E(\lambda_{e_1}) \lambda_{e_2}) + \cdots$$

$$+ \sum_{\alpha(e_1) \circ \cdots \circ \alpha(e_n) = \gamma} R_E(\cdots (R_E(\lambda_{e_1}) \cdots) \lambda_{e_n}))$$

- case of Merge derivations: target functions on a graph values in semiring $(\mathbb{R} \cup \{-\infty\}, \max, +)$ with R ReLU or in Viterbi $([0,1], \max, \cdot)$ with $R = c_\lambda$ threshold
- map Φ assigns a possible *diagram of Merge derivations*
- *checking all possible ways of realizing some given chain of Merge derivations γ through compositions coming from the chosen diagram, weighted by elements in the given semiring and filtered by R*
  - $\mathcal{R} = (\mathbb{R} \cup \{-\infty\}, \max, +)$ with R =ReLU: all possibilities with weights of each step above the ReLU threshold
  - $\mathcal{R} = ([0,1], \max, \cdot)$ with the threshold $R = c_\lambda$ all possibilities with probabilities of each step above threshold
  - Boolean semiring $\mathcal{B} = (\{0,1\}, \max, \cdot)$ with $R = \mathrm{id}$: derivations γ realized through diagram G with truth values on each edge and composition of arrows = AND operation on truth values, different paths of derivations = OR operation on truth values

Example

- consider the chain of Merge derivations for the sentence "many people praise many people" (example from "Merge & SMT" §3.4) and choice of model diagram G for parsing



Pietroski's semantics and Merge

- independent existence *within* semantics of a *Combine* binary operation that parallels the functioning of Merge in syntax
- in Pietroski's formulation *Combine* = *Label* ∘ *Concatenate* two operations

$$\mathit{Concatenate}(\alpha, \beta) = \{\alpha, \beta\} = \overset{\frown}{\alpha \quad \beta}$$

$$\mathit{Combine}(\alpha, \beta) = \mathit{Label} \circ \mathit{Concatenate}(\alpha, \beta)$$
$$= \mathit{Label}(\overset{\frown}{\alpha \quad \beta}) = h(\alpha, \beta)$$
$$\overset{\frown}{\alpha \quad \beta}$$

- binary operation *Combine* is not symmetric because of the *head* label
- if compositional operation takes place in semantic space $\mathcal{S}$, then $\mathcal{S}$ needs to have own computational system (at least partially defined): two systems each with "Merge" type operation, one for syntax one for semantics
- different from other conceptual spaces (perceptual manifolds for vision)

Main claim: Merge suffices

all computational structure is on the side of syntax

- start with map $s : \mathcal{SO}_0 \to \mathcal{S}$ extended to $s : \mathrm{Dom}(h) \to \mathcal{S}$ (using some property like geodesic convexity on $\mathcal{S}$)
- the i-concept *Combine*(α, β) where $\alpha = s(T_1)$ and $\beta = s(T_2)$ is well defined if $T = \mathfrak{M}(T_1, T_2) \in \mathrm{Dom}(h)$ and given by

$$\textit{Combine}(\alpha, \beta) := s(\mathfrak{M}(T_1, T_2)) \in \mathcal{S}$$

  with $s(T)$ constructed using geodesic arcs and a semantic proximity $\mathcal{P}$ as discussed before
- Note: no need to separate *Combine* into *Label* and *Concatenate*
- check that this is OK with some potential issues (idempotents, rule out improper inferences)

idempotents

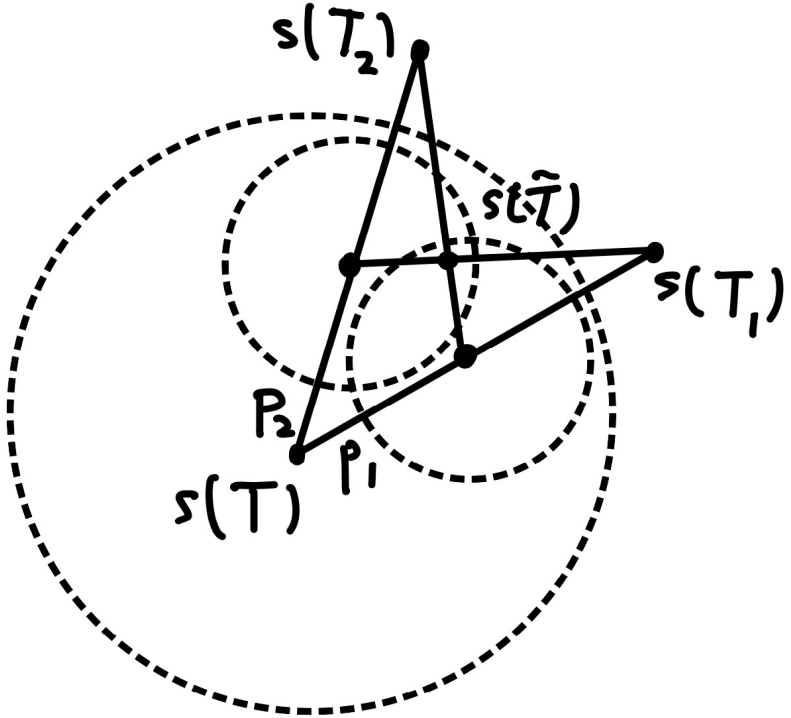
- on the semantics side expect possible idempotent structures $\mathit{Combine}(\alpha, \alpha) = \alpha$
- never have this on the syntax side: $\mathfrak{M}(T, T) = \overbrace{T \quad T} \neq T$
- is this a problem with $\mathit{Combine}(\alpha, \beta) := s(\mathfrak{M}(T_1, T_2))$?
- no: it just means $s : \mathrm{Dom}(h) \to \mathcal{S}$ not always an embedding
- location of the point $s(\mathfrak{M}(T, T'))$ on geodesic arc between $s(T)$ and $s(T')$ depends on $\mathbb{P}(s(T), s(T'))$
- if $\mathbb{P}(s, s') = 0$ or $\mathbb{P}(s, s') = 1$ obtain cases where

$$\mathit{Combine}(\alpha, \beta) = \alpha \quad \text{or} \quad \mathit{Combine}(\alpha, \beta) = \beta$$

even if $\mathfrak{M}(T, T') \neq T$ and $\mathfrak{M}(T, T') \neq T'$

inference: Example (adjuncts to verb)

- consider sentences: "John ate a sandwich in the basement" and "John ate a sandwich at noon",
- these two sentences clearly do *not* imply that "John ate a sandwich in the basement at noon"
- one can see this in terms of the construction of $s(\mathfrak{M}(T_1, T_2))$



how to fit in this model predicate saturation in Pietroski's compositional semantics?

- revisit mathematical structure of syntactic objects (free nonassociative commutative magma)
- another aspect: operad and algebra over an operad (operad action)
- operad action on syntactic objects and compatibly on semantic spaces
- compositional semantics (including predicate saturation) in this compatibility relation

operads

- operad (in category of Sets) is a collection $\mathcal{O} = \{\mathcal{O}(n)\}_{n\geq 1}$ of sets of n-ary operations (with n inputs and one output), with composition laws

$$\gamma : \mathcal{O}(n) \times \mathcal{O}(k_1) \times \cdots \times \mathcal{O}(k_n) \to \mathcal{O}(k_1 + \cdots + k_n)$$

