Lecture 7: Merge and Hopf Algebras
Ma 191c: Mathematical Models of Generative Linguistics

Matilde Marcolli

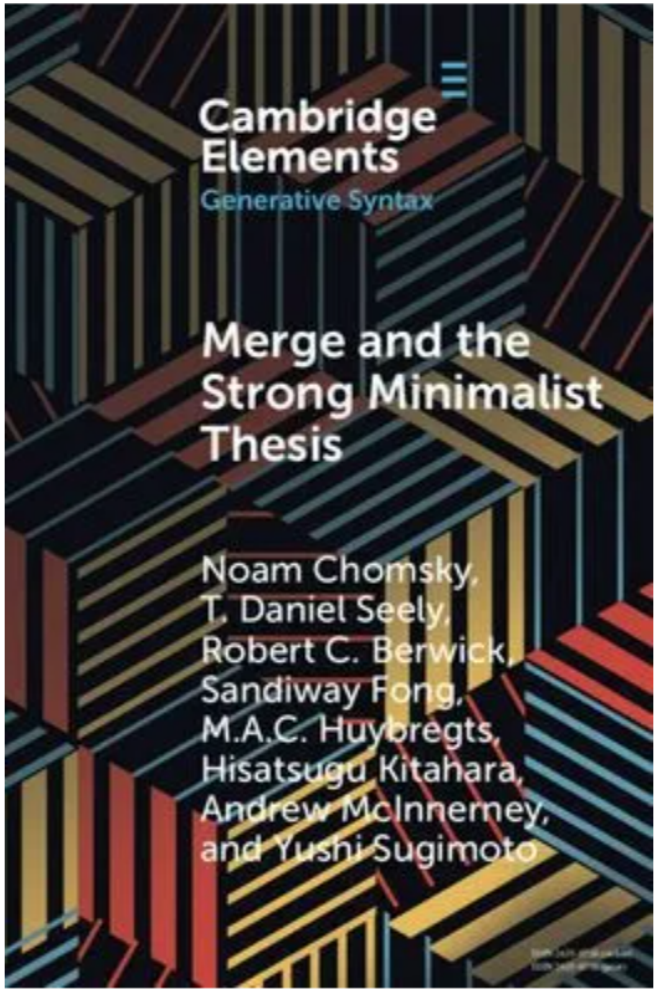
Caltech, Spring 2024

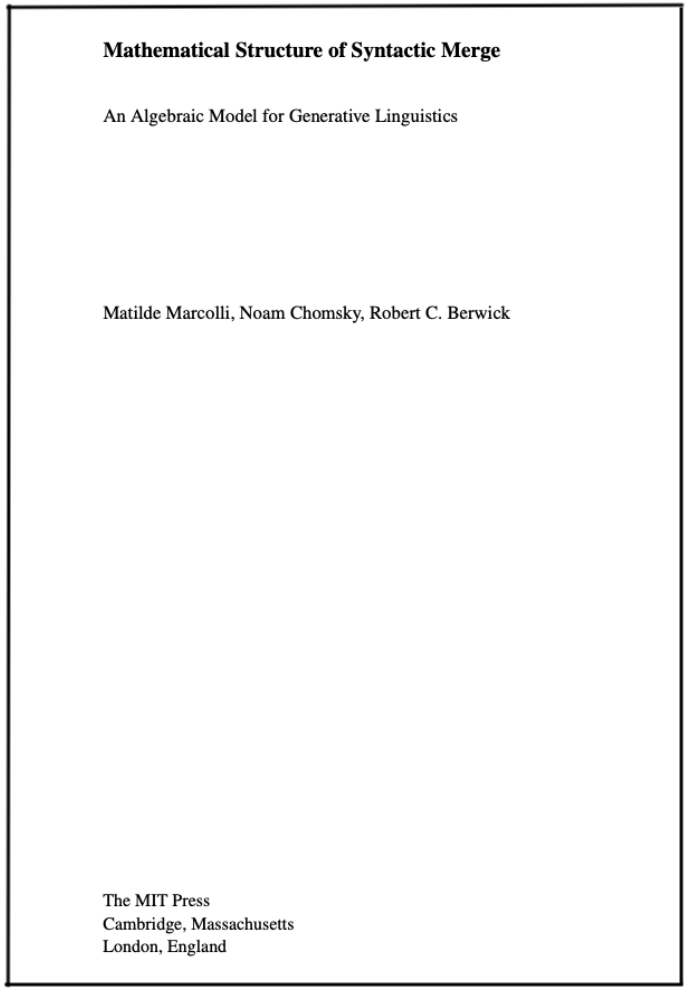
Based on joint work with Noam Chomsky and Robert Berwick

- 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

to appear as our forthcoming monograph "Mathematical structure of syntactic Merge", MIT Press

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

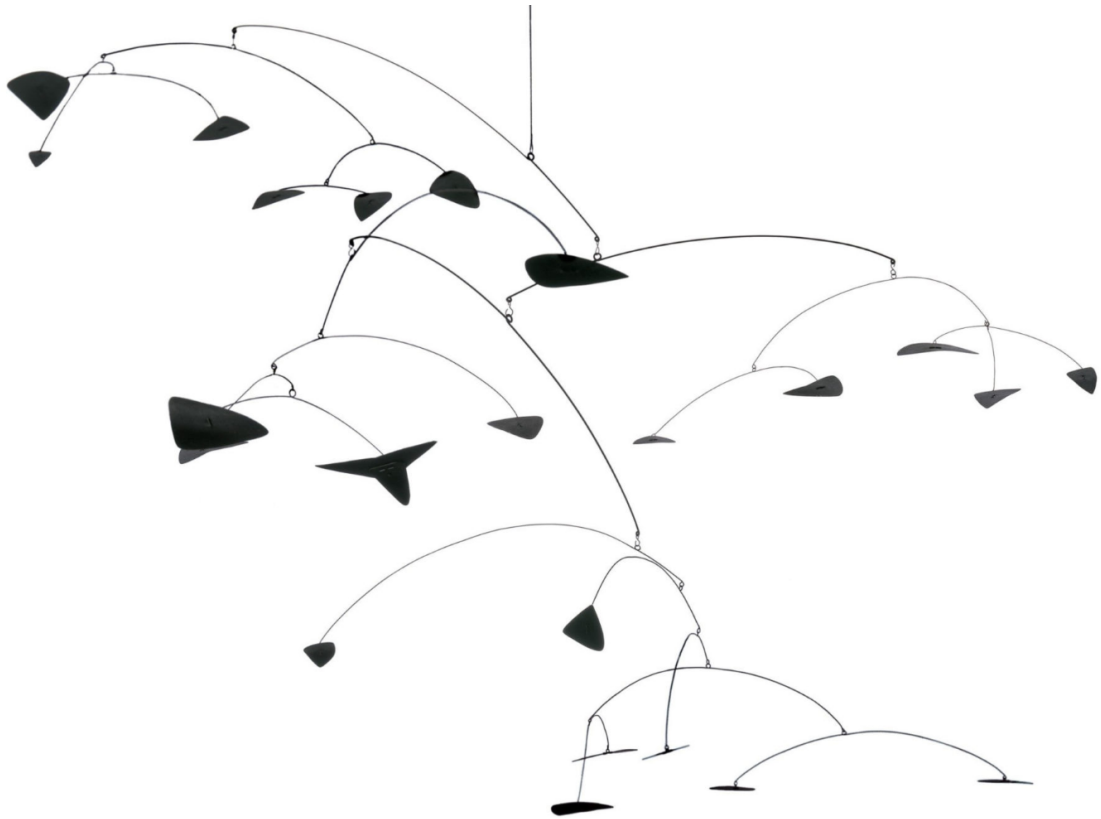
Key point: all aspects of this linguistic model have a mathematical formulation and properties of the model fall into place by *structural necessity* of the corresponding algebraic formalism

Change of perspective: Strings versus Structures

- what language appears to look like

$$00220100211120001212200002002 11\ldots$$

- what language actually looks like



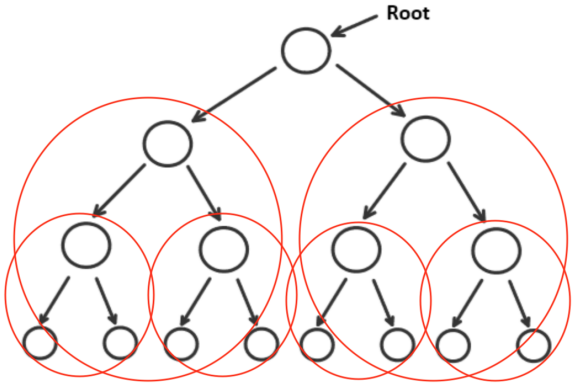
Main aspects of the Merge model of syntax (in its most recent formulation: 2013 onward)

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

all these notions have a precise mathematical formulation that shows many aspects of the linguistic model that have empirical grounds in fact follow by constraints from the algebraic structure

Core computational process of structure formation

- *simple* core computational structure: Merge

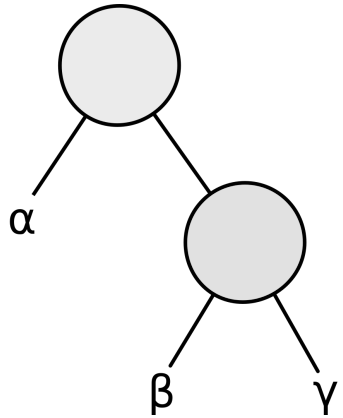


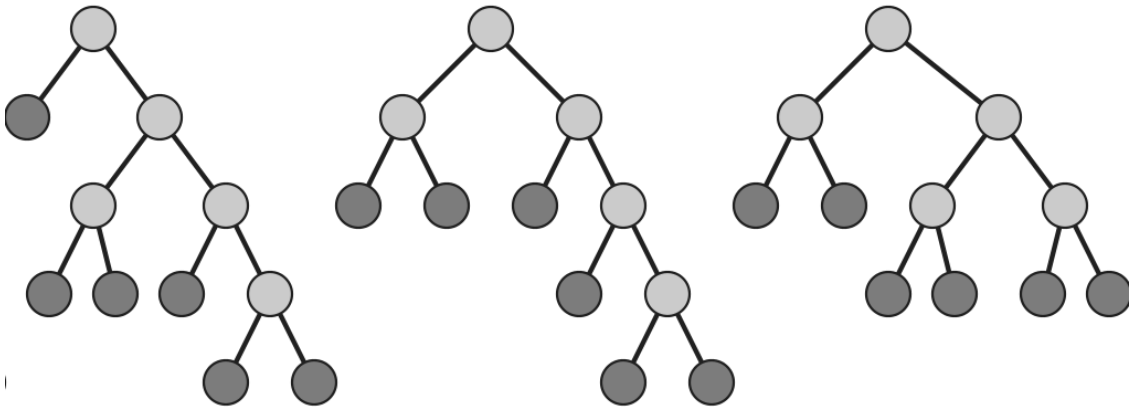
This process generates structures

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

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

- a computational process of *structure formation*





- similar computational structures are natural in the context of fundamental physics (Feynman graphs)

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)

magma of syntactic objects: free nonassociative commutative magma

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

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

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

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

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

free magma and abstract binary rooted trees

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

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

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

Note: "binary trees" here means "full binary trees"

tree-notation versus set-notation

- "tree notation" versus "set notation" (linguists prefer "set notation" while "tree notation" is common in mathematics)
- examples

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

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

- of course choice of notation is irrelevant and does not change anything, but notation can be suggestive of some rather than other types of operations on trees or can be potentially misleading (suggesting planarity when not assumed)

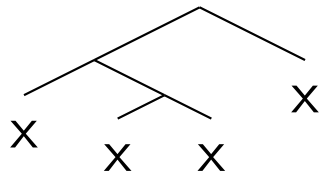
Core computational structure of Merge

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

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

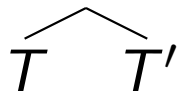
  where α, β are two such balanced bracketed expressions
- equivalent description: elements are finite binary rooted trees (no assignment of planar structure)

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



- 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

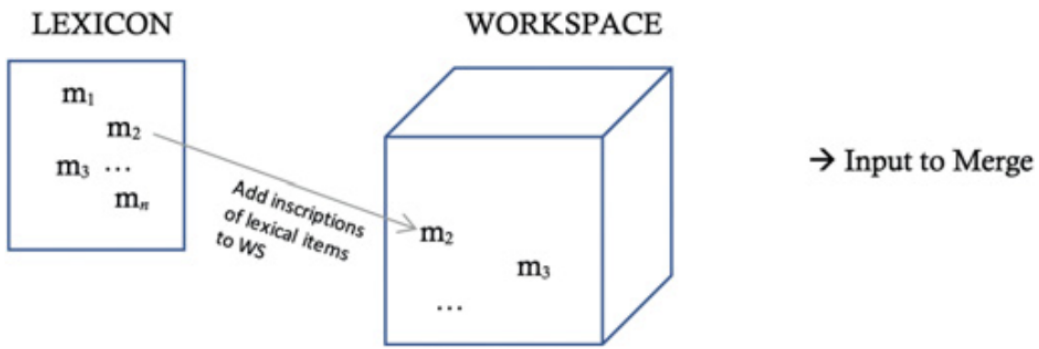
$$X = \mathfrak{B}(P(X))$$

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

(more on this later!)

workspaces



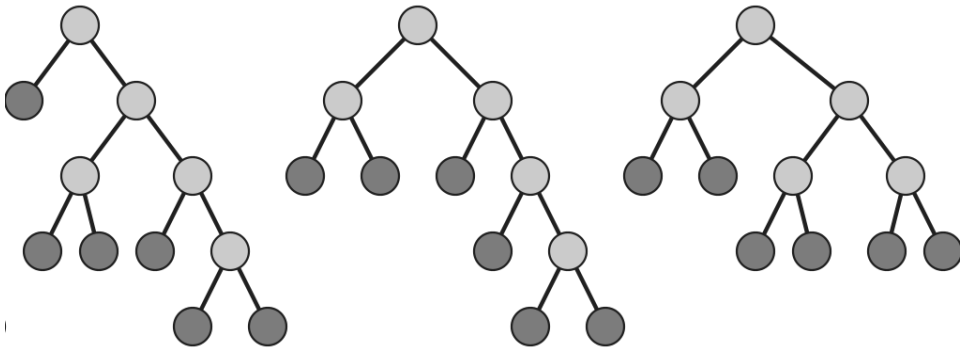
- sentences built by repeated applications of Merge (this process is called a "derivation")
- starting from an initial set of lexical items, syntactic features
- the operations take place in a kind of "computational scratchpad," called a workspace (WS)
- workspace is the set of available computational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

