Lecture 10: Comparison with Physics
Ma 191c: Mathematical Models of Generative Linguistics

Matilde Marcolli

Caltech, Spring 2024

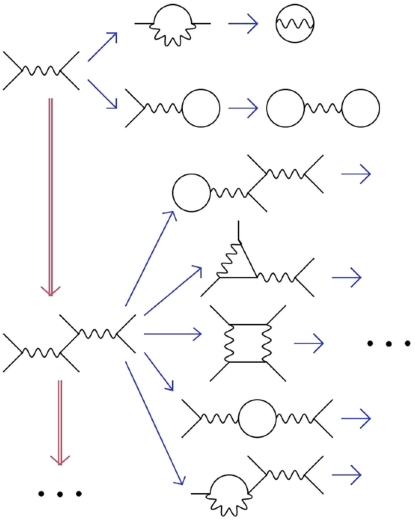
Generative structures in physics:
perturbative Quantum Field Theory (QFT)

- perturbative expansion of computation of Feynman integrals
- contributions labelled by graphs (Feynman graphs of the QFT)
- order of the expansion = loop order = first Betti number of graphs
- contribution of each graph an integral in momenta associated to (internal) edges; external edges incoming/outgoing momenta; momentum conservation at vertices
- integral often divergent: renormalization to extract finite *meaningful physical values*
- *consistency over substructures* for renormalization

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

Formal languages formulation (graph grammars)



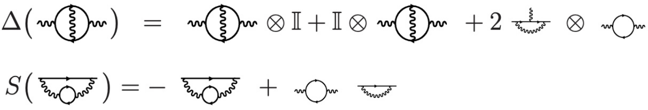
Example: formal languages approach – the generative grammar for the Feynman graphs of the $\phi^2 A$ physical theory (graph grammars: usually context sensitive)

Formal languages formulation

- M. Marcolli, A. Port, *Graph grammars, insertion Lie algebras, and quantum field theory*, Math. Comput. Sci. 9 (2015), no. 4, 391–408.

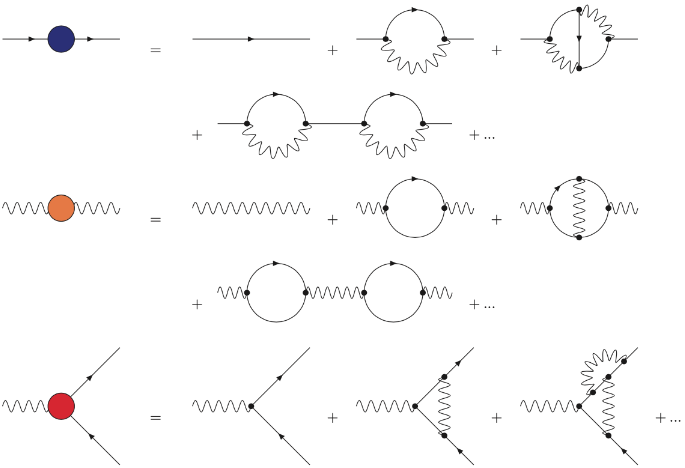
Hopf algebra formulation

- D. Kreimer, *On the Hopf algebra structure of perturbative quantum field theories*, Adv. Theor. Math. Phys. 2 (1998), no. 2, 303–334
- A. Connes, D. Kreimer, *Hopf algebras, renormalization and noncommutative geometry*, Comm. Math. Phys. 199 (1998), no. 1, 203–242.



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

Dyson-Schwinger equations also formulated in terms of the Hopf algebra structure



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

From Renormalization to Syntax-Semantics Interface

- 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

Quick overview of the physics setting for comparison

Setting of Perturbative Quantum Field Theory

- Action functional in D dimensions

$$S(\phi) = \int \mathcal{L}(\phi) d^D x = S_0(\phi) + S_{int}(\phi)$$

- Lagrangian density

$$\mathcal{L}(\phi) = \frac{1}{2}(\partial\phi)^2 - \frac{m^2}{2}\phi^2 - \mathcal{L}_{int}(\phi)$$

- Perturbative expansion: Feynman rules and Feynman diagrams

$$S_{eff}(\phi) = S_0(\phi) + \sum_{\Gamma} \frac{\Gamma(\phi)}{\#\mathrm{Aut}(\Gamma)} \quad \text{(1PI graphs)}$$

Algebraic renormalization in perturbative QFT

- A. Connes, D. Kreimer, *Renormalization in quantum field theory and the Riemann-Hilbert problem*, I and II, hep-th/9912092, hep-th/0003188
- A. Connes, M. Marcolli, *Renormalization, the Riemann-Hilbert correspondence, and motivic Galois theory*, hep-th/0411114
- K. Ebrahimi-Fard, L. Guo, D. Kreimer, *Integrable Renormalization II: the general case*, hep-th/0403118

Two step procedure:

• Regularization: replace divergent integral $U(\Gamma)$ by function with poles

• Renormalization: pole subtraction with consistency over subgraphs (Hopf algebra structure)

• Kreimer, Connes–Kreimer, Connes–Marcolli: Hopf algebra of Feynman graphs and BPHZ renormalization method in terms of Birkhoff factorization and differential Galois theory

• Ebrahimi-Fard, Guo, Kreimer: algebraic renormalization in terms of Rota–Baxter algebras