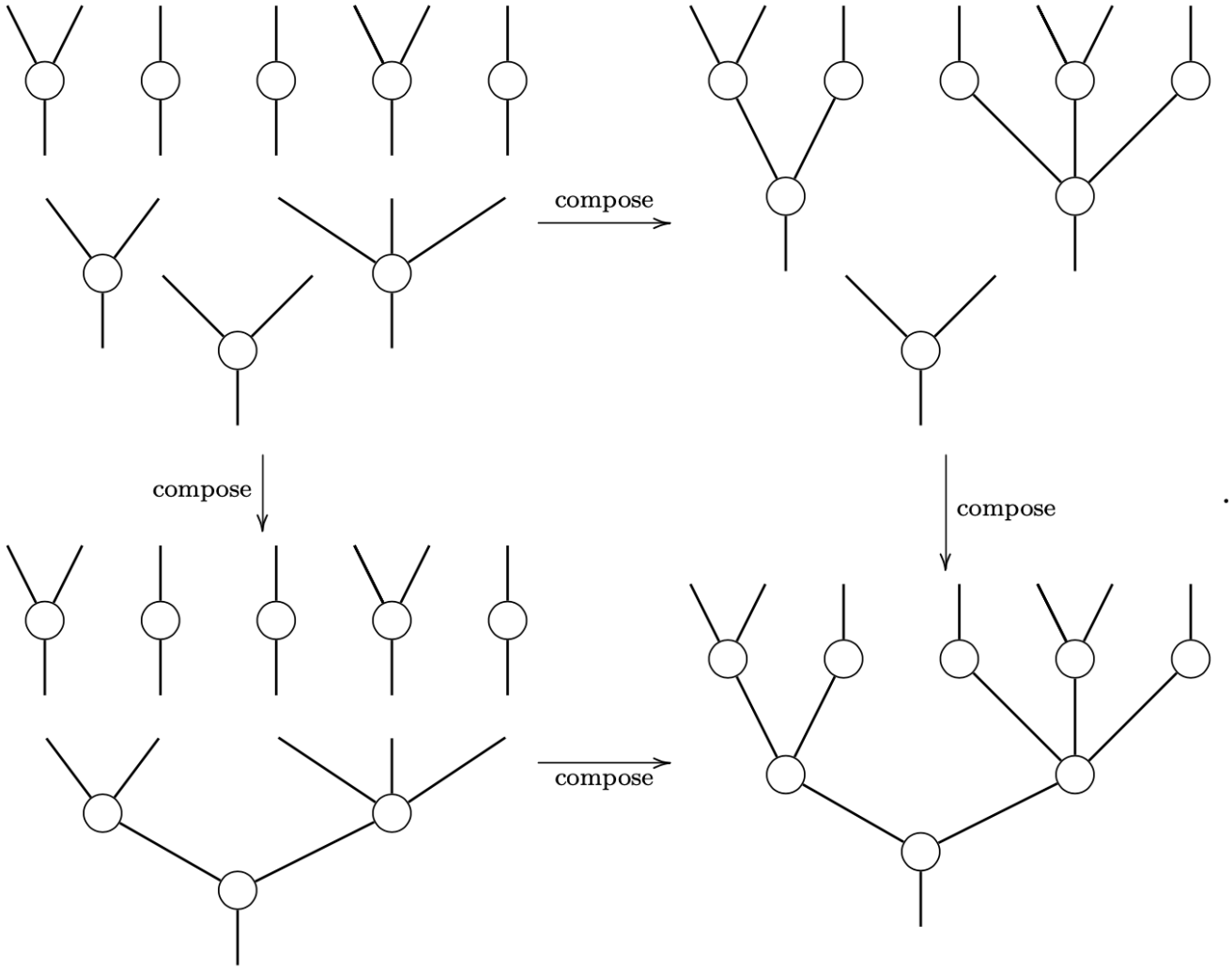
  plugging output of operations in $\mathcal{O}(k_i)$ into the i-th input of operations in $\mathcal{O}(n)$

- associativity of operad composition

$$\gamma(\gamma(T, T_1, \ldots, T_m), T_{1,1}, \ldots, T_{1,n_1}, \ldots, T_{m,1}, \ldots, T_{m,n_m}) =$$

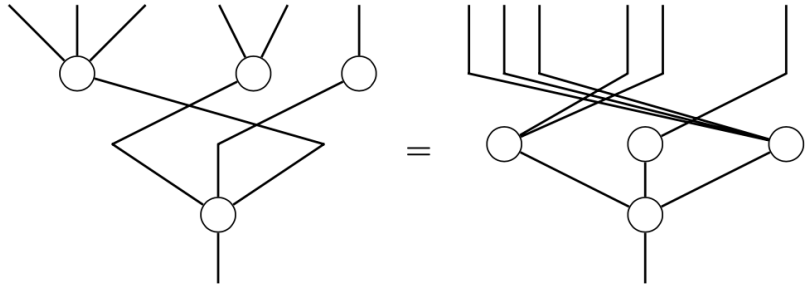
$$\gamma(T, \gamma(T_1, T_{1,1}, \ldots, T_{1,n_1}), \ldots, \gamma(T_m, T_{m,1}, \ldots, T_{m,n_m}))$$

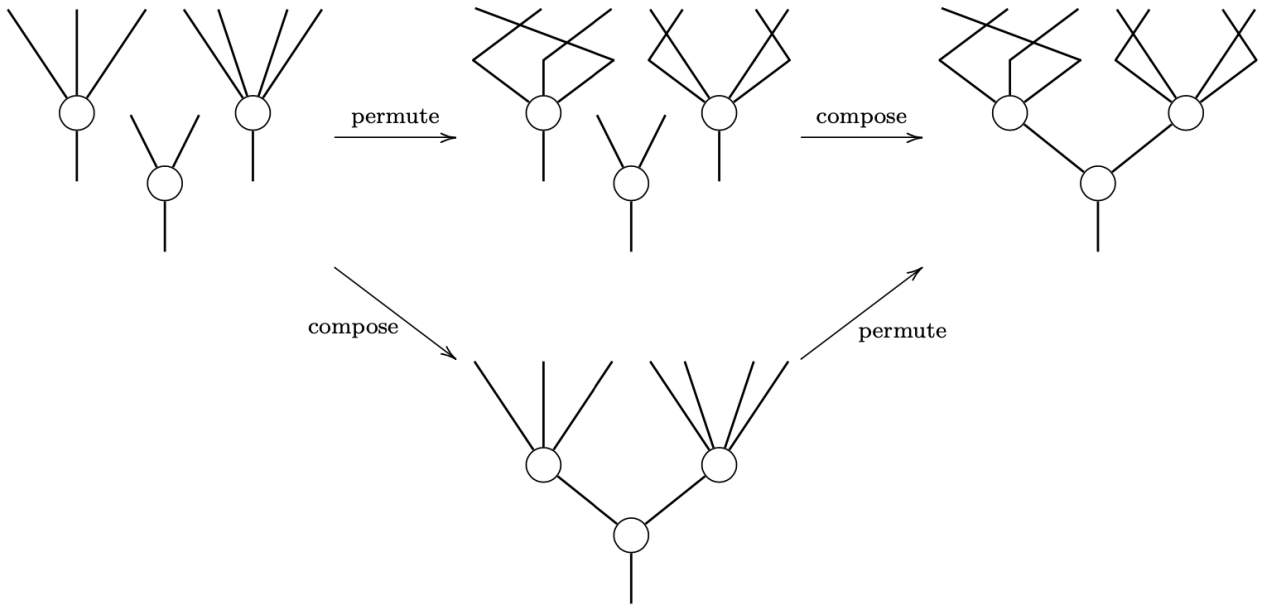
- unit $1 \in \mathcal{O}(1)$ for composition



associativity of operad composition

case of symmetric operad: two equivariance conditions with respect to actions of symmetric groups





algebras over operads

- algebra $\mathcal{A}$ over an operad $\mathcal{O}$ (in Sets) is a set $\mathcal{A}$ on which the operations of $\mathcal{O}$ act
- can use elements of $\mathcal{A}$ as inputs for operations in $\mathcal{O}(n)$

$$\gamma_{\mathcal{A}} : \mathcal{O}(n) \times \mathcal{A}^n \to \mathcal{A}$$

- compositionality: for $T \in \mathcal{O}(m)$, $T_i \in \mathcal{O}(n_i)$, $\{a_{i,j}\}_{j=1}^{n_i} \subset \mathcal{A}$

$$\gamma_{\mathcal{A}}(\gamma_{\mathcal{O}}(T, T_1, \ldots, T_m), a_{1,1}, \ldots, a_{1,n_1}, \ldots, a_{m,1}, \ldots, a_{m,n_m}) =$$

$$\gamma_{\mathcal{A}}(T, \gamma_{\mathcal{A}}(T_1, a_{1,1}, \ldots, a_{1,n_1}), \ldots, \gamma_{\mathcal{A}}(T_m, a_{m,1}, \ldots, a_{m,n_m})) .$$

with $\gamma_{\mathcal{O}}$ composition in operad and $\gamma_{\mathcal{A}}$ operad action

