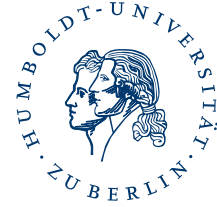


HUMBOLDT-UNIVERSITÄT ZU BERLIN



---

FACULTY OF MATHEMATICS AND NATURAL  
SCIENCES I  
DEPARTMENT OF PHYSICS

---

**Algorithmization of the Hopf algebra of  
Feynman graphs**

MASTER THESIS

A Thesis Submitted in Partial Fulfillment of the Requirements for  
the Degree of

Master of Science (M.Sc.)

in Physics

by

Michael Borinsky

Supervisors: Prof. Dr. Dirk Kreimer  
Dr. Oliver Schnetz

submitted the



# Contents

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Introduction</b>  | <b>3</b>  |
| <b>2</b> | <b>Basic definitions</b>   | <b>5</b>  |
| 2.1      | Graph theoretic concepts . . . . .   | 5         |
| 2.1.1    | Multigraphs . . . . .  | 5         |
| 2.2      | Feynman graphs . . . . .   | 7         |
| 2.2.1    | Definition . . . . .   | 7         |
| 2.2.2    | Properties of Feynman graphs . . . . .   | 10        |
| 2.2.3    | Contractions of subgraphs . . . . .  | 11        |
| 2.2.4    | Residues of graphs . . . . .   | 12        |
| 2.2.5    | Weight $\omega_D$ of a Feynman graph . . . . .                                 | 12        |
| <b>3</b> | <b>The Hopf Algebra of Feynman graphs</b>                                      | <b>15</b> |
| 3.1      | Definition . . . . .   | 15        |
| 3.2      | Properties of $\mathcal{H}_D$ . . . . .  | 18        |
| 3.2.1    | The Hopf algebra $\mathcal{H}_D$ . . . . .                                     | 20        |
| 3.3      | Sum of the coproducts of all 1PI graphs . . . . .                              | 21        |
| 3.3.1    | Numbers of vertices, edges and connected components of certain types . . . . . | 21        |
| 3.3.2    | Permuting external legs . . . . .  | 22        |
| 3.3.3    | Insertions . . . . .   | 22        |
| 3.3.4    | Automorphism group of products of graphs . . . . .                             | 24        |
| 3.3.5    | Sum formula for 1PI graphs . . . . .   | 25        |
| <b>4</b> | <b>Zero-dimensional QFT</b>  | <b>29</b> |
| 4.1      | $\varphi^k$ -theory . . . . .  | 29        |
| 4.1.1    | Generating function . . . . .  | 29        |
| 4.1.2    | Connected graphs . . . . .   | 32        |
| 4.2      | $ \phi ^2 A$ -theory . . . . .   | 34        |

|          |   |           |
|----------|---|-----------|
| <b>5</b> | <b>Diagram generation</b>                                 | <b>37</b> |
| 5.1      | Overview . . . . .  | 37        |
| 5.2      | Sketch of the implementation of <b>feynngen</b> . . . . . | 37        |
| 5.3      | Check of validity . . . . .                               | 38        |
| 5.4      | Manual of <b>feynngen</b> . . . . .                       | 38        |
| 5.4.1    | Overview . . . . .  | 38        |
| 5.4.2    | Options and Parameters . . . . .                          | 39        |
| 5.4.3    | Output of $\varphi^k$ -graphs . . . . .                   | 40        |
| 5.4.4    | Output of QED graphs . . . . .                            | 42        |
| 5.4.5    | Labeled and unlabeled legs . . . . .                      | 44        |
| <b>6</b> | <b>Coproduct computation</b>                              | <b>45</b> |
| 6.1      | Overview . . . . .  | 45        |
| 6.2      | Sketch of the implementation of <b>feyncop</b> . . . . .  | 45        |
| 6.3      | Check of validity . . . . .                               | 46        |
| 6.4      | Manual of <b>feyncop</b> . . . . .                        | 46        |
| 6.4.1    | Overview . . . . .  | 46        |
| 6.4.2    | Option and Parameters . . . . .                           | 47        |
| 6.4.3    | Reference to edges . . . . .                              | 48        |
| 6.4.4    | Output of subgraphs . . . . .                             | 48        |
| 6.4.5    | Output of cographs . . . . .                              | 49        |
| 6.4.6    | Output of tensor products . . . . .                       | 51        |
| 6.4.7    | Usage in conjunction with <b>feynngen</b> . . . . .       | 52        |
| 6.4.8    | Filtering for primitive graphs . . . . .                  | 54        |
| <b>7</b> | <b>Conclusion and future prospects</b>                    | <b>57</b> |

# Chapter 1

## Introduction

Quantum field theory is a mathematical scheme to describe elementary particles and forces. It forms the framework for the standard model of particle physics which gives the most accurate account of matter at the atomic scale. The status quo strategy to perform explicit calculations in quantum field theory, perturbation theory, requires renormalization to yield finite results. Recently renormalization has been given an illuminating formulation by Dirk Kreimer and collaborators. This formulation of renormalization, which emerged from the BPHZ scheme, is based on the notion of the Hopf algebra of Feynman graphs. It provides a clear distinction between analytic and combinatorial aspects of the renormalization procedure. One of the central objects of this algebra is the coproduct which encodes the combinatorial aspects of renormalization as the forest formula does in the BPHZ scheme.

The purpose of this thesis was to develop a program, **feyncop**, which automatizes the calculation of the coproduct of Feynman graphs. Additionally to the study of renormalization in general, this program is aimed to be used as input for parametric integration techniques to evaluate Feynman amplitudes as described for instance in [2].

These upcoming techniques initiated an interest in relatively large loop order Feynman diagrams. Therefore, an additional program, **feynngen**, to generate high loop order diagrams needed to be developed to fulfill this demand.

An introduction to the properties of the Hopf algebra of Feynman graphs preceded by a definition of Feynman graphs and their properties is given in the chapters 2 and 3. These properties were used to derive theorem 1 which yielded an identity suitable to check the coproduct computation of **feyncop**. The combinatorial proof of this theorem constitutes a simplification of the proof given in [13].

A short treatment of zero dimensional quantum field theory, which gives useful formulas to check the Feynman graph generation with **feynngen**, is layed out in

chapter 4.

Therefore, both programs are validated. Details to the Feynman graph generation program **feynngen** are given in chapter 5 and the coproduct computation program, **feyncop** is described in chapter 6.

**feynngen** is capable of generating  $\varphi^k$ -theory and QED Feynman graphs and of filtering these graphs for the properties of connectedness, one-particle-irreducibility, 2-vertex-connectivity and tadpole-freeness. It can handle graphs with fixed external legs as those without fixed external legs. For instance, all 130516 1PI,  $\varphi^4$ , 8-loop diagrams with four external legs can be generated, together with their symmetry factor, within eight hours and all 342430 1PI, QED, vertex residue type, 6-loop diagrams can be generated in three days both on a standard end-user computer.

The output of **feynngen** can be used in conjunction with **feyncop** which can compute the coproduct of a graph with weights assigned to its edges for an arbitrary dimension.

# Chapter 2

## Basic definitions

### 2.1 Graph theoretic concepts

For the graph generation and the calculation of the coproduct, the graph theoretical tool **nauty**, described in [10], is used. Therefore, some definitions of basic graph theoretical terms are required. Here, the focus is on the transition from the treatment of graphs in combinatorics and graph theory to a quantum field theoretic context. On the graph theoretical side, the definitions are based on [1], whereas the quantum field theoretical view point is based on [9] and [5], which can be consulted for a more detailed discussion of the following notions. The central objects of perturbative QFT are Feynman graphs. These are special cases of multigraphs.

#### 2.1.1 Multigraphs

**Definition 1.** *A multiset is a generalization of the notion of a set in which elements are allowed to appear more than once.*

**Definition 2.** *A multigraph  $G$  is a pair  $(V, E)$ , where  $V$  denotes the vertex set of finite cardinality and  $E$  is a multiset, the edge multiset, with elements  $e \in V \times V$ .*

**Adjacency and incidence** For any edge  $e = (v_1, v_2) \in E$  the vertices  $v_1$  and  $v_2$  are called adjacent to each other and incident to the edge  $e$ .

**Self-loop** An edge  $(v, v)$  incident to only one vertex is called a self-loop.

**Valency** The number of incident edges to a vertex  $v \in V$  is called the valency of  $v$ . Self-loops incident to  $v$  are counted twice.

**Simple graph** A multigraph  $G = (V, E)$  is a simple graph if it has no self-loops and every edge is only present once in  $E$ .

**Directed, undirected and mixed graphs** Depending on whether the elements of the edge multiset  $e \in E$  are ordered or unordered pairs of vertices, the multigraph is called directed or undirected. A multigraph may also contain directed and undirected edges.

**Isomorphism** Two graphs  $G = (V, E)$  and  $G' = (V', E')$  are considered as isomorphic,  $G \simeq G'$ , if there is a bijection  $\phi : G \rightarrow G'$  that maps the vertex set  $V$  onto  $V'$  and the edge multiset  $E$  onto  $E'$ , such that adjacency is respected:

$$\phi(e) = (\phi(v_1), \phi(v_2)) \quad \forall e \in E \text{ with } (v_1, v_2) = e.$$

$\phi$  is called an isomorphism between  $G$  and  $G'$ .

**Automorphism group** The set of all isomorphisms of a graph onto itself  $\phi : G \rightarrow G$  is the automorphism group,  $\text{Aut}(G)$ , of  $G$ . The inverse of the cardinality of the automorphism group is called the symmetry factor of the graph.

Handling Feynman graphs, a small pathology arises in connection with self-loops. For the notions of isomorphism and the automorphism group unoriented self-loops are to be considered as edges consisting of two half-edges which can be permuted freely. The overall effect of this additional freedom is that every unoriented self-loop is assigned an additional symmetry generator of the automorphism group of order two. This generator commutes with all other elements of  $\text{Aut}(G)$ .

**Subgraphs** A graph  $G' = (V', E')$  is a subgraph of  $G = (V, E)$  if  $E' \subseteq E$  and  $V' \subseteq V$ . This relation is denoted as  $G' \subseteq G$ .

**Connectedness** A multigraph  $G$  is disconnected if the vertex set and the edge multiset can be split into disjoint sets and multisets  $V = V_1 \cup V_2$  and  $E = E_1 \cup E_2$ , such that  $g_1 = (V_1, E_1)$  and  $g_2 = (V_2, E_2)$  are again multigraphs in the sense of Definition 2. Implicitly, this statement is also expressed as  $G$  being a disjoint union of the graphs  $g_1$  and  $g_2$ :  $G = g_1 \cup g_2$ .

Applying this procedure successively, eventually yields a set of multigraphs,  $\{G_1, \dots, G_n\}$ , which are connected. These are subgraphs of  $G = \bigcup_i G_i$  and are called the connected components of  $G$ .



**Loop number or first Betti number** The loop number or first Betti number  $h_1(G)$  is the number of independent loops or cycles of  $G$ .

The identity

$$h_1(G) = |E| - |V| + f_G \quad (2.1)$$

is valid for every graph, where  $|E|$  and  $|V|$  are the appropriate cardinalities of the vertex sets and edge multisets of  $G$  and  $f_G$  denotes the number of connected components of  $G$ .

*Proof.* (Sketch) Obviously, the formula holds for the empty graph  $G = \emptyset$  without vertices or edges.

For an arbitrary graph  $G$ , adding a vertex will increase  $|V|$  and the number of connected components  $f_G$  by one. Therefore, the formula holds for all graphs without edges.

Adding an edge, will either reduce the number of connected components  $f_G$  by one or it will add a loop to the graph. So, identity (2.1) stays valid after this operation. Every graph can be constructed by subsequently adding vertices or edges. Therefore equation (2.1) is valid for all graphs.  $\square$

## 2.2 Feynman graphs

To capture the structure of a Feynman graph in general, more information than the one of a multigraph is necessary. Viewed from a graph theoretical view point, Feynman graphs are edge-colored multigraphs. But for the treatment of the Hopf algebra of Feynman graphs, a Taylor-made definition in resemblance to [5] is more convenient.

### 2.2.1 Definition

Let  $G = (V, E)$  be a multigraph with the vertex set and edge multiset being disjoint unions of the sets and multisets  $\Gamma_{\text{int}}^{[0]}$ ,  $\Gamma_{\text{ext}}^{[0]}$ ,  $\Gamma_{\text{int}}^{[1]}$  and  $\Gamma_{\text{ext}}^{[1]}$ , such that

$$V = \Gamma_{\text{int}}^{[0]} \cup \Gamma_{\text{ext}}^{[0]}$$

and

$$E = \Gamma_{\text{int}}^{[1]} \cup \Gamma_{\text{ext}}^{[1]}.$$

The vertices in  $\Gamma_{\text{ext}}^{[0]}$  shall have valency 1 and those in  $\Gamma_{\text{int}}^{[0]}$  valency  $\geq 3$ . The edges in  $\Gamma_{\text{int}}^{[1]}$  must be incident only to vertices  $v \in \Gamma_{\text{int}}^{[0]}$  and those in  $\Gamma_{\text{ext}}^{[1]}$  are incident to at least one vertex  $v \in \Gamma_{\text{ext}}^{[0]}$ .

The vertices in  $\Gamma_{\text{int}}^{[0]}$  are called internal vertices and those in  $\Gamma_{\text{ext}}^{[0]}$  are called external or source vertices. The edges in  $\Gamma_{\text{int}}^{[1]}$  are called internal edges and the ones in  $\Gamma_{\text{ext}}^{[1]}$  are called external edges or legs.

**Definition 3.** A Feynman graph  $\Gamma = (G, \text{res})$  is a pair of a multigraph  $G$  with the above properties and a coloring  $\text{res}$ ,

$$\text{res} : \Gamma^{[1]} \rightarrow \mathcal{R}_E, \quad (2.2)$$

which assigns a color or type from a set of allowed edge types  $\mathcal{R}_E$  to every edge in  $G$ .

The map  $\text{res}$  can be extended to the internal vertices of  $\Gamma$ ,

$$\text{res} : \Gamma_{\text{int}}^{[0]} \cup \Gamma^{[1]} \rightarrow \mathcal{R}_V \cup \mathcal{R}_E, \quad (2.3)$$

by assigning a vertex type  $r_V \in \mathcal{R}_V$  to every internal vertex  $v \in \Gamma_{\text{int}}^{[0]}$ , such that the vertex types are determined uniquely by the edges incident to  $v$ . The elements  $r \in \mathcal{R}_V \cup \mathcal{R}_E$  are also called the allowed residue types of the theory under inspection.