Connes–Kreimer Hopf algebra $\mathcal{H} = \mathcal{H}(\mathcal{T})$ (depends on theory)

- Free commutative algebra in generators Γ 1PI Feynman graphs
- Grading: loop number (or internal lines)

$$\deg(\Gamma_1 \cdots \Gamma_n) = \sum_i \deg(\Gamma_i), \quad \deg(1) = 0$$

- Coproduct:

$$\Delta(\Gamma) = \Gamma \otimes 1 + 1 \otimes \Gamma + \sum_{\gamma \in \mathcal{V}(\Gamma)} \gamma \otimes \Gamma/\gamma$$

- Antipode: inductively

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

for $\Delta(X) = X \otimes 1 + 1 \otimes X + \sum X' \otimes X''$

Rota–Baxter algebra of weight $\lambda = -1$

$\mathcal{R}$ commutative unital algebra
$T : \mathcal{R} \to \mathcal{R}$ linear operator with

$$T(x)T(y) = T(xT(y)) + T(T(x)y) + \lambda T(xy)$$

- typical case: $\mathcal{R} = \mathbb{C}[[z]][z^{-1}]$ Laurent series and $T =$ projection on the polar part
- T determines splitting $\mathcal{R}_+ = (1 - T)\mathcal{R}$, $\mathcal{R}_- =$ unitization of $T\mathcal{R}$; both $\mathcal{R}_\pm$ are algebras

Feynman rule

• $\phi : \mathcal{H} \to \mathcal{R}$ *commutative algebra homomorphism*

• assignment of *regularized* but not yet *renormalized* values to Feynman graphs: regularized value is a Laurent series, original divergent integral is the pole at $z = 0$

from CK Hopf algebra $\mathcal{H}$ to Rota–Baxter algebra $\mathcal{R}$ weight -1

$$\phi \in \mathrm{Hom}_{\mathrm{Alg}}(\mathcal{H}, \mathcal{R})$$

• Note: ϕ does *not know* that $\mathcal{H}$ Hopf and $\mathcal{R}$ Rota-Baxter, only commutative algebras

- Birkhoff factorization $\exists \phi_\pm \in \mathrm{Hom}_{\mathrm{Alg}}(\mathcal{H}, \mathcal{R}_\pm)$

$$\phi = (\phi_- \circ S) \star \phi_+$$

where $\phi_1 \star \phi_2(X) = \langle \phi_1 \otimes \phi_2, \Delta(X) \rangle$

- Connes-Kreimer inductive formula for Birkhoff factorization:

$$\phi_-(X) = -T(\phi(X) + \sum \phi_-(X')\phi(X''))$$

$$\phi_+(X) = (1 - T)(\phi(X) + \sum \phi_-(X')\phi(X''))$$

where $\Delta(X) = 1 \otimes X + X \otimes 1 + \sum X' \otimes X''$

- Recovers what known in physics as BPHZ renormalization procedure in physics

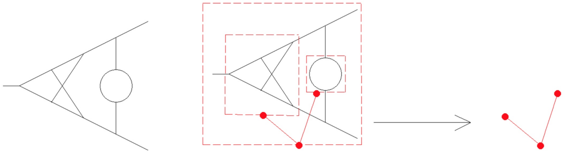
- case of Laurent series $\Phi_+(X)(z)$ is in $\mathbb{C}[[z]]$ so $\Phi_+(0)$ exists and is the renormalized value; $\Phi_-(X)(z)$ is divergent at $z = 0$: counterterms, subtraction of divergences...

Connes–Kreimer Hopf algebra of rooted trees

- polynomial algebra generated by the planar rooted trees T
- coproduct: sum over all admissible cuts

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

- grading by span of the planar rooted trees with k internal vertices
- antipode defined inductively on graded bialgebras
- used as reformulation of the Connes–Kreimer Hopf algebra of Feynman graphs in perturbative QFT



Combinatorial Dyson–Schwinger equations

- C. Bergbauer and D. Kreimer, *Hopf algebras in renormalization theory: locality and Dyson-Schwinger equations from Hochschild cohomology*, hep-th/0506190
- K. Yeats, *Rearranging Dyson-Schwinger Equations*, AMS 2011.
- L. Foissy, *Classification of systems of Dyson-Schwinger equations of the Hopf algebra of decorated rooted trees*, Adv. Math. 224 (2010), no. 5, 2094–2150
- L. Foissy, *Lie algebras associated to systems of Dyson-Schwinger equations*, Adv. Math. 226 (2011), no. 6, 4702–4730.

Dyson–Schwinger equations and Hopf subalgebras

- If grafting operator satisfies *cocycle condition*, then solutions of Dyson–Schwinger equations form a *Hopf subalgebra*

Insertion and Hochschild 1-cocycles

- T =forest: *grafting operator* $B^+_\delta(T)$ = sum of planar trees with new root vertex added with incoming flags equal number of trees in T and a single output flag and decoration δ

- cocycle condition:

$$\Delta B^+_\delta = (id \otimes B^+_\delta)\Delta + B^+_\delta \otimes 1$$

equivalent to $\tilde{\Delta} B^+_\delta = (id \otimes B^+_\delta)\tilde{\Delta} + id \otimes B^+_\delta(1)$ with $\tilde{\Delta}(x) := \sum x' \otimes x''$ (non-primitive part) and $B^+_\delta(1) = v_\delta$ (single vertex, label δ): first term admissible cuts root vertex attached to $\rho_C(T)$, second term admissible cut separating root vertex.