Merge operad and action on syntactic objects

- operad $\mathcal{M}$ freely generated by a single commutative binary operation $\mathfrak{M}$
- have $\mathcal{M}(1) = \{\mathrm{id}\}$, $\mathcal{M}(2) = \{\mathfrak{M}\}$, $\mathcal{M}(3) = \{\mathfrak{M} \circ (\mathrm{id} \times \mathfrak{M}), \mathfrak{M} \circ (\mathfrak{M} \times \mathrm{id})\}$, etc.
- action $\gamma_{\mathcal{SO}} : \mathcal{M}(n) \times \mathcal{SO}^n \to \mathcal{SO}$ with $\gamma_{\mathcal{SO}}(T, T_1, \ldots, T_n)$ with $T \in \mathcal{M}(n)$ and $T_i \in \mathcal{SO}$ for $i = 1, \ldots, n$ abstract binary rooted tree in $\mathfrak{T}_{\mathcal{SO}_0} = \mathcal{SO}$ obtained by grafting root of the syntactic object T_i to the i-th leaf of $T \in \mathcal{M}(n)$
- if syntactic objects T_i have n_i leaves, then syntactic object $\gamma(T, T_1, \ldots, T_n)$ obtained in this way has $n_1 + \cdots + n_k$ leaves

a subtlety about abstract trees

- no choice of ordering of leaves, so better way of writing action
- $\mathfrak{T}_{\mathcal{SO}_0,k}$, $\mathfrak{T}_k$ abtract binary trees with k leaves (with/without leaves lexical items)
- for $T \in \mathfrak{T}_n$ and $T_\ell \in \mathfrak{T}_{\mathcal{SO}_0,k_\ell}$

$$\gamma(T, \{T_\ell\}_{\ell \in L(T)})$$

root of the tree T_ℓ is grafted to the leaf $\ell \in L(T)$

- associativity of compositions

$$\gamma(\gamma(T, \{T_\ell\}_{\ell \in L(T)}), \{T'_{\ell'}\}_{\ell' \in L(\gamma(T, \{T_\ell\}_{\ell \in L(T)}))})$$

$$= \gamma(T, \{\gamma(T_\ell, \{T'_{\ell'}\}_{\ell' \in L(T_\ell)})\}_{\ell \in L(T)})$$

action of permutation groups

- first equivariance condition

$$\gamma(T \circ \tau, \{T_{\tau(\ell)}\}) = \gamma(T, \{T_\ell\}) \circ \tau'$$

$\tau \in \mathrm{Sym}(L(T)) \simeq S_n$, with $n \in \#L(T)$, and S_n the symmetric group, and with $\tau' \in \mathrm{Sym}(L(\gamma(T, \{T_\ell\})) \simeq S_{\sum_\ell k_\ell}$ that permutes the n blocks of k_ℓ leaves, leaving each block unchanged

- second equivariance condition

$$\gamma(T, \{T_\ell \circ \sigma_\ell\}_{\ell \in L(T)}) = \gamma(T, \{T_\ell\}_{\ell \in L(T)}) \circ \underline{\sigma}$$

$\sigma_\ell \in \mathrm{Sym}(L(T_\ell)) \simeq S_{k_\ell}$ with
$\underline{\sigma} \in \mathrm{Sym}(L(\gamma(T, \{T_\ell\})) \simeq S_{\sum_\ell k_\ell}$ that permutes the leaves within each block of k_ℓ leaves, leaving the position of the blocks unchanged

- for simplicity of notation use $\gamma(T, T_1, \ldots, T_n)$ instead of $\gamma(T, \{T_\ell\}_{\ell \in L(T)})$

compatibility with head function

- $\mathcal{M}_h(n)$ set of pairs (T, h_T) an abstract binary rooted tree $T \in \mathfrak{T}_n$ (with no labeling at the n leaves) and a *head function* $h_T : V^{int}(T) \to L(T)$
- composition in $\mathcal{M}$ induces operad structure on $\mathcal{M}_h = \{\mathcal{M}_h(n)\}$
- operad compositiom

$$\gamma_{\mathcal{M}_h} : \mathcal{M}_h(n) \times \mathcal{M}_h(k_1) \times \cdots \times \mathcal{M}_h(k_n) \to \mathcal{M}_h(k_1 + \cdots + k_n)$$

- data $h_T, h_{T_1}, \ldots, h_{T_n}$ combine to define a *head function* on $T' = \gamma_{\mathcal{M}}(T, T_1, \ldots, T_n)$:
  - all vertices of T' that are vertices of one of the trees T_i: set $h_{T'}(v) = h_{T_i}(v)$
  - vertices of T' that are non-leaf vertices of T, we define $h_{T'}(v)$ by: *head function* on T determines a leaf $h_T(v) \in L(T)$; $T_{i(\ell)}$ denote the tree that is grafted to the leaf $\ell \in L(T)$; take $h_{T'}(v) := h_{T_{i(h_T(v))}}$

operad action and semantic spaces

- $\mathcal{S}^+ = \mathcal{S} \cup \{s_\infty\}$ be the Alexandrov one-point compactification of a topological semantic space $\mathcal{S}$
- action of the operad $\mathcal{M}$ on syntactic objects together with function $s : \mathrm{Dom}(h) \to \mathcal{S}$ determine an action of the operad $\mathcal{M}$ on $\mathcal{S}^+$

$$\gamma_{\mathcal{S}}(T, s_1, \ldots, s_n) := \begin{cases} s(\gamma_{\mathcal{SO}}(T, T_1, \ldots, T_n)) & \text{if } \begin{array}{l} s_i = s(T_i) \text{ and} \\ \gamma_{\mathcal{SO}}(T, T_1, \ldots, T_n) \in \mathrm{Dom}(h) \end{array} \\ s_\infty & \text{otherwise} \end{cases}$$

for $T \in \mathcal{M}(n)$ and $s_1, \ldots, s_m \in \mathcal{S}^+$

- induces by restriction operad action of $\mathcal{M}_h$ on $\mathcal{S}$

compositional semantic spaces

- $\mathcal{S}$ is a *compositional semantic space* if

  1. There is a map $s : \mathrm{Dom}(h) \to \mathcal{S}$ extending $s : \mathcal{SO}_0 \to \mathcal{S}$.
  2. There is an action of the operad $\mathcal{M}$ on $\mathcal{S}$

  $$\gamma_{\mathcal{S}} : \mathcal{M}(n) \times \mathcal{S}^n \to \mathcal{S}$$

  3. For $T \in \mathcal{M}(n)$ and for $T_1, \ldots, T_n \in \mathrm{Dom}(h) \subset \mathcal{SO}$ such that

  $$\gamma_{\mathcal{SO}}(T, T_1, \ldots, T_n) \in \mathrm{Dom}(h)$$

  we have

  $$\gamma_{\mathcal{S}}(T, s(T_1), \ldots, s(T_n)) = s(\gamma_{\mathcal{SO}}(T, T_1, \ldots, T_n))$$

- the operation $\gamma_{\mathcal{S}} : \mathcal{M}(n) \times \mathcal{S}^n \to \mathcal{S} \Rightarrow$ predicate saturation
- same setting also accounts for theta-theory

Theta theory and operads

- so far only used *head function*: more refined information
  theta-theory: identify and remove the ill-formed sentences by structure of dependants (complements) of the *head*, in particular assignment of θ-roles
- theta-theory models thematic relations between predicates and their arguments
- predicates assign *θ-roles* to their arguments
- θ-roles of arguments of predicates: "theme", "agent", "experiencer", "locative", "instrument", "possessor", etc
- there should be a one-to-one correspondence between theta roles and arguments these are assigned to
- dichotomy in semantics:
  1. External Merge (EM) sole responsible for assignment of theta-roles: argument structure, propositional domain
  2. Internal Merge (IM) does no theta-structure: clausal domain, information-related, non-argument structure, displacement
- in our setting indeed theta-theory related to operad structure of syntactic objects (which only uses EM)

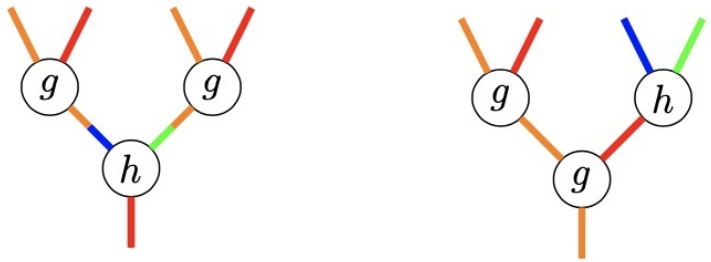
colored operads

- notion of colored operads similar to passing from group to groupoid etc: make operad compostions defined under some matching conditions (matching "colors" in a set Θ)
- collection $\mathcal{O} = \{\mathcal{O}(c, c_1, \ldots, c_n)\}$ of sets, with $c, c_i \in \Theta$ for $i = 1, \ldots, n$, c_i are color labels of inputs and c color label of output
- composition laws have to match colors

$$\gamma : \quad \mathcal{O}(c, c_1, \ldots, c_n) \times \mathcal{O}(c_1, c_{1,1}, \ldots, c_{1,k_1}) \times \cdots \times \mathcal{O}(c_n, c_{n,1}, \ldots, c_{n,k_n}) \\ \to \mathcal{O}(c, c_{1,1}, \ldots, c_{1,k_1}, \ldots, c_{n,1}, \ldots, c_{n,k_n})$$