("Merge & SMT" §1)

workspaces

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

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

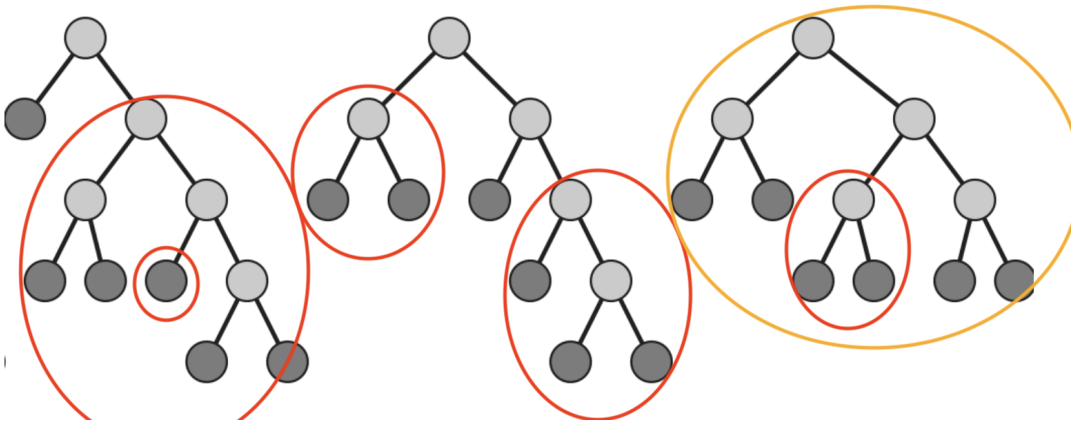


- set of workspaces = set of binary forests $\mathfrak{F}_{\mathcal{SO}_0}$
- Merge operations map the set $\mathfrak{F}_{\mathcal{SO}_0}$ to itself (transform workspaces into new workspaces)

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

accessible terms

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



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

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

- 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 workspaces and accessible terms?
- answer: Workspaces form a Hopf algebra

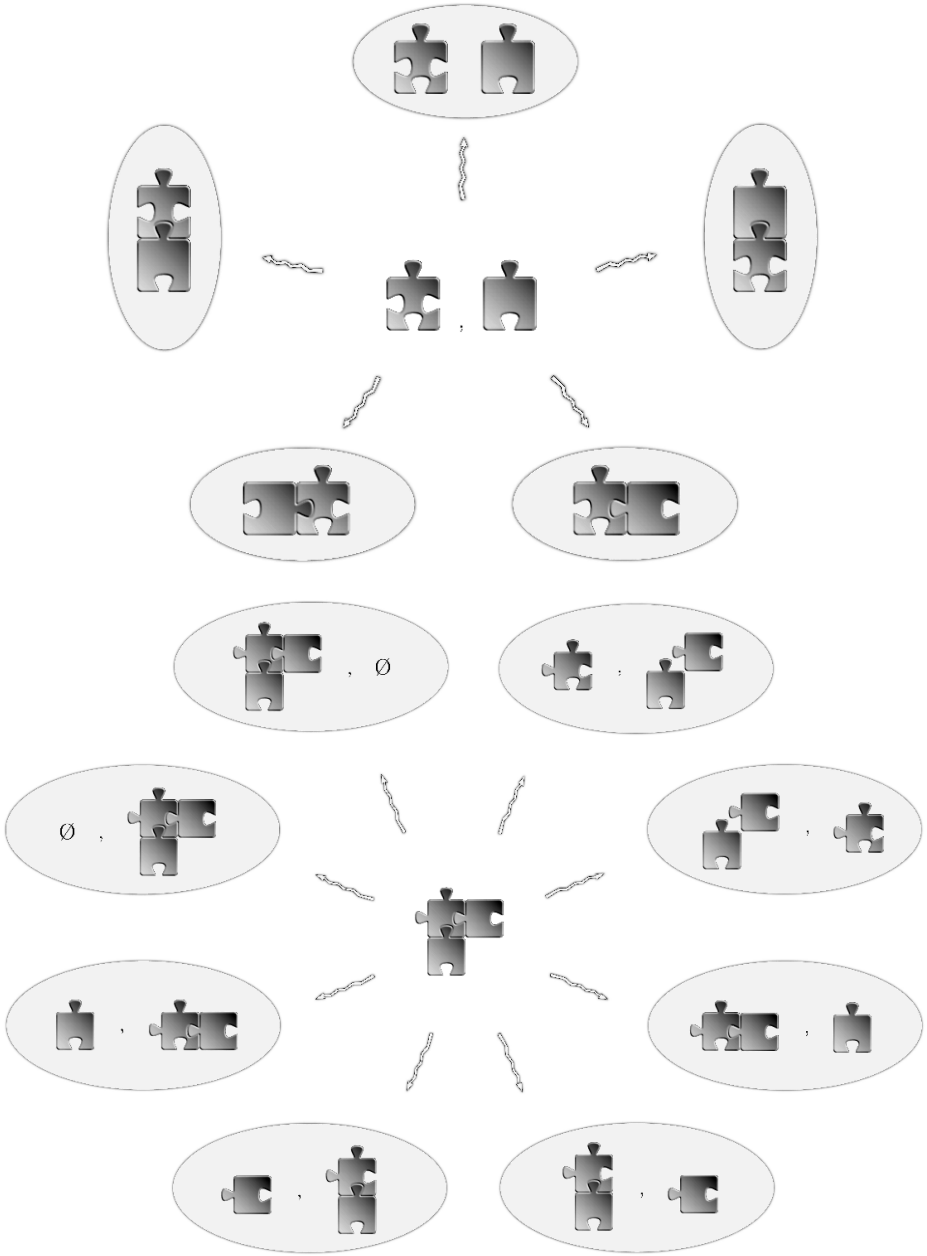
algebraic operations governing workspaces and accessible terms

- assemblying of workspaces: *product* operation $\sqcup$ that groups together a collection of accessible terms into a workspace
- extracting accessible terms (disassemble operation): *coproduct* operation Δ that extracts computational material from the workspace for use by Merge operations
- an algebraic structure that has compatible product/coproduct operations that compose/decompose combinatorial objects is the kind of structure modelled by Hopf algebras
- important aspect of the coproduct extracting accessible terms (as part of Merge operation): this makes Merge both a structure builder and a parser (we'll discuss this more in modeling the syntax-semantics interface)

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

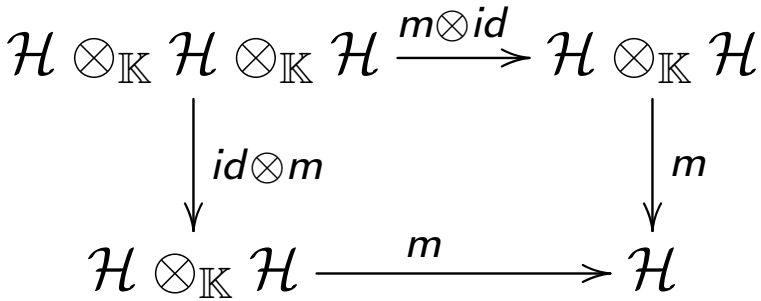
  (a relation when interchanging order of product/coproduct)

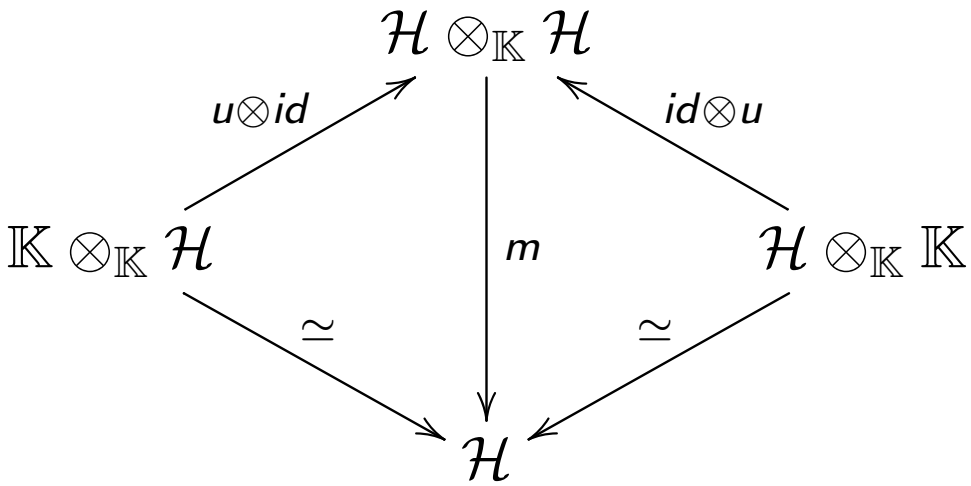

Ø
Ø

A formal definition of Hopf algebra

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

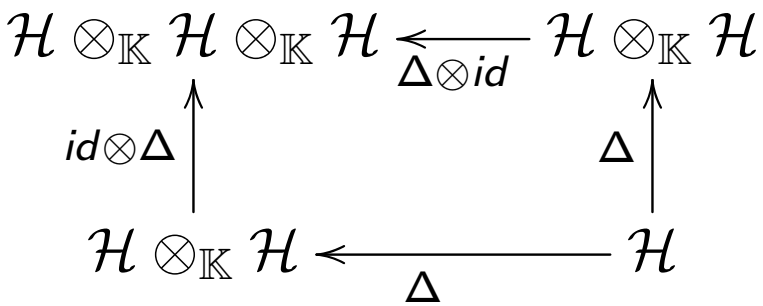
multiplication: associativity and unit

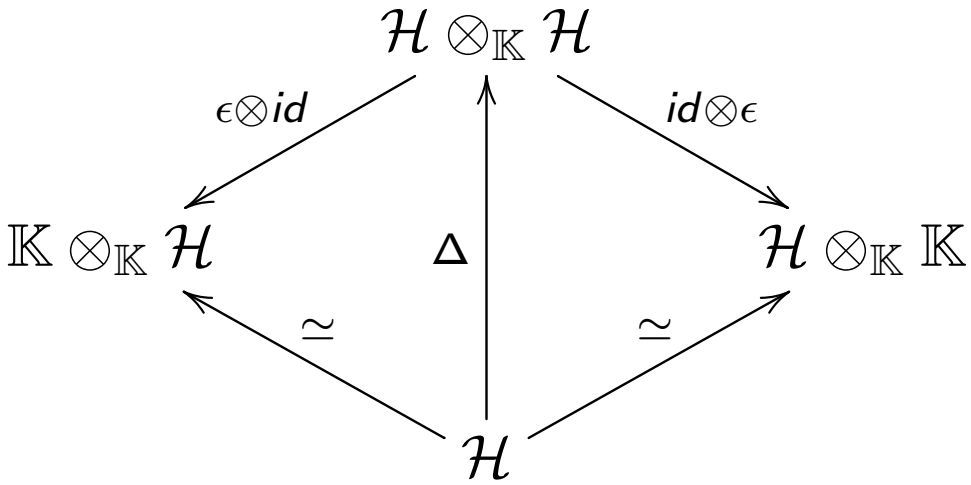




commutativity of these diagrams

comultiplication: coassociativity and counit





commutativity of these diagrams: coassociativity

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

compatibility of product and coproduct

compatibility between product and coproduct:

$$\begin{array}{ccccc} \mathcal{H}\otimes\mathcal{H} & \xrightarrow{m} & \mathcal{H} & \xrightarrow{\Delta} & \mathcal{H}\otimes\mathcal{H} \\ \downarrow{\scriptstyle \Delta\otimes\Delta} & & & & \uparrow{\scriptstyle m\otimes m} \\ \mathcal{H}\otimes\mathcal{H}\otimes\mathcal{H}\otimes\mathcal{H} & & \xrightarrow[id\otimes\tau\otimes id]{} & & \mathcal{H}\otimes\mathcal{H}\otimes\mathcal{H}\otimes\mathcal{H} \end{array}$$

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

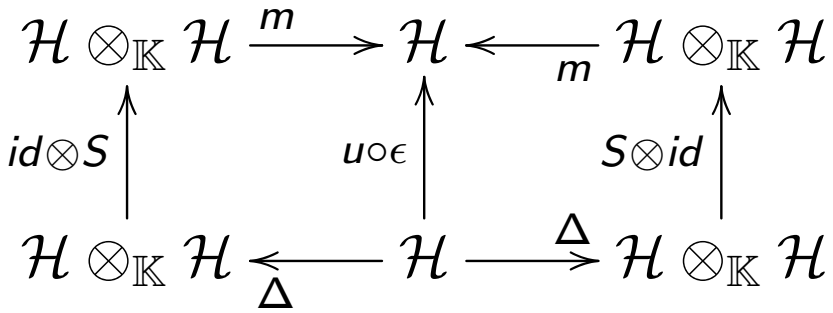
$$\Delta \circ \sqcup = (\sqcup \otimes \sqcup) \circ \tau \circ (\Delta \otimes \Delta)$$

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

$$\begin{array}{ccc} \mathcal{H}\otimes\mathcal{H} & \xrightarrow{m} & \mathcal{H} \\ & {\scriptstyle \epsilon\otimes\epsilon} \searrow \quad \swarrow {\scriptstyle \epsilon} & \\ & \mathbb{K} & \end{array} \qquad \text{and} \qquad \begin{array}{ccc} \mathcal{H}\otimes\mathcal{H} & \xleftarrow{\Delta} & \mathcal{H} \\ & {\scriptstyle u\otimes u} \nwarrow \quad \nearrow {\scriptstyle u} & \\ & \mathbb{K} & \end{array}$$

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

antipode: further compatibility, commutativity of diagram



- usually the antipode is an additional constraint
- withoug antipode only have bialgebra not Hopf algebra

graded connected Hopf algebras

- Hopf algebra is graded $\mathcal{H} = \oplus_{\ell \geq 0} \mathcal{H}_\ell$
- and connected: $\mathcal{H}_0 = \mathbb{K}$
- m, Δ compatible with grading

$$m : \mathcal{H}_\ell \otimes \mathcal{H}_k \to \mathcal{H}_{\ell+k}$$

$$\Delta : \mathcal{H}_\ell \to \bigoplus_{a+b=\ell} \mathcal{H}_a \otimes \mathcal{H}_b$$

then antipode comes for free (determined by the bialgebra)

- S is determined by the rest of the structure: is not an additional constraint
- S constructed inductively using coproduct and product

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

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

Commutative Hopf Algebras and Affine Group Schemes

- Commutative algebra $\mathcal{A}$ over $\mathbb{K}$: functor

$$X : \mathrm{CommAlg}_{\mathbb{K}} \to \mathrm{Sets}$$

$$X(\mathcal{R}) = \underset{\mathrm{CommAlg}_{\mathbb{K}}}{\mathrm{Hom}}(\mathcal{A}, \mathcal{R})$$

- $X(\mathcal{R})$ set of $\mathcal{R}$-points of the *affine scheme* X dual to $\mathcal{A}$
- but if $\mathcal{H} \in \mathrm{CommAlg}_{\mathbb{K}}$ is also a Hopf algebra then functor

$$G : \mathrm{CommAlg}_{\mathbb{K}} \to \mathrm{Groups}$$

$$G(\mathcal{R}) = \underset{\mathrm{CommAlg}_{\mathbb{K}}}{\mathrm{Hom}}(\mathcal{A}, \mathcal{R})$$

with group operation

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

and inverse given by the antipode

$$\phi^{-1}(x) = \phi(S(x))$$

and unit from the counit of $\mathcal{H}$

- G is an *affine group scheme*

simple examples of affine group schemes

- *additive group* $\mathbb{G}_a$ dual to Hopf algebra $\mathcal{H} = \mathbb{K}[t]$ with $\Delta(t) = t \otimes 1 + 1 \otimes t$ (primitive)

$$(\phi_1 \star \phi_2)(t) = (\phi_1 \otimes \phi_2)\Delta(t) = \phi_1(t) + \phi_2(t)$$