Dyson–Schwinger equations and Hopf subalgebras

(Bergbauer–Kreimer)

• Dyson–Schwinger equations in a Hopf algebra of the form

$$X = 1 + \sum_{n=1}^{\infty} c_n B_{\delta}^{+}(X^{n+1})$$

• associative algebra $\mathcal{A}$ (subalgebra of $\mathcal{H}$) generated by components x_n of unique solution of DS equation

• using cocycle condition for B_{δ}^{+} get

$$\Delta(x_n) = \sum_{k=0}^{n} \Pi_k^n \otimes x_k, \quad \text{where} \quad \Pi_k^n = \sum_{j_1+\cdots+j_{k+1}=n-k} x_{j_1} \cdots x_{j_{k+1}}$$

$\Rightarrow$ Hopf subalgebra

• generalized by Foissy for broader class of DS equations in Hopf algebras, including systems

Variant: Hopf ideals

- DS equation $X = 1 + \sum_{n=1}^{\infty} c_n \, B_\delta^+(X^{n+1})$
- *ideal* $\mathcal{I}$ generated by the components x_n (with $n \geq 1$) of solution
- cocycle condition for $B_\delta^+ \Rightarrow \mathcal{I}$ Hopf ideal

elements of $\mathcal{I}$ finite sums $\sum_{m=1}^{M} h_m x_{k_m}$ with $h_m \in \mathcal{H}$ and x_k components of unique solution of DS equation

Hopf ideal condition: $\Delta(\mathcal{I}) \subset \mathcal{I} \otimes \mathcal{H} \oplus \mathcal{H} \otimes \mathcal{I}$

coproduct $\Delta(x_k)$: primitive part $1 \otimes x_k + x_k \otimes 1$ in $\mathcal{H} \otimes \mathcal{I} \oplus \mathcal{I} \otimes \mathcal{H}$; other terms in $\mathcal{I} \otimes \mathcal{I}$, so coproducts $\Delta(h_m x_{k_m})$ in $\mathcal{H} \otimes \mathcal{I} \oplus \mathcal{I} \otimes \mathcal{H}$.

$\Rightarrow$ *quotient Hopf algebra* $\mathcal{H}_{\mathcal{I}} = \mathcal{H}/\mathcal{I}$

Note: commutative Hopf algebra; if noncommutative use two-sided ideals

Hopf algebras and Lie algebras in QFT

- Connes-Kreimer Hopf algebra $\mathcal{H}_{CK}$ of Feynman graphs is graded connected and commutative
- dual to an affine group scheme G_{CK}
- Milnor–Moore theorem: dual Hopf algebra is the universal enveloping algebra of the Lie algebra of primitive elements

$$\mathcal{H}_{CK}^{\vee} = U(\mathfrak{g}_{CK})$$

- The Lie bracket of the Lie algebra $\mathfrak{g}_{CK}$ is described by insertions at vertices

Lie algebras and pre-Lie structures

• Lie algebra: vector space V with bilinear bracket $[\cdot,\cdot]$ operation with $[x,y] = -[y,x]$ and Jacobi identity

$$[x,[y,z]] + [z,[x,y]] + [y,[z,x]] = 0.$$

• tangent space at the identity of a Lie group is a Lie algebra

• pre-Lie structure: a bilinear map $\star : V \otimes V \to V$ on a vector space V

$$(x \star y) \star z - x \star (y \star z) = (x \star z) \star y - x \star (z \star y)$$

identity of associators under $y \leftrightarrow z$

• Given a pre-Lie structure

$$[x,y] = x \star y - y \star x$$

is a Lie bracket (pre-Lie identity $\Rightarrow$ Jacobi identity)

Lie Algebra of an Affine Group Scheme

• functor $\mathfrak{g} : \mathrm{Alg}_{\mathbb{K}} \to \mathrm{Lie}$ from category of commutative algebras over $\mathbb{K}$ to category of Lie algebras

- $\mathfrak{g}(A)$ linear maps $L : \mathcal{H} \to A$ such that

$$L(xy) = L(x)\epsilon(y) + \epsilon(x)L(y), \quad \forall x, y \in \mathcal{H}$$

- Lie bracket

$$[L_1, L_2](x) = (L_1 \otimes L_2 - L_2 \otimes L_1)(\Delta(x))$$

• Milnor–Moore theorem: for a commutative graded connected ($\mathcal{H}_0 = \mathbb{K}$) Hopf algebra the affine group scheme G dual to $\mathcal{H}$ is completely determined by its Lie algebra $\mathfrak{g}$

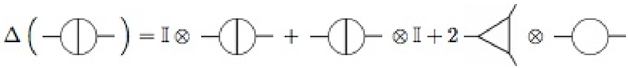
Hopf algebra of Feynman graphs

• commutative algebra generated by all the 1PI graphs G of the QFT (polynomial algebra in the G)

• comultiplication $\Delta : \mathcal{H} \to \mathcal{H} \otimes \mathcal{H}$ (coassociative, non-cocommutative)

$$\Delta(G) = G \otimes 1 + 1 \otimes G + \sum_{\gamma \subset G} \gamma \otimes G/\gamma$$

• Example:



• antipode (related algebra and coalgebra structure) constructed inductively on number of edges (or loops)

Hopf algebra and Lie algebra

• $\mathcal{H} = \oplus_{n\geq 0}\mathcal{H}_n$ with $\mathcal{H}_0 = \mathbb{C}$ connected commutative graded Hopf algebra

• $A =$ commutative algebra, $\mathrm{Hom}(\mathcal{H}, A) = \mathcal{G}(A)$ is a group

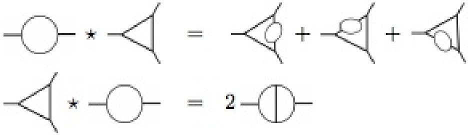
• the Hopf algebra $\mathcal{H}$ is determined by the Lie algebra $\mathcal{L}$ of $\mathcal{G}(\mathbb{C})$

• insertion of graphs is a pre-Lie operator $\Rightarrow$ Lie algebra

• insertion Lie algebra of Feynman graphs

• given two graphs G_1, G_2: count in how many ways can insert one into the other at a vertex (so that external edges glued to corolla of edges at the vertex)

• Examples of graph insertions:



gives pre-Lie structure

Lie algebra of Feynman graphs

- Lie bracket

$$[G, G'] = \sum_{v \in V(G)} G \circ_v G' - \sum_{v' \in V(G')} G' \circ_{v'} G,$$

sum over vertices and counting all possible ways of inserting the other graph at that vertex matching external edges

Comparison between Hopf algebra and Formal Languages viewpoint in Physics

- Matilde Marcolli, Alexander Port, *Graph grammars, insertion Lie algebras, and quantum field theory*, Math. Comput. Sci. 9 (2015), no. 4, 391–408.

Graph Grammars and Quantum Field Theory

• Example of a different setting where formal languages can be applied, with a different class of formal grammars (graph grammars)

Graph Grammars main results:

1. Any context free graph grammar determines an insertion Lie algebra and a commutative Hopf algebra
2. Feynman graphs of a QFT are a graph language

Graph Grammars

Formal languages adapted to parallelism in computation

- instead of linear languages: strings in an alphabet obtained by production rules of a grammar
- grammars that produce a language consisting of a family of graphs
- production rules that substitute parts of a graph with other parts (gluing)
- an initial graph as starting point
- edge and vertex labels by terminal and non-terminal symbols

Graphs

Two main ways of thinking about graphs:

First description:

• $V(G)$ = set of vertices; $E(G)$ = set of edges; $\partial : E(G) \to V(G) \times V(G)$

• if G is oriented (directed) then source and target $s, t : E(G) \to V(G)$

• Σ_V, Σ_E sets of vertex and edge labels; $L_{V,G} : V(G) \to \Sigma_V$, $L_{E,G} : E(G) \to \Sigma_E$ assignment of labels

Second description:

- $C(G)$ = set of corollas with assigned valences
(a vertex with n half-edges)
- $\mathcal{F}(G)$ = set of all half-edges
- involution: $\mathcal{I} : \mathcal{F}(G) \to \mathcal{F}(G)$
- edges: pairs (f, f') with $f \neq f'$ in $\mathcal{F}(G)$ with $\mathcal{I}(f) = f'$
(an edge is a gluing of two half edges)
- external edges: $f \in \mathcal{F}(G)$ fixed by the involution $\mathcal{I}$
(half-edges not matched to anything else)
- assignment of labels $L_{\mathcal{F},G} : \mathcal{F}(G) \to \Sigma_{\mathcal{F}}$ and $L_{V,G} : C(G) \to \Sigma_V$

$$L_{\mathcal{F},G} \circ \mathcal{I} = L_{\mathcal{F},G}$$

(the involution must match labels)

Graph Grammar

$$(N_E, N_V, T_E, T_V, P, G_S)$$

- edge labels: $\Sigma_E = N_E \cup T_E$ non-terminal and terminal
- vertex labels: $\Sigma_V = N_V \cup T_V$ non-terminal and terminal
- G_S = start graph
- P = production rules: a finite set

Production rules of a Graph Grammar

$$P = (G_L, G_R, H)$$

- G_L = labelled graph (l.h.s. of production)
- G_R = labelled graph (r.h.s. of production)
- H = labelled graph with label preserving isomorphisms

$$\phi_L : H \stackrel{\simeq}{\to} \phi_L(H) \subset G_L, \quad \phi_R : H \stackrel{\simeq}{\to} \phi_R(H) \subset G_R$$

(isomorphic subgraphs in G_L and G_R)

Meaning: the production rule searches for a copy of G_L inside a given graph G and glues in a copy of G_R by identifying them along the common subgraph H

Context-free Graph Grammars

• when all production rules $P = (G_L, G_R, H)$ have G_L (hence H) a single vertex

• Chomsky hierarchy for Graph Grammar (different from the one for linear languages) was identified in

- M. Nagl, *Graph-Grammatiken: Theorie, Implementirung, Anwendung*, Vieweg, 1979

References on Graph Grammars

- H. Ehrig, K. Ehrig, U. Prange, G. Taentzer, *Fundamentals of algebraic graph transformation*. New York: Springer, 2010.
- H. Ehrig, H.J. Kreowski, G. Rozenberg, *Graph-grammars and their application to computer science*, Lecture Notes in Computer Science, Vol. 532, Springer, 1990.
- G. Rozenberg, *Handbook of Graph Grammars and Computing by Graph Transformation. Volume 1: Foundations*, World Scientific, 1997.