- similar associativity, unity (one unit 1_c per color), and symmetric properties

main idea: use colors for different θ-roles and matching rules of colored operad composition ensure correct consistent assignment of θ-roles



Example of compositions with mismatched and with matched color assignments (assignment of θ-roles)

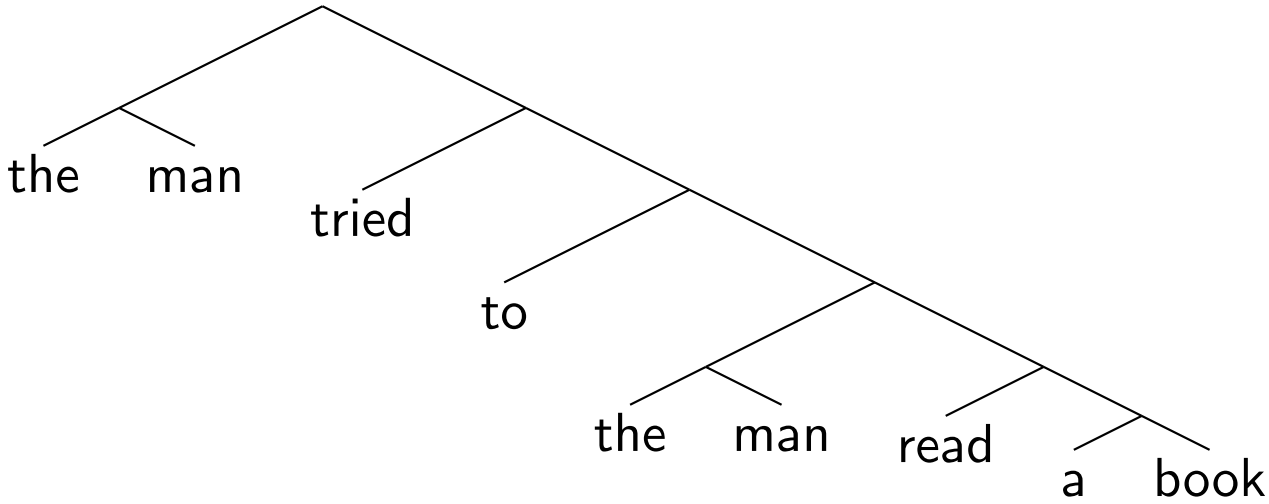
Theta theory and colored operads

- set Θ of θ-roles and θ relations: labels "predicate" or "argument" and for arguments θ-roles labels "theme", "agent", "experiencer", "locative", "instrument", "possessor", etc.
- $\mathrm{Dom}_\Theta(h) \subset \mathrm{Dom}(h)$ set of syntactic objects $T \in \mathcal{SO}$ in domain of *head function h* admitting assignment of labels in Θ to edges of T compatible, on each substructure accessible term T_v with *head* and complement determined by *head function h*
- set $\mathrm{Dom}_\Theta(h) \subset \mathcal{SO}$ determines a colored operad $\mathcal{M}_{h,\Theta} = \{\mathcal{M}_{h,\Theta}(\theta, \theta_1, \ldots, \theta_n)\}$
- consequence: all n-ary theta-structures (elements $(T, h_T, \theta_T) \in \mathcal{M}_{h,\Theta}(\theta, \theta_1, \ldots, \theta_n)$) are composition of binary theta-structures through repeated application of binary External Merge building elements of $\mathcal{M}$

deriving obligatory control: an example ("Merge and SMT" §5.3)

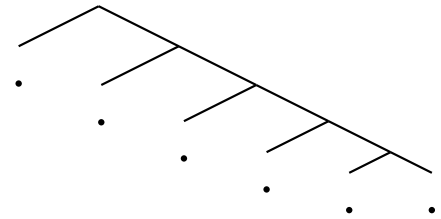
- sentence "the man tried to read a book"

$$\{\{\text{the}, \text{man}\}, \{\text{tried}, \{\text{to}, \{\{\text{the}, \text{man}\}, \{\text{read}, \{\text{a}, \text{book}\}\}\}\}\}\} =$$

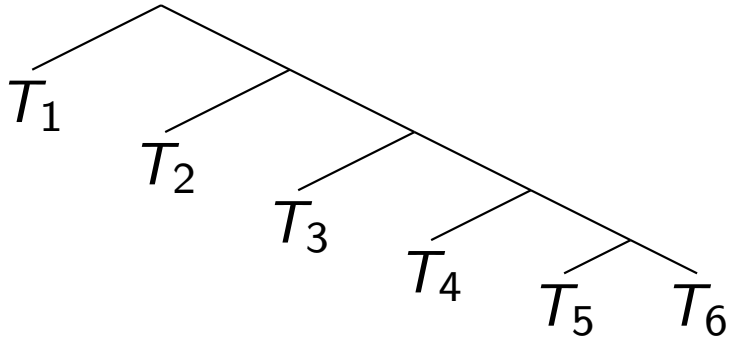


$$\mathfrak{M}(\text{the man}, \mathfrak{M}(\text{tried}, \mathfrak{M}(\text{to}, \mathfrak{M}(\text{the man}, \text{read a book})))))$$

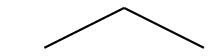
- in terms of operad action: element $T \in \mathcal{M}(6)$ of the form

$T =$ 

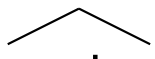
- inputs $T_1, \ldots, T_6$ in $\mathcal{SO}^6$, output syntactic object



- here take

$T_1 = T_4 =$ 

$T_2 =$ tried $T_3 =$ to

$T_5 =$ read $T_6 =$ 

Operad action: restriction to diagonals (copies)

- for subset $\mathcal{I} \subset \{1, \ldots, n\}$ diagonal

$$\mathrm{Diag}_{\mathcal{I}} = \{(T_1, \ldots, T_n) \in \mathcal{SO}^n \mid T_i = \hat{T} \in \mathcal{SO}, \forall i \in \mathcal{I}\}$$

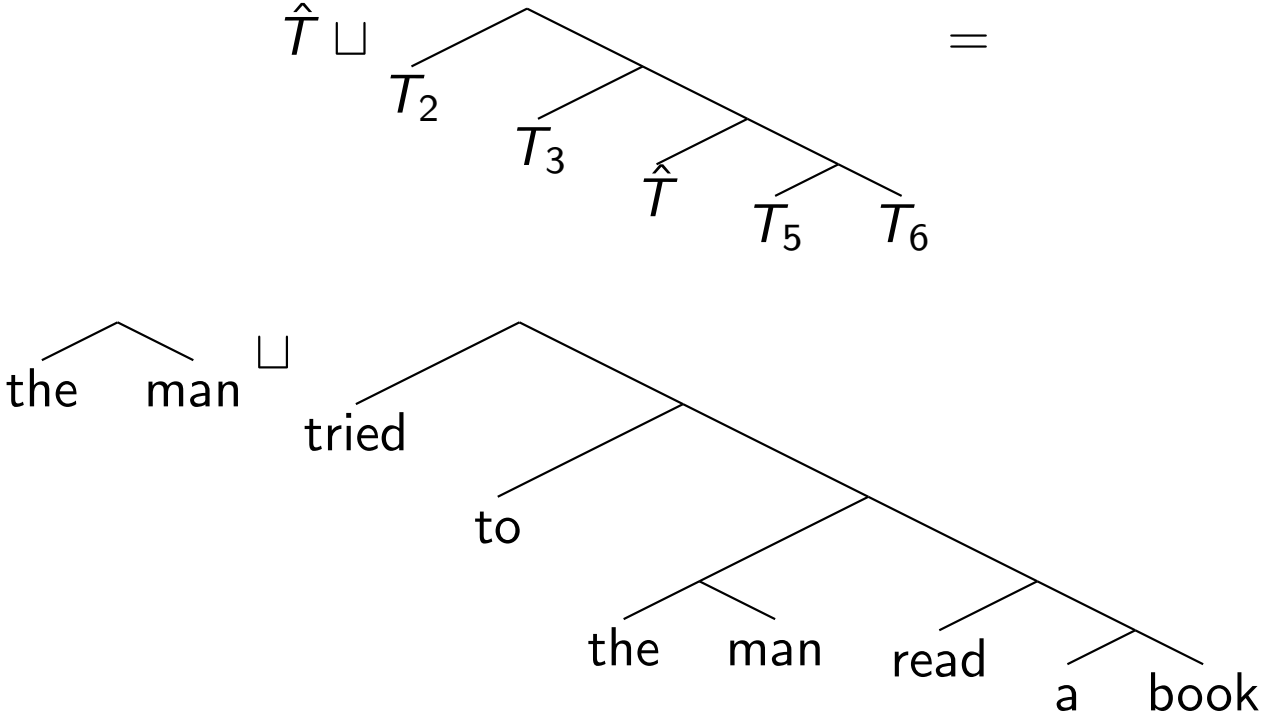
- in this example $\mathrm{Diag}_{1,4} \subset \mathcal{SO}^6$
- distinction between case of *repetitions* and *copies*: repetitions are *isomorphic* syntactic objects but not *identical*, copies are the same syntactic object
- effect: if repetitions, usual coproduct Δ in action of Merge, if copies identified so can only extract all at once (or same accessible term from all) or none