Generally, the Feynman rules restrict the sets  $\mathcal{R}_V$  and  $\mathcal{R}_E$ , such that only Feynman graphs with certain vertex and edge types are allowed. For instance,  $\varphi^4$  theory has one vertex type,  $\mathcal{R}_V^{\varphi^4} = \{\text{X}\}$ , and one edge type,  $\mathcal{R}_E^{\varphi^4} = \{—\}$ , whereas quantum electro dynamics (QED) allows one vertex type,  $\mathcal{R}_V^{\text{QED}} = \{\text{~}\}$ , and two edge types,  $\mathcal{R}_E^{\text{QED}} = \{\text{→}, \text{~}\}$ .

This definition differs slightly from the one in [9] and [5], because of the additional external vertices in  $\Gamma_{\text{ext}}^{[0]}$ . These external or source vertices are added for simplicity of the representation of graphs as edge lists, for the determination of the isomorphism class of a graph and for the transition from multigraphs to simple graphs needed as input for the **nauty** package.

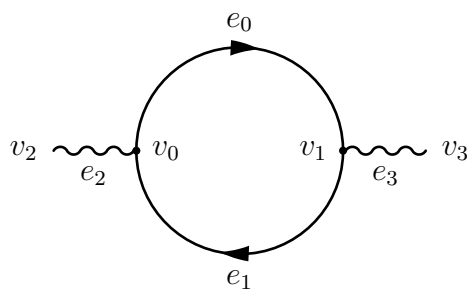
In the following, Feynman graphs will be referred to as graphs or diagrams if the distinction from other types of graphs is clear from the context.

Examples of Feynman graphs of QED and  $\varphi^4$ -theory are depicted in figure 2.1.

### Feynman graphs with fixed source vertices

Usually, handling green's functions in quantum field theory, the external edges and the source vertices of Feynman graphs are considered as fixed. That means, there is an additional bijective map,

$$\delta : \Gamma_{\text{ext}}^{[0]} \rightarrow \{1, \dots, |\Gamma_{\text{ext}}^{[0]}|\},$$

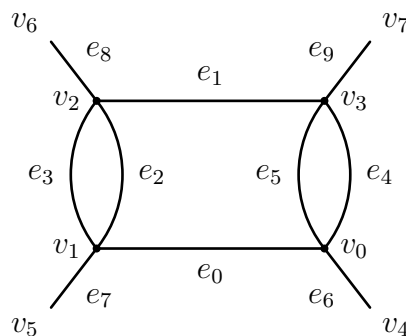


$$\begin{aligned} \Gamma_{\text{int}}^{[0]} &= \{v_0, v_1\} & \Gamma_{\text{ext}}^{[0]} &= \{v_2, v_3\} \\ \Gamma_{\text{int}}^{[1]} &= \{e_0, e_1\} & \Gamma_{\text{ext}}^{[1]} &= \{e_2, e_3\} \end{aligned}$$

$$\begin{aligned} e_0 &= (v_0, v_1) & e_1 &= (v_1, v_0) \\ e_2 &= (v_0, v_2) & e_3 &= (v_1, v_3) \end{aligned}$$

$$\begin{aligned} \text{res}(e_0) &= \text{res}(e_1) = \rightarrow \\ \text{res}(e_2) &= \text{res}(e_3) = \sim \\ \text{res}(v_0) &= \text{res}(v_1) = \text{fermion vertex symbol} \end{aligned}$$

(a) A QED Feynman graph



$$\begin{aligned} \Gamma_{\text{int}}^{[0]} &= \{v_0, v_1, v_2, v_3\} \\ \Gamma_{\text{ext}}^{[0]} &= \{v_4, v_5, v_6, v_7\} \\ \Gamma_{\text{int}}^{[1]} &= \{e_0, e_1, e_2, e_3, e_4, e_5\} \\ \Gamma_{\text{ext}}^{[1]} &= \{e_6, e_7, e_8, e_9\} \end{aligned}$$

$$\begin{aligned} e_0 &= (v_0, v_1) & e_1 &= (v_2, v_3) \\ e_2 &= e_3 = (v_1, v_2) & e_4 &= e_5 = (v_0, v_3) \\ e_6 &= (v_0, v_4) & e_7 &= (v_1, v_5) \\ e_8 &= (v_2, v_6) & e_9 &= (v_3, v_7) \end{aligned}$$

$$\begin{aligned} \text{res}(e) &= \text{---} & \forall e \in \Gamma_{\text{int}}^{[1]} \cup \Gamma_{\text{ext}}^{[1]} \\ \text{res}(v) &= \text{X} & \forall v \in \Gamma_{\text{int}}^{[0]} \end{aligned}$$

(b) A  $\varphi^4$ -theory Feynman graph

Figure 2.1: Examples of Feynman graphs after definition 3.

associated to a graph  $\Gamma$ , giving a unique numbering of the external vertices. Note, that this map also induces a unique numbering on the external edges, because every external vertex is incident to at least one external edge.

In the scope of this thesis, a graph with fixed external edges and vertices is referred to as leg-fixed graph.

### Isomorphy

Although, the concepts of adjacency, incidence and connectedness apply transparently to Feynman graphs using the appropriate properties of the underlying multigraph, care must be taken considering isomorphy of Feynman graphs. Two Feynman graphs  $\Gamma = (G, \text{res})$  and  $\Gamma' = (G', \text{res}')$  are isomorphic if there is an isomorphism  $\phi$  between the two underlying multigraphs  $\phi : G \rightarrow G'$ , such that  $\phi$  preserves the edge and vertex types:  $\text{res} \circ \phi = \text{res}'$ .

For leg-fixed graphs, the isomorphism  $\phi$  also needs to preserve the numbering of the legs:  $\delta \circ \phi = \delta'$ .

Figure 2.2 depicts some examples of Feynman graphs with the corresponding orders of the automorphism groups for the leg-fixed (lf) and the not leg-fixed (nlf) case.

### Subgraphs of Feynman graphs

A Feynman graph  $\gamma$  is a subgraph of a graph  $\Gamma$ ,  $\gamma \subseteq \Gamma$ , if  $\gamma_{\text{int}}^{[1]} \subseteq \Gamma_{\text{int}}^{[1]}$ ,  $\gamma_{\text{int}}^{[0]} \subseteq \Gamma_{\text{int}}^{[0]}$  and every vertex  $v \in \gamma_{\text{int}}^{[0]} \subseteq \Gamma_{\text{int}}^{[0]}$  has the same vertex type in  $\gamma$  as in  $\Gamma$ :  $\text{res}_{\gamma}(v) = \text{res}_{\Gamma}(v)$ . That means, possible deficiencies of incident edges in a subgraph are fixed by adding additional external legs. Note that a subgraph is given uniquely by its internal edges and vertices. The external edges and vertices can be reconstructed from missing edges incident to a vertex to fulfill the vertex type requirement.

A canonical ordering of the external legs can not be defined easily. Therefore subgraphs are considered as non-leg-fixed graphs in the scope of this thesis.

## 2.2.2 Properties of Feynman graphs

**1PI - one particle irreducibility** A graph, which is still connected if any internal edge is removed, is called a 1PI graph.

Figure 2.2 (b), 2.2 (c) and 2.2 (d) show examples of 1PI graphs. In figure 2.2 (a) a disconnected graph is depicted, which is thereby not 1PI.

**2-vertex-connectivity** A graph, which has no self-loops and is still connected if any vertex  $v$  is removed together with the edges incident to  $v$ , is called 2-vertex-connected.

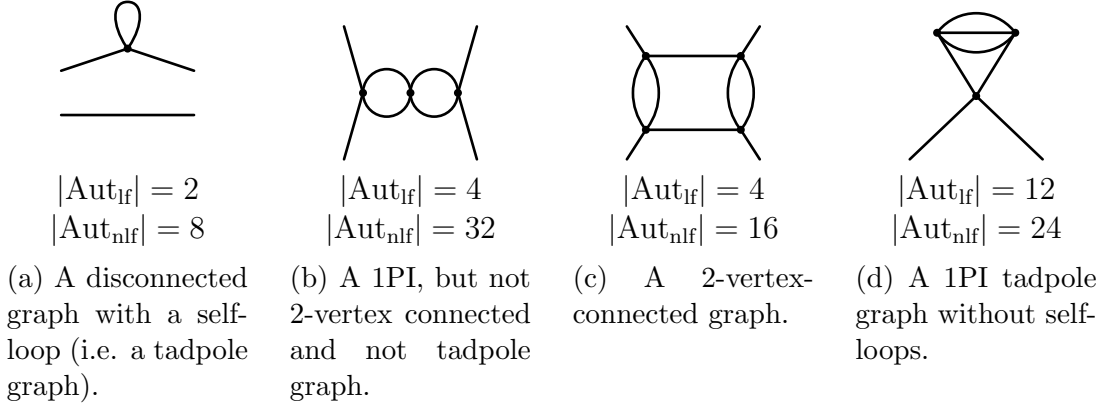


Figure 2.2: Examples of Feynman graphs, which fulfill certain properties, with the orders of their automorphism groups.

In figure 2.2 (c) a 2-vertex-connected graph is shown. The other graphs in figure 2.2 are not 2-vertex-connected.

**Tadpole graphs** A not 2-vertex-connected graph  $\Gamma$ , which either has self-loops or can be split into connected components upon removal of any vertex, such that one of the new connected components is not connected to an external vertex, is called a tadpole graph.

For graphs without external legs, this notion coincides with 2-vertex connectivity.

Figure 2.2 (a) depicts a disconnected tadpole graph and figure 2.2 (d) shows an example of a tadpole graph without self-loops.

### 2.2.3 Contractions of subgraphs

An important notion in connection to the Hopf algebra of Feynman graphs is the contraction of subgraphs. If  $\gamma \subseteq \Gamma$ , the contraction,  $\Gamma/\gamma$ , of  $\gamma$  in  $\Gamma$  is obtained by removing all edges  $e \in \gamma_{\text{int}}^{[1]} \subseteq \Gamma_{\text{int}}^{[1]}$  of  $\Gamma$  and by merging all vertices connected to these edges to one new vertex. If the new vertex is of valency 2, it is also removed and the two edges incident to it are joined to one edge.

Contractions are commutative operations if disjoint subgraphs are contracted. For a graph  $\Gamma$  with  $\gamma_1, \gamma_2 \subseteq \Gamma$  and  $\gamma_1 \cap \gamma_2 = \emptyset$ :

$$(\Gamma/\gamma_1)/\gamma_2 = (\Gamma/\gamma_2)/\gamma_1 = \Gamma/(\gamma_1 \cup \gamma_2). \quad (2.4)$$

Furthermore, contractions can be “canceled”, for instance for  $\delta \subseteq \gamma \subseteq \Gamma$

$$(\Gamma/\delta)/(\gamma/\delta) = \Gamma/\gamma. \quad (2.5)$$



This weight corresponds to the negative power of the momenta in the associated Feynman rule.

In QED for example, a weight of 2 is assigned to photon edges,  $\omega(\sim) = 2$ , and 1 is assigned to fermion edges,  $\omega(\rightarrow) = 1$ . Because QED vertices do not depend on any momenta, their weight is 0,  $\omega(\text{vertex}) = 0$ .

In  $\varphi^k$ -theory all edges are assigned the weight 2,  $\omega(-) = 2$  and the vertices have weight 0.

**The map  $\omega_D$  on Feynman graphs**

Using the map  $\omega$  for given Feynman rules to assign a weight to every vertex and edge type, an additional map  $\omega_D$  can be defined, giving a weight to a Feynman graph:

**Definition 4.**

$$\omega_D(\Gamma) := \sum_{v \in \Gamma_{int}^{[0]}} \omega(\text{res}(v)) + \sum_{e \in \Gamma_{int}^{[1]}} \omega(\text{res}(e)) - Dh_1(\Gamma) \tag{2.7}$$

where  $h_1(\Gamma)$  denotes the loop number of  $\Gamma$  and  $D$  is a parameter that will be associated with the spacetime dimension. Neglecting possible infrared divergencies, the value of  $\omega_D$  coincides with the degree of divergence of the integral associated to the graph in a  $D$ -dimensional quantum field theory. To the empty graph the weight 0 is assigned:  $\omega_D(\emptyset) = 0$ . A 1PI graph  $\Gamma$  with  $\omega_D(\Gamma) \leq 0$  is called superficially divergent in  $D$  dimensions.

**Examples**

$$\begin{aligned} \omega_4(\text{photon loop}) &= \omega_4(\text{fermion loop}) = \omega_6(\text{photon tadpole}) = -2 \\ \omega_4(\text{photon triangle}) &= \omega_4(\text{fermion box}) = \omega_4(\text{photon box}) = \omega_6(\text{photon tadpole}) = 0 \\ \omega_4(\text{fermion box}) &= \omega_4(\text{photon tadpole}) = \omega_6(\text{photon box}) = 2 \end{aligned}$$





# Chapter 3

## The Hopf Algebra of Feynman graphs

### 3.1 Definition

Let  $\mathcal{T}$  be the set of all non-isomorphic 1PI Feynman graphs of a given quantum field theory, including the empty graph.  $\mathcal{F}$  is the free commutative monoid generated by the elements in  $\mathcal{T}$ . The empty graph is associated with the neutral element  $\mathbb{I} \in \mathcal{F}$ .

Following [6],  $\mathcal{H}_D$  is the vector space spanned by the elements of  $\mathcal{F}$  with the multiplication  $m$  on  $\mathcal{F}$  extended to linear combinations of products of graphs.

Additionally,  $\mathcal{H}_D$  is equipped with a linear map called the coproduct. For 1PI graphs  $\Gamma \in \mathcal{T} \subset \mathcal{H}_D$  it is defined as

**Definition 5.**

$$\Delta_D \Gamma := \sum_{\gamma \trianglelefteq \Gamma} \gamma \otimes \Gamma/\gamma \quad : \quad \mathcal{T} \rightarrow \mathcal{H}_D \otimes \mathcal{H}_D \quad (3.1)$$

where

$$\gamma \trianglelefteq \Gamma \Leftrightarrow \gamma \in \left\{ \delta \subseteq \Gamma \mid \delta = \bigcup_i \delta_i, \text{ such that } \delta_i \in \mathcal{T} \text{ and } \omega_D(\delta_i) \leq 0 \right\} \quad (3.2)$$

denotes the membership of  $\gamma$  in the set of subgraphs of  $\Gamma$ , whose connected components are superficially divergent 1PI graphs. Disconnected graphs  $\gamma = \bigcup_i \gamma_i$  are

identified with the product  $\left( \prod_i \gamma_i \right) \in \mathcal{F} \subset \mathcal{H}_D$ . The cograph  $\Gamma/\Gamma$  and the empty graph  $\gamma = \emptyset$  in (3.1) are identified with  $\mathbb{I} \in \mathcal{H}_D$ .