Comparing generative processes in QFT

- A. Connes, D. Kreimer, *Insertion and elimination: the doubly infinite Lie algebra of Feynman graphs*. Ann. Henri Poincaré 3 (2002), no. 3, 411–433.
- K. Ebrahimi-Fard, J.M. Gracia-Bondia, F. Patras. *A Lie theoretic approach to renormalization.* Commun. Math. Phys 276, no. 2 (2007): 519-549.
- M. Bachmann, H. Kleinert, A. Pelster, *Recursive graphical construction for Feynman diagrams of quantum electrodynamics*, Phys.Rev. D61 (2000) 085017.
- H. Kleinert, A. Pelster, B. Kastening, M. Bachmann, *Recursive graphical construction of Feynman diagrams and their multiplicities in ϕ^4 and in $\phi^2 A$ theory*, Phys.Rev. E62 (2000) 1537–1559.

From context free Graph Grammars to Insertion Lie Algebras

• Insertion Graph Grammar consists of data

$$(N_E, N_V, T_E, T_V, P, G_S)$$

edge labels $\Sigma_E = N_E \cup T_E$, nonterminal and tarminal, vertex labels is given $\Sigma_V = N_V \cup T_V$, start graph is G_S and production rules $P = (G_L, H, G_R)$, with G_L and G_R labelled graphs and H a labelled graph with isomorphisms

$$\phi_L : H \stackrel{\simeq}{\to} \phi_L(H) \subset G_L, \quad \phi_R : H \stackrel{\simeq}{\to} \phi_R(H) \subset G_R.$$

ϕ_L label preserving

• production $P = (G_L, H, G_R)$ searches for a copy of G_L inside a graph G and glues in a copy of G_R identifying them along common subgraph H, new labels matching those of $\phi_R(H)$

• context free if $G_L = \{v\}$ (hence $H = \{v\}$ also)

• formulation in terms of graphs as corollas and matched half-edges

• production rules $P = (G_L, H, G_R)$ as before with additional requirement that $\phi_L(E_{ext}(H, G_R)) \subset E_{ext}(G_L, G)$ and $\phi_R(E_{ext}(H, G_L)) \subset E_{ext}(G_R)$, for any G the production rule is applied to, $G_L \subset G$

• here gluing two graphs $G_L \cup_H G_L$ along common subgraph H by corollas

$$C_{G_L \cup_H G_L} = C_{G_L} \cup_{C_H} C_{G_R},$$

identifying corollas around each vertex of H in G_L and G_R and matching half-edges by involution

• here context-free: $G_L = H = C(v)$ corolla of a vertex v, and all vertices of graphs in the graph language have same valence

Insertion Operator and Lie Algebra

• given a context-free insertion graph grammar $\mathcal{G}$

• $\mathcal{V}$ vector space spanned by set $\mathcal{W}_{\mathcal{G}}$ of all the graphs obtained by repeated application of production rules starting with G_S (not same as graph language $\mathcal{L}_{\mathcal{G}}$ because also nonterminal labels)

• insertion operator $\triangleleft : \mathcal{V} \otimes \mathcal{V} \to \mathcal{V}$

$$G_1 \triangleleft G_2 = \sum_{v \in V(G_1)} P(v, v_2, G_2)(G_1) = \sum_{v \in V(G_1)} G_1 \triangleleft_v G_2$$

defines a pre-Lie structure on $\mathcal{V}$

• Lie algebra $\mathrm{Lie}_{\mathcal{G}}$ vector space $\mathcal{V}$ spanned graphs of $\mathcal{W}_{\mathcal{G}}$ with Lie bracket $[G_1, G_2] = G_1 \triangleleft G_2 - G_2 \triangleleft G_1$

• there is also a version using corollas, and there are versions for context-sensitive cases

The Graph Language of a Quantum Field Theory

- Note: this is not the same construction as the Lie algebra of the Connes–Kreimer Hopf algebra (because that would require an infinite number of production rules: generated by all primitive elements of the Hopf algebra

- This method based on just finitely many production rules that realize all Feynman graphs of a given QFT as the graph language of a graph grammar

Example: Feynman graph language of ϕ^4-theory

- quantum field theory with Lagrangian

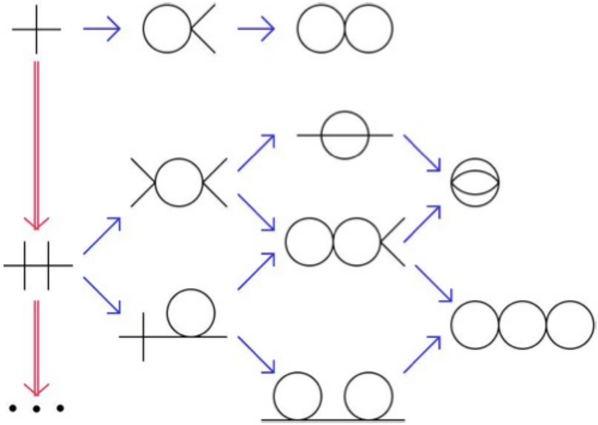
$$\mathcal{L}(\phi) = \frac{1}{2}(\partial\phi)^2 + \frac{1}{2}m^2\,\phi^2 + \frac{1}{4!}\lambda\,\phi^4$$