$$\Delta_{\mathcal{I}}(T) = \sum F_{\underline{v}} \otimes T/F_{\underline{v}} + \sum (F_{\underline{v}} \sqcup \hat{T}_v) \otimes (T/F_{\underline{v}})//\hat{T}_v$$

sums are over subforests $F_{\underline{v}} \subset T$ such that $\hat{T} \cap F_{\underline{v}} = \emptyset$, and where we write $T//\hat{T}_v$ to denote the quotient with respect to all occurrences of $\hat{T}_v$ in T as accessible terms of all the identified copies in $\mathrm{Diag}_{\mathcal{I}}$

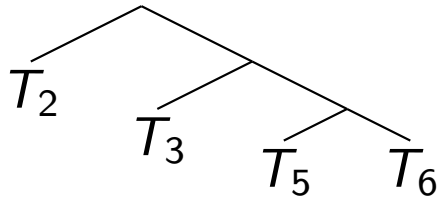
- still coassociative coproduct

- in the example considered start with a workspace forest

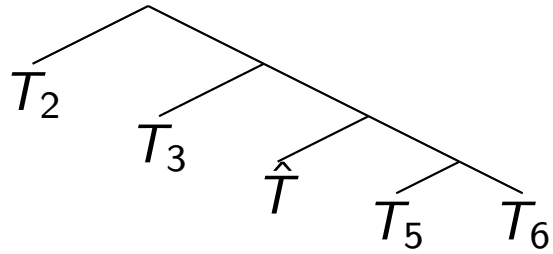


- a term in the coproduct $\Delta_{\mathcal{I}}$

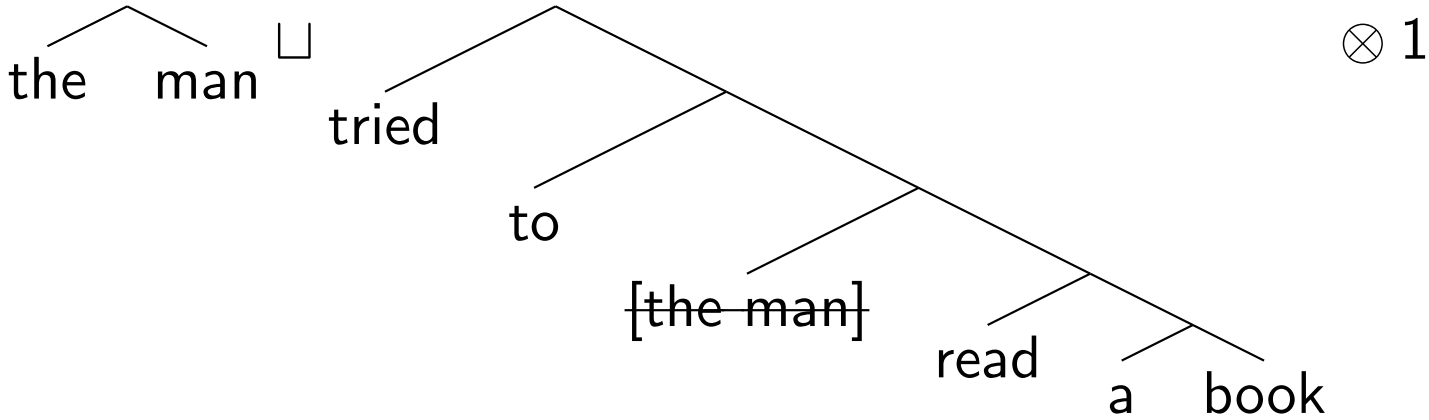
$$\hat{T} \sqcup T//\hat{T} \otimes 1, \quad \text{with} \quad T//\hat{T} =$$