- *multiplicative group* $\mathbb{G}_m$ dual to Hopf algebra $\mathcal{H} = \mathbb{K}[t, t^{-1}]$ with $\Delta(t) = t \otimes t$ (grouplike)

$$(\phi_1 \star \phi_2)(t) = (\phi_1 \otimes \phi_2)\Delta(t) = \phi_1(t) \cdot \phi_2(t)$$

Combinatorial Hopf Algebras

- graded connected $\mathcal{H} = \oplus_{\ell \geq 0} \mathcal{H}_\ell$ with $\mathcal{H}_0 = \mathbb{K}$
- linear basis $\mathcal{B}_k$ of $\mathcal{H}_k$ consists of *combinatorial objects* (e.g. trees, graphs, matroids, etc)
- grading is a measure of "size" of the objects (e.g. number of leaves in a binary tree; number of edges in a graph, etc)
- coproduct describes decomposition operations: typically terms $x' \otimes x''$ in $\Delta(x)$ are pairs subobject–quotient object
- usually asymmetric role of two sides (two channels) of the coproduct (non-cocommutative)
- $\mathcal{H}_0 = \mathbb{K}$: only one object of *size zero*

- G.C. Rota, *Hopf algebra methods in combinatorics*, Colloq. Internat. CNRS 260, CNRS, Paris 1978, pp. 363–365.
- J.L. Loday, M. Ronco, *Combinatorial Hopf algebras*, in "Quanta of maths", Clay Math. Proc. 11, pp. 347–383, Amer. Math. Soc., 2010

back to workspaces: composition and decomposition

- vector space $\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0})$ spanned by forests
- grading $\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}) = \oplus_k \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0,k})$ with $\mathfrak{T}_{\mathcal{SO}_0,k}$ binary rooted trees with k leaves; $\mathfrak{F}_{\mathcal{SO}_0,k}$ forests with k leaves,

$$F = \sqcup_a T_a \quad \text{with } T_a \in \mathfrak{T}_{\mathcal{SO}_0,k_a} \quad \text{and } \sum_a k_a = k$$

- $\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0,0}) = \mathbb{K}$ formal "empty forest" $\mathbf{1}$
- product: assemble workspaces (forests) by disjoint union of syntactic objects (trees)

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

- product is compatible with grading

$$\sqcup : \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0,k}) \otimes \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0,\ell}) \to \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0,k+\ell})$$

- coproduct: disassembles workspaces into constituent parts
- need a disassembling (coproduct) operation Δ that is compatible with $\sqcup$ by

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

- two different choices:
  1. simplest one: decompose workspace into constituent syntactic objects
  2. more interesting decomposition: extract accessible terms

simplest coproduct: partitions of the workspace

- coproduct where all trees are primitive elements $\Delta(T_a) = T_a \otimes 1 + 1 \otimes T_a$ get for $\Delta(F) = \sqcup_a \Delta(T_a)$

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

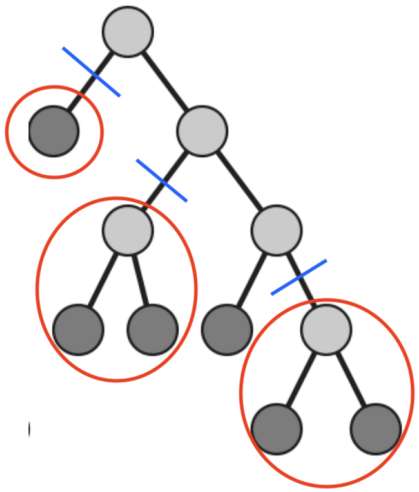
- behaves like the additive group (in variables T_a)
- would get only *partitions of the workspace*, no access to substructures (accessible terms)
- would get External Merge but no movement (Internal Merge)

coproduct: admissible cuts

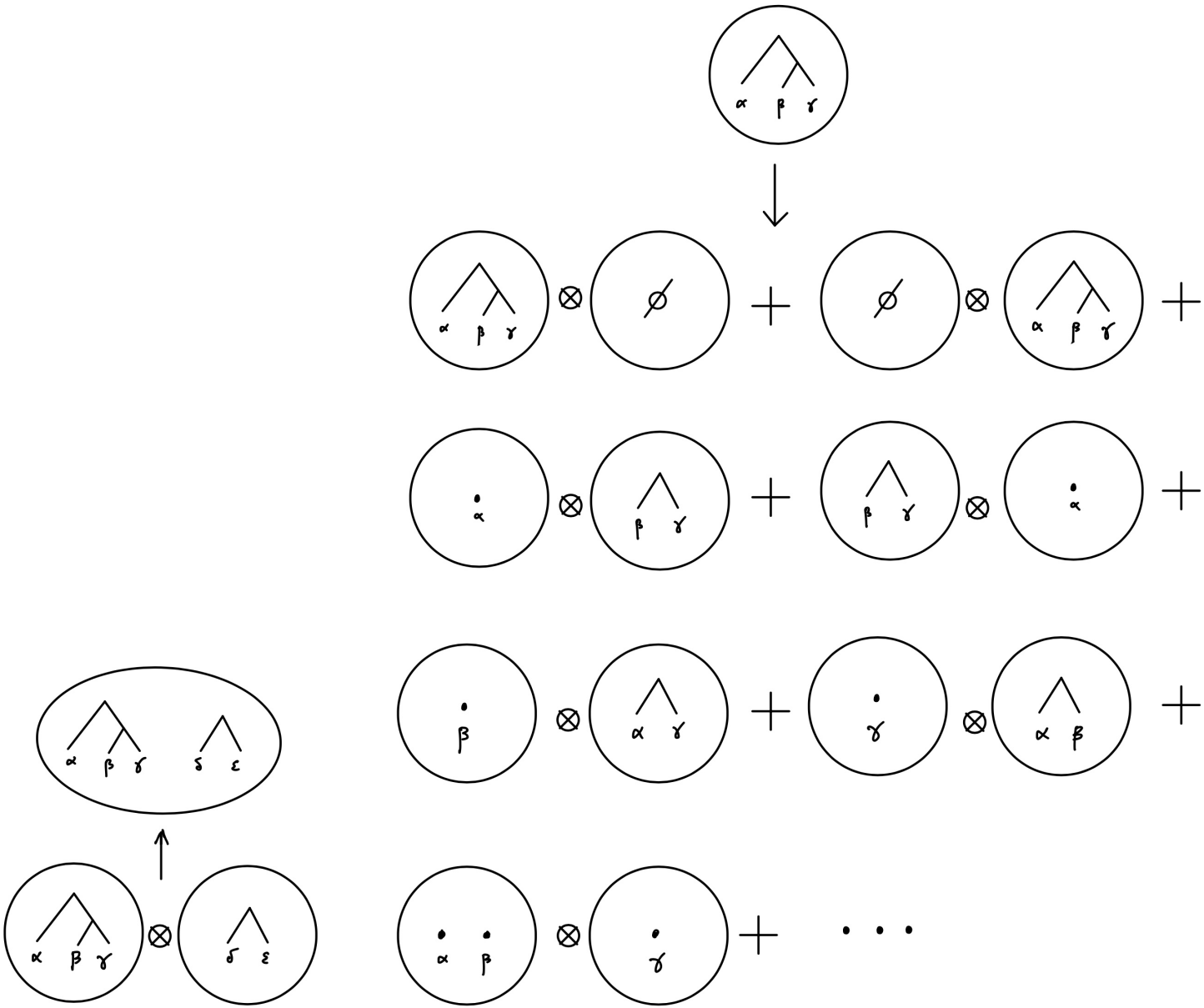
- in a tree $T \in \mathfrak{T}_{\mathcal{SO}_0}$ consider forests $F_{\underline{v}} \subset T$ $F_{\underline{v}} = T_{v_1} \sqcup \cdots \sqcup T_{v_n}$ of accessible terms
- coproduct

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

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



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


α β γ
α β γ
∅
∅
α β γ
α
β γ
β γ
α
β
α γ
γ
α β
α β γ
δ ε
α β γ
δ ε
α β
γ
• • •

admissible cuts

- coproduct in terms of admissible cuts

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

$$\pi_C(T) = F_{\underline{v}} = T_{v_1} \sqcup \cdots \sqcup T_{v_n}$$

  collection of accessible terms that are extracted by the cut

- $\rho_C(T)$ tree that remains attached to the root of T after cut
- note that $\rho_C(T)$ is *not* a (full) binary tree: it has non-branching vertices

admissible cuts: higher arity

- let $\tilde{\mathfrak{F}}_{\mathcal{SO}_0}$ denote the set of all forests (not necessarily binary) with leaves labels in the set $\mathcal{SO}_0$
- $\mathcal{V}(\tilde{\mathfrak{F}}^{\leq n}_{\mathcal{SO}_0})$ subspace of $\mathcal{V}(\tilde{\mathfrak{F}}_{\mathcal{SO}_0})$ spanned by "at most n-ary" forests (i.e. components are possibly non-full n-ary trees)
- admissible cuts give coproduct $\Delta : \mathcal{V}(\tilde{\mathfrak{F}}_{\mathcal{SO}_0}) \to \mathcal{V}(\tilde{\mathfrak{F}}_{\mathcal{SO}_0}) \otimes \mathcal{V}(\tilde{\mathfrak{F}}_{\mathcal{SO}_0})$ which preserves these subspaces

$$\Delta : \mathcal{V}(\tilde{\mathfrak{F}}^{\leq n}_{\mathcal{SO}_0}) \to \mathcal{V}(\tilde{\mathfrak{F}}^{\leq n}_{\mathcal{SO}_0}) \otimes \mathcal{V}(\tilde{\mathfrak{F}}^{\leq n}_{\mathcal{SO}_0})$$

- in particular $\Delta : \mathcal{V}(\tilde{\mathfrak{F}}^{\leq 2}_{\mathcal{SO}_0}) \to \mathcal{V}(\tilde{\mathfrak{F}}^{\leq 2}_{\mathcal{SO}_0}) \otimes \mathcal{V}(\tilde{\mathfrak{F}}^{\leq 2}_{\mathcal{SO}_0})$ and if applied to(full) binary trees

$$\Delta : \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}) \to \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}) \otimes \mathcal{V}(\tilde{\mathfrak{F}}^{\leq 2}_{\mathcal{SO}_0})$$

admissible cuts: Hopf algebra and comodule

- Notation: write Δ^ρ for coproduct with admissible cuts
- $(\mathcal{V}(\tilde{\mathfrak{F}}^{\leq 2}_{\mathcal{SO}_0}), \sqcup, \Delta^\rho)$ is a graded connected Hopf algebra
- all properties are simple to check: only point that requires some discussion is *coassociativity*

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

  - lef: admissible cut C' on $\rho_C(T)$ of previous cut C
  - right: admissible cut C on $\pi_{C'}(T)$ of previous cut C'

admissible cuts: comodule

- $\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}) \subset \mathcal{V}(\tilde{\mathfrak{F}}^{\leq 2}_{\mathcal{SO}_0})$ is subalgebra and right comodule
- right comodule M over a coalgebra $\mathcal{H}$ is a $\mathbb{K}$-vector space with a linear map

$$\xi : M \to M \otimes \mathcal{H}$$

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

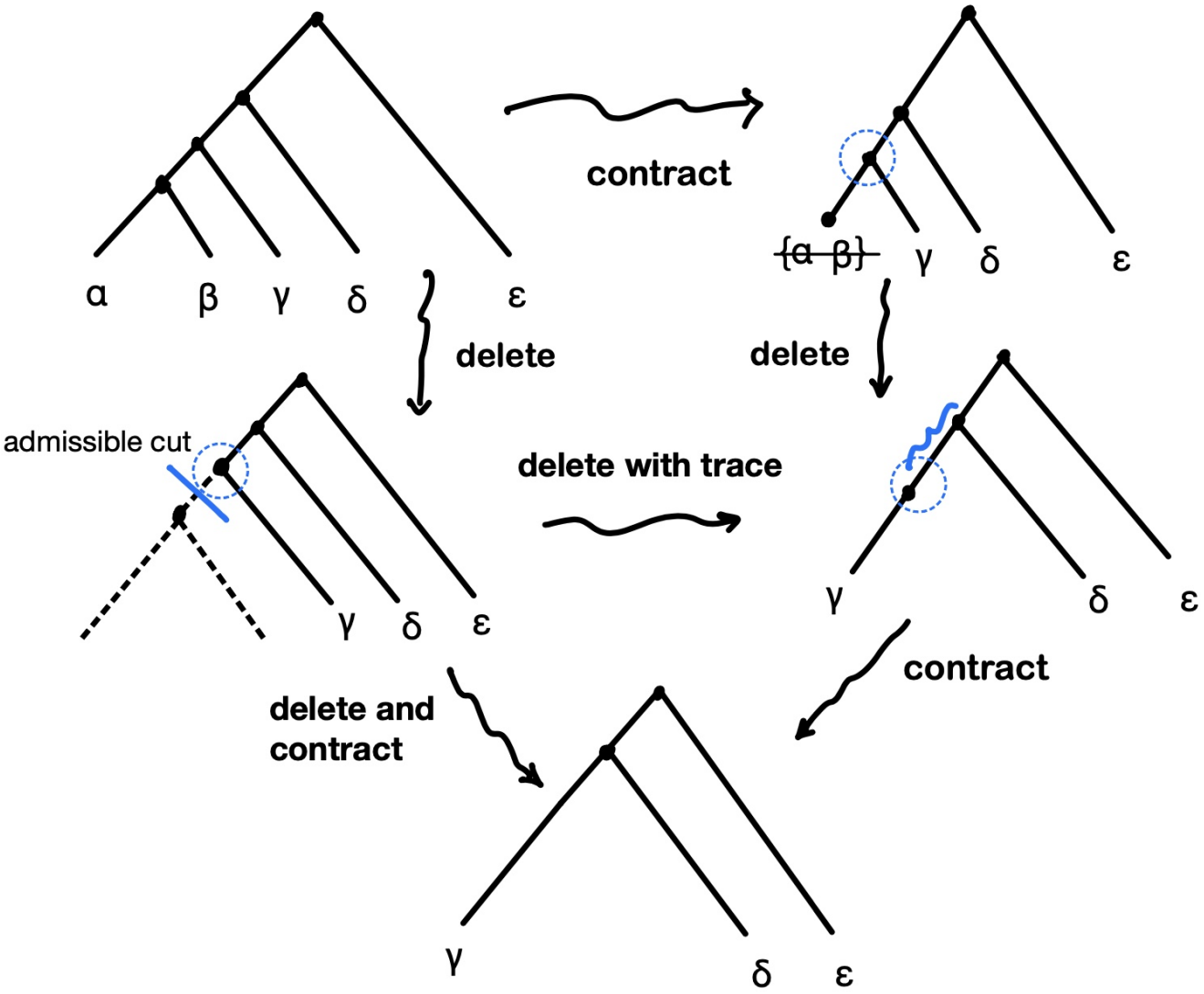
$$(\mathrm{id} \otimes \epsilon) \circ \xi = \mathrm{id}$$