Feynman graphs have all vertices of valence four

- graph language $\mathcal{L}_{\mathcal{G}}$ generated by a graph grammar $\mathcal{G}$ with G_S a 4-valent corolla and two production rules:

1. $P(G_S, \{f, f'\} \subset \mathcal{F}_{G_S}, G_e)$ glues two external edges of G_S
2. $P(G_S, \{f\} \subset \mathcal{F}_{G_S}, G_S \cup_{f'} G_e)$ glues two copies of G_S along an edge

Example: Feynman graph language of ϕ^4-theory



Example: other scalar field theory examples ϕ^3 and ϕ^4

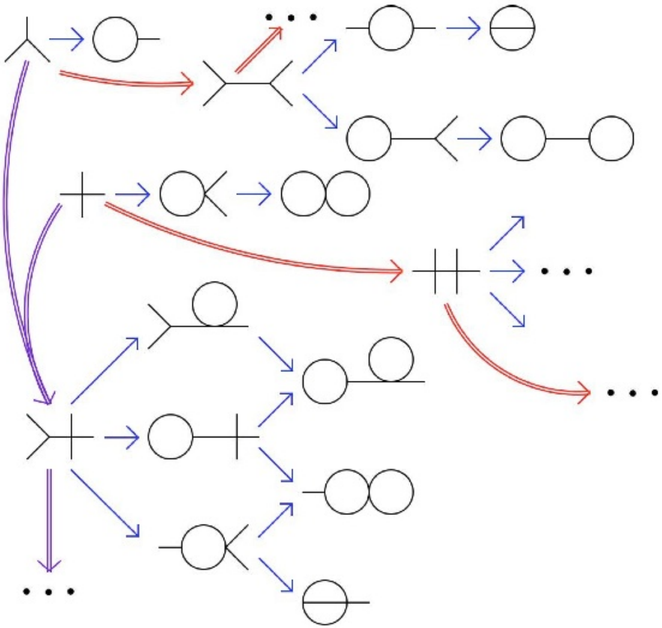
• scalar field theory with Lagrangian

$$\mathcal{L}(\phi) = \frac{1}{2}(\partial\phi)^2 + \frac{1}{2}m^2\,\phi^2 + \frac{1}{6}\lambda_3\,\phi^3 + \frac{1}{24}\lambda_4\,\phi^4$$

• start graph G_S given by a k-valent corolla, for smallest k in interaction Lagrangian (here $k = 3$) and production rules

1. $P(G_S, \{f, f'\} \subset \mathcal{F}_{G_S}, G_e)$ gluing two external edges of G_S
2. $P(G_S, G_e, G_{S,f})$ gluing a copy of G_e (edge propagator) to start graph G_S one half-edge of G_e with one half-edge of G_S other half edge f as new external edge
3. $P(G_{S,f_1,\ldots,f_r}, \{f_i\} \subset \mathcal{F}_{G_S}, G_{S,f_1,\ldots,f_r} \cup_{f_i=f'_j} G_{S,f'_1,\ldots,f'_s})$ gluing along an edge two corollas $G_{S,f_1,\ldots,f_r}$ and $G_{S,f'_1,\ldots,f'_s}$

Example: ϕ^3 and ϕ^4 terms



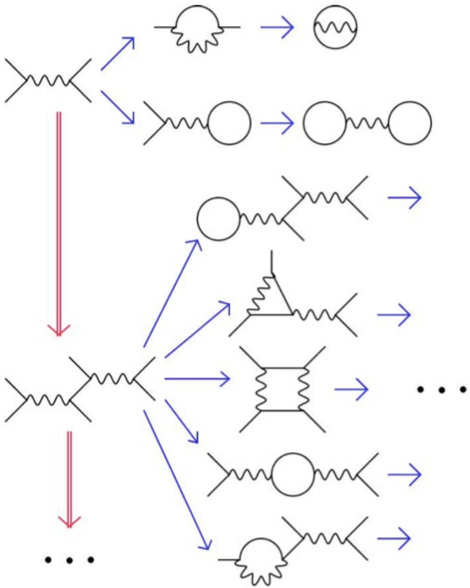
Example: Feynman graph language of $\phi^2 A$-theory

• similar to a ϕ^3 theory but with two fields A and ϕ (similar to electrodynamics) with a cubic interaction term $\phi^2 A$

• all graphs have trivalent vertices: corolla with one A-labelled half-edge and two ϕ-labelled half-edges

• graph grammar with initial graph that is more complicated than a corolla: two vertices connected by one A-edge and each with two ϕ half-edges and two production rules (gluing two ϕ half-edges; gluing two copies of initial graph along a ϕ-edge)

Example: Feynman graph language of $\phi^2 A$-theory



Intermediate step Manin's Renormalization and Computation

- Yu.I. Manin, *Renormalization and computation I: motivation and background*. OPERADS 2009, 181–222, Sémin. Congr., 26, Soc. Math. France, Paris, 2013
- Yu.I. Manin, *Infinities in quantum field theory and in classical computing: renormalization program*, Programs, proofs, processes, 307–316, Lecture Notes in Comput. Sci., 6158, Springer, 2010.
- Yu.I. Manin, *Renormalization and computation II: time cut-off and the halting problem*, Math. Struct. in Comp. Science, vol. 22, pp. 729–751, Cambridge University Press, 2012
- C. Delaney, M. Marcolli, *Dyson-Schwinger equations in the theory of computation*, Feynman amplitudes, periods and motives, pp. 79–107, Contemp. Math., 648, Amer. Math. Soc., 2015.