Note, that while  $\Gamma/\gamma$ , the so-called cographs, in (3.1) can inherit a numbering of the external edges from  $\Gamma$ , it is not possible to assign a numbering to the external legs of the subgraphs  $\gamma$ . That means, if the corresponding spaces of leg fixed and non-leg fixed 1PI graphs,  $\mathcal{T}^{\text{leg-fixed}}$ ,  $\mathcal{T}^{\text{non-leg-fixed}}$ ,  $\mathcal{H}_D^{\text{leg-fixed}}$ ,  $\mathcal{H}_D^{\text{non-leg-fixed}}$  are distinguished,  $\Delta_D$  maps the spaces as follows:

$$\begin{aligned} \Delta_D & : \mathcal{T}^{\text{non-leg-fixed}} & \rightarrow & \mathcal{H}_D^{\text{non-leg-fixed}} & \otimes & \mathcal{H}_D^{\text{non-leg-fixed}} \\ \Delta_D & : \mathcal{T}^{\text{leg-fixed}} & \rightarrow & \mathcal{H}_D^{\text{non-leg-fixed}} & \otimes & \mathcal{H}_D^{\text{leg-fixed}}. \end{aligned}$$

Strictly speaking, only a coaction not a coproduct is obtained in the leg-fixed case. A coproduct can be obtained by defining a map  $\mathcal{T}^{\text{non-leg-fixed}} \rightarrow \mathcal{T}^{\text{leg-fixed}}$  which maps a non-leg-fixed graph to the sum of all leg-fixed graphs which correspond to the non-leg-fixed graph if the ordering of the external edges is ignored. Therefore,  $\Delta_D$  can still be promoted to a coproduct in the leg-fixed case.

Because here, the subgraphs themselves and not the algebra elements associated with them are of most interest, this ambiguity will not play a major role in this thesis. For most of the properties of  $\mathcal{H}_D$ , it is not relevant whether the set of leg-fixed or the one of non-leg-fixed 1PI graphs is taken as generators. An exception is section 3.3, which will only cope with non-leg-fixed graphs for simplicity.

The map  $\Delta_D$  can be extended recursively to products of graphs,  $\Delta_D : \mathcal{F} \rightarrow \mathcal{H}_D \otimes \mathcal{H}_D$ , by defining for  $(\Gamma_1\Gamma_2) \in \mathcal{F}$  with  $\Gamma_1 \in \mathcal{T}$  and  $\Gamma_2 \in \mathcal{F}$  as

$$\Delta_D (\Gamma_1\Gamma_2) := (\Delta_D\Gamma_1) (\Delta_D\Gamma_2). \quad (3.3)$$

In accordance to this,  $\Delta_D\mathbb{I} := \mathbb{I} \otimes \mathbb{I}$  is set. Eventually,  $\Delta_D$  can be extended linearly to all elements  $h \in \mathcal{H}_D$ , equipping  $\mathcal{H}_D$  with a coproduct  $\Delta_D : \mathcal{H}_D \rightarrow \mathcal{H}_D \otimes \mathcal{H}_D$ .

Additionally, the reduced coproduct  $\tilde{\Delta}_D$  is defined as

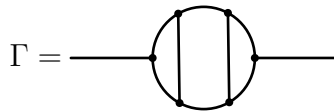
$$\tilde{\Delta}_D := \Delta_D - \text{id} \otimes \mathbb{I} - \mathbb{I} \otimes \text{id} \quad : \quad \mathcal{H}_D \rightarrow \mathcal{H}_D \otimes \mathcal{H}_D, \quad (3.4)$$

giving rise to the space of primitive elements of  $\mathcal{H}_D$ :

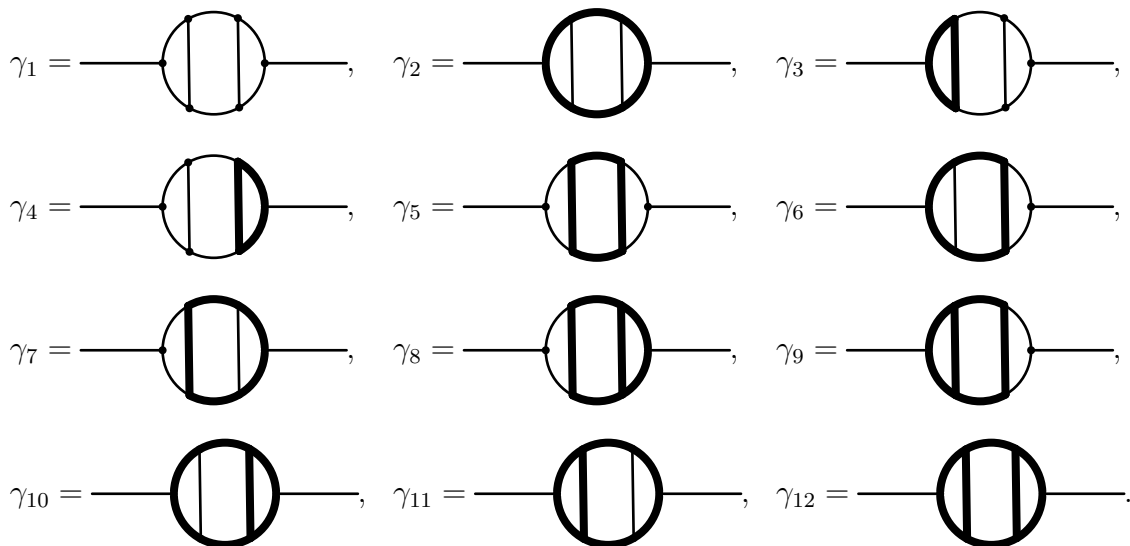
$$\text{Prim}(\mathcal{H}_D) := \ker \tilde{\Delta}. \quad (3.5)$$

### Example

Suppose the coproduct in 6 dimensions of the  $\varphi^3$ -theory graph



shall be calculated. The graph has the following 1PI subgraphs, represented as thick lines in  $\Gamma$ :



Of these subgraphs only  $\gamma_1$ ,  $\gamma_3$ ,  $\gamma_4$ ,  $\gamma_8$ ,  $\gamma_9$  and  $\gamma_{12}$  are superficially divergent in 6 dimensions, as can be checked by applying  $\omega_6$  on them.

$\gamma_3$  and  $\gamma_4$  are disjoint. Therefore, their disjoint union must be included in the set of subgraphs composed of superficially divergent 1PI graphs:

$$\left\{ \delta \subseteq \Gamma \mid \delta = \bigcup_i \delta_i, \text{ such that } \delta_i \in \mathcal{T} \text{ and } \omega_D(\delta_i) \leq 0 \right\} = \{ \gamma_1, \gamma_3, \gamma_4, \gamma_8, \gamma_9, \gamma_3 \cup \gamma_4, \gamma_{12} \}.$$

The relevant cographs can be calculated by contracting the corresponding subgraphs,

$$\Gamma/\gamma_3 = \Gamma/\gamma_4 = \text{---} \bigcirc \text{---}$$

$$\Gamma/\gamma_8 = \Gamma/\gamma_9 = \Gamma/(\gamma_3 \cup \gamma_4) = \text{---} \bigcirc \text{---},$$

and subgraphs can be decorated with external legs promoting them to elements

of  $\mathcal{F}$ :

$$\begin{aligned} \gamma_3 \text{ and } \gamma_4 &\rightarrow \text{---} \langle \text{---} \rangle \in \mathcal{F} & \gamma_8 \text{ and } \gamma_9 &\rightarrow \text{---} \langle \text{---} \rangle \in \mathcal{F} \\ \gamma_3 \cup \gamma_4 &\rightarrow \left( \text{---} \langle \text{---} \rangle \right)^2 \in \mathcal{F} \end{aligned}$$

Note, that  $\gamma_1 = \emptyset$  and  $\Gamma/\gamma_{12} = \Gamma/\Gamma$  are identified with the unit  $\mathbb{I} \in \mathcal{F}$ .

Therefore, the coproduct can be given:

$$\begin{aligned} \Delta_D \left( \text{---} \langle \text{---} \rangle \text{---} \right) &= \sum_{\gamma \trianglelefteq \Gamma} \gamma \otimes \Gamma/\gamma = \sum_{\gamma \in \{\gamma_1, \gamma_3, \gamma_4, \gamma_8, \gamma_9, \gamma_3 \cup \gamma_4, \gamma_{12}\}} \gamma \otimes \Gamma/\gamma = \\ &= \mathbb{I} \otimes \text{---} \langle \text{---} \rangle \text{---} + \text{---} \langle \text{---} \rangle \otimes \mathbb{I} + \\ &+ 2 \langle \text{---} \rangle \otimes \text{---} \langle \text{---} \rangle \text{---} + 2 \langle \text{---} \rangle \otimes \text{---} \langle \text{---} \rangle \text{---} + \\ &+ \left( \langle \text{---} \rangle \right)^2 \otimes \text{---} \langle \text{---} \rangle \text{---}. \end{aligned}$$

## 3.2 Properties of $\mathcal{H}_D$

**Proposition 1.**  $\Delta_D$  is a coassociative map,

$$(\Delta_D \otimes \text{id}) \Delta_D = (\text{id} \otimes \Delta_D) \Delta_D, \quad (3.6)$$

and thereby  $\mathcal{H}_D$  is a bialgebra.

*Proof.* It is sufficient to prove the coassociativity for  $\Delta_D$  acting on a graph  $\Gamma \in \mathcal{T}$ . Applying the coproduct twice on  $\Gamma$  yields

$$(\text{id} \otimes \Delta_D) \Delta_D \Gamma = \sum_{\gamma \trianglelefteq \Gamma} \gamma \otimes \left[ \sum_{\delta \trianglelefteq \Gamma/\gamma} \delta \otimes (\Gamma/\gamma)/\delta \right],$$

where  $\delta$  can be substituted by  $\delta'/\gamma$ ,

$$= \sum_{\gamma \trianglelefteq \Gamma} \gamma \otimes \sum_{\gamma \trianglelefteq \delta' \trianglelefteq \Gamma} \delta'/\gamma \otimes (\Gamma/\gamma)/(\delta'/\gamma),$$

and  $(\Gamma/\gamma)/(\delta'/\gamma)$  reduces to  $\Gamma/\delta'$ :

$$= \sum_{\gamma \trianglelefteq \Gamma} \gamma \otimes \sum_{\gamma \trianglelefteq \delta' \trianglelefteq \Gamma} \delta'/\gamma \otimes \Gamma/\delta'.$$

Changing the order of summation and using transitivity of the relation  $\trianglelefteq$ ,

$$= \sum_{\delta' \trianglelefteq \Gamma} \left[ \sum_{\gamma \trianglelefteq \delta'} \gamma \otimes \delta'/\gamma \right] \otimes \Gamma/\delta',$$

establishes:

$$= (\Delta_D \otimes \text{id}) \Delta_D \Gamma.$$

□

**Proposition 2.**  $\mathcal{H}_D$  is a bialgebra graded by the loop number. That means,  $m$  and  $\Delta_D$  are maps between the relevant spaces:

$$\mathcal{H}_D = \bigoplus_{L \geq 0} \mathcal{H}_D^{(L)} \quad \text{and} \quad (3.7)$$

$$m : \mathcal{H}_D^{(L_1)} \otimes \mathcal{H}_D^{(L_2)} \rightarrow \mathcal{H}_D^{(L_1+L_2)} \quad (3.8)$$

$$\Delta_D : \mathcal{H}_D^{(L)} \rightarrow \bigoplus_{\substack{L_1, L_2 \geq 0 \\ L_1+L_2=L}} \mathcal{H}_D^{(L_1)} \otimes \mathcal{H}_D^{(L_2)}. \quad (3.9)$$

*Proof.* Graphs with different loop numbers are non-isomorphic. Therefore, (3.7) is trivial. (3.8) follows from the association of products with disjoint union of graphs, for which the loop number is additive:

$$h_1\left(\prod_i \Gamma_i\right) = \sum_i h_1(\Gamma_i) \quad \forall \Gamma_i \in \mathcal{T}$$

It is enough to proof the compatibility of  $\Delta_D$  with the grading for graphs  $\Gamma \in \mathcal{T}$ . Suppose  $\gamma$  is a 1PI subgraph  $\gamma \subseteq \Gamma$ . Because  $\gamma$  is a 1PI graph, it can assumed that  $|\gamma_{\text{ext}}^{[1]}| = |\gamma_{\text{ext}}^{[0]}|$ .

If  $\gamma$  is of edge residue type, the cograph  $\Gamma/\gamma$  fulfills

$$\begin{aligned} |(\Gamma/\gamma)^{[1]}| &= |\Gamma^{[1]}| - |\gamma_{\text{int}}^{[1]}| - 1 \\ |(\Gamma/\gamma)^{[0]}| &= |\Gamma^{[0]}| - |\gamma_{\text{int}}^{[0]}|, \end{aligned}$$

because all internal edges and vertices of  $\gamma$  are removed from  $\Gamma$  and one pair of edges is joined to a single one. Subtracting the second from the first equation and using  $h_1(G) = |E| - |V| + f_G$  with  $f_\Gamma = f_\gamma = f_{\Gamma/\gamma} = 1$ , yields

$$\begin{aligned} h_1(\Gamma/\gamma) - 1 &= h_1(\Gamma) - 1 - (h_1(\gamma) - 1) - 1 \\ &\Rightarrow h_1(\Gamma/\gamma) = h_1(\Gamma) - h_1(\gamma), \end{aligned}$$

which is the desired result.

If the 1PI  $\gamma \subseteq \Gamma$  is of vertex residue type, all internal edges of  $\gamma$  are removed from  $\Gamma$ , but one vertex will be left after the contraction. Therefore,

$$\begin{aligned} |(\Gamma/\gamma)^{[1]}| &= |\Gamma^{[1]}| - |\gamma_{\text{int}}^{[1]}| \\ |(\Gamma/\gamma)^{[0]}| &= |\Gamma^{[0]}| - |\gamma_{\text{int}}^{[0]}| + 1, \end{aligned}$$

which by subtraction of the second from the first equation yields equation (3.10) again.