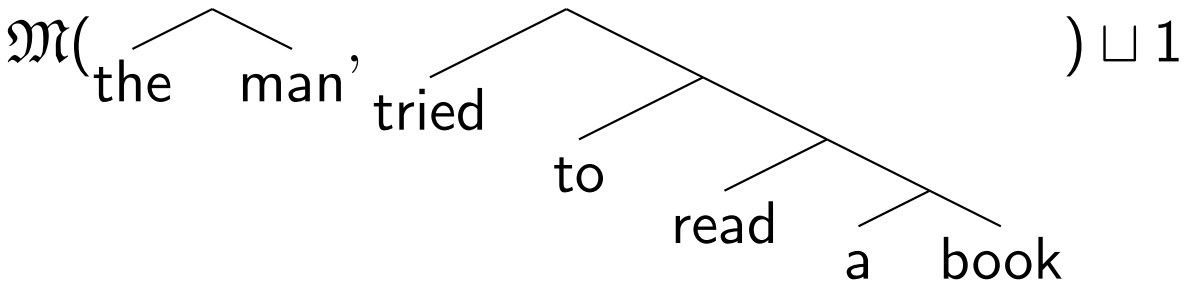
$$T =$$



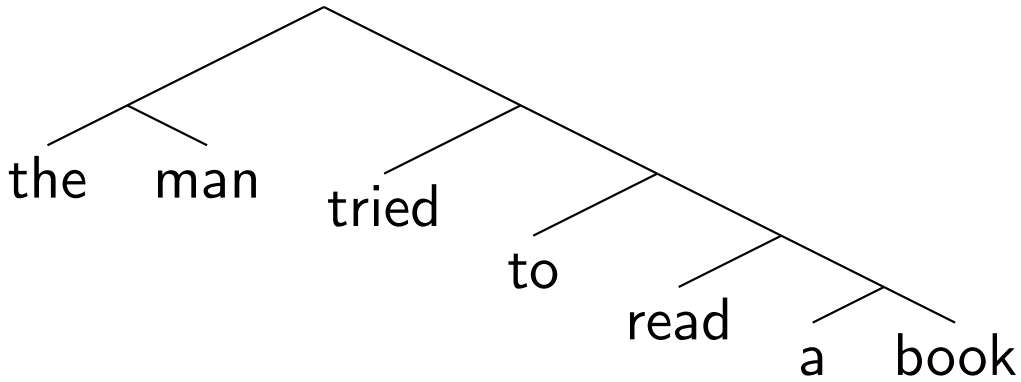
- this means coproduct term of the form



- coproduct term that is targeted by the External Merge producing



=

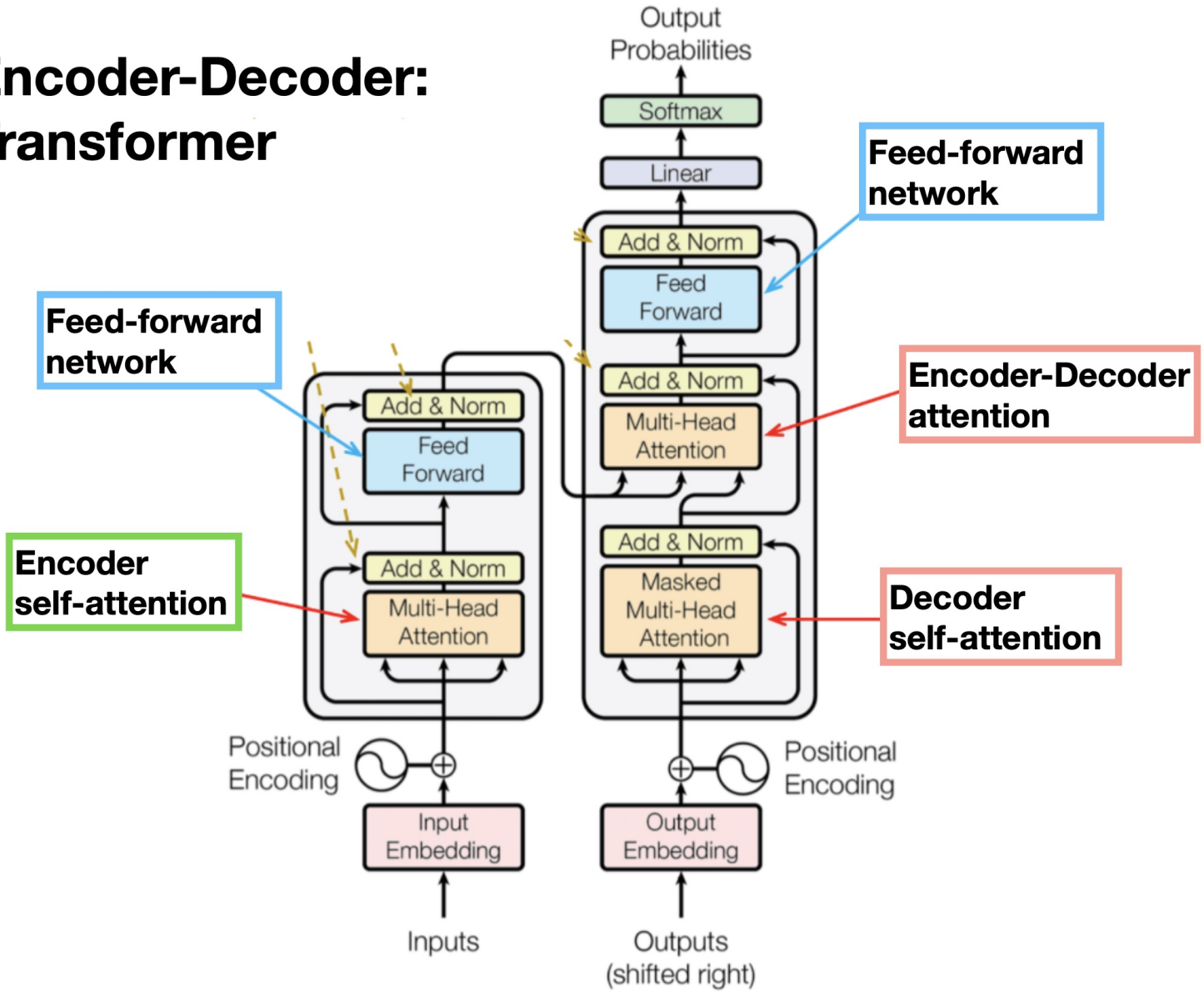


- if repetitions instead of copies (example "many people like many people") then the two accessible terms would be extracted independently by Δ not simultaneously by $\Delta_{\mathcal{I}}$ (so would not have cancellation of deeper copy)

Attention modules and transformer architectures

- despite widespread vacuous claims of incompatibility between current large language models (LLMs) and generative linguistics, the attention modules of transformer architectures of LLMs fit as another example of the Hopf algebra characters of our syntax-semantics interface model
- given map function $s : \mathcal{SO}_0 \to \mathcal{S}$ of lexical items to a (vector space) model of semantics
- focus here on attention modules, in the case of self-attention in transformer architectures

Encoder-Decoder: Transformer



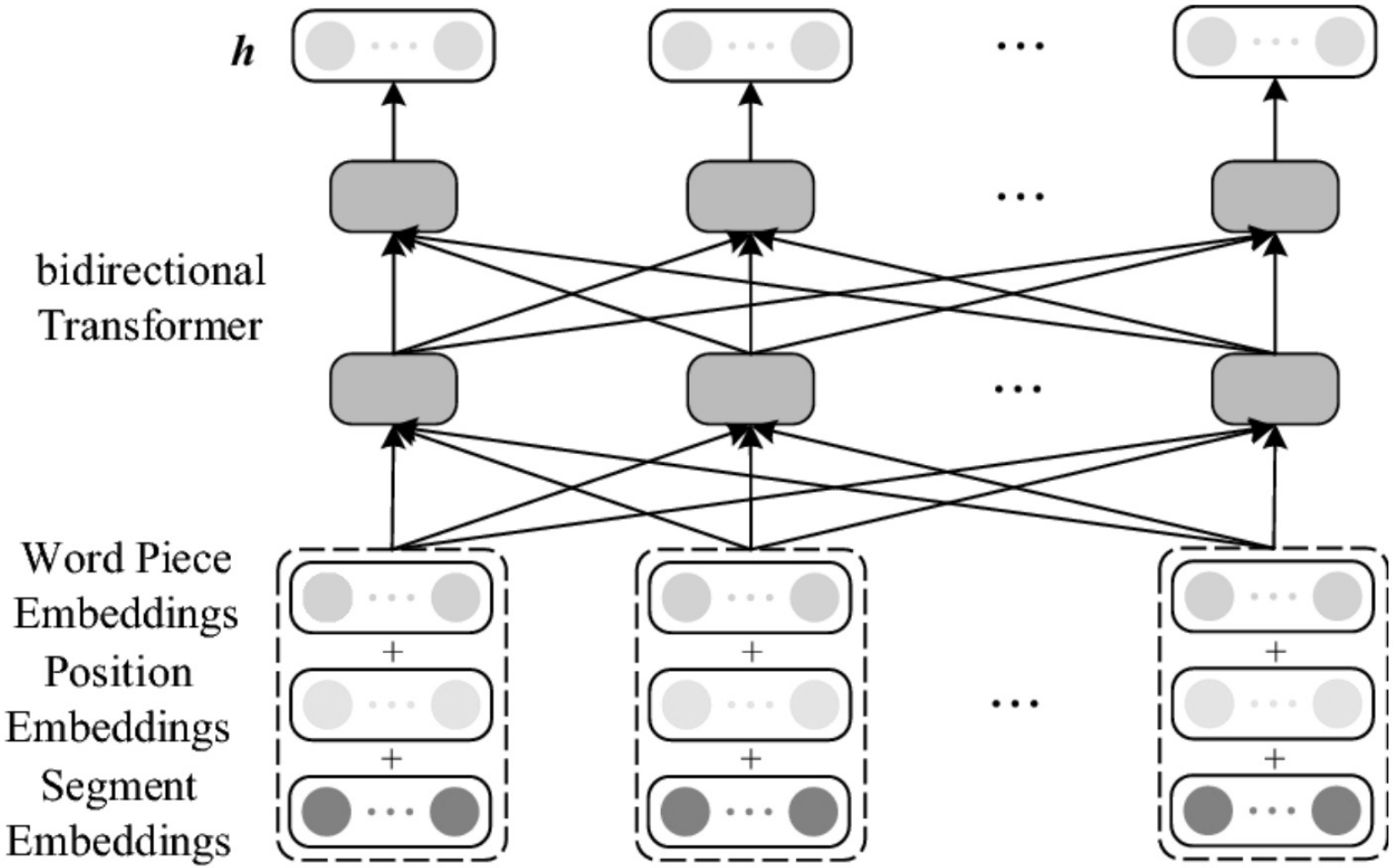
- in self-attention modules one considers three linear transformations: Q (queries), K (keys), and V (values), $Q, K \in \mathrm{Hom}(\mathcal{S}, \mathcal{S}')$ and $V \in \mathrm{Hom}(\mathcal{S}, \mathcal{S}'')$, where $\mathcal{S}'$ and $\mathcal{S}''$ are themselves vector spaces of semantic vectors (in general of dimensions not necessarily equal to that of $\mathcal{S}$)
- these encode (statistically) other words that are structurally related to ("called by" or "calling for") the given word
- fixed identifications $\mathcal{S} \simeq \mathbb{R}^n$, $\mathcal{S}' \simeq \mathbb{R}^m$, $\mathcal{S}'' \simeq \mathbb{R}^d$ with Euclidean vector spaces, with assigned bases, and one works with the corresponding matrix representations of $Q, K \in \mathrm{Hom}(\mathbb{R}^n, \mathbb{R}^m)$ and $V \in \mathrm{Hom}(\mathbb{R}^n, \mathbb{R}^d)$
- target Euclidean space $\mathcal{S}'$ is endowed with an inner product $\langle \cdot, \cdot \rangle$, that can be used to estimate semantic similarity