Manin's "renormalization and computation"

• Idea: treat noncomputable functions like infinities in QFT

• Renormalization = extraction of finite part from divergent Feynman integrals; extraction of "computable part" from noncomputables

• First step: build a Hopf algebra (flow charts) and a Feynman rule that detects the presence of noncomputability (infinities)

• Second step: BPHZ type Birkhoff factorization procedure to identify where undecidable part of computation is located in substructures and which quotient structures remain computable after "removal of infinities"

Primitive recursive functions

- generated by *basic functions*
  - Successor $s : \mathbb{N} \to \mathbb{N}$, $s(x) = x + 1$;
  - Constant $c^n : \mathbb{N}^n \to \mathbb{N}$, $c^n(x) = 1$ (for $n \geq 0$);
  - Projection $\pi_i^n : \mathbb{N}^n \to \mathbb{N}$, $\pi_i^n(x) = x_i$ (for $n \geq 1$);
- with *elementary operations*
  - Composition
  - Bracketing
  - Recursion

Elementary operations:

- Composition $\mathfrak{c}_{(m,m,p)}$: for $f : \mathbb{N}^m \to \mathbb{N}^n$, $g : \mathbb{N}^n \to \mathbb{N}^p$,

$$g \circ f : \mathbb{N}^m \to \mathbb{N}^p, \quad \mathcal{D}(g \circ f) = f^{-1}(\mathcal{D}(g));$$

- Bracketing $\mathfrak{b}_{(k,m,n_i)}$: for $f_i : \mathbb{N}^m \to \mathbb{N}^{n_i}$, $i = 1, \dots, k$,

$$f = (f_1, \dots, f_k) : \mathbb{N}^m \to \mathbb{N}^{n_1 + \cdots + n_k}, \quad \mathcal{D}(f) = \mathcal{D}(f_1) \cap \cdots \cap \mathcal{D}(f_k);$$

- Recursion $\mathfrak{r}_n$: for $f : \mathbb{N}^n \to \mathbb{N}$ and $g : \mathbb{N}^{n+2} \to \mathbb{N}$,

$$h(x_1, \dots, x_n, 1) := f(x_1, \dots, x_n),$$

$$h(x_1, \dots, x_n, k+1) := g(x_1, \dots, x_n, k, h(x_1, \dots, x_n, k)), \quad k \geq 1,$$

where recursively $(x_1, \dots, x_n, 1) \in \mathcal{D}(h)$ iff $(x_1, \dots, x_n) \in \mathcal{D}(f)$ and $(x_1, \dots, x_n, k+1) \in \mathcal{D}(h)$ iff $(x_1, \dots, x_n, k, h(x_1, \dots, x_n, k)) \in \mathcal{D}(g)$.

Manin's Hopf algebra of flow charts

- planar labelled rooted trees (bracketing and recursion are ordered: need planar)
- label set of vertices $\mathcal{D}_V = \{\mathfrak{c}_{(m,n,p)}, \mathfrak{b}_{(k,m,n_i)}, \mathfrak{r}_n\}$ (composition, bracketing, recursion)
- label set of flags $\mathcal{D}_F$ primitive recursive functions
- *admissible* labelings:

  - $\phi_V(v) = \mathfrak{c}_{(m,n,p)}$: v valence 3; labels $h_1 = \phi_F(f_1)$, $h_2 = \phi_F(f_2)$ incoming flags with domains and ranges $h_1 : \mathbb{N}^m \to \mathbb{N}^n$ and $h_2 : \mathbb{N}^n \to \mathbb{N}^p$; outgoing flag composition $h_2 \circ h_1 = \mathfrak{c}_{(m,n,p)}(h_1, h_2)$.

  - $\phi_V(v) = \mathfrak{r}_n$: v valence 3; labels $h_1 = \phi_F(f_1)$, $h_2 = \phi_F(f_2)$ incoming flags with domains and ranges $h_1 : \mathbb{N}^n \to \mathbb{N}$ and $h_2 : \mathbb{N}^{n+2} \to \mathbb{N}$, outgoing flag recursion $h = \mathfrak{r}_n(h_1, h_2)$.

  - $\phi_V(v) = \mathfrak{b}_{(k,m,n_i)}$: v must have valence $k+1$; labels $h_i = \phi_F(f_i)$ incoming flags with domain $\mathbb{N}^m$; outgoing flag bracketing $f = (f_1, \dots, f_k) = \mathfrak{b}_{(k,m,n_i)}(f_1, \dots, f_k)$.