This generalizes to disconnected graphs  $\gamma = \bigcup \gamma_i \subseteq \Gamma$  by subsequent computation of the contraction, because  $\Gamma/(\gamma_1 \cup \gamma_2) = (\Gamma/\gamma_1)/\gamma_2$ :

$$\begin{aligned} h_1(\Gamma/\gamma_1) &= h_1(\Gamma) - h_1(\gamma_1) \\ h_1((\Gamma/\gamma_1)/\gamma_2) &= h_1(\Gamma/\gamma_1) - h_1(\gamma_2) = h_1(\Gamma) - h_1(\gamma_1) - h_1(\gamma_2) \\ &\dots \\ \Rightarrow h_1\left(\Gamma/\left(\bigcup_i \gamma_i\right)\right) &= h_1(\Gamma) - \sum_i h_1(\gamma_i) \end{aligned}$$

□

### 3.2.1 The Hopf algebra $\mathcal{H}_D$

With the additional algebra morphisms  $\epsilon$ , the counit, and  $S$  the antipode,

$$\epsilon(\mathbb{I}) := 1 \tag{3.10}$$

$$\epsilon(\Gamma) := 0 \quad \forall \Gamma \neq \emptyset, \Gamma \in \mathcal{T} \tag{3.11}$$

$$S(X) := -X - m(\text{id} \otimes S)\tilde{\Delta}_D(X) \quad \forall X \in \mathcal{H}_D, \tag{3.12}$$

$\mathcal{H}_D$  becomes a Hopf algebra.

### 3.3 Sum of the coproducts of all 1PI graphs

Having established the basic properties of the Hopf algebra of Feynman graphs, an additional identity, suitable to test the functionality of **feyncomp**, will be derived in this section. To do so, some further properties of Feynman graphs must be introduced. For simplicity, this section will only take graphs without fixed external legs into account.

#### 3.3.1 Numbers of vertices, edges and connected components of certain types

For every residue type  $r \in \mathcal{R}_V \cup \mathcal{R}_E$  and graph  $\Gamma \in \mathcal{T}$ ,

$$m_r(\Gamma) := \left| \left\{ t \in \Gamma_{\text{int}}^{[0]} \cup \Gamma_{\text{int}}^{[1]} \mid \text{res}(t) = r \right\} \right| \quad (3.13)$$

gives the number of internal vertices or edges in  $\Gamma$  of type  $r$ .

Suppose,  $\gamma$  is a product of graphs,  $\gamma = \left( \prod_i \gamma_i \right) \in \mathcal{F}$ , then

$$n_r(\gamma) := |\{\gamma_i \in \mathcal{T} \text{ such that } \text{res}(\gamma_i) = r\}| \quad (3.14)$$

denotes the number of factors of  $\gamma$  with residue  $r$ .

Let  $\mathcal{R}'_E$  be the set of edge residue types taking different orientations into account. That means that for each oriented edge type in  $r_E \in \mathcal{R}_E$  there are two edge types in  $\mathcal{R}'_E$  each corresponding to one of the possible orientations. Edge types without orientation appear once in  $\mathcal{R}'_E$  as in  $\mathcal{R}_E$ . For QED for instance this means  $\mathcal{R}'_E^{\text{QED}} = \{\rightarrow, \leftarrow, \sim\}$ .

Given some edge type  $r_E \in \mathcal{R}'_E$  and a vertex type  $r_V \in \mathcal{R}_V$ ,

$$N_{r_E}(r_V) \quad (3.15)$$

is the number of edges of type  $r_E$  incident to vertices of type  $r_V$  in the given orientation. For QED the values are,

$$N_{\rightarrow}(\curvearrowright) = N_{\leftarrow}(\curvearrowleft) = N_{\sim}(\curvearrowright) = 1. \quad (3.16)$$

On the other hand for  $\varphi^3$ -theory, the only relevant value is

$$N_{-}(\curvearrowleft) = 3. \quad (3.17)$$

### 3.3.2 Permuting external legs

Let  $\text{Perm}_{\text{ext}}(\Gamma)$  for  $\Gamma \in \mathcal{T}$  be the group of permutations of external vertices of  $\Gamma$ , preserving the external vertex types.

By permuting the external vertices of  $\Gamma$  new graphs can be obtained. In the case of non-leg-fixed graphs, these new graphs will be isomorphic to the original one.

For  $\Gamma \in \mathcal{T}$  the number of different permutations is given as

$$\prod_{r_E \in \mathcal{R}'_E} N_{r_E}(\text{res}(\Gamma))!. \quad (3.18)$$

For a product of graphs  $\gamma = \left( \prod_i \gamma_i \right) \in \mathcal{F}$  with some numbering of the factors chosen,  $\text{Perm}_{\text{ext}}(\gamma)$  is defined as the product group:

$$\text{Perm}_{\text{ext}}(\gamma) = \text{Perm}_{\text{ext}}(\gamma_1) \times \text{Perm}_{\text{ext}}(\gamma_2) \times \dots \quad (3.19)$$

The cardinality of this group is

$$|\text{Perm}_{\text{ext}}(\gamma)| = \prod_{r \in \mathcal{R}_V \cup \mathcal{R}_E} \left( \prod_{r_E \in \mathcal{R}'_E} N_{r_E}(r_V)! \right)^{n_r(\gamma)}. \quad (3.20)$$

### 3.3.3 Insertions

Inserting a graph into another can be interpreted as an inverse to the operation of contracting a subgraph. Suppose  $\gamma, \Gamma \in \mathcal{T}$  and  $\gamma$  should be inserted into  $\Gamma$ . If  $\gamma$  has an edge type residue, it can be glued in an internal edge  $e \in \Gamma_{\text{int}}^{[1]}$  with  $\text{res}(e) = \text{res}(\gamma)$  and if it has a vertex type residue, it can be inserted to replace an internal vertex  $v \in \Gamma_{\text{int}}^{[0]}$  with  $\text{res}(v) = \text{res}(\gamma)$ .

This can be extended to products of graphs  $\gamma = \left( \prod_i \gamma_i \right) \in \mathcal{F}$  being inserted into  $\Gamma \in \mathcal{T}$ , by inserting all the graphs  $\gamma_i$  subsequently into different vertices or edges of  $\Gamma$ . Note that it is possible to insert multiple graphs of edge residue type into the same edge of  $\Gamma$ , whereas it is not possible to insert more than one of vertex residue type into the same vertex of  $\Gamma$ .

The set of insertions of  $\gamma \in \mathcal{F}$  into  $\Gamma \in \mathcal{T}$ , yielding possibly isomorphic graphs, is denoted by  $\mathcal{I}(\Gamma|\gamma)$ .

The graph obtained, when  $\gamma$  is inserted into  $\Gamma$  using an insertion  $i \in \mathcal{I}(\Gamma|\gamma)$ , is denoted by  $\Gamma \circ_i \gamma \in \mathcal{T}$ .



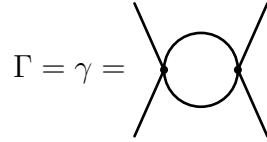
The cardinality of  $\mathcal{I}(\Gamma|\gamma)$  is,

$$|\mathcal{I}(\Gamma|\gamma)| = |\text{Perm}_{\text{ext}}(\gamma)| \prod_{r_V \in \mathcal{R}_V} n_{r_V}(\gamma)! \binom{m_{r_V}(\Gamma)}{n_{r_V}(\gamma)} \times \prod_{r_E \in \mathcal{R}_E} n_{r_E}(\gamma)! \binom{m_{r_E}(\Gamma) + n_{r_E}(\gamma) - 1}{n_{r_E}(\gamma)}, \quad (3.21)$$

where the first term describes the freedom to insert  $\gamma$  into  $\Gamma$  in all permutations of  $\gamma$ 's external legs, the second term represents the numbers of choices of insertion places for the factors of vertex residue type and the last term stands for the number of choices to insert factors of edge type into  $\Gamma$ . The factorials count the number of ways to choose a suitable order of the factors to insert. The difference in the vertex and edge insertions results from the fact that edges can be used for multiple insertions and vertices only for one. The special cases are declared as

$$\begin{aligned} |\mathcal{I}(\Gamma|\mathbb{I})| &= |\mathcal{I}(\mathbb{I}|\Gamma)| = |\mathcal{I}(\mathbb{I}|\mathbb{I})| = 1 & \forall \Gamma \in \mathcal{T} \\ \text{and } |\mathcal{I}(\mathbb{I}|\gamma)| &= 0 & \forall \gamma \in \mathcal{F}, \gamma \notin \mathcal{T}. \end{aligned} \quad (3.22)$$

**Examples** Suppose

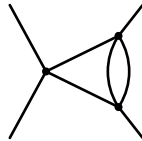


and  $\gamma$  shall be inserted into  $\Gamma$ . Dealing with a  $\varphi^4$ -theory only one vertex and one edge type need to be considered.  $\gamma$  is connected and of vertex residue type. Therefore,  $n_{\times}(\gamma) = 1$  and  $n_{-}(\gamma) = 0$ .  $|\text{Perm}_{\text{ext}}(\gamma)| = 4! = 24$ , because of the four external legs of  $\gamma$  of similar type.

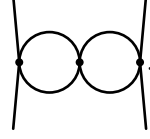
$\Gamma$  has two internal edges and two internal vertices. Consequently,  $m_{\times}(\Gamma) = 2$  and  $m_{-}(\Gamma) = 2$ . The total number of insertions is then, according to formula (3.21),

$$|\mathcal{I}(\Gamma|\gamma)| = 24 \cdot 1! \cdot \binom{2}{1} \cdot 0! \cdot \binom{2-1}{0} = 48.$$

Only two non isomorphic graphs will be yielded if  $\Gamma \circ_i \gamma$  is formed for all  $i \in \mathcal{I}(\Gamma|\gamma)$ . Considering the possible permutations of the external legs of  $\gamma$ , there will be 32 insertions giving graphs of the same isomorphy class as



and 16 insertions will give graphs of the same isomorphism class as



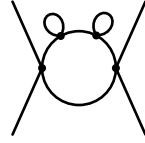
If on the other hand the product of graphs

$$\gamma' = \left( \text{diagram} \right)^2$$

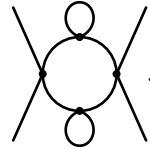
shall be inserted into  $\Gamma$ , the edge insertion places of  $\Gamma$  need to be considered. Clearly,  $n_{\times}(\gamma') = 0$ ,  $n_{-}(\gamma') = 2$  and  $|\text{Perm}_{\text{ext}}(\gamma')| = (2!)^2 = 4$ , because both factors of  $\gamma'$  have two external legs of the same type. Plugging into formula (3.21) yields

$$|\mathcal{I}(\Gamma|\gamma')| = 4 \cdot 0! \cdot \binom{2}{0} \cdot 2! \cdot \binom{2+2-1}{2} = 24.$$

Forming  $\Gamma \circ_i \gamma'$  for all  $i \in \mathcal{I}(\Gamma|\gamma')$  will yield 16 graphs isomorphic to



and 8 graphs isomorphic to



### 3.3.4 Automorphism group of products of graphs

Guided by the association of products of graphs with disconnected graphs the notion of the automorphism group can be extended. The extension to products of non-isomorphic graphs is easy, because all factors can be distinguished:

$$\text{Aut} \left( \prod_i \gamma_i \right) = \text{Aut}(\gamma_1) \times \text{Aut}(\gamma_2) \times \dots \quad \text{with } \gamma_i \in \mathcal{T} \text{ and } \gamma_i \neq \gamma_j \quad \forall i, j \quad (3.23)$$

$$\Rightarrow \left| \text{Aut} \left( \prod_i \gamma_i \right) \right| = \prod_i |\text{Aut}(\gamma_i)| \quad \text{with } \gamma_i \in \mathcal{T} \text{ and } \gamma_i \neq \gamma_j \quad \forall i, j \quad (3.24)$$

If the product contains more than one graph of the same isomorphism class, additional symmetry generators are obtained given by the permutations of the graphs of the same isomorphism class. Therefore, with given multiplicities  $n_i$  for every isomorphism class the cardinality of the automorphism group is:

$$\left| \text{Aut} \left( \prod_i (\gamma_i)^{n_i} \right) \right| = \prod_i (n_i! |\text{Aut}(\gamma_i)|^{n_i}) \quad \text{with } \gamma_i \in \mathcal{T} \text{ and } \gamma_i \not\cong \gamma_j \quad \forall i, j. \quad (3.25)$$

### 3.3.5 Sum formula for 1PI graphs

Making use of the preceding definitions and properties of Feynman graphs an identity of 1PI graphs sums involving the coproduct is proved. The statement is also proved in [13]. Here, a different argument is layed out based on the following lemma, a variant of a theorem in [3]:

**Lemma 1.** *For three given graphs  $\gamma \in \mathcal{F}$ ,  $\Gamma \in \mathcal{T}$  and  $\tilde{\Gamma} \in \mathcal{T}$ , the set of pairs  $(j_1, j_2)$  of an embedding  $j_1$  of  $\gamma$  into  $\Gamma$  and an isomorphism  $j_2$  between  $\Gamma/j_1(\gamma)$  and  $\tilde{\Gamma}$ ,*

$$A(\gamma, \tilde{\Gamma}, \Gamma) = \left\{ (j_1, j_2) \mid j_1 : \gamma \rightarrow \Gamma \text{ and } j_2 : \Gamma/j_1(\gamma) \rightarrow \tilde{\Gamma} \right\}, \quad (3.26)$$

*has the same cardinality as the set of all pairs  $(i, j)$  of insertions  $i \in \mathcal{I}(\tilde{\Gamma}|\gamma)$  and isomorphisms  $j$  between  $\tilde{\Gamma} \circ_i \gamma$  and  $\Gamma$ ,*

$$B(\gamma, \tilde{\Gamma}, \Gamma) = \left\{ (i, j) \mid i \in \mathcal{I}(\tilde{\Gamma}|\gamma) \text{ and } j : \tilde{\Gamma} \circ_i \gamma \rightarrow \Gamma \right\}. \quad (3.27)$$

*Proof.* A bijection between the sets  $A(\gamma, \tilde{\Gamma}, \Gamma)$  and  $B(\gamma, \tilde{\Gamma}, \Gamma)$  is constructed.