- query vector $Q(s(\ell))$, for $\ell \in \mathcal{SO}_0$, can be thought of performing a role analogous to the *semantic probes* discussed before
- think of queries as elements $q \in \mathcal{S}^\vee$ dual vector space $\mathcal{S}^\vee = \mathrm{Hom}(\mathcal{S}, \mathbb{R})$, so query matrix in

$$\mathcal{S}^\vee \otimes \mathbb{R}^m \simeq \mathcal{S}^\vee \otimes \mathcal{S}' = \mathrm{Hom}(\mathcal{S}, \mathcal{S}')$$

  m-fold probe Q evaluated on the given semantic vector $s(\ell)$
- similarly key vector $K(s(\ell))$, for $\ell \in \mathcal{SO}_0$, in $K \in \mathrm{Hom}(\mathcal{S}, \mathcal{S}')$, creating an m-fold probe out of the given vector $s(\ell)$
- dual role of $\mathcal{S}'$: (m-fold) probes to be evaluated on input semantic vector $s(\ell)$, or new probes generated by semantic vector $s(\ell)$ (reflected in terminology "query" and "key")
- values vector $V(s(\ell))$ representation of semantic vectors $s(\ell)$ in a vector space $\mathcal{S}''$ dimension lower than $\mathcal{S}$ ($d = \dim \mathcal{S}''$ embedding dimension)

- a set $L \subset \mathcal{SO}_0$: usually seen as an ordered set, but *in fact it should not be* (can use bi-directional architectures like BERT)



Neural network architecture of BERT. The input word piece, position and segment embeddings are summed

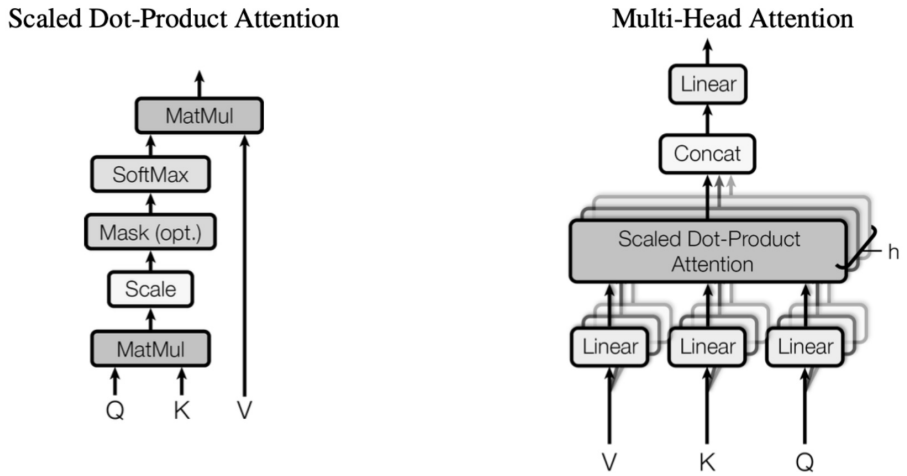
- to an element $\ell \in L$ assign attention operator $A_\ell : L \subset \mathcal{S} \to \mathcal{S}'$ given by

$$A_\ell(s(\ell')) = \sigma(\langle Q(s(\ell)), K(s(\ell'))\rangle)$$

where σ softmax $\sigma(x)_i = \exp(x_i)/\sum_j \exp(x_j)$, for $x = (x_i)$

- Note: ignoring usual rescaling by $\sqrt{d}$, no influence on algebraic structure
- $A_{\ell,\ell'} := A_\ell(s(\ell'))$ attention matrix
- $A_{\ell,\ell'}$ a probability measure on how attention from position ℓ is distributed towards other positions ℓ' in the set L
- assign an output (in $\mathcal{S}''$) to input $s(L) \subset \mathcal{S}$, as vectors $y_\ell = \sum_{\ell'} A_{\ell,\ell'} V(s(\ell'))$: for each $\ell \in L$, have $y_\ell = (y_\ell)_{i=1}^d \in \mathcal{S}'' \simeq \mathbb{R}^d$
- matrix representation $A_{\ell,\ell'}$ uses ordering of $\ell \in L$ but underlying linear operator does not; resulting y_ℓ also symmetric in ordering

- usually several such attention modules running in parallel: multi-head attention

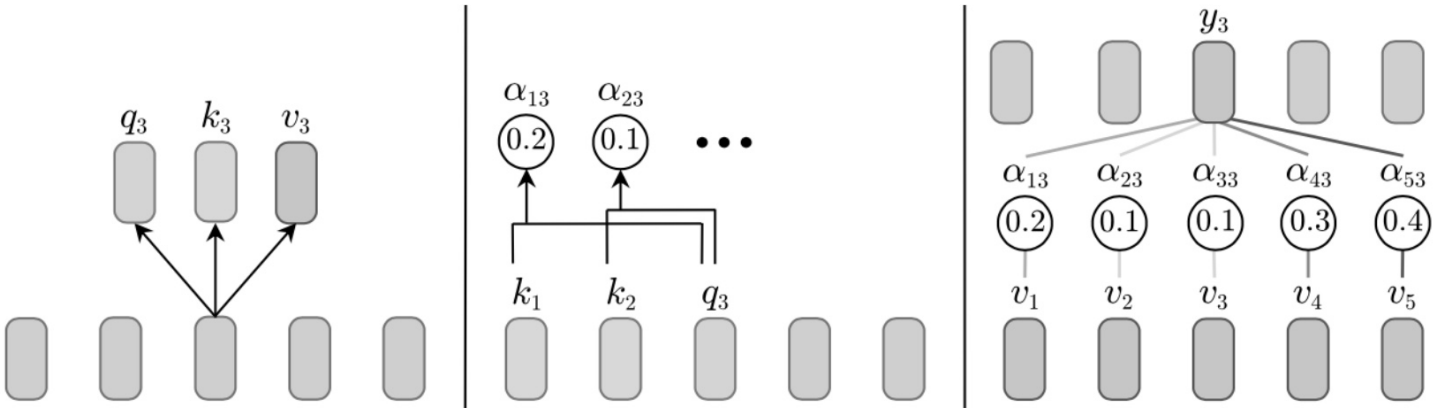


- vectors $Q(s(\ell)) = \oplus_i Q(s(\ell))_i$, $K(s(\ell)) = \oplus_i K(s(\ell))_i$, and $V(s(\ell)) = \oplus_j V(s(\ell))_j$ are split into blocks of decomposition $\mathcal{S}' = \oplus_{i=1}^{N} \mathcal{S}'_i$
- attention matrices, for $i = 1, \ldots, N$,

$$A^{(i)}_{\ell,\ell'} = \sigma(\langle Q(s(\ell))_i, K(s(\ell))_i \rangle_{\mathcal{S}'_i})$$

attention distribution with *attention head i*

- not consider here multiple attention heads: enough to use a single one to see the structure



queries, keys, values from input semantic vectors, attention matrices, probabilities, and weighted output

attention as character

- Hopf algebra character

$$\phi_A(T) = \max_{\ell \in L(T)} A_{h(T),\ell}$$

  if $T \in \mathrm{Dom}(h)$ and zero otherwise

- *syntactic relation*: collection $\rho = \rho_T$ of relations $\rho_T \subset L(T) \times L(T)$; equiv with $\rho_T(\ell, \ell') = 1$ is $\ell, \ell' \in L(T)$ in the relation and $\rho_T(\ell, \ell') = 0$ otherwise
- ρ *exactly attention-detectable* if $\exists$ query/key linear maps $Q_\rho, K_\rho \in \mathrm{Hom}(\mathcal{S}, \mathcal{S}')$ and *head function* h_ρ

$$\rho_T(h_\rho(T), \ell_{\max,h_\rho}) = 1$$

  for $T \in \mathrm{Dom}(h_\rho)$ with

$$\ell_{\max,h_\rho} = \mathrm{argmax}_{\ell \in L(T)} A_{h_\rho(T),\ell}$$

  $A =$ attention matrix built from Q_ρ, K_ρ

- syntactic relation ρ is *approximately attention-detectable* if $\exists$ query/key linear maps $Q_\rho, K_\rho \in \mathrm{Hom}(\mathcal{S}, \mathcal{S}')$ and *head function* h_ρ

$$\frac{1}{\#\mathcal{D}} \sum_{T \in \mathcal{D}} \rho(h_\rho(T), \ell_{\max, h_\rho}) \sim 1$$

  for some sufficiently large set $\mathcal{D} \subset \mathrm{Dom}(h_\rho)$ of trees
- existence of query/key linear maps Q_ρ, K_ρ is relative to specified context (a corpus, a dataset, etc)
- threshold Rota-Baxter operator c_λ

$$\begin{aligned}\phi_{A,-}(T) = \quad & c_\lambda(\max\{\phi_A(T), c_\lambda(\phi_A(F_{\underline{v}})) \cdot \phi_A(T/F_{\underline{v}}), \ldots, \\ & c_\lambda(\phi_A(F_{\underline{v_N}})) \cdot \phi_A(F_{\underline{v}_{N-1}}/F_{\underline{v_N}}) \cdots \phi_A(T/F_{\underline{v}_1})\}) .\end{aligned}$$