- here $M = \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0})$ with $\xi = \Delta^{\rho}|_{\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0})}$

Other possible forms of "remainder term" $T/F_{\underline{v}}$

- case seen above $T/^{\rho}F_{\underline{v}} := \rho_C(T)$ for admissible cut C with $\pi_C(T) = F_{\underline{v}}$
- different possibilities for what $T/F_{\underline{v}}$
  1. contraction $T/^{c}F_{\underline{v}}$
  2. keep non-branch nodes (as trace) $T/^{\rho}F_{\underline{v}}$
  3. deletion $T/^{d}F_{\underline{v}}$

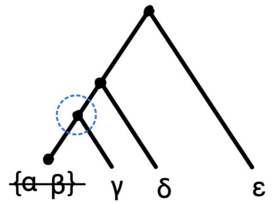
  all these play different roles in the linguistic model



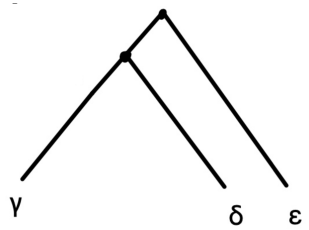
different roles: "deeper copies are interpreted at CI interface but not expressed at SM interface"

different roles (and different alg properties):

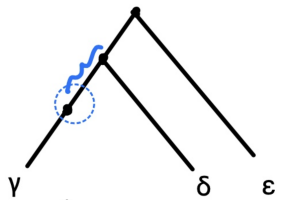
1. Quotient $T/^{c}T_{v}$: can go to the CI (conceptual-intensional; syntax-semantics) interface for interpretation



2. Quotient $T/^{d}T_{v}$: can go to SM (sensory-motor) interface for externalization



3. Quotient $T/^{\rho}T_{v}$: intermediate relating them algebraically and useful in recursive parsing



Contraction coproduct

- when using the contraction quotient $T/^c T_v$ get coproduct Δ^c

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

- in $T/^c F_{\underline{v}}$ contract every component T_{v_i} of $F_{\underline{v}} = T_{v_1} \sqcup \cdots \sqcup T_{v_n}$ to its root vertex v_i with new label $\not{T}_{v_i}$
- Note: because root vertex v remains in $T/^c T_v$ grading by number of leaves not compatible
- a (full) binary rooted tree with n leaves has $2n-1$ vertices and $2n-2$ edges
- can use number of edges as grading, then compatible with contraction coproduct Δ^c (and with product $\sqcup$)
- but now graded but *not connected*: single leaves are of degree zero (isolated lexical items)

Bialgebra vs Hopf algebra

- consider $\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0})$ with contraction coproduct Δ^c
- coassociativity holds (argument like for admissible cuts)

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

- compatibility of product and coproduct also, so get bialgebra
- the fact that degree zero includes subspace dimension $\#\mathcal{SO}_0 > 1$ (not connected) affects existence of antipode: $\Delta(\alpha) = \alpha \otimes \alpha$ non-invertible grouplike elements
- $(\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}), \sqcup, \Delta^c)$ bialgebra dual to an *affine semigroup scheme*
- have antipode (hence Hopf algebra) on quotient by ideal generated by $\mathbf{1} - \alpha$ for $\alpha \in \mathcal{SO}_0$

Note on $\mathcal{SO}_0$-labels at the leaves

- $\mathcal{SO}_0$ is the set of lexical items and syntactic features
- syntactic features carry information on role of lexical items in sentence
- example: in the "buffalos" sentence all same lexical items but with different syntactic features
- the set $\mathcal{SO}_0$ is possibly large but finite
- so how to interpret the label $\not{T}_v$ at the new leaf v of the contraction quotient $T/^c T_v$?
- it seems now have a possible label $\not{T}$ for every $T \in \mathcal{SO}$ (an infinite set!)
- *but...* don't need to remember all of T in the label $\not{T}$ only a syntactic feature (already available in the finite set $\mathcal{SO}_0$) that describes the *role* of the structure T in the larger structure in which the quotient is taken
- so $T/^c T_v$ is still an object in $\mathcal{SO} = \mathfrak{T}_{\mathcal{SO}_0}$

Deletion quotient $T/^d T_v$

- first consider the "at most binary" tree $\rho_C(T) = T/^\rho T_v$ (here C is an elementary cut of a single edge) and then take the maximal binary rooted tree in $\mathfrak{T}_{\mathcal{SO}_0}$ determined by $T/^\rho T_v$ (by contraction of some edges)
- same for $T/^d F_{\underline{v}}$ obtained from $\rho_C(T) = T/^\rho F_{\underline{v}}$
- Note: this may be the empty tree:

$$T = \overbrace{\alpha \quad \beta} \quad \text{with admissible cut } C \text{ of both edges}$$

single root with no label is not in $\mathfrak{T}_{\mathcal{SO}_0}$ so in this case $T/^d F_{\underline{v}} = \mathbf{1}$

Brief discussion of coassociativity for deletion:

- for Δ^d list of terms on the two sides of

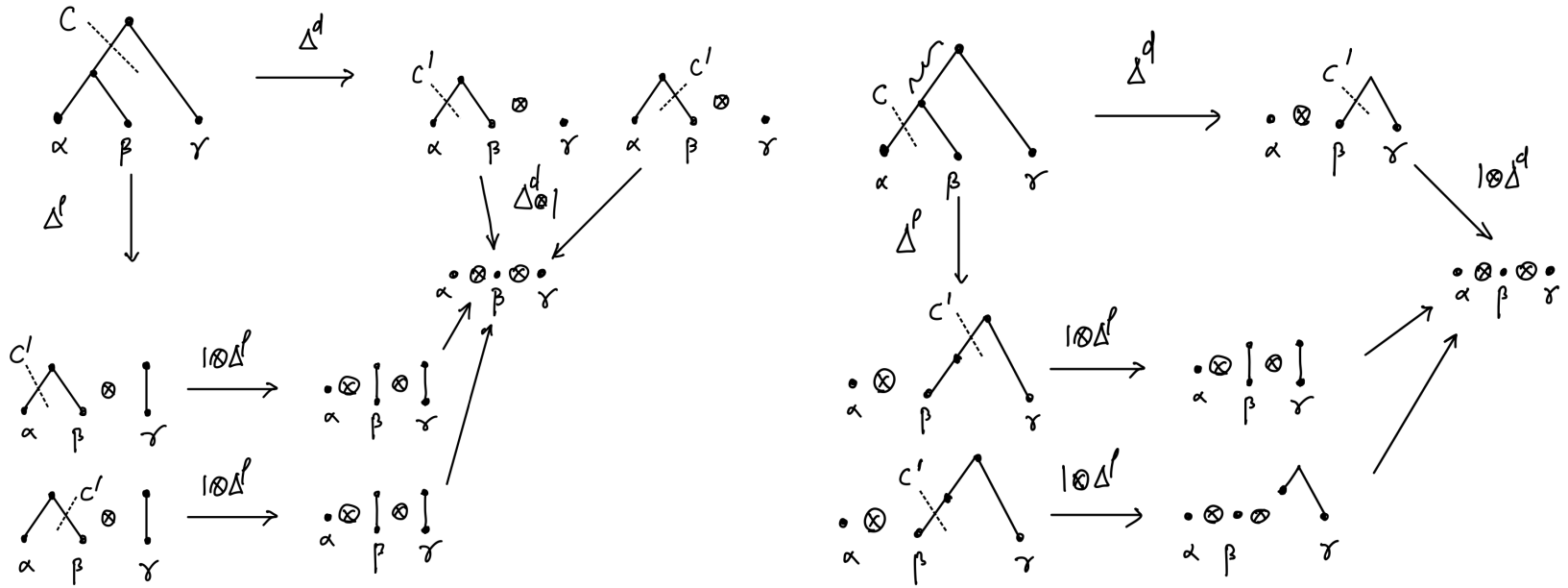
$$(\mathrm{id} \otimes \Delta^d) \circ \Delta^d \neq (\Delta^d \otimes \mathrm{id}) \circ \Delta^d$$

  generally match but wrong multiplicities

- can describe as

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

  for A an endomorphism (outer) that assign multiplicities to edges and vertices



cutting first C leaves two cuts C' in $(\Delta^d \otimes 1)\Delta^d(T)$ producing two identical terms but only one in $(1 \otimes \Delta^d)\Delta^d(T)$

Digression: Drinfeld quasi-Hopf algebras

- $\mathcal{H}$ unital associative algebra
- coalgebra non-coassociative but lack of coassociativity measured by an invertible element $\Phi \in \mathcal{H} \otimes \mathcal{H} \otimes \mathcal{H}$

$$((1 \otimes \Delta) \circ \Delta(x)) = \Phi((\Delta \otimes 1) \circ \Delta(x))\Phi^{-1}$$

- satisfying a pentagon identity

$$(1 \otimes 1 \otimes \Delta)(\Phi)\,(\Delta \otimes 1 \otimes 1)(\Phi) = (1 \otimes \Phi)\,(1 \otimes \Delta \otimes 1)(\Phi)\,(\Phi \otimes 1)$$

- and unit and counit identities also modified by Φ
- quasi-Hopf algebras in context of *quantum groups*
- Question: case where instead of $\Phi \in \mathcal{H} \otimes \mathcal{H} \otimes \mathcal{H}$ have an automorphism A of $\mathcal{H} \otimes \mathcal{H} \otimes \mathcal{H}$

$$(1 \otimes \Delta) \circ \Delta = A \circ (\Delta \otimes 1) \circ \Delta \circ A^{-1}$$

or an endomorphism by taking

$$(1 \otimes \Delta) \circ \Delta \circ A = A \circ (\Delta \otimes 1) \circ \Delta$$

Workspaces and Hopf algebra: summary

- Bialgebra of workspaces: $\mathcal{H} = (\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}), \sqcup, \Delta^c)$, Hopf algebra modulo $\mathbf{1} - \mathcal{SO}_0$
- Hopf algebra and comodule: $\mathcal{H}^{\leq 2} = (\mathcal{V}(\mathfrak{F}^{\leq 2}_{\mathcal{SO}_0}), \sqcup, \Delta^\rho)$ with $(M = \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}), \xi = \Delta^\rho|_{\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0})}$ comodule
- Deletion coproduct: $\mathcal{H} = (\mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}), \sqcup, \Delta^d)$ with correction to co-associativity

loosely refer to "Hopf algebra of workspaces" to denote any of the above as needed

Discussion of these differences for Hopf algebras of rooted trees:

- D.Calaque, K.Ebrahimi-Fard, D.Manchon, *Two interacting Hopf algebras of trees: A Hopf-algebraic approach to composition and substitution of B-series*, Advances in Applied Mathematics 47 (2011) 282–308

Projection maps

- an "at most binary" tree to the unique maximal binary tree by edge contraction

$$\Pi_{d,\rho} : \mathcal{V}(\tilde{\mathfrak{F}}^{\leq 2}_{\mathcal{SO}_0}) \to \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0})$$

- delete edge to leaf with label $\not{\tau}$

$$\Pi_{\rho,c} : \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}) \to \mathcal{V}(\tilde{\mathfrak{F}}^{\leq 2}_{\mathcal{SO}_0})$$

- projection $\Pi_{d,c} = \Pi_{d,\rho} \circ \Pi_{\rho,c}$

$$\Pi_{d,c} : \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}) \to \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0})$$

- these projections satisfy

$$\Delta^d = (\mathrm{id} \otimes \Pi_{d,\rho}) \circ \Delta^\rho = (\mathrm{id} \otimes \Pi_{d,c}) \circ \Delta^c$$

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

action of Merge on workspaces

- Merge operations $\mathfrak{M}_{S,S'}$ pairs S, S' of syntactic objects

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

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

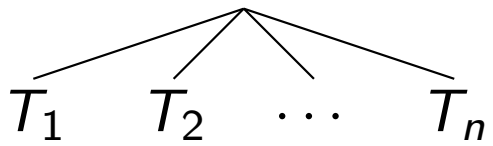
(written as $\mathrm{WS}' = \mathrm{Merge}(S, S', \mathrm{WS})$ in the notation of "Merge & SMT")

- Note: this action contains various forms of Merge (external, internal, sideward)

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

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

- different forms of the coproduct

$$\Delta^d = (\mathrm{id} \otimes \Pi_{d,\rho}) \circ \Delta^\rho = (\mathrm{id} \otimes \Pi_{d,c}) \circ \Delta^c$$

- correspondingly on Merge operation

$$\begin{aligned} \mathfrak{M}_{S,S'} = & \quad \sqcup \circ (\mathcal{B} \otimes \mathrm{id}) \circ (\mathrm{id} \otimes \Pi_{d,\rho}) \circ \delta_{S,S'} \circ \Delta^\rho \\ = & \quad \sqcup \circ (\mathcal{B} \otimes \mathrm{id}) \circ (\mathrm{id} \otimes \Pi_{d,c}) \circ \delta_{S,S'} \circ \Delta^c \,. \end{aligned}$$

- quotient $T/^c T_v$ with $\cancel{T_v}$ at the contraction vertex is what goes to the CI (syntax-semantics) interface for parsing, while $T/^d T_v = \Pi_{d,c}(T/^c T_v)$ goes to externalization
- externalization after all algebraic structure-building operations have been done (so Δ^d not as good as Δ^c not problematic)

matching terms

- two syntactic objects $S, S' \in \mathfrak{T}_{\mathcal{SO}_0}$
- operator $\gamma_{S,S'} : \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0}) \to \mathcal{V}(\mathfrak{F}_{\mathcal{SO}_0})$

$$\gamma_{S,S'}(F) = \begin{cases} F & F = S \sqcup S' \\ 0 & \text{otherwise} \end{cases}$$

- operator $\delta_{S,S'}$

$$\delta_{S,S'} = \gamma_{S,S'} \otimes \mathrm{id}$$

- matching of terms (in the left channel of the coproduct)
- in particular keeps only terms where forest with two components in left-channel (for a binary Merge)

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

$$F' = \mathfrak{M}(T_v, T_w) \sqcup F/(T_v \sqcup T_w)$$

different cases according to where T_v, T_w located in F

- External Merge: accessible terms used are full *components*

$$F' = \mathfrak{M}(T_a, T_b) \sqcup \hat{F}_{a,b}$$

with $F = \sqcup_i T_i = T_a \sqcup T_b \sqcup \hat{F}_{a,b}$ with $\hat{F}_{a,b} = \sqcup_{i \neq a,b} T_i$

- Internal Merge: accessible terms used are an accessible term T_v of a component T_a and the remainder T_a/T_v

$$F' = \mathfrak{M}(T_v, T_a/T_v) \sqcup \hat{F}_a$$

with $F = \sqcup_i T_i = T_a \sqcup \hat{F}_a$ with $\hat{F}_a = \sqcup_{i \neq a} T_i$