Given is a pair  $(j_1, j_2) \in A(\gamma, \tilde{\Gamma}, \Gamma)$ .  $j_1$  gives rise to a subgraph  $j_1(\gamma) \subseteq \Gamma$ . This subgraph can be contracted to yield an insertion  $i' \in \mathcal{I}(\Gamma/j_1(\gamma)|j_1(\gamma))$ . Using  $j_1$  and  $j_2$  an insertion  $i \in \mathcal{I}(\tilde{\Gamma}|\gamma)$  can be won. Clearly, there is an isomorphism  $j : \tilde{\Gamma} \circ_i \gamma \rightarrow \Gamma$ , induced by  $j_1$  and  $j_2$  because  $(\Gamma/j_1(\gamma)) \circ_{i'} j_1(\Gamma) = \Gamma$ .

This construction is reversible, because with these  $(i, j) \in B(\gamma, \tilde{\Gamma}, \Gamma)$ , the isomorphism  $j : \tilde{\Gamma} \circ_i \gamma \rightarrow \Gamma$  can be used to reconstruct  $j_1$  by restricting it on  $\gamma \subseteq \tilde{\Gamma} \circ_i \gamma$ . Contracting  $\Gamma$  to  $\Gamma/j_1(\gamma)$  and applying  $j^{-1}$  to get  $j^{-1}(\Gamma)/j^{-1}(j_1(\gamma))$  gives  $\tilde{\Gamma}$ . Therefore,  $j_2$  is also retrieved. This procedure is defined for all  $(i, j) \in B(\gamma, \tilde{\Gamma}, \Gamma)$  and establishes that this is a one-to-one.  $\square$

**Corollary 1.** *For  $\gamma \in \mathcal{F}$  and  $\tilde{\Gamma} \in \mathcal{T}$ ,*

$$\frac{|\mathcal{I}(\tilde{\Gamma}|\gamma)|}{|\text{Aut}(\gamma)| |\text{Aut}(\tilde{\Gamma})|} = \sum_{\Gamma \in \mathcal{T}} \frac{|\{ \gamma' \subseteq \Gamma \mid \gamma' \simeq \gamma \text{ and } \Gamma/\gamma' \simeq \tilde{\Gamma} \}|}{|\text{Aut}(\Gamma)|}. \quad (3.28)$$

*Proof.* Because the total number of automorphisms of a graph  $\gamma$  is given by  $|\text{Aut}(\gamma)|$

$$|A(\gamma, \tilde{\Gamma}, \Gamma)| = |\text{Aut}(\gamma)| |\text{Aut}(\tilde{\Gamma})| \left| \left\{ \gamma' \subseteq \Gamma \mid \gamma' \simeq \gamma \text{ and } \Gamma/\gamma' \simeq \tilde{\Gamma} \right\} \right| \quad (3.29)$$

and

$$|B(\gamma, \tilde{\Gamma}, \Gamma)| = |\text{Aut}(\Gamma)| \left| \left\{ i \mid i \in \mathcal{I}(\tilde{\Gamma}|\gamma) \text{ and } \tilde{\Gamma} \circ_i \gamma \simeq \Gamma \right\} \right|. \quad (3.30)$$

Using

$$\sum_{\Gamma \in \mathcal{T}} \left| \left\{ i \mid i \in \mathcal{I}(\tilde{\Gamma}|\gamma) \text{ and } \tilde{\Gamma} \circ_i \gamma \simeq \Gamma \right\} \right| = |\mathcal{I}(\tilde{\Gamma}|\gamma)|, \quad (3.31)$$

and plugging in  $\frac{|B(\gamma, \tilde{\Gamma}, \Gamma)|}{|\text{Aut}(\Gamma)|}$  yields

$$\begin{aligned} |\mathcal{I}(\tilde{\Gamma}|\gamma)| &= \sum_{\Gamma \in \mathcal{T}} \frac{|B(\gamma, \tilde{\Gamma}, \Gamma)|}{|\text{Aut}(\Gamma)|} \\ &= \sum_{\Gamma \in \mathcal{T}} \frac{|\text{Aut}(\gamma)| |\text{Aut}(\tilde{\Gamma})| \left| \left\{ \gamma' \subseteq \Gamma \mid \gamma' \simeq \gamma \text{ and } \Gamma/\gamma' \simeq \tilde{\Gamma} \right\} \right|}{|\text{Aut}(\Gamma)|}, \end{aligned} \quad (3.32)$$

which confirms (3.28).  $\square$

Using this corollary, it is possible to prove the theorem, which is the purpose of this section:

**Theorem 1.**

$$\sum_{\Gamma \in \mathcal{T}} \frac{\Delta_D \Gamma}{|\text{Aut}(\Gamma)|} = \sum_{\substack{\gamma = \left( \prod_i \gamma_i \right) \in \mathcal{F} \\ \omega_D(\gamma_i) \leq 0}} \sum_{\tilde{\Gamma} \in \mathcal{T}} \frac{|\mathcal{I}(\tilde{\Gamma}|\gamma)|}{|\text{Aut}(\gamma)| |\text{Aut}(\tilde{\Gamma})|} \gamma \otimes \tilde{\Gamma}. \quad (3.33)$$

*Proof.* The right hand side of the statement can be rewritten using corollary 1:

$$\sum_{\Gamma \in \mathcal{T}} \sum_{\substack{\gamma = \left( \prod_i \gamma_i \right) \in \mathcal{F} \\ \omega_D(\gamma_i) \leq 0}} \sum_{\tilde{\Gamma} \in \mathcal{T}} \frac{\left| \left\{ \gamma' \subseteq \Gamma \mid \gamma' \simeq \gamma \text{ and } \Gamma/\gamma' \simeq \tilde{\Gamma} \right\} \right|}{|\text{Aut}(\Gamma)|} \gamma \otimes \tilde{\Gamma}. \quad (3.34)$$

Interchanging sums is permissible, because the grading guaranties that the left hand side of the statement, restricted on a certain loop number, will be a finite sum of tensor products. Performing the two inner sums and taking definition of the relation  $\leq$  in (3.2) into account, gives the result

$$= \sum_{\Gamma \in \mathcal{T}} \sum_{\gamma' \leq \Gamma} \frac{1}{|\text{Aut}(\Gamma)|} \gamma' \otimes \Gamma/\gamma', \quad (3.35)$$

where the inner sum coincides with the definition of the coproduct on Feynman graphs.  $\square$

This formula gives a non-trivial check of the coproduct computation in **feyn-cop**. Clearly, identity (3.33) can be restricted to graphs with a certain residue  $r$ , because the cographs carry the same residue as the original graph, and a certain loop number  $L$  using the grading:

$$\sum_{\substack{\Gamma \in \mathcal{T} \\ h_1(\Gamma)=L \\ \text{res}(\Gamma)=r}} \frac{\Delta_D \Gamma}{|\text{Aut}(\Gamma)|} = \sum_{\substack{l_1, l_2 \geq 0 \\ l_1 + l_2 = L}} \sum_{\substack{\gamma = \left( \prod_i \gamma_i \right) \in \mathcal{F} \\ \omega_D(\gamma_i) \leq 0 \\ h_1(\gamma) = l_1}} \sum_{\substack{\tilde{\Gamma} \in \mathcal{T} \\ h_1(\tilde{\Gamma}) = l_2 \\ \text{res}(\tilde{\Gamma}) = r}} \frac{|\mathcal{I}(\tilde{\Gamma}|\gamma)|}{|\text{Aut}(\gamma)| |\text{Aut}(\tilde{\Gamma})|} \gamma \otimes \tilde{\Gamma}. \quad (3.36)$$

**feyn-cop** implements a unit test which uses this formula to check the computation if all graphs of a given loop number and a certain residue type are given as input. This check was performed for QED,  $\varphi^3$  and  $\varphi^4$  graphs up to at least fourth loop order and the residue types in  $\mathcal{R}_E \cup \mathcal{R}_V$ . Additional checks were also performed with graphs of residue types not in  $\mathcal{R}_E \cup \mathcal{R}_V$ .

**Example** To illustrate the check, which can be performed using identity (3.33), an example is given.

Consider the sum of all two loop, propagator residue type  $\varphi^4$ -graphs weighted by their symmetry factor:

$$X_2^- := \sum_{\substack{\Gamma \in \mathcal{T} \\ h_1(\Gamma)=2 \\ \text{res}(\Gamma)=-}} \frac{\Gamma}{|\text{Aut}(\Gamma)|} = \frac{1}{8} \text{diagram}_1 + \frac{1}{12} \text{diagram}_2 .$$

The coproducts of both graphs in four dimensions are

$$\begin{aligned} \Delta_4 \left( \text{diagram}_1 \right) &= \mathbb{I} \otimes \text{diagram}_1 + \text{diagram}_1 \otimes \mathbb{I} + \text{diagram}_3 \otimes \text{diagram}_4 + \text{diagram}_5 \otimes \text{diagram}_6 \\ \Delta_4 \left( \text{diagram}_2 \right) &= \mathbb{I} \otimes \text{diagram}_2 + \text{diagram}_2 \otimes \mathbb{I} + 3 \text{diagram}_5 \otimes \text{diagram}_6 . \end{aligned}$$

Therefore, applying the coproduct to the sum yields

$$\Delta_4 X_2^- = \mathbb{I} \otimes X_2^- + X_2^- \otimes \mathbb{I} + \frac{1}{8} \text{L} \otimes \text{L} + \frac{3}{8} \text{X} \otimes \text{L}.$$

On the other hand identity (3.36) gives:

$$\begin{aligned} \Delta_4 X_2^- &= \sum_{\substack{\Gamma \in \mathcal{T} \\ h_1(\Gamma)=2 \\ \text{res}(\Gamma)=-}} \frac{\Delta_D \Gamma}{|\text{Aut}(\Gamma)|} = \mathbb{I} \otimes X_2^- + X_2^- \otimes \mathbb{I} + \\ &+ \sum_{\substack{\gamma = \left( \prod_i \gamma_i \right) \in \mathcal{F} \\ \omega_4(\gamma_i) \leq 0 \\ h_1(\gamma)=1}} \sum_{\substack{\tilde{\Gamma} \in \mathcal{T} \\ h_1(\tilde{\Gamma})=1 \\ \text{res}(\tilde{\Gamma})=-}} \frac{|\mathcal{I}(\tilde{\Gamma}|\gamma)|}{|\text{Aut}(\gamma)| |\text{Aut}(\tilde{\Gamma})|} \gamma \otimes \tilde{\Gamma}, \end{aligned}$$

where the trivial summands were already evaluated in accordance to the special values given in (3.22).

The only relevant graphs for the sum over the subgraphs  $\gamma$  on the right hand side are  $\text{L}$  and  $\text{X}$ , because these are the only 1PI,  $\varphi^4$ , one-loop graphs with  $\omega_4(\gamma) \leq 0$ . For the sum over the cographs  $\tilde{\Gamma}$  only  $\text{L}$  has to be considered, because it is the only 1PI, one-loop graph with two external legs.

Calculating the numbers of insertions,

$$|\mathcal{I}(\text{L}|\text{L})| = 2 \qquad |\mathcal{I}(\text{L}|\text{X})| = 24,$$

which in this case only depend on the possible external leg permutations and the orders of the automorphism groups,

$$|\text{Aut}(\text{L})| = 4 \qquad |\text{Aut}(\text{X})| = 16,$$

yields the same result as the direct calculation above:

$$\begin{aligned} &\sum_{\substack{\gamma = \left( \prod_i \gamma_i \right) \in \mathcal{F} \\ \omega_4(\gamma_i) \leq 0 \\ h_1(\gamma)=1}} \sum_{\substack{\tilde{\Gamma} \in \mathcal{T} \\ h_1(\tilde{\Gamma})=1 \\ \text{res}(\tilde{\Gamma})=-}} \frac{|\mathcal{I}(\tilde{\Gamma}|\gamma)|}{|\text{Aut}(\gamma)| |\text{Aut}(\tilde{\Gamma})|} \gamma \otimes \tilde{\Gamma} = \\ &= \frac{2}{4 \cdot 4} \text{L} \otimes \text{L} + \frac{24}{16 \cdot 4} \text{X} \otimes \text{L}. \end{aligned}$$

# Chapter 4

## Zero-dimensional QFT

Although, most interacting quantum field theories are hard to solve in general, zero-dimensional quantum field theories form an exception, which is worth studying, especially for the analysis of Feynman diagram topologies. The topologies of graphs are independent of the dimension of spacetime. A solution of the zero-dimensional theory can be used to check the validity of the graph topologies for the  $D$ -dimensional case. To validate the Feynman graph generation with the program **feyngen**, the perturbation expansion of a zero dimensional quantum field theory was used. The following chapter sketches the procedure to obtain a perturbation expansion from such a field theory guided by explicit examples of the necessary calculations.

### 4.1 $\varphi^k$ -theory

#### 4.1.1 Generating function

For a zero-dimensional  $\varphi^k$ -theory on a single point the generating function of the graphs is given, similar to [8], as,

$$Z_{\varphi^k}(a, \lambda, j) := \int_{\mathbb{R}} \frac{d\varphi}{\sqrt{2\pi a}} e^{-\frac{\varphi^2}{2a} + \lambda \frac{\varphi^k}{k!} + j\varphi}, \quad (4.1)$$

where  $a$  counts the number of edges (including external ones),  $j$  counts the number of source vertices and  $\lambda$  counts the number of internal vertices. The measure is chosen such that  $Z_{\varphi^k}(a, 0, 0) = 1 \quad \forall a > 0$ , meaning that the empty graph is assigned the symmetry factor 1. Of course, there is no propagation in a zero dimensional field theory of a single point. Therefore, the propagator is simulated by the constant self interaction amplitude  $a$ .

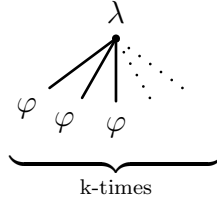
For general, possibly odd,  $k \geq 3$ , this integral is only convergent if  $\lambda = 0$ . But from a perturbative view point, a formal power series can still be obtained by expanding the  $j$ ,  $\lambda$  factors and pulling the summations to the front:

$$Z_{\varphi^k}(a, \lambda, j) = \int_{\mathbb{R}} \frac{d\varphi}{\sqrt{2\pi a}} \left\{ e^{-\frac{\varphi^2}{2a}} \sum_{n \geq 0} \frac{1}{n!} \left( \frac{\lambda \varphi^k}{k!} \right)^n \sum_{m \geq 0} \frac{1}{m!} (j\varphi)^m \right\} \quad (4.2)$$

$$\Rightarrow \tilde{Z}_{\varphi^k}(a, \lambda, j) := \sum_{n, m \geq 0} \int_{\mathbb{R}} \frac{d\varphi}{\sqrt{2\pi a}} \left\{ e^{-\frac{\varphi^2}{2a}} \frac{1}{n! m!} \left( \frac{\lambda \varphi^k}{k!} \right)^n (j\varphi)^m \right\}. \quad (4.3)$$

$\tilde{Z}_{\varphi^k}$  gives a more suitable definition, because the integral converges, resulting in a well defined formal power series. The series is not convergent for  $\lambda > 0$ , but in the present case only the enumerative properties of the series are of interest and questions of convergence can be ignored.