- Coproduct, grading, antipode from Hopf algebra of rooted trees

Variants on the Hopf algebra of flow charts

- noncommutative Hopf algebra $\mathcal{H}^{nc}_{\mathrm{flow},\mathcal{P}}$
- Hopf algebra with only vertex labels $\mathcal{H}^{nc}_{\mathrm{flow},\mathcal{V}}$
- Use only binary operations (valence 3 vertices): express bracketing as a composition of binary operations

$$\mathfrak{b}_{(k,m,n_i)} = \mathfrak{b}_{(2,m,n_1,n_2+\cdots+n_k)} \circ \cdots \circ \mathfrak{b}_{(2,m,n_{k-1},n_k)}$$

- Extend composition and recursion to k-ary operations
  - k-ary compositions $\mathfrak{c}_{(k,m,n_i)}(h_i) = h_k \circ \cdots \circ h_1$ of functions $h_i : \mathbb{N}^{n_{i-1}} \to \mathbb{N}^{n_i}$, for $i = 1, \ldots, k$, with $n_0 = m$
  - $(k+1)$-ary recursions with k initial conditions:

$$\begin{array}{l} h(x_1,\ldots,x_n,1) = h_1(x_1,\ldots,x_n),\ldots \\ h(x_1,\ldots,x_n,k) = h_k(x_1,\ldots,x_n), \\ h(x_1,\ldots,x_n,k+\ell) = \\ h_{k+1}(x_1,\ldots,x_n,h_1(x_1,\ldots,x_n),\ldots,h_k(x_1,\ldots,x_n),k+\ell-1), \\ \text{for } \ell \geq 1 \end{array}$$

Partial recursive functions and the Hopf algebra

- enlarge from primitive recursive to partial recursive: same elementary operations $\mathfrak{c}, \mathfrak{b}, \mathfrak{r}$ of composition, bracketing and recursion but additional μ operation
- μ operation: input function $f : \mathbb{N}^{n+1} \to \mathbb{N}$, output

$$h : \mathbb{N}^n \to \mathbb{N}, \quad h(x_1, \ldots, x_n) = \min\{x_{n+1} \,|\, f(x_1, \ldots, x_{n+1}) = 1\},$$

with domain $\mathcal{D}(h)$ those $(x_1, \ldots, x_n)$ such that $\exists x_{n+1} \geq 1$

$$f(x_1, \ldots, x_{n+1}) = 1, \text{ with } (x_1, \ldots, x_n, k) \in \mathcal{D}(f), \forall k \leq x_{n+1}$$

- Church's thesis: get all semi-computable functions, for which $\exists$ program computing $f(x)$ for $x \in \mathcal{D}(f)$ and never stops for $x \notin \mathcal{D}(f)$
- Hopf algebra: additional vertex decoration by μ operations, extended to arbitrary valence by combining with bracketing; edge decorations by partial recursive functions

Manin's proposal of possible types of Feynman rules for computation

• $\mathcal{B}$ algebra of functions $\Phi : \mathbb{N}^k \to \mathcal{M}(D)$ from $\mathbb{N}^k$, for some k, to algebra $\mathcal{M}(D)$ of analytic functions in unit disk $D = \{z \in \mathbb{C} \,:\, |z| < 1\}$.

• Rota–Baxter operator T on $\mathcal{B}$ componentwise projection onto polar part at $z = 1$

• For any tree τ that computes f set

$$\Phi_\tau(\underline{k}, z) = \Phi(\underline{k}, f, z) := \sum_{n \geq 0} \frac{z^n}{(1 + n\bar{f}(\underline{k}))^2}$$

$\bar{f} : \mathbb{N}^m \to \mathbb{Z}_{\geq 0}$ computes $f(x)$ at $x \in \mathcal{D}(f)$ and 0 at $x \notin \mathcal{D}(f)$.

• $\Phi_\tau(\underline{k}, z)$ pole at $z = 1$ iff $\underline{k} \notin \mathcal{D}(f)$

• this Φ is algebraic Feynman rule: commutative algebra homomorphism from enlarged Hopf algebra of flow charts to Rota–Baxter algebra $\mathcal{B}$

Birkhoff factorization

- negative part of Birkhoff factorization becomes

$$\Phi_-(\underline{k}, f_\tau, z) = -T(\Phi(\underline{k}, f_\tau, z) + \sum_C \Phi_-(\underline{k}, f_{\pi_C(\tau)}, z)\Phi(\underline{k}, f_{\rho_C(\tau)}, z))$$

- Note: $f = f_\tau$ label of outgoing flag of τ: then $f_{\rho_C(\tau)} = f_\tau$

$$\Phi_-(\underline{k}, f_\tau, z) = -T\left(\Phi(\underline{k}, f_\tau, z)(1 + \Phi_-(\underline{k}, \sum_C f_{\pi_C(\tau)}, z))\right)$$

- What is happening here? Like in QFT, looking not only at "divergences" of program τ but also of *all subprograms* $\pi_C(\tau)$ and $\rho_C(\tau)$ determined by admissible cuts (the problem of subdivergences in renormalization)

Why subdivergences in computation?

- $\Phi_-(\underline{k}, f_\tau, z)$ detects not only if τ has infinities but if any subroutine does
- Note: $\Phi(\underline{k}, f_\tau, z)$ only depends on $f = f_\tau$ not on τ, but $\Phi_-(\underline{k}, f_\tau, z)$ really *depends on* τ
- Unlike QFT there are programs without divergences that do have subdivergences

Note: *Useful viewpoint*: every partial recursive function can be computed by a Hopf-primitive program: Kleene normal form as μ of a total function

- this general idea on Renormalization and Computation remains to be developed... *but* it serves as a useful conceptual intermediate step between the physics of QFT and the syntax-semantics interface model for generative linguistics