- Sideward Merge: accessible terms from different components

$$F' = \mathfrak{M}(T_v, T_w) \sqcup T_a/T_v \sqcup T_b/T_w \sqcup \hat{F}_{a,b}$$

$$F' = \mathfrak{M}(T_v, T_b) \sqcup T_a/T_v \sqcup \hat{F}_{a,b}$$

$$F' = \mathfrak{M}(T_v, T_w) \sqcup T_a/(T_v \sqcup T_w) \sqcup \hat{F}_a$$

Selection of External and Internal Merge

- empirical linguistic reasons: only External/Internal Merge
- Example: Sideward Merge would generate non-grammatical constructions like
  "Which sister of John wonders who likes a picture of"
- so want the Sideward Merge cases to be eliminated by some optimality principle (more costly, less efficient)
- what is the cost function? that makes Sideward Merge more costly than EM and IM?
- Minimal Search, Minimal Yield (discussion below)

Cases of 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

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

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

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

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

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

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

(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

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

- grafting

$$(\mathcal{B} \otimes \mathrm{id})(\alpha \sqcup T_2 \otimes 1) = \begin{array}{ccc} & \wedge & \\ \alpha & & T_2 \end{array} \otimes 1$$

- 1 is unit of product

$$\begin{array}{ccc} & \wedge & \\ \alpha & & T_2 \end{array} \sqcup 1 = \begin{array}{ccc} & \wedge & \\ \alpha & & T_2 \end{array}$$

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

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

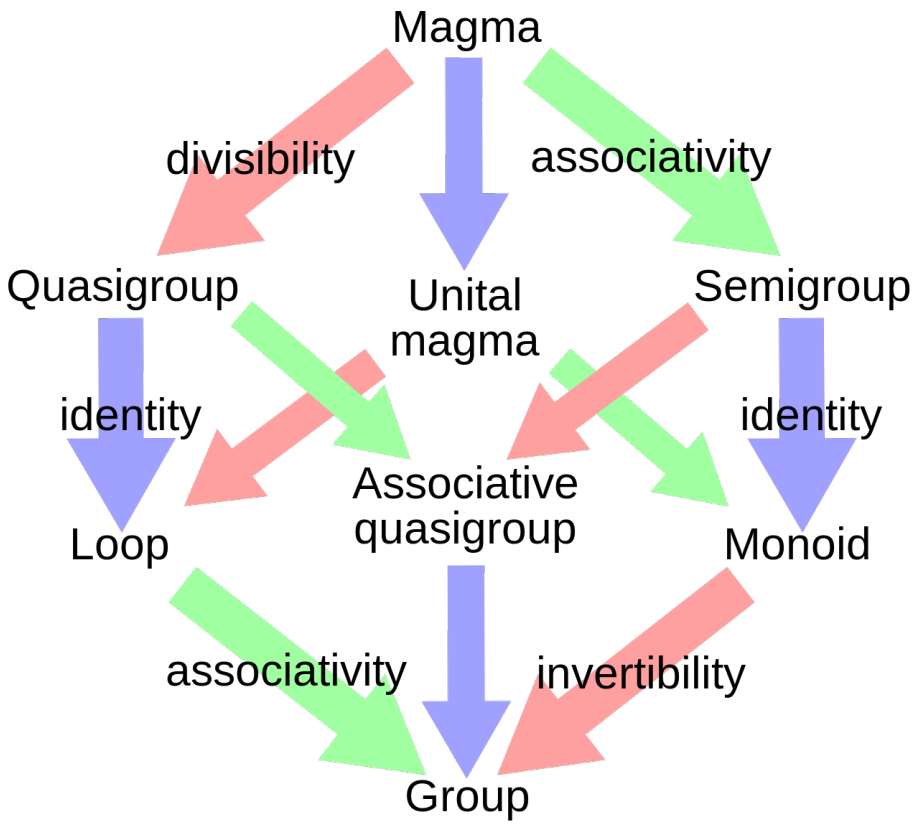
Internal Merge: preliminary discussion

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

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

- so can extend magma of syntactic object to unital magma

observation: algebraic structure "below groups":



- so can have an operator $\mathfrak{M}_{S,\mathbf{1}}$ acting on workspace $F = \sqcup_k T_k$

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

- but here $\delta_{S,\mathbf{1}} = \delta_S$ taking term

$$T_v \otimes T_a / T_v \sqcup \hat{F}_a = T_v \sqcup \mathbf{1} \otimes T_a / T_v \sqcup \hat{F}_a$$

  of coproduct with $T_v \simeq S$
- and $\mathfrak{B}(T_v \sqcup \mathbf{1}) = \mathfrak{M}(T_v, \mathbf{1}) = T_v$
- so new workspace $F' = \mathfrak{M}_{S,\mathbf{1}}(F)$ is

$$T_v \sqcup T_a / T_v \ \sqcup \hat{F}_a$$

Note: one of linguistic reasons for introducing workspaces, Internal Merge deposits a copy T_v of the extracted accessible term in the workspace... this is done here by the operation $\mathfrak{M}_{S,\mathbf{1}}$

Cases of Merge: Internal Merge

- once have workspace $T_v \sqcup T_a/T_v \sqcup \hat{F}_a$
- proceed with $\mathfrak{M}_{S,S'}$ where $S \simeq T_v$ and $S' \simeq T_a/T_v$
- obtain new workspace

$$\mathfrak{M}_{T_v,T_a/T_v} \circ \mathfrak{M}_{T_v,\mathbf{1}}(F) = \mathfrak{M}(T_v, T_a/T_v) \sqcup \hat{F}_a$$

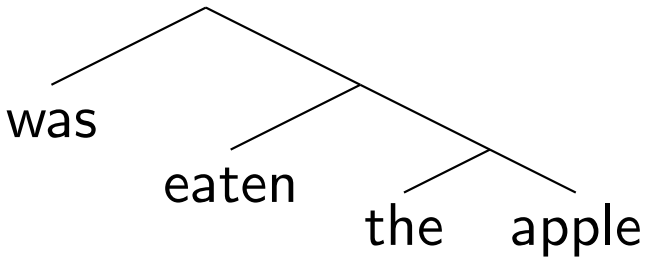
- Internal Merge is $\mathfrak{M}_{T_v,T_a/T_v} \circ \mathfrak{M}_{T_v,\mathbf{1}}$

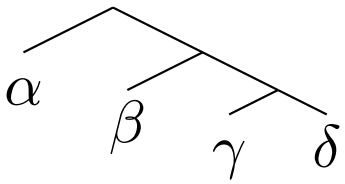
Comment on Internal Merge

- the operation $\mathfrak{M}_{\beta,\mathbf{1}}$ does not really "exist in isolation" only in composition as Internal Merge (has wrong effect on WS's size etc to exist on its own: see later!)
- so IM not really a "composite operation"
- Note: could eliminate $\mathfrak{M}_{\beta,\mathbf{1}}$ step entirely if make grafting $\mathcal{B}$ act on terms $T_v \otimes T/T_v$ instead of terms $T_v \sqcup T/T_v \otimes 1$, but this requires a "coindexing" problem, where need to keep track of which component in a forest (on left or right of coprod) will use for input.

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

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

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

$$\overset{\frown}{\gamma \quad \delta} \otimes \overset{\frown}{\alpha \quad \beta}$$

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

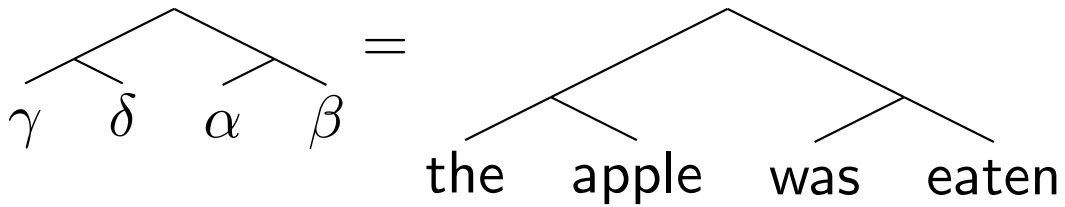
- read this coproduct term as

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

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

$$\overset{\frown}{\gamma \quad \delta} \sqcup \overset{\frown}{\alpha \quad \beta}$$

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



- this is the new worspace

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

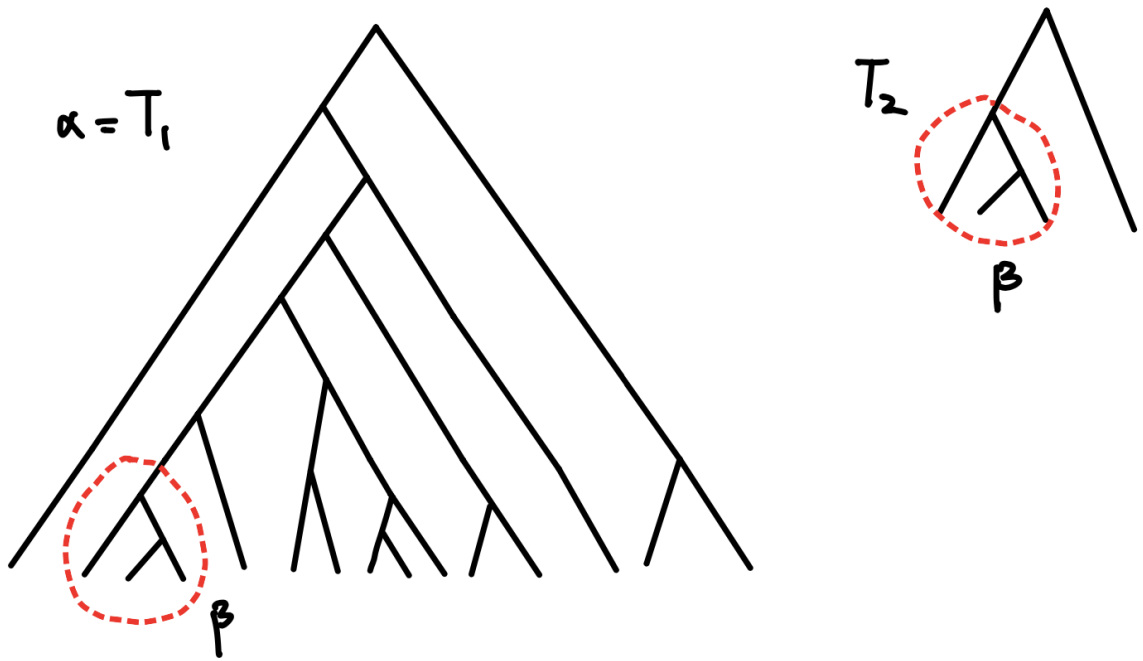
- so our description of Internal Merge matches what expected

Internal Merge: an analogy with Peano arithmetic

- von Neumann construction of the natural numbers: $X \mapsto \{X, \{X\}\}$
- $0 = \emptyset$, $1 = \{\emptyset\}$, $2 = \{\emptyset, \{\emptyset\}\}$, $3 = \{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$, etc
- looks like the binary Internal Merge *but* with a copy of the whole X as "accessible term"
- also difference at first step with unary $\emptyset \mapsto \{\emptyset\}$

Reducing the cases of Merge

- want only External Merge $\mathfrak{M}(T_a, T_b) \sqcup \hat{F}_{a,b}$ and Internal Merge $\mathfrak{M}(T_a, T_a/T_v) \sqcup \hat{F}_a$ to be the *optimal cases*
- want the other cases with $T_v \subsetneq T_a$ and $T_w \subsetneq T_b$ (with T_a and T_b different or same) to be less likely (more costly, less efficient)
- what is the cost function? that makes these more costly than EM and IM?
- Minimal Search, Minimal Yield
- idea of Minimal Search: it is less expensive to locate either two components (External) or one component and a subcomponent of the same (Internal) than other configurations of subcomponents across different components
- *is this true?* (.... yes, even if it seems not)



Minimal Search for $\mathfrak{M}_{S,S'}$ with $S = T_1$ and $S' = \beta$ should assign lower cost to the copy of β inside T_1 than the one inside T_2

Cost function for Minimal Search

1. *Weight of extracted accessible terms*: the cost increases with the depth at which the accessible term is located so weight of ϵ^{d_v}, with $d_v = \mathrm{dist}(v, v_T)$ for v_T the root of T
2. *Weight of quotient terms*: cost of performing the operation $T \mapsto T/T_v$ has weight of ϵ^{-d_v} because for larger d_v more similar T and T/T_v smaller less costly change
3. *Weight of multiple extractions/quotients*: if $F_{\underline{v}} = T_{v_1} \sqcup \cdots \sqcup T_{v_n}$ weight $\epsilon^{d_{\underline{v}}}$ with $d_{\underline{v}} = d_{v_1} + \cdots + d_{v_n}$
4. *Cost of grafting*: cost $c(\mathfrak{M}(T, 1)) = 0$ as no change and if T has weight ϵ^d and T' has weight $\epsilon^{d'}$, then we set $c(\mathfrak{M}(T, T')) = d + d'$ and weight $\epsilon^d \cdot \epsilon^{d'} = \epsilon^{c(\mathfrak{M}(T,T'))}$
5. *Cost/weight of derivations*: $\varphi = \mathfrak{M}_{S_n, S'_n} \circ \cdots \circ \mathfrak{M}_{S_1, S'_1}$ cost $c(\varphi) = \sum_i c(\mathfrak{M}_{S_i, S'_i})$ weight $\epsilon^{c(\phi)} = \prod_i \epsilon^{c(\mathfrak{M}_{S_i, S'_i})}$

Minimal Search

- incorporate keeping track of cost/weight of a Merge operation

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

$$\mathcal{B}^{\epsilon}(\alpha \sqcup \beta) = \epsilon^{c(\mathfrak{M}(\alpha,\beta))}\, \mathcal{B}(\alpha \sqcup \beta)$$

- The only zero-cost Merge operations are Internal and External Merge. All other forms of Merge have higher cost.
- For $\epsilon < 1$, Internal and External Merge are the leading order terms in any derivation.
- In the limit $\epsilon \to 0$ only derivations in which all the Merge operations are Internal and External Merge remain.