The sum over monomials in  $\varphi$  has the combinatorial interpretation of corollas, entities consisting of a vertex (mirroring the  $\lambda$  factor) with  $k$  nodes connected to it (associated with the  $\varphi^k$  factor coming with each  $\lambda$ ),



and source nodes connected to source vertices  $\varphi$  and  $j$  factors:

$$\varphi \longrightarrow \times j$$

The integration can be interpreted as application of Wick's theorem acting on the monomials in  $\varphi$  by gluing the nodes associated to the  $\varphi$ -factors together in all possible ways, so called Wick contractions. By gluing corollas and source vertices together all Feynman graphs can be obtained.

The factorials in the denominator of the summands correspond to a division by the total size of the relevant symmetry group acting on these corollas and the source vertices. The number of Wick contractions yielding isomorphic graphs corresponds to the orbit size of these graphs under this symmetry group. Applying the orbit stabilizer theorem, every term obtains the inverse of the size of the subgroup stabilizing the graph under the action of the full symmetry group, such that  $\tilde{Z}_{\varphi^k}$  is equal to a sum over all non-isomorphic graphs weighted by the inverse of that stabilizer (i.e. their symmetry factor).



In this case, Wick's Theorem takes the simple form,

$$\int_{\mathbb{R}} \frac{d\varphi}{\sqrt{2\pi a}} \varphi^{2l} e^{-a\varphi^2} = \frac{1}{a^l} (2l-1)!! \quad \forall l \in \mathbb{N} \quad (4.4)$$

$$\int_{\mathbb{R}} \frac{d\varphi}{\sqrt{2\pi a}} \varphi^{2l+1} e^{-a\varphi^2} = 0 \quad \forall l \in \mathbb{N}, \quad (4.5)$$

with which the formal power series can be written as

$$\tilde{Z}_{\varphi^k}(a, \lambda, j) = \sum_{l \geq 0} \sum_{\substack{n, m \geq 0 \\ nk+m=2l}} \frac{(2l-1)!!}{n!m!(k!)^n} a^l \lambda^n j^m. \quad (4.6)$$

and a closed form for the coefficient in every order if  $a$ ,  $\lambda$  and  $j$  is won.

**Example** For a  $\varphi^3$  theory the terms up to  $a^3$  are:

$$\begin{aligned} \tilde{Z}_{\varphi^3}(a, \lambda, j) = & 1 + \frac{1}{2}j^2a + \left(\frac{1}{8}j^4 + \frac{1}{2}j\lambda\right)a^2 + \\ & + \left(\frac{1}{48}j^6 + \frac{5}{12}j^3\lambda + \frac{5}{24}\lambda^2\right)a^3 + \dots \end{aligned} \quad (4.7)$$

In accordance to the combinatorial interpretation stated above, every term in this expansion can be identified with a sum over the symmetry factors of all non-isomorphic  $\varphi^3$  Feynman graphs (without fixed legs) with the number of source vertices, internal vertices and edges corresponding to the powers in  $j$ ,  $\lambda$  and  $a$ .

In this case the corresponding series in diagrammatic notation is:

$$\begin{aligned} \tilde{Z}_{\varphi^3}(a, \lambda, j) = & 1 + \underbrace{\frac{1}{2} \text{---} \text{---}}_{\frac{1}{2}j^2a} + \underbrace{\frac{1}{8} \text{---} \text{---}}_{\frac{1}{8}j^4a^2} + \underbrace{\frac{1}{2} \text{---} \text{---}}_{\frac{1}{2}j\lambda a^2} + \underbrace{\frac{1}{48} \text{---} \text{---}}_{\frac{1}{48}j^6a^3} + \\ & + \underbrace{\frac{1}{6} \text{---} \text{---}}_{\frac{5}{12}j^3\lambda a^3} + \underbrace{\frac{1}{4} \text{---} \text{---}}_{\frac{5}{24}\lambda^2 a^3} + \frac{1}{12} \text{---} \text{---} + \frac{1}{8} \text{---} \text{---} + \dots \end{aligned} \quad (4.8)$$

Crossed source vertices correspond to  $j$ -factors, dotted vertices to  $\lambda$ -factors and every edge corresponds to a power in  $a$ .

Note, the closed form of the perturbative expansion of a zero dimensional field theory in (4.6) can be interpreted as an expansion of a general field theory with Feynman rules assigning the amplitude 1 to every graph.

If all the non-isomorphic graphs for a  $\varphi^k$  quantum field theory shall be generated, equation (4.6) can be used to check the validity of the sum of the computed symmetry factors by comparing it with a term in the series as done in the example.

**feyngen** performs this check automatically if all graphs contributing to a certain summand in  $\tilde{Z}$  are computed.

### 4.1.2 Connected graphs

Usually, only more restricted sets of diagrams are of interest. Because  $\tilde{Z}_{\varphi^k}(a, \lambda, j)$  effectively counts labeled objects, a consequence of the weighting with the symmetry factor, the exponential formula from the field of labeled counting, as stated for instance in [14] or [7], can be applied to obtain the power series that enumerates connected graphs.

The exponential formula gives the simple correspondence between the generating function  $\tilde{Z}_{\varphi^k}(a, \lambda, j)$  of all graphs and the generation function  $W_{\varphi^k}(a, \lambda, j)$  of the connected graphs,

$$\tilde{Z}_{\varphi^k}(a, \lambda, j) = \tilde{Z}_{\varphi^k}(0, 0, 0)e^{W_{\varphi^k}(a, \lambda, j)} \quad (4.9)$$

$$\Rightarrow W_{\varphi^k}(a, \lambda, j) = \log \left( \frac{\tilde{Z}_{\varphi^k}(a, \lambda, j)}{\tilde{Z}_{\varphi^k}(0, 0, 0)} \right). \quad (4.10)$$

The expansion of  $Z_{\varphi^k}$  starts with a 1, therefore using,

$$\log(1 + x) = - \sum_{n \geq 1} \frac{(-1)^n}{n} x^n, \quad (4.11)$$

a formal power series for  $W_{\varphi^k}(a, \lambda, j)$  could be obtained.

Unfortunately, the above expansion (4.11) is not suitable for a practical calculation of  $W_{\varphi^k}(a, \lambda, j)$ , because calculating higher powers of  $\tilde{Z}_{\varphi^k}$  becomes a highly exhaustive task for higher orders. For a more efficient explicit calculation of the connected diagrams the reciprocal of the generating function  $\tilde{Z}_{\varphi^k}$  is needed.

Abbreviating,

$$\tilde{Z}_{\varphi^k}(a, \lambda, j) = \sum_{l \geq 0} a^l z_l(\lambda, j) \quad (4.12)$$

with

$$z_l(\lambda, j) := \sum_{\substack{n, m \geq 0 \\ nk+m=2l}} \frac{(2l-1)!!}{n!m!(k!)^n} \lambda^n j^m \quad (4.13)$$

and

$$\frac{1}{\tilde{Z}_{\varphi^k}(a, \lambda, j)} =: \sum_{l \geq 0} a^l z_l^{\text{rec}}(\lambda, j), \quad (4.14)$$

the coefficients of the reciprocal power series can be computed following the standard procedure:

$$\begin{aligned} 1 &= \sum_{l \geq 0} a^l z_l^{\text{rec}}(\lambda, j) \sum_{l' \geq 0} a^{l'} z_{l'}(\lambda, j) = \\ &= \sum_{L \geq 0} a^L \sum_{l=0}^L z_l^{\text{rec}}(\lambda, j) z_{L-l}(\lambda, j), \end{aligned} \quad (4.15)$$

which yields a recursion relation for  $z_l^{\text{rec}}(\lambda, j)$ :

$$\begin{aligned} z_0^{\text{rec}}(\lambda, j) &= 1 \\ z_L^{\text{rec}}(\lambda, j) &= - \sum_{l=0}^{L-1} z_l^{\text{rec}}(\lambda, j) z_{L-l}(\lambda, j) \quad \forall L \geq 1. \end{aligned} \quad (4.16)$$

With this result, a recursion relation for the coefficients of

$$W_{\varphi^k}(a, \lambda, j) = \sum_{l \geq 1} a^l w_l(\lambda, j) \quad (4.17)$$

can be given. Differentiating equation (4.10) on both sides by  $a$ ,

$$\frac{\partial W_{\varphi^k}(a, \lambda, j)}{\partial a} = \frac{\frac{\partial \tilde{Z}_{\varphi^k}(a, \lambda, j)}{\partial a}}{\tilde{Z}_{\varphi^k}(a, \lambda, j)}, \quad (4.18)$$

and plugging in the power series yields the relation,

$$\begin{aligned} \sum_{l \geq 0} (l+1) a^l w_{l+1}(\lambda, j) &= \sum_{l \geq 0} (l+1) a^l z_{l+1}(\lambda, j) \sum_{l' \geq 0} a^{l'} z_{l'}^{\text{rec}}(\lambda, j) = \\ &= \sum_L a^L \sum_{l=0}^L (l+1) z_{l+1}(\lambda, j) z_{L-l}^{\text{rec}}(\lambda, j), \end{aligned} \quad (4.19)$$

from which a recursion can be read off,

$$\Rightarrow (L+1) w_{L+1}(\lambda, j) = \sum_{l=0}^L (l+1) z_{l+1}(\lambda, j) z_{L-l}^{\text{rec}}(\lambda, j) \quad \forall L \geq 0 \quad (4.20)$$

$$\Rightarrow w_L(\lambda, j) = \frac{1}{L} \sum_{l=1}^L l z_l(\lambda, j) z_{L-l}^{\text{rec}}(\lambda, j) \quad \forall L \geq 1 \quad (4.21)$$

**Example** For  $\varphi^3$  theory the first coefficients  $z_l^{\text{rec}}(\lambda, j)$ , calculated using the recursion formula (4.16), are:

$$\begin{aligned} z_0^{\text{rec}}(\lambda, j) &= 1 \\ z_1^{\text{rec}}(\lambda, j) &= -\frac{1}{2}j^2 \\ z_2^{\text{rec}}(\lambda, j) &= -\frac{1}{8}j^4 - \frac{1}{2}j\lambda + \left(\frac{1}{2}j^2\right)^2 = \frac{1}{8}j^4 - \frac{1}{2}j\lambda \end{aligned} \quad (4.22)$$

Using recursion (4.21), the first coefficients  $w_l(\lambda, j)$  can be calculated,

$$\begin{aligned} w_1(\lambda, j) &= \frac{1}{2}j^2 \\ w_2(\lambda, j) &= \frac{1}{2} \left( -\frac{1}{4}j^4 + 2 \left( \frac{1}{8}j^4 + \frac{1}{2}j\lambda \right) \right) = \frac{1}{2}j\lambda \\ w_3(\lambda, j) &= \frac{1}{3} \left( \frac{1}{16}j^6 - \frac{1}{4}j^3\lambda + 2 \left( -\frac{1}{16}j^6 - \frac{1}{4}j^3\lambda \right) + \right. \\ &\quad \left. + 3 \left( \frac{1}{48}j^6 + \frac{5}{12}j^3\lambda + \frac{5}{24}\lambda^2 \right) \right) = \frac{1}{6}j^3\lambda + \frac{5}{24}\lambda^2 \end{aligned} \quad (4.23)$$

This corresponds to the diagrammatic series of connected graphs:

$$W_{\varphi^3}(a, \lambda, j) = \underbrace{\frac{1}{2} \text{---} \text{---} \text{---}}_{\frac{1}{2}j^2 a} + \underbrace{\frac{1}{2} \text{---} \text{---} \text{---} \text{---}}_{\frac{1}{2}j\lambda a^2} + \underbrace{\frac{1}{6} \text{---} \text{---} \text{---} \text{---}}_{\frac{1}{6}j^3\lambda a^3} + \underbrace{\frac{1}{12} \text{---} \text{---} \text{---} \text{---}}_{\frac{5}{24}\lambda^2 a^3} + \frac{1}{8} \text{---} \text{---} \text{---} \text{---} + \dots \quad (4.24)$$

## 4.2 $|\phi|^2 A$ -theory

Additionally, QED graphs with two types of edges, one directed and one undirected, will be generated. To check this computation, another generating function needs to be introduced. The topology of the diagram is not influenced by the spin characteristics of the theory. Therefore, it is enough to take the generating function of a theory with one charged scalar field  $\phi$  and a neutral scalar field  $A$ , interacting via a  $q|\phi|^2 A$  term, mimicking the  $q\bar{\psi}A\psi$  interaction.

The generating function

$$Z_{|\phi|^2 A}(a, q, j, \eta, \bar{\eta}) = \int_{\mathbb{R}} \frac{dA}{\sqrt{2\pi a}} \int_{\mathbb{C}} \frac{d^2\phi}{\pi a} e^{-\frac{A^2 + 2|\phi|^2}{2a} + q|\phi|^2 A + jA + \eta\bar{\phi} + \bar{\eta}\phi} \quad (4.25)$$

is normalized such that  $Z_{|\phi|^2 A}(a, 0, 0, 0, 0) = 1 \quad \forall a > 0$ . The  $a$  again gives a constant self-interaction amplitude. The expansion around  $q = 0$  under the

integral is performed and integration and summation are interchanged, yielding a formal power series:

$$\begin{aligned} \tilde{Z}_{|\phi|^2 A}(a, q, j, \eta, \bar{\eta}) &= \sum_{n, m, s, t \geq 0} \int_{\mathbb{R}} \frac{dA}{\sqrt{2\pi a}} \int_{\mathbb{C}} \frac{d^2\phi}{\pi a} \left\{ e^{-\frac{A^2 + 2|\phi|^2}{2a}} \times \right. \\ &\quad \left. \times \frac{q^n}{n!} (A|\phi|^2)^n \frac{j^m}{m!} A^m \frac{\eta^s}{s!} \bar{\phi}^s \frac{\bar{\eta}^t}{t!} \phi^t \right\}. \end{aligned} \quad (4.26)$$

Using

$$\int_{\mathbb{C}} \frac{d^2\phi}{\pi a} |\phi|^{2l} e^{-\frac{|\phi|^2}{a}} = l! a^l \quad \forall l \in \mathbb{N} \quad (4.27)$$

$$\int_{\mathbb{C}} \frac{d^2\phi}{\pi a} \phi^s \bar{\phi}^t e^{-\frac{|\phi|^2}{a}} = 0 \quad \forall s, t \in \mathbb{N} \text{ and } s \neq t, \quad (4.28)$$

the integration can be performed term wise:

$$\tilde{Z}_{|\phi|^2 A}(a, q, j, \eta, \bar{\eta}) = \sum_{l \geq 0} a^l \sum_{\substack{n, m, r \geq 0 \\ m+n+2r=2l}} \frac{(m+n+1)!(2n+2r)!}{n!m!(r!)^2} q^n j^m |\eta|^{2r}. \quad (4.29)$$

The connected diagrams can be obtained using the exponential formula as in the last section. The explicit calculation is performed in the same way as depicted for  $\varphi^k$ -theory.