- for simplicity focusing on the case of chains of subtrees $T_{v_N} \subset T_{v_{N-1}} \subset \cdots \subset T_{v_1} \subset T$

attention along syntactic substructures

- for quotient given by contraction $h(T/T_v) = h(T)$ so

$$\max_{\ell \in L(T/T_v)} A_{h(T),\ell} \leq \max_{\ell \in L(T)} A_{h(T),\ell}$$

- $\phi_-(T)$ identifies chains of accessible terms of T for which
  1. all values

$$\phi_A(T_{v_i}) = \max_{\ell \in L(T_{v_i})} A_{h(T_{v_i}),\ell}$$

     are above threshold λ
  2. all the quotients $T_{v_{i-1}}/T_{v_i}$ have

$$\phi_A(T_{v_{i-1}}/T_{v_i}) = \max_{\ell \in L(T_{v_{i-1}}/T_{v_i})} A_{h(T_{v_{i-1}}),\ell} =$$

$$\max_{\ell \in L(T_{v_{i-1}})} A_{h(T_{v_{i-1}}),\ell} = \phi_A(T_{v_{i-1}})$$

- tracking where attention concentrates over substructures: first condition max attention from the *head* of each subtree sufficiently large; second guarantees that when considering next nested subtree trying to maximize its attention value, one does not spoil optimizations achieved at previous steps for larger subtrees

incorporating syntactic relations as character

- syntactic relation ρ Boolean valued $\mathcal{B} = (\{0,1\}, \max, \cdot)$

$$\phi_\rho(T) = \max_{\ell \in L(T)} \rho(h(T), \ell)$$

  detects whether ρ is realized in T or not
- can combine characters, values in Viterbi $\mathcal{P} = ([0,1], \max, \cdot)$

  (commonly used in NLP for probabilistic values)

$$\phi_{A,\rho}(T) = \max_{\ell \in L(T)} \rho(h(T), \ell) \cdot A_{h(T),\ell}$$

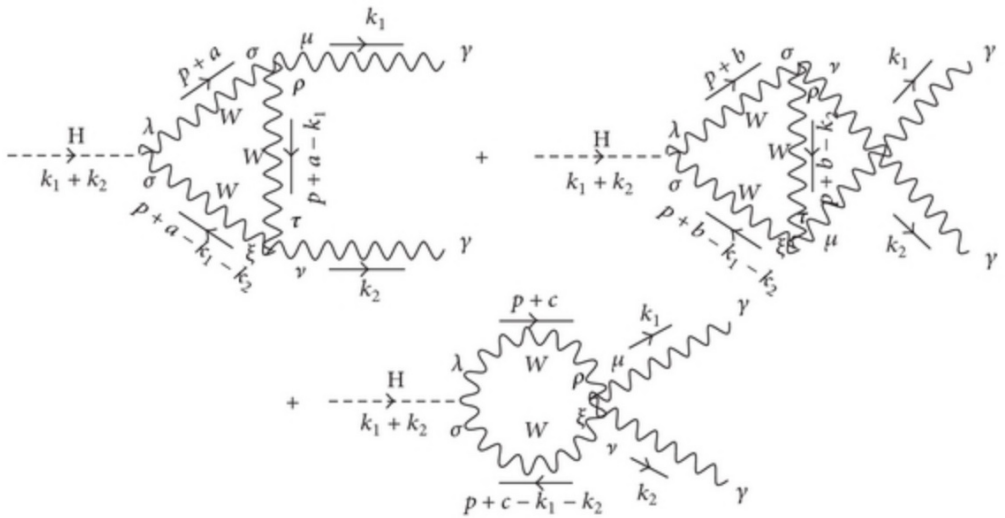
- here one maximizes attention from syntactc *head* over set of $\ell \in L(T)$ that *already satisfy* syntactic relation with the *head*
- Birkhoff factorization as before but subtrees with $\phi_\rho(T_v) = 0$ do not contribute even if their $\max_\ell A_{h(T),\ell}$ is large
- comparison between ϕ_A and $\phi_{\rho,A}$ identifies attention-detectability of ρ
- if detectability fails, identifies where in substructures attention matrix maximum happens outside of where the syntactic relation holds

syntax as inverse problem

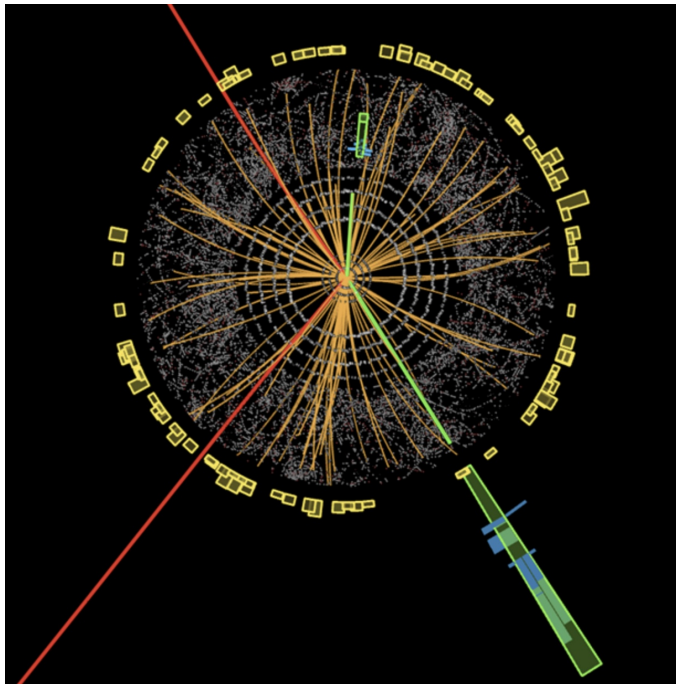
- attention modules fit (through characters as above) in our model of syntax semantics interface
- in this light: what is happening in LLMs?
  - *very large corpora* of text *produced by human language*
  - encodings of *semantic relatedness* (in the form of vectors)
  - *but*.... syntax casts a shadow copy of itself upon semantics (embedding of syntax inside semantics discussed earlier)
  - computational architecture: multi-head attention modules
  - the keys and queries are a *statistical proxy* for syntactic relationship (theta-theory, c-command): *structural relations*
  - they "see syntax" through a probabilistic smear of how it is mapped into semantics
  - *very large parallel computing* trying to solve a *computationally very hard* and *imperfect* inverse problem
- results like C.D. Manning e al. PNAS 2020, finding syntactic trees in weights of attention modules are *not* surprising

physics as metaphor

- Quantum Field Theory: generative process of Feynman diagrams, assignment of meaningful physical values (renormalization) $\Rightarrow$ perturbative computation of Higgs boson production cross sections



- Particle accelerators and detectors: solving an *inverse problem* that identifies inside enormous set of data traces of the correct diagrams/processes involving creation/decay of a Higgs particle through interactions of other particles



sees "an image" of the QFT objects embedded into the set of data collected by detectors, against a noise background of a huge number of other simultaneous events

- the generative process of syntax is embedded in LLMs in a conceptually similar way: its image is scattered in a probabilistic smear across large number of weights and vectors, trained over large data sets
- signals of linguistic structures detectable against a background of probabilistic noise
- LLMs do not "invalidate" generative syntax any more than particle detectors would "invalidate" Quantum Field Theory: quite the opposite
  consequently:
- LLMs are *not* a language theory, generative syntax is
- LLMs are an *experimental apparatus* for the study of the inverse problem of the syntax-semantic interface
- data and technology without *theory* do *not* constitute science

The purpose of science is to obtain a coincise conceptual explanation of natural phenomena, that should be testable, predictive, and essential (*entia non sunt multiplicanda praeter necessitatem*)

Generative linguistics aims at producing such models of the structure and functioning of language

renouncing the explanatory capacity of theoretical science spells humanity's self-defeat

Wir müssen wissen. Wir werden wissen
(David Hilbert)

Thanks Everybody!

with special thanks to Noam and Bob for all of this!

(photo from our long zoom discussion about a year ago

that got all this work together)