Minimal Yield and Complexity

- Measures of size of workspaces: number b_0 of connected components, number α of accessible terms $\mathrm{Acc}(F)$, number $\sigma = b_0 + \alpha$ of accessible terms $\mathrm{Acc}'(F)$
- Minimal Yield:

$$\begin{array}{ll} \sigma(\Phi(F)) = \sigma(F) + 1 & \text{(minimality of yield)} \\ b_0(\Phi(F)) \leq b_0(F) & \text{(no divergence)} \\ \alpha(\Phi(F)) \geq \alpha(F) & \text{(no information loss)} \end{array}$$

Counting for different quotients

$\alpha = \#$ acc terms, $\sigma = b_0 + \alpha$

- $\alpha(\mathfrak{M}(T, T')) = \alpha(T) + \alpha(T') + 2$
- $\sigma(\mathfrak{M}(T, T')) = \sigma(T) + \sigma(T') + 1$
- $\alpha(T) = \alpha(T_v) + \alpha(T/^d T_v) + 2$
- $\sigma(T) = \sigma(T_v) + \sigma(T/^d T_v) + 1$
- $\alpha(T) = \alpha(T_v) + \alpha(T/^c T_v) + 1$
- $\sigma(T) = \sigma(T_v) + \sigma(T/^c T_v)$

External and Internal Merge

Type of Merge	*Coproduct*	b_0	α	σ
External	Δ^c and Δ^d	-1	$+2$	$+1$
Internal	Δ^c	0	$+1$	$+1$
Internal	Δ^d	0	0	0

But... the $\mathfrak{M}_{S,1}$ does not exist on its own

Merge	*Coproduct*	b_0	α	σ
$\mathfrak{M}_{S,1}$	Δ^c	$+1$	-1	0
$\mathfrak{M}_{S,1}$	Δ^d	$+1$	-2	-1

Other forms of Merge

(2b): $\mathfrak{M}(T_v, T') \sqcup T/T_v$

(3b): $\mathfrak{M}(T_v, T_w) \sqcup T/T_v \sqcup T'/T_w$

(3a): $\mathfrak{M}(T_v, T_w) \sqcup T/(T_v \sqcup T_w)$

Merge	*Coproduct*	b_0	α	σ
(3b)	Δ^c	$+1$	0	$+1$
(3b)	Δ^d	$+1$	-2	-1
(2b)	Δ^c	0	$+1$	$+1$
(2b)	Δ^d	0	0	0
(3a)	Δ^c	$+1$	0	$+1$
(3a)	Δ^d	$+1$	-2	-1

Minimal Yield eliminates all but (2b)

No Complexity Loss

- No complexity loss: $\Phi : \mathfrak{F}_{\mathcal{SO}_0} \to \mathfrak{F}_{\mathcal{SO}_0}$, for $F = \sqcup_{a \in \mathcal{I}} T_a$ set of components $\pi_0(F) \simeq \mathcal{I}$ with induced map $\Phi_0 : \pi_0(F) \to \pi_0(\Phi(F))$ ($a \in \pi_0(F)$ to $\Phi_0(a)$ component of $\pi_0(\Phi(F))$ that contains the image of the root vertex of the component T_a of F)
- *No Complexity Loss* for Φ if for all $a \in \pi_0(F)$

$$\deg(\Phi_0(a)) \geq \deg(a)$$

$\deg(a)$ degree of component T_a in the Hopf algebra.

External Merge: $\deg(\mathfrak{M}(T_i, T_j)) = \deg(T_i) + \deg(T_j)$, OK
Internal Merge: $\deg(T_v, T/T_v) = \deg(T)$, OK
Other forms all have components mapping to lower degree: e.g. in $\mathfrak{M}(T_v, T_w) \sqcup T/T_v \sqcup T'/T_w$ root vertices of T and T' map to T/T_v and T'/T_w lower degree

Merge must be binary

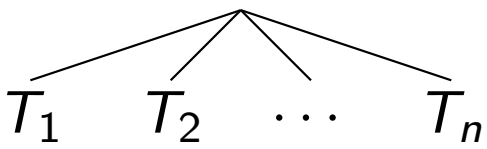
- Also an optimization: a Merge with any other $n \geq 3$ arity would both undergenerate and overgenerate with respect to binary Merge (observed by Riny Huijbregts)
- syntactic objects of a hypothetical n-ary Merge

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

- rooted n-ary trees (without planar structure)

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

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

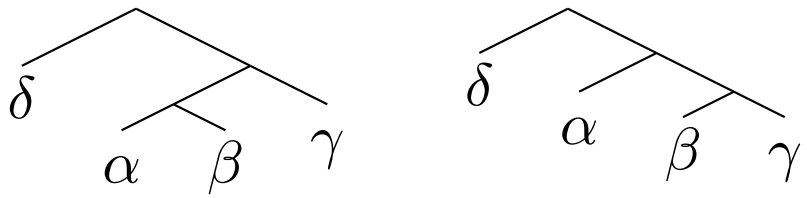


- by number of leaves

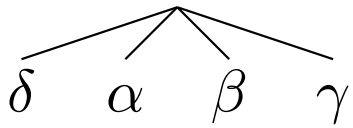
$$\mathcal{SO}^{(n)} = \bigsqcup_{k \geq 1} \mathcal{SO}^{(n)}_{k(n-1)+1}$$

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 $\mathfrak{B}$)

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

- 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)

Merge is Markovian

- usually assumed "the operations of syntax are Markovian"
- in the course of Merge derivations at each step the Merge operations have access to only the current state of the workspace
- does this assumption *follow* from this formulation of Merge?
- is there a stronger sense in which "Merge is Markovian"?

Markov chains

- $\mathcal{V} = \oplus_\ell \mathcal{V}_\ell$ graded vector space, basis $\mathcal{B} = \cup_\ell \mathcal{B}_\ell$
- $\mathcal{K} : \mathcal{H} \to \mathcal{H}$ be a *linear operator*
- preserves graded subspaces, $\mathcal{K}_\ell : \mathcal{H}_\ell \to \mathcal{H}_\ell$
- matrix $K_{\mathcal{B}_\ell}$ representing $\mathcal{K}$ in basis $\mathcal{B}_\ell$

$$K_{\mathcal{B}_\ell}(x, y) \geq 0 \quad \forall x, y \in \mathcal{B}_\ell$$

- $\forall x \in \mathcal{B}_\ell \; \exists y \in \mathcal{B}_\ell$ such that $K_{\mathcal{B}_\ell}(x, y) > 0$

Then associated Markov chain:

- set of states $\mathcal{B}_\ell$
- transition matrix: stochastic matrix

$$\tilde{K}_{\mathcal{B}_\ell}(x, y) = c(x)^{-1} K_{\mathcal{B}_\ell}(x, y)$$

with $c(x) = \sum_y K_{\mathcal{B}_\ell}(x, y) > 0$ so

$$\tilde{K}_{\mathcal{B}_\ell}(x, y) \geq 0 \; \forall x, y \in \mathcal{B}_\ell \quad \text{and} \quad \sum_y \tilde{K}_{\mathcal{B}_\ell}(x, y) = 1 \; \forall x \in \mathcal{B}_\ell$$

stronger form: Markov chain from linear operator

- $\mathcal{K}_\ell : \mathcal{H}_\ell \to \mathcal{H}_\ell$ gives a stochastic matrix after a rescaling of the base $\mathcal{B}$
- suppose there is a Perron-Frobenius eigenfunction

$$\sum_y K_{\mathcal{B}_\ell}(x,y)\eta(y) = \eta(x)$$

with $\eta(x) > 0$ for all $x \in \mathcal{B}_\ell$

- then

$$\hat{K}_{\mathcal{B}_\ell}(x,y) = \frac{\eta(y)}{\eta(x)} K_{\mathcal{B}_\ell}(x,y)$$

is the transition matrix of a Markov chain (i.e. stochastic matrix)

$$\hat{K}_{\mathcal{B}_\ell}(x,y) \geq 0 \ \forall x,y \in \mathcal{B}_\ell \quad \text{and} \quad \sum_y \hat{K}_{\mathcal{B}_\ell}(x,y) = 1 \ \forall x \in \mathcal{B}_\ell$$

Perron–Frobenius theorem

- A square matrix with $A_{ij} \geq 0$
- irreducible: directed graph with n vertices and edge $i \to j$ if $A_{ij} \neq 0$ is strongly connected (any two vertices connected by a directed path)
- then $\exists$ Perron–Frobenius eigenfunction η (left/right) with eigenvalue $\lambda = \rho(A)$ spectral radius and all $\eta_i > 0$

$$\sum_j A_{ij}\, \eta_j = \lambda\, \eta_i$$

- $\tilde{A}$ normalized by the spectral radius gives stochastic matrix

$$\hat{A}_{ij} = \eta_i^{-1}\, \eta_j\ \tilde{A}_{ij}$$

Hopf algebras Markov chains

- $\mathcal{H}$ combinatorial Hopf algebra (graded, connected, commutative) with linear basis $\mathcal{B}$
- linear operators $\mathcal{K} = \sqcup \circ \mathcal{Q} \circ \Delta$ preserving grading
- such that a global rescaling $\mathcal{K}_\rho = \rho^{-1}\mathcal{K}$ has Perron–Frobenius eigenfunction with $\eta(x) > 0$

$$\sum_y K_{\mathcal{B}_\ell,\rho}(x,y)\eta(y) = \eta(x)$$

- then in rescaled basis $\hat{\mathcal{B}}_\ell = \{\eta(x)^{-1}x\,|, x \in \mathcal{B}_\ell\}$ the matrix $K_{\hat{\mathcal{B}}_\ell,\rho}$ is stochastic (transition matrix of a Markov chain)
- Examples: $\sqcup \circ \Delta$; $\sqcup^a \circ \Delta^a$, $\sqcup \circ \Pi_d \circ \Delta$

introduced in

- P. Diaconis, C.Y.A. Pang, A. Ram, *Hopf algebras and Markov chains: two examples and a theory*. J. Algebraic Combin. 39 (2014), no. 3, 527–585.
- C.Y.A. Pang, *Markov chains from descent operators on combinatorial Hopf algebras*, arXiv:1609.04312.

weaker form

- linear operators $\mathcal{K} = \sqcup \circ \mathcal{Q} \circ \Delta$ preserving grading
- such that $\forall x \in \mathcal{B}_\ell \ \exists y \in \mathcal{B}_\ell$ with $K_\ell(x, y) > 0$
- then local rescaling (dependent on x)
  $\tilde{K}_\ell(x, y) = c_\ell(x)^{-1} K_\ell(x, y)$ is stochastic (transition matrix of a Markov chain)

invariant subspaces

- start with a (multi)set Ω of lexical items and syntactic features in $\mathcal{SO}_0$ with $\ell = \#\Omega > 2$

$$\Omega = \alpha_1 \sqcup \cdots \sqcup \alpha_\ell$$

- take span $\mathcal{W}_\Omega$ of $F \in \mathfrak{F}_{\mathcal{SO}_0}$ with same set of decorated leaves

$$L(F) = \Omega$$

- subspace $\mathcal{V}_\Omega \subset \mathcal{W}_\Omega$ spanned by F with non-empty set of edges
- invariant under Merge operations

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

Merge action

- all possible Merge operations on a given F: take

$$\mathcal{K} = \sum_{S,S'} \mathfrak{M}_{S,S'}$$

(finite sum when applied to a given F) agrees with operator

$$\mathcal{K} = \sqcup \circ (\mathfrak{B} \otimes \mathrm{id}) \circ \Pi_{(2)} \circ \Delta$$

$\Pi_{(2)}$ projection of $\mathcal{H} \otimes \mathcal{H}$ onto span of $S \sqcup S' \otimes F''$
$(S, S' \in \mathfrak{T}_{\mathcal{SO}_0})$

- also consider

$$\Xi := \sqcup \circ \Pi_{(1)} \circ \Delta$$

$\Pi_{(1)}$ projection onto span of $T \otimes F'$, $T \in \mathfrak{T}_{\mathcal{SO}_0}$

- for all possible Internal Merge also operator $\mathcal{K} \circ \Xi$

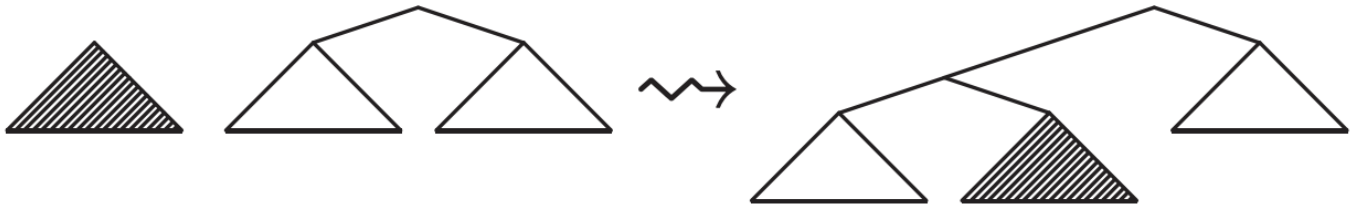
irreducibility (strong connectedness)

- $K_{\Omega,\ell}(F, F')$ matrix elements of $\mathcal{K}$ and $K\Xi_{\Omega,\ell}(F, F')$ matrix elements of $\mathcal{K} \circ \Xi$ on $\mathcal{V}_\Omega$
- graph $G_{\Omega,\mathcal{K}_\ell}$ vertices $F \in \mathcal{B}_{\Omega,\ell}$ basis of $\mathcal{V}_\Omega$ directed edge when $K_{\Omega,\ell}(F, F') > 0$ (same for $\mathcal{K} \circ \Xi$)
- take two vertices (for simplicity T and T' case of forests same) want a path between them: disassemble T' and reassemble T
- locate α_i, α_j in $L(T')$ that are joined to $\mathfrak{M}(\alpha_i, \alpha_j)$ in T
- use $\alpha_i \sqcup \alpha_j \otimes T'/(\alpha_i \sqcup \alpha_j)$ term of coproduct to get an arrow $K_{\Omega,\ell}(T', F_1) > 0$ to $F_1 = \mathfrak{M}(\alpha_i, \alpha_j) \sqcup T'/(\alpha_i \sqcup \alpha_j)$

- continue to all such pairs and to higher structures T_v, T_w that occur in T' that appear as $\mathfrak{M}(T_v, T_w)$ in T
- get chain of arrows $T' \to F_1 \to \cdots \to F_r$
- since $L(T) = L(T')$ all terms occur this way until no leaves of T' left
- then use another chain $F_r \to F_{r+1} \to \cdots \to F_k = T$ of External Merges to assemble these together to T
- strong connectedness: so get strong form of Hopf algebra Markov chain
- note use of Sideward/Countercyclic Merge (for disassembling) here not just EM/IM
- using only EM/IM get only the weaker form (and with only EM not even that)

Other proposed operations in generative inguistics can be compared with this Merge model using the algebraic structure
Example: tree insertions that describe "countercyclic movement" and "Late Merge"

- the use of insertion operations at inner vertices of trees has been suggested in the form of "Late Merge" (and also criticized)
- is this really an "extension" of Merge? is it incompatible? is it already obtainable from the usual Merge operations?