# Chapter 5

## Diagram generation

### 5.1 Overview

The python program **feynngen** can generate  $\varphi^k$  for  $k \geq 3$  and QED diagrams ready to be used in green's function calculations. The main purpose of **feynngen** is to provide input for the coproduct calculation in **feyncop**.

Developing **feynngen**, the focus was on the generation of Feynman diagrams with comparatively large loop orders. For instance, all 130516 1PI,  $\varphi^4$ , 8-loop diagrams with four external legs can be generated, together with their symmetry factor, within eight hours and all 342430 1PI, QED, vertex residue type, 6-loop diagrams can be generated in three days both on a standard end-user computer.

Additionally to the computation of non-isomorphic diagrams, **feynngen** calculates the symmetry factors of the resulting graphs. Handling of leg-fixed and non-leg-fixed graphs is implemented. Furthermore, options are available to filter for connected, 1PI, vertex-2-connected and tadpole free graphs.

To achieve the high speed for the computation **feynngen** relies on the established **nauty** package.

### 5.2 Sketch of the implementation of feynngen

The graphs in **feynngen** are represented as edge lists or respectively as an ordered sequence representing the multiset  $\Gamma^{[1]}$  of a graph  $\Gamma$  as defined in definition 3. The edges themselves are stored as pairs of vertices. Vertices are represented by integers  $\geq 0$ . The vertex set is not explicitly stored, but every vertex is given implicitly by the edges incident to it. This is possible, because vertices with valency 0 are not required. Internal and external edges are stored equally. External vertices are only characterized by their valency being 1.

In the first step of the computation of non-isomorphic Feynman diagrams, the

**nauty** programs **geng** and **multig**, whose implementation is discussed in [11], are used to compute non-isomorphic multigraphs without self-loops.

**feynngen** decorates these diagrams with self-loops if the generation of tadpole diagrams is requested, until every vertex has the desired vertex type.

If the generation of QED graphs is required, the possible ways to color and direct the graph's edges as photon or fermions, to yield a valid QED graph, are computed.

Afterwards, the filters requested by the given options are tested on the graph. If the graph fulfills all desired properties, the algorithm continues. Next, if not otherwise stated by the parameters, the different topologies resulting from possible permutations of the external legs are computed.

To calculate the order of the automorphism group, the edge and vertex colored multigraph must be converted to a vertex colored, simple graph. This conversation is achieved by adding auxiliary colored vertices, representing the colored or multiple edges.

In the last step, using **nauty**, the order of the automorphism group of the graph is computed and it is ensured that by coloring, adding self-loops or permuting the external legs no isomorphic graphs were generated.

### 5.3 Check of validity

To check the validity of the diagram generation, the results of chapter 4 were used. If **feynngen** calculates all diagrams or all connected diagrams of a given class, the sum of the symmetry factors is compared to an corresponding term in a power series, associated to a zero dimensional theory as shown in the referred chapter.

Another test for the computation of 1PI graphs is possible in conjunction with **feynncop**. It is described in section 6.3.

## 5.4 Manual of feynngen

### 5.4.1 Overview

Because **feynngen** is a program optimized to generate non-isomorphic high loop Feynman diagrams, the only mandatory parameter for **feynngen** is the order in  $\hbar$  in the perturbation series of the class of diagrams to be generated. This corresponds to the number

$$h_1(\gamma) - f_\gamma + 1,$$



where  $f_\gamma$  is the number of connected components of the graphs  $\gamma$  to generate. For connected graphs,  $f_\gamma = 1$ , this number is equivalent to the loop number  $h_1(\gamma)$ .

For example, the call to **feynngen**,

```
$ ./feynngen 3 4 5
```

will generate all connected 3, 4 and 5 loop vacuum diagrams and all disconnected diagrams which correspond to these loop orders in respect to their order in  $\hbar$  in the perturbation series of  $\varphi^4$  theory.

### 5.4.2 Options and Parameters

Additionally to the loop number, the generation can be controlled by various options. **feynngen** called with the option **--help** prints the list of all possible program options together with a short description.

```
$ ./feynngen --help
```

For instance, only graphs with certain properties can be generated. Such restrictions on the graphs generated can be set by the following options:

**-c / --connected**

Generate only connected graphs.

**-p / --1PI**

Generate only 1PI graphs.

**-v / --vtx2cntd**

Generate only 2-vertex connected graphs.

**-t / --notadpoles**

Generate only non-tadpole graphs.

By default, **feynngen** generates  $\varphi^4$  graphs. The two options:

**-k# / --valence=#**

Generate graphs with vertex valence  $\#$  for a scalar  $\varphi^\#$  theory.

**--qed**

Generate graphs for QED.

can be used to alter the type of graphs generated.

The external leg structure of the graphs is determined by the parameters,

**-j# / --ext.legs=#**

Set the total number of external legs  $\#$  of  $\varphi^k$  graphs.

**-b# / --ext\_photon\_legs=#**

Set the number of external photon legs # of QED graphs.

**-f# / --ext\_fermion\_legs=#**

Set the number of external fermion legs # of QED graphs.

depending on whether graphs for  $\varphi^k$ -theory or for QED are generated. By default, only graphs without external legs are given as output.

Additionally, the behaviour under graph isomorphisms of the external legs can be controlled:

**-u / --no\_ext\_label**

External legs of graphs are not considered as fixed if this option is set. This option influences isomorphism testing and symmetry factor calculation of graphs.

If not stated otherwise, leg-fixed diagrams are generated.

### 5.4.3 Output of $\varphi^k$ -graphs

#### Representation of graphs

Graphs are represented as edge lists. An edge is represented by a pair of vertices or a triple of two vertices and an integer, representing the weight of the edge. The pairs or triples are embraced by brackets. Vertices are labeled by integers.

So for example

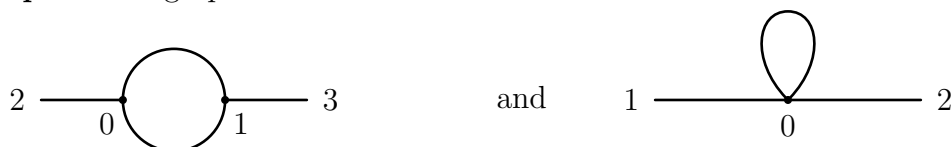
[2,3]                      and                      [6,4,1]

represent one edge without weight between the vertices 2 and 3 and one of weight 1 between the vertices 4 and 6. For QED graphs edges with weight 1 will depict oriented fermion edges, so [6,4,1] can be interpreted as an fermion pointing from vertex 6 to vertex 4.

External edges are not distinguished from internal edges, except that they are incident to an external one-valent vertex as in definition 3.

The edge list is embraced by brackets and prefixed by a **G** to simplify the usage of the output with **maple**.

**Example** The graphs



are represented as

$$G[[1,0],[1,0],[2,0],[3,1]] \quad \text{and} \quad G[[0,0],[1,0],[2,0]],$$

where in the first diagram 0 and 1 are the internal vertices and 2 and 3 are external vertices. In the second diagram 0 is the only internal vertex with 1 and 2 depicting external vertices. The labeling of the vertices is auxiliary and is used to give a representative of the isomorphism class of the graph. The labeling is assigned using **nauty**, which chooses a canonical labeling unique for every isomorphism class of graphs. The labeling is chosen, such that the external vertices carry the highest labels.

### Output of graph sums

The output format of **feynngen** is designed to be readable by a **maple** program. Therefore, the graphs generated by **feynngen** are written as a sum of graphs with every graph weighted by its symmetry factor.

For instance, the sum of all  $\varphi^3$  2-loop graphs without external legs, with each graph weighted by its symmetry factor, depicted diagrammatically as

$$\frac{1}{8} \begin{array}{c} \text{---} \\ \circ \quad \text{---} \quad \text{---} \quad \circ \\ \text{---} \end{array} + \frac{1}{12} \begin{array}{c} \text{---} \\ \circ \quad \text{---} \quad \circ \\ \text{---} \end{array}$$

can be generated by **feynngen** as follows:

```
$ ./feynngen 2 -k3
phi3_j0_h2 :=
+G[[0,0],[1,0],[1,1]]/8
+G[[1,0],[1,0],[1,0]]/12
;
```

Where the **2** in the command line stands for diagram generation of order  $\hbar^2$  and **-k3** for  $\varphi^3$ -theory, such that only graphs with three valent vertices are generated.

**Symmetry factors** As can be seen in the above example, the graphs are given weighted by their symmetry factor. The format is

$G[\dots]/\text{Aut}$ ,

where **Aut** is the order of the automorphism group of the graph.

Note that in general, the calculation of the symmetry factor depends on the **-u** option if external legs are present, depending on whether external legs are labeled or not. An explicit example for the behaviour of the **-u** option is given in section 5.4.5.

### Distinguished name for maple usage

The sum of graphs is given a name, which indicates the loop number(s), the number of external edges and the theory type.

That means, the output is always of the form:

```
phi(k)_j(m)_h(L) :=
+G[...]/Aut1
+G[...]/Aut2
+...
...
;
```

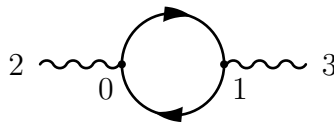
Where  $(k)$  is replaced with the appropriate theory degree,  $(m)$  is substituted by the number of external edges and  $(L)$  is given by the order in  $\hbar$  of the graphs.

#### 5.4.4 Output of QED graphs

QED diagrams carry additional information, because they have two different edge types. These edge types are associated with different weights. Consequently, an QED edge  $[v1, v2, w]$  is depicted as a pair of vertices  $v1, v2$  together with the edge's weight  $w$ .

Weight 1 is assigned to a fermion propagator and weight 2 is assigned to a photon propagator. QED Feynman diagrams are represented as edge lists as in the last section. Additionally, fermion propagators carry orientation information, therefore fermion edges are depicted as ordered pairs of vertices.

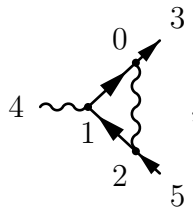
**Example** For example, the graph



is represented as

$$G[[0, 1, 1], [1, 0, 1], [2, 0, 2], [3, 1, 2]]$$

and



is depicted as

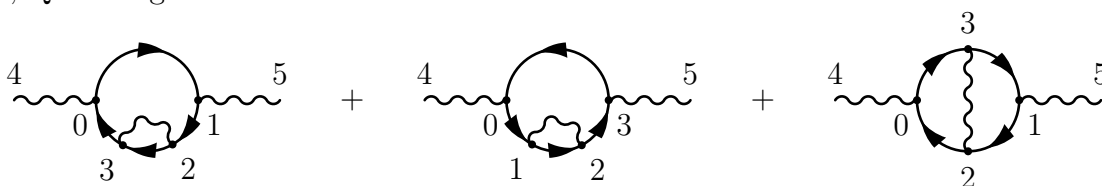
$$G[[1,0,1],[2,1,1],[2,0,2],[0,3,1],[5,2,1],[4,1,2]].$$

The orientation of the fermion lines matches the ordering of the vertices in the edges. **feyngen** only generates graphs with valid QED vertex types as described in definition 3.

### Output of graph sums

The output of graph sums is similar to the output of  $\varphi^k$  graphs. As for the treatment of  $\varphi^k$  graphs, an example for the case of QED diagram generation is given.

**Example** Consider the sum of all two loop, photon propagator residue type, 1PI, QED diagrams



**feyngen** generates them if it is called with the command line

```
$ ./feyngen --qed 2 -b2 -p
qed_f0_b2_h2 :=
+G[[0,1,1],[1,2,1],[2,3,1],[3,0,1],[3,2,2],[4,0,2],[5,1,2]]/1
+G[[0,1,1],[1,2,1],[2,3,1],[3,0,1],[2,1,2],[4,0,2],[5,3,2]]/1
+G[[0,3,1],[1,2,1],[2,0,1],[3,1,1],[3,2,2],[4,0,2],[5,1,2]]/1
;
```

**--qed** indicates QED graph generation, **2** stands for 2-loop diagrams ( $\hbar^2$ ), **-b2** makes **feyngen** generate graphs with 2 photon legs and the **-p** option filters out non 1PI graphs.

### Distinguished name for maple usage

The name of the graph sum for QED diagrams is slightly modified:

```
qed_f(m1)_b(m2)_h(L) :=
...
;
```

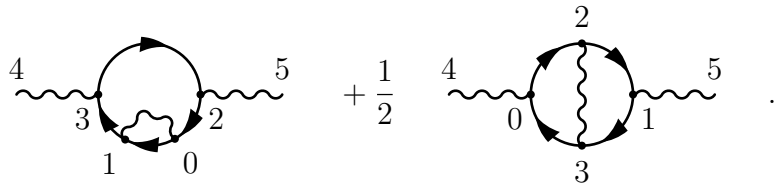
Instead of  $j(m)$  indicating the total number of legs as in the case of  $\varphi^k$  graphs,  $f(m1)$  and  $b(m2)$  are given with  $(m1)$  being replaced by the number of fermion legs and  $(m2)$  by the number of photon legs.

### 5.4.5 Labeled and unlabeled legs

Again, without the `-u` option the external legs are considered as fixed. For this reason the first and the second graphs in the last example are not isomorphic and all diagrams carry a symmetry factor of 1.

The `-u` option controls this behaviour and influences the generation of non-isomorphic graphs and the calculation of symmetry factors appropriately.

**Example** Consider again the two loop, photon propagator, 1PI, QED diagrams, but without fixed external legs,



Because of the additional freedom in permuting the edges and vertices, the first two diagrams of the last example belong to the same isomorphy class.

Furthermore, the last diagram gains an additional symmetry generator of index 2, such that the order of the automorphism group increases from 1 to 2.