Lie algebras and Hopf algebras

- Lie algebra: vector space $\mathcal{L}$ with bilinear operation $[\cdot,\cdot] : \mathcal{L} \otimes \mathcal{L} \to \mathcal{L}$ satisfying $[L_1, L_2] = -[L_2, L_1]$ and Jacobi identity

$$[L_1, [L_2, L_3]] + [L_2, [L_3, L_1]] + [L_3, [L_1, L_2]] = 0$$

- *right pre-Lie structure* (or left pre-Lie) $\lhd : \mathcal{L} \otimes \mathcal{L} \to \mathcal{L}$

$$(L_1 \lhd L_2) \lhd L_3 - L_1 \lhd (L_2 \lhd L_3) = (L_1 \lhd L_3) \lhd L_2 - L_1 \lhd (L_3 \lhd L_2)$$

  from which get Lie bracket

$$[L_1, L_2] := L_1 \lhd L_2 - L_2 \lhd L_1$$

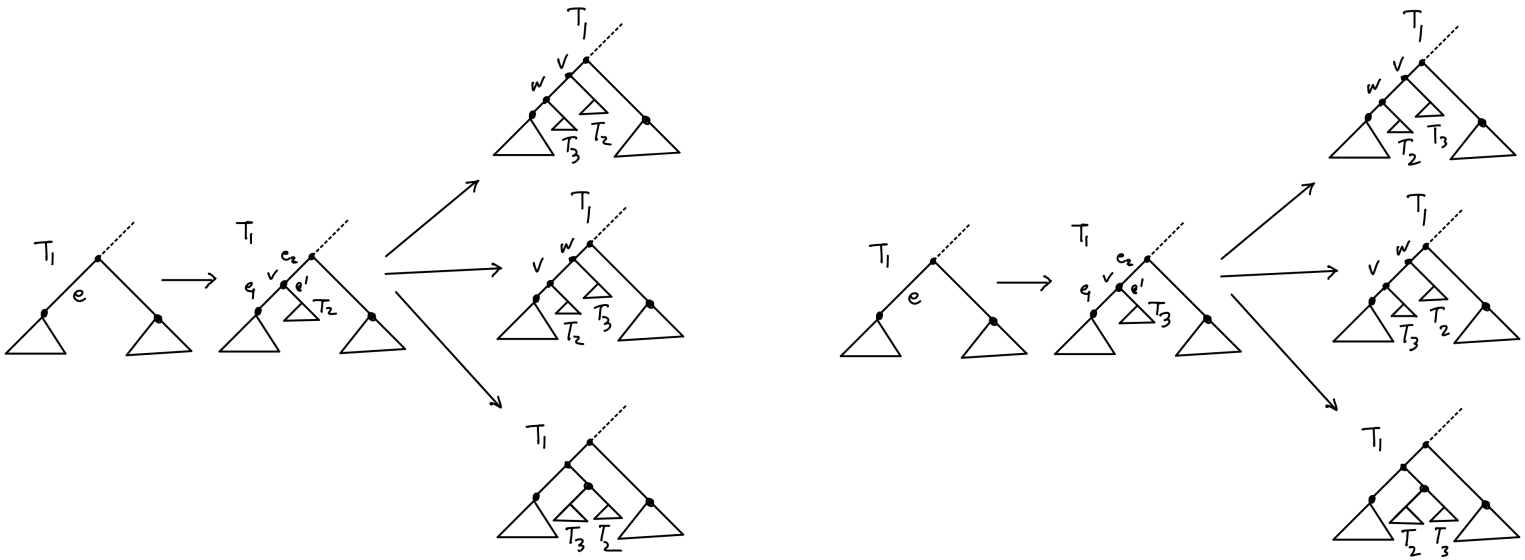
- a graded connected Hopf algebra has an associated Lie algebra (primitive elements of the dual Hopf algebra)

Insertion Lie algebra dual to Hopf algebra of workspaces

- $T_1 \lhd_e T_2$ denote binary rooted tree obtained by splitting edge e with new vertex v and attaching to v a new edge e' connected to root of T_2
- pre-Lie structure

$$T_1 \lhd T_2 = \sum_{e \in E(T_1)} T_1 \lhd_e T_2$$

satisfies identity because



Milnor-Moore theorem

dual Hopf algebra $\mathcal{H}^\vee$ of a graded connected commutative Hopf algebra $\mathcal{H}$ is the universal enveloping algebra of the Lie algebra $\mathcal{L}$ of the primitive elements of $\mathcal{H}^\vee$

$$\mathcal{H} = U(\mathcal{L})^\vee$$

- $\mathcal{H}$ and dual $\mathcal{H}^\vee$ with dual basis for trees generators of $\mathcal{H}$

$$T \mapsto Z_T$$

- indecomposable for product of $\mathcal{H}$ primitive for coproduct of $\mathcal{H}^\vee$
- Lie algebra structure

$$(Z_T \star Z_S - Z_S \star Z_T)(F) =$$

$$\sum_{F_{\underline{v}}} Z_T(F_{\underline{v}}) Z_S(F/F_{\underline{v}}) - \sum_{F_{\underline{w}}} Z_S(F_{\underline{w}}) Z_T(F/F_{\underline{w}})$$

- vanishes on forests with $b_0 > 0$ and leaves just insertion Lie algebra bracket

- what one sees formulated as "countercyclic movement" is the insertion operations $T \lhd_e T'$ and $T \rhd_{e'} T'$
- these are not actually a new structure (an extension of Merge) but are determined by the structure underlying the usual Merge

Example of how "Late Merge" is used: sentence like

[These pictures of John$_i$]$_j$ seemed to him$_i$ [–$_j$ to be very good].

apparent problem: condexing and violation of "condition C" of Binding Theory, interpreted as *of John* is late-merged into its position, but not needed as this sentence is a *single phase* (so no violation of binding conditions), does not require a different form of Merge

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

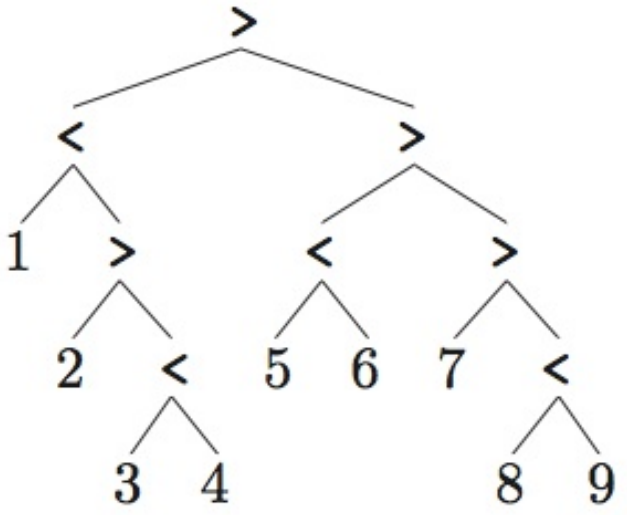
Note: this notion of *head function* is equivalent to the properties of *head* defined in Chomsky's "Bare phrase structure", 1995

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*

Example of a head function on a tree



- *abstract head function* h defined on a subdomain $\mathrm{Dom}(h) \subset \mathfrak{T}_{\mathcal{SO}_0}$, that assigns to a $T \in \mathrm{Dom}(h)$ a $h : T \mapsto h_T$ with $h_T : V^o(T) \to L(T)$ a head function as above
- such $\mathrm{Dom}(h) \subset \mathfrak{T}_{\mathcal{SO}_0}$ is in general not a submagma: can have $T_1, T_2 \in \mathrm{Dom}(h)$ but $\mathfrak{M}(T_1, T_2) \notin \mathrm{Dom}(h)$
- this happens with syntactic head: *exocentric constructions* when $\mathfrak{M}(T_1, T_2) \notin \mathrm{Dom}(h)$

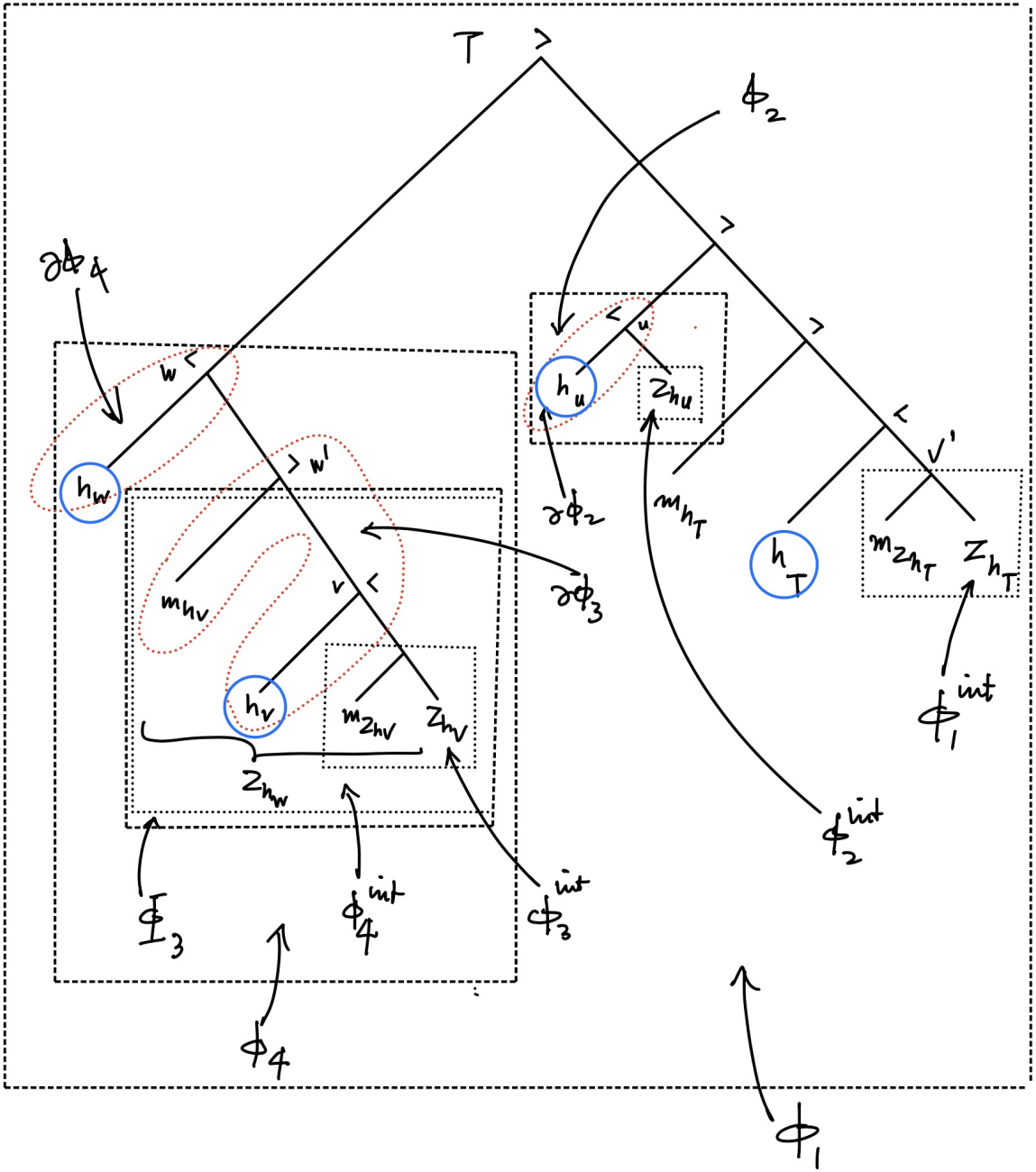
head functions and planar embeddings

- a *head function* $h_T : V^{int}(T) \to L(T)$ determines a planar embedding $T^{\pi^{h_T}}$ of T: put the marked edge below each vertex to the left of the other
- Kayne's "linear correspondence axiom": 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

Phase Theory



head and complement

- *complement of head*: elements head *must* combine with
- *modifiers*: structures the head does not necessarily have to combine with
- complemented abstract head function

$$h_{T,Z} : V^o(T) \to L(T) \times (\mathrm{Acc}(T) \cup \{1\})$$

from non-leaf vertices (with $1 = \emptyset$)

$$h_{T,Z}(v) = (h_T(v), Z_v)$$

$v \mapsto h_T(v)$ abstract head function and complement Z_v (possibly empty) $Z_v \subset T_{s_{h_T(v)}}$, with $s_{h_T(v)}$ the sister vertex of $h_T(v)$ in T

- rest of $T_{s_{h_T(v)}}$: modifiers

Phase algorithm

- head function h_T partitions vertices of T into path $\{\gamma_\ell\}_{\ell \in L(T)}$ (following the head: highest vertex v_ℓ of each path "maximal projection")
- set of phases of T

$$L_\Phi(T) = \{\ell \in L(T) \,|\, \#V(\gamma_\ell) > 1\}$$

- Φ_ℓ phase associated to $\ell \in L_\Phi(T)$

$$\Phi_\ell = \{T_v \in \mathrm{Acc}'(T) \,|\, T_v \subseteq T_{v_\ell}\}$$

- phase interior: $\ell \in L_\Phi(T)$ and v mother vertex above ℓ on path γ_ℓ and s_ℓ sister vertex of ℓ under v; if $Z_\ell = \emptyset$ then $\Phi^\circ_\ell = \emptyset$; if $Z_\ell \neq \emptyset$

$$\Phi^\circ_\ell := \{T_v \in \mathrm{Acc}(T) \,|\, T_v \subseteq T_{s_\ell}\}$$

- phase edge: if $Z_\ell = \emptyset$, take $\partial\Phi_\ell = \Phi_\ell$ and if $Z_\ell \neq \emptyset$

$$\partial\Phi_\ell := \{T_v \in \mathrm{Acc}'(T) \,|\, T_w \subseteq T_{v_\ell} \text{ and } T_w \not\subseteq T_{s_\ell}\}$$

all accessible terms of T_{v_ℓ} not in interior of phase.

- partial ordering on set $L_\Phi(T)$ of phases induced by inclusion: $\ell \prec \ell'$ if $\Phi_\ell \subset \Phi_{\ell'}$, so Φ_ℓ *lower phase* and $\Phi_{\ell'}$ *higher phase*
- *inaccessible terms* at phase Φ_ℓ: interiors of lower phases

$$\Upsilon_\ell := \left\{ T_v \in \mathrm{Acc}(T) \,|\, T_v \in \bigcup_{\ell' \prec \ell} \Phi^\circ_{\ell'} \right\}$$

- complement $\Phi_\ell \smallsetminus \Upsilon_\ell$ *available for computation* at phase Φ_ℓ
- counting head at edge of phase, if excluding head movement also count as not accessible
- action of Internal Merge takes an accessible term $T_{u,a} \subset T_u$ that is *in the interior* of the (current) phase Φ°_ℓ and move it to the edge of the (current) phase of the resulting object
- when next phase is formed (by External Merge of some additional structure) what has remained in interior of previous phase becomes unaccessible

- subspace $\mathcal{V}^h(\mathfrak{F}_{\mathcal{SO}}) \subset \mathcal{V}(\mathfrak{F}_{\mathcal{SO}})$ spanned by forests $F = \sqcup_a T_a$ with all the components $T_a \in \mathrm{Dom}(h) \subset \mathfrak{T}_{\mathcal{SO}}$
- Phase coproduct:

$$\Delta^c_\Phi(T) = \sum_{\underline{v} \in \Phi_{h_T} \setminus \Upsilon_{h_T}} F_{\underline{v}} \otimes T/^\omega F_{\underline{v}}$$

only extracting compatibly with the phase structure: significant reduction in size

labeling algorithm

- designed to make these objects interpretable at the syntax-semantics interface (assignment of labels at non-leaf vertices)
- head function h is *raising* (Moro's dynamical asymmetry) if
  - for $T \in \mathrm{Dom}(h)$ and accessible term $T_v \subset T$ with $h(T) = h(T/^d T_v)$ Internal Merge

    $\mathfrak{M}(T_v, T/^c T_v) \in \mathrm{Dom}(h)$ with $h(\mathfrak{M}(T_v, T/^c T_v)) = h(T/^d T_v)$

  - for $T \in \mathfrak{T}_{\mathcal{SO}_0}$ and accessible term $T_v \subset T$ with Internal Merge $\mathfrak{M}(T_v, T/^c T_v) \in \mathrm{Dom}(h)$ and $T/^d T_v \in \mathrm{Dom}(h)$

    $$h(\mathfrak{M}(T_v, T/^c T_v)) = h(T/^d T_v)$$

- so if $T = \mathfrak{M}(T_1, T_2)$ and either T_1 or T_2 raises through IM then $T \in \mathrm{Dom}(h)$ and can label via h_T
- can also extend labeling when $h(T_1)$ and $h(T_2)$ share feature
- some objects still remain unlabelled, rejected as non-parsable at semantic interface

Phases as "block-spin renormalization"

- blocking the interior of previous phases in further steps of derivation is similar idea to "block-spin" renormalization in physics
- aggregate components at shorter distances (lower level in structure construction)
- key point here: this is not just fixing a size of substructures but building a hierarchy of substructures following data of a head function and its complement structure

some work already tried comparing Merge to MERA-type renormalization based on tensor networks... but critically missing the structure of Phases

- A.J. Gallego, R. Orús, *Language design as information renormalization*, arXiv:1708.01525v5
- V. Pestun, Y. Vlassopoulos, *Tensor network language model*, arXiv:1710.10248

FormSet

- "primitive coproduct" on workspaces $\Delta_P(T) = T \otimes 1 + 1 \otimes T$ and $\Delta_P(F) = \sqcup_a \Delta_P(T_a)$
- $\mathcal{B}$ grafting operations

$$\sqcup \circ (\mathcal{B} \otimes \mathrm{id}) \circ \Delta_P$$

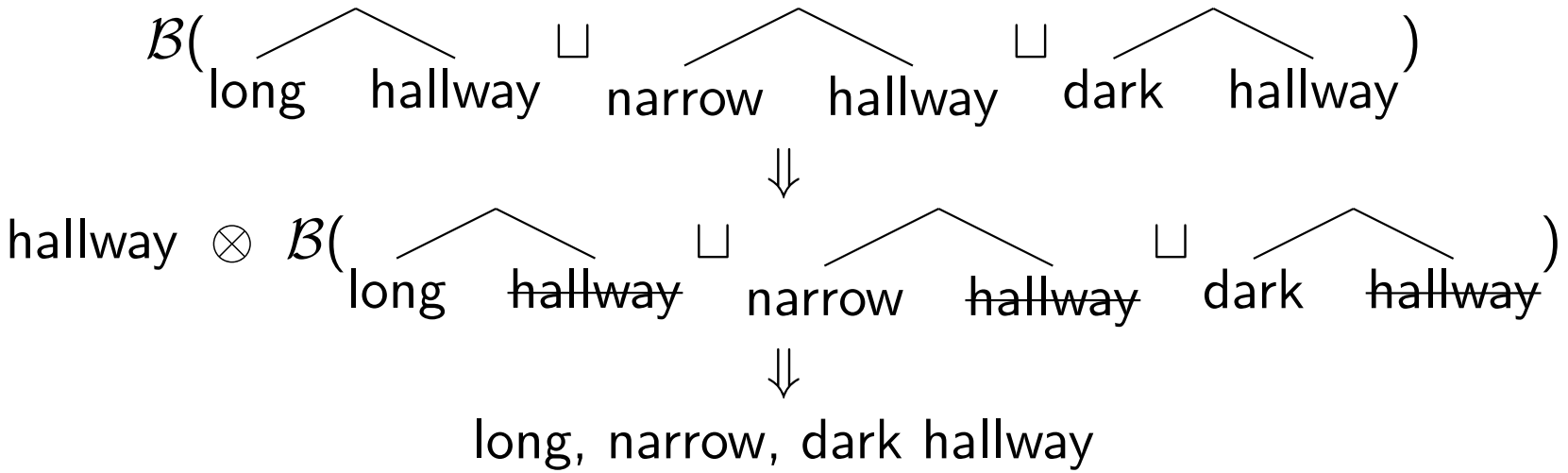
- it is *not* an n-ary Merge (very different algebraic properties from *n*-ary Merge)
- responsible for unbounded unstructured sequences (see Fong-Oishi)

$$\mathcal{B}(\text{John} \sqcup \text{Bill} \sqcup \text{my friends} \sqcup \text{the actor who won the Oscar})$$

$$\mathcal{B}(\text{ran} \sqcup \text{danced} \sqcup \text{took a vacation})$$

- selects "diagonals" for FormCopy operation

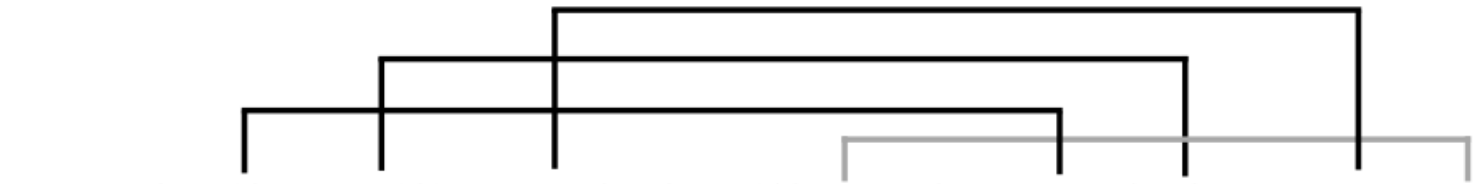
selects "diagonals" for FormCopy operation:



coming up next: Externalization

- in preparation for that: a look back at cross serial dependencies
- how does one build with EM and IM a tree that accounts for the cross serial case

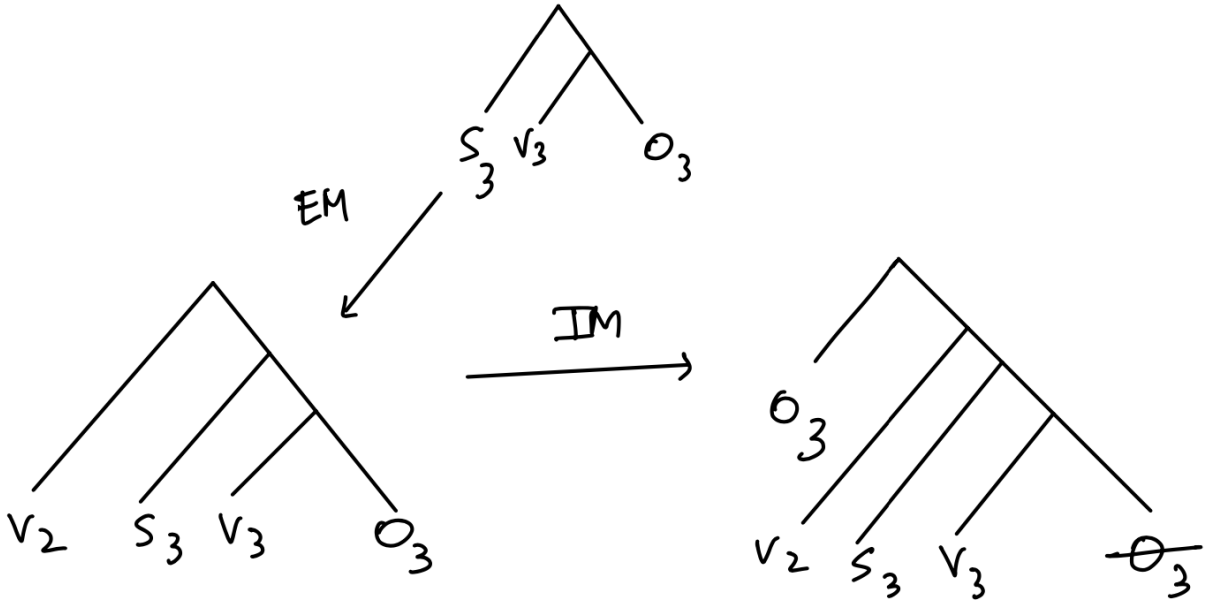
Example



...omdat ik Cecilia Henk de nijlpaarden zag helpen voeren
...because I Cecilia Henk the hippos saw help feed
'...because I saw Cecilia help Henk feed the hippos'

...omdat ik Cecilia Henk de nijlpaarden zag-0 help-en voer-en
C S_1 S_2 S_3 O_3 V_1 T_1 V_2 T_2 V_3 T_3

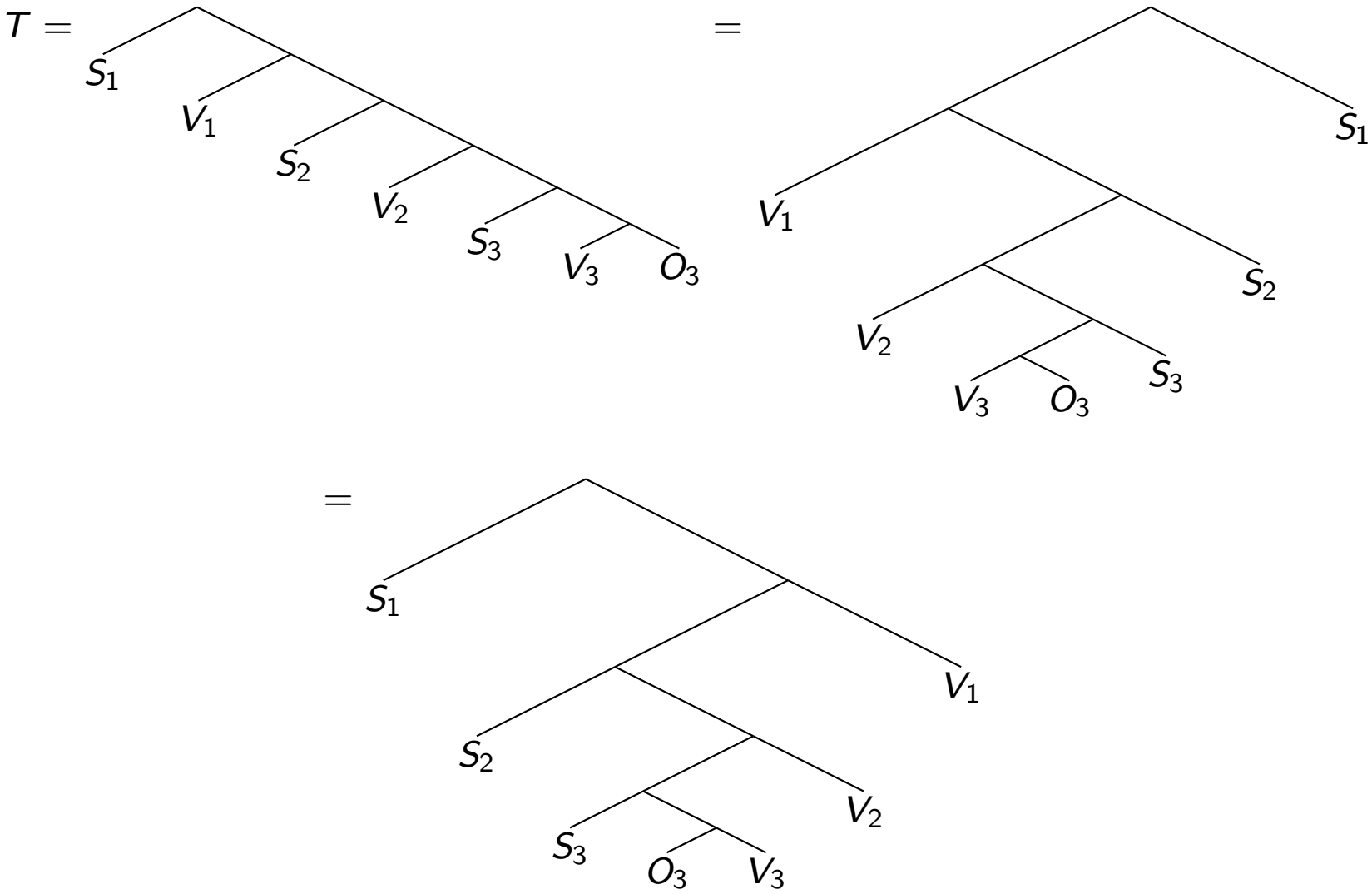
T= tense (morphology), V=verb, S=subject, O=object, C=complementizer



the crossed dependence $V_3 - O_3$ may be ascribed to more flexible word order in the presence of more morphology
issue with Phases (movement from interior of lower phase)

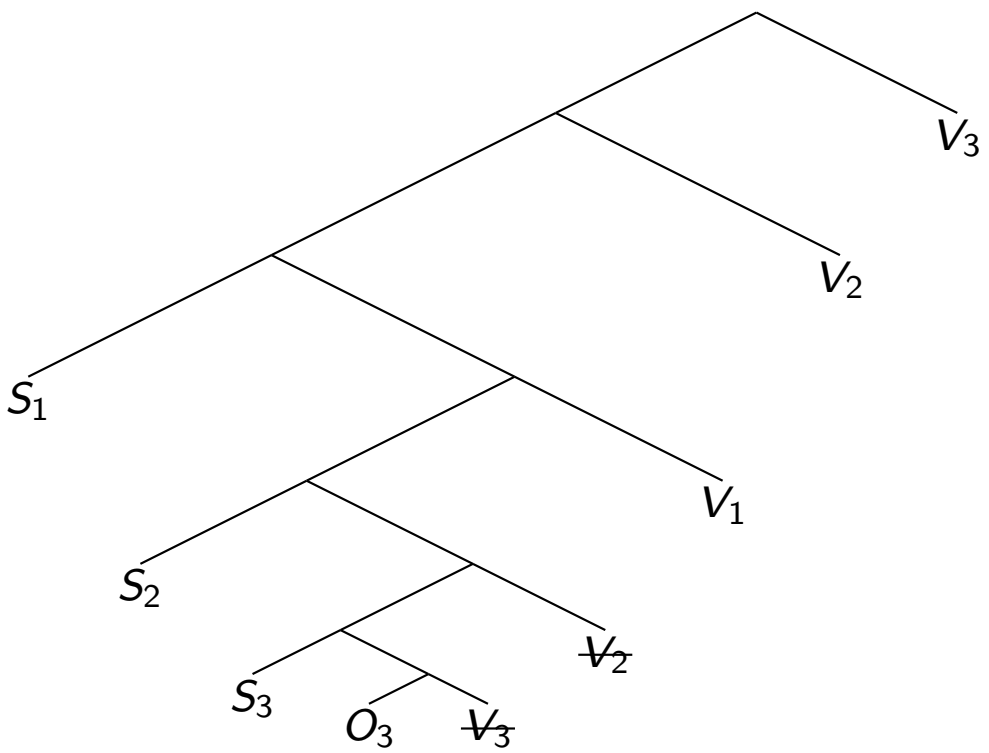
but if look at cross-serial dependence $S_i - V_i$

- start with non-planar tree



- verb raising movement (edge of phase) followed by choice of planar embedding

$T' =$



- Merge generates all these structures: both T and T'
- in externalization process some are selected and some rejected depending on language specific constraints and planar structure is assigned
- planar embedding + quotient selection