This can be reproduced using `feynngen` with the `-u` option:

```
$ ./feynngen --qed 2 -b2 -p -u
qed_f0_b2_h2_unlab_ext_legs :=
+G[[0,1,1],[1,3,1],[2,0,1],[3,2,1],[1,0,2],[4,3,2],[5,2,2]]/1
+G[[0,2,1],[1,3,1],[2,1,1],[3,0,1],[3,2,2],[4,0,2],[5,1,2]]/2
;
```

The text `unlab_ext_legs` succeeding the name of the graph sum acknowledges this behaviour for later reproducibility.

# Chapter 6

## Coproduct computation

### 6.1 Overview

The python program **feyncop** can be used to compute the reduced coproduct  $\widetilde{\Delta}_D$  of given 1PI graphs as defined in (3.4). The output of **feynngen** can be piped into **feyncop** to calculate the reduced coproduct of all 1PI graphs of a given loop order and residue type.

By default, the subgraphs composed of superficially divergent, 1PI graphs of the input graphs are computed and given as output. These correspond to the left factor of the tensor product originating the coproduct. Optionally, the complementary cographs, giving account to the right factor of the tensor product, can be computed. Furthermore, there is the option to identify the sub- and cographs with unlabeled 1PI graphs.

Additionally, the input graphs can be filtered for primitive graphs.

The coproduct calculation does only take the degree of divergence obtained by power counting, formulated by the map  $\omega_D$  into account. Further information, as gained by Furry's theorem in the case of QED, is not used.

### 6.2 Sketch of the implementation of feyncop

First **feyncop** reads the input graphs, piped into the python program, in a manner compatible with **feynngen**.

Internally, graphs are represented as edge lists as in the implementation of **feynngen**. Optionally, a weight can be assigned to every edge. By default, a weight of 2 for every edge is assumed.

To calculate the coproduct, the subgraphs of the given graph are computed. They are tested for one particle irreducibility and superficial divergence of their connected components. Therefore, the computation relies on a fast 1PI

testing function build upon a DFS algorithm as for instance described in [4]. If required, the edges of the cograph are computed by shrinking the internal subgraph edges subsequently.

If the sub- and cographs are to be identified with full fledged Feynman graphs, two-valent vertices are removed from the cograph and external legs are added in accordance to the vertex types of the graph.

To identify the subgraphs with unlabeled ones the **nauty** package is used to compute a canonical labeling. Similar types of resulting tensor products are collected and given as output.

### 6.3 Check of validity

The reduced coproduct computation was checked for validity by testing the results for the coassociativity and the gradedness of the coproduct derived in chapter 3.

Additionally, the coproduct of a sum of 1PI graphs of a given graded class of graphs has been checked for accordance with theorem 1. This check also confirms the validity of the computation of 1PI diagrams by **feynngen**.

## 6.4 Manual of feynncop

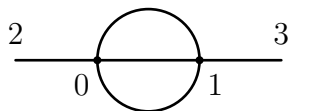
### 6.4.1 Overview

A graph or a sum of graphs, for which the reduced coproduct shall be calculated, must be piped as input into **feynncop**.

For example,

```
$ echo "G[[1,0],[1,0],[1,0],[2,0],[3,1]]" | ./feynncop -D4
```

where the **echo** command is used to pipe the input into **feynncop**, will give the three proper non empty subgraphs of the diagram

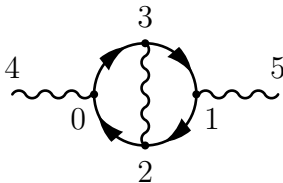


which are, in four dimensions, composed of superficially divergent 1PI graphs. Details to the output format will be given in section 6.4.4 and the following ones with elaborate examples.

QED graphs are handled the same way, except for the weights that must be given with the edges.



For instance, the subgraphs for the reduced coproduct  $\tilde{\Delta}_4$  of the graph



or given in the G-format as

```
G[[0,3,1],[1,2,1],[2,0,1],[3,1,1],[3,2,2],[4,0,2],[5,1,2]]
```

can be calculated using the command line

```
$ echo "G[[0,3,1],[1,2,1],[2,0,1],[3,1,1],[3,2,2],[4,0,2],[5,1,2]]"
| ./feynkop -D4
```

.

## 6.4.2 Option and Parameters

Similar to `feynngen`,

```
$ ./feynkop --help
```

prints a list of the available options and parameters with a short summary.

The parameter  $D$ , altering the dimension parameter of the reduced coproduct  $\Delta_D$ , is controlled by the option

**-D# / --dimension=#**

Set the dimension for the calculation of the coproduct.

By default  $D = 4$  is assumed.

By default, `feynkop` calculates only the subgraphs, which are composed of superficially divergent 1PI graphs, of the input graphs. This behaviour can be changed by the options:

**-c / --cographs**

Calculate and print the cographs additionally to the corresponding subgraphs. Output format: Sum of graphs with a list of the subgraphs, composed of superficially divergent, 1PI graphs, and their cograph.

**-u / --unlabeled**

Transform the subgraphs and the cographs to unlabeled graphs and identify similar tensor products. Output format: Sum of tensor products.

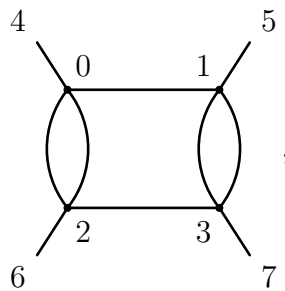
**-p / --primitives**

Only filter the input graphs for primitive graphs. Output format: Sum of graphs.

### 6.4.3 Reference to edges

`feynco` represents subgraphs as a set of edges of the original graph. This has the advantage that the location of the subgraph in the original graph is implicitly included in the output. This information is crucial for some applications of the coproduct of a Feynman diagram. The edges are referred to by their index in the edge list of the `G`-format, as described in the last chapter, starting with 0.

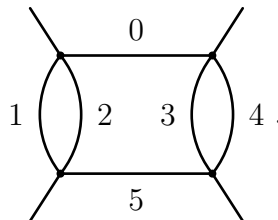
**Example** The graph, with an auxiliary vertex labeling,



represented in the `G`-format by

$$G[[1,0], [2,0], [2,0], [3,1], [3,1], [3,2], [4,0], [5,1], [6,2], [7,3]],$$

will be assigned the following internal edge labels,



Here, labels for external legs were omitted for simplicity. They are not of interest for the description of subgraphs.

### 6.4.4 Output of subgraphs

By default, only the subgraphs composed of superficially divergent 1PI graphs are outputted. The output is given as pairs of the original graph in the `G`-format and of the subgraphs, each split into its connected components.


These pairs are preceded by an `D` to mark an new object suitable to be read by `maple`.

Therefore, giving a single graph as input, the output will take the form:

```

D[ ( original graph ), [
{ { 1. subgraph's 1. connected component's edges },
{ 1. subgraph's 2. connected component's edges },
...
},
{ { 2. subgraph's 1. connected component's edges }, ... }
...
] ]

```

**Example** Taking again the graph , represented in the G-format as above, the output to the call,

```

$ echo "G[[1,0],[2,0],[2,0],[3,1],[3,1],[3,2],
[4,0],[5,1],[6,2],[7,3]]" | ./feynccop -D4

```

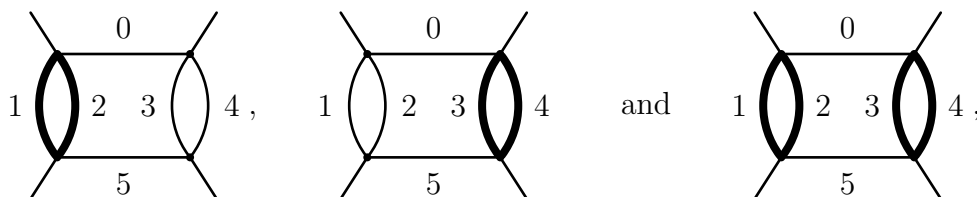
will look as follows:

```

+ D[G[[1,0],[2,0],[2,0],[3,1],[3,1],[3,2],[4,0],[5,1],[6,2],[7,3]],
[{{1,2}}, {{3,4}}, {{1,2},{3,4}}]]
;

```

The original graph is paired with the information about the subgraphs composed of superficially divergent components in  $[{\{1,2\}}, {\{3,4\}}, {\{1,2\},\{3,4\}}]$ . The three sets in the list correspond to the three subgraphs, indicated by thick lines,



represented as the sets of sets,

$\{\{1,2\}\}$ ,  $\{\{3,4\}\}$  and  $\{\{1,2\},\{3,4\}\}$ .

The subgraphs are split into their connected components and every connected component is given as a set of edge references.

### 6.4.5 Output of cographs


Called with the **-c** option, **feynccop** also outputs the cographs for the reduced coproduct.

With this option the output is a triple with the original graph, the subgraphs as described above and with the corresponding cograph in the G-format.

The contracted edges of the cographs are not removed from the original edge list, but replaced by the dummy edge  $[-1,-1]$  to simplify reference to the edges by index.

The output will be of the form

```
D[ ( original graph ), [
[ { { 1. subgraph's 1. connected component's edges },
{ 1. subgraph's 2. connected component's edges },
...
},
( cograph to 1. subgraph ) ],
[ { { 2. subgraph's 1. connected component's edges }, ... },
( cograph to 2. subgraph ) ],
...
] ].
```

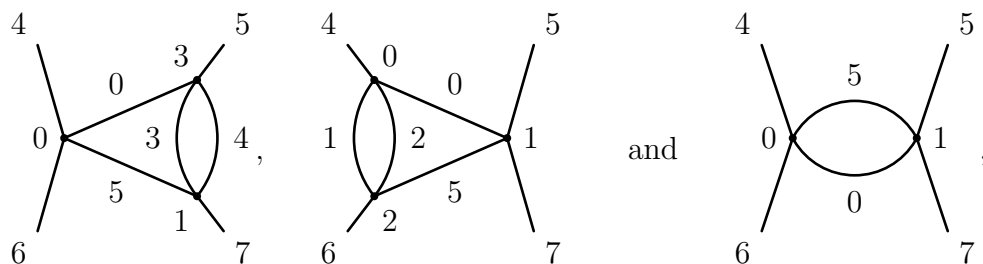
**Example** With  as input,

```
$ echo "G[[1,0],[2,0],[2,0],[3,1],[3,1],[3,2],
[4,0],[5,1],[6,2],[7,3]]" | ./feyncop -D4 -c
```

the following outcome will be produced:

```
+ D[G[[1,0],[2,0],[2,0],[3,1],[3,1],[3,2],
[4,0],[5,1],[6,2],[7,3]], [
[{{1,2}},
G[[1,0],[-1,-1],[-1,-1],[3,1],[3,1],[3,0],
[4,0],[5,1],[6,0],[7,3]]],
[{{3,4}},
G[[1,0],[2,0],[2,0],[-1,-1],[-1,-1],[1,2],
[4,0],[5,1],[6,2],[7,1]]],
[{{1,2},{3,4}},
G[[1,0],[-1,-1],[-1,-1],[-1,-1],[-1,-1],[1,0],
[4,0],[5,1],[6,0],[7,1]]]]
]
;
```

The cographs, with vertex and edge labeling, corresponding to the subgraphs in the last section



are represented in the G-format as

```
G[[1,0],[−1,−1],[−1,−1],[3,1],[3,1],[3,0],
[4,0],[5,1],[6,0],[7,3]]
```

,

```
G[[1,0],[2,0],[2,0],[−1,−1],[−1,−1],[1,2],
[4,0],[5,1],[6,2],[7,1]]
```

and

```
G[[1,0],[−1,−1],[−1,−1],[−1,−1],[−1,−1],[1,0],
[4,0],[5,1],[6,0],[7,1]].
```


During the calculation of the cographs, vertices are removed. Therefore, the vertex labeling of the cographs is not compatible with the one of the original graph. Whenever two vertices are merged, the new vertex will carry the smaller label.

### 6.4.6 Output of tensor products

Called with the option **-u**, **feynccop** identifies the subgraphs and cographs with unlabeled graphs, groups them in tensor products and sums tensor products of similar form.

Called with this option **feynccop** will handle all graphs as non-leg-fixed graphs. Giving leg-fixed graphs as input will result in less terms with higher factors than expected.

The tensor products are given as pairs of a product of superficially divergent connected components of the subgraphs and the cograph. The pairs are preceded by an T to indicate the tensor product type of output. The output is a sum of these tensor products.

**Example** Giving again  as input,

```
$ echo "G[[1,0],[2,0],[2,0],[3,1],[3,1],[3,2],[4,0],[5,1],[6,2],[7,3]]
" | ./feynccop -D4 -u
```

the output will be:

```
+ 2/1 * T[ G[[1,0],[1,0],[2,0],[3,0],[4,1],[5,1]],
G[[1,0],[1,0],[2,0],[2,1],[3,2],[4,2],[5,0],[6,1]] ]
+ T[ (G[[1,0],[1,0],[2,0],[3,0],[4,1],[5,1]])^2,
G[[1,0],[1,0],[2,0],[3,0],[4,1],[5,1]] ]
;
```

This output corresponds to the reduced coproduct calculation,

$$\tilde{\Delta}_4(\text{Diagram}) = 2 \text{Diagram} \otimes \text{Diagram} + (\text{Diagram})^2 \otimes \text{Diagram}.$$

Where,

$$2 \text{Diagram} \otimes \text{Diagram},$$

is represented as

```
2/1 * T[ G[[1,0],[1,0],[2,0],[3,0],[4,1],[5,1]],
G[[1,0],[1,0],[2,0],[2,1],[3,2],[4,2],[5,0],[6,1]] ]
```

and

$$(\text{Diagram})^2 \otimes \text{Diagram},$$

is denoted as

```
T[ (G[[1,0],[1,0],[2,0],[3,0],[4,1],[5,1]])^2,
G[[1,0],[1,0],[2,0],[3,0],[4,1],[5,1]] ].
```

Note that the labelings of the vertices of the subgraphs and cographs in this mode of **feyncop** carry no resemblance to the vertex labeling of the input graph. The labelings are again chosen using **nauty**'s canonical labeling algorithm. Therefore, the graphs are representatives of the corresponding graph isomorphy class.

### 6.4.7 Usage in conjunction with feynngen

**feyncop** can also be used in conjunction with **feynngen**. The coproduct of every input graph is computed and the sum of the coproducts in a format depending on the options is given as output. The output of **feynngen** can be piped into **feyncop** as input.

The instance,

```
$ ./feynngen 2 -j2 -k4 -pu | ./feyncop -D4
```

will yield the sum of the reduced coproducts, given in the D-format, of all non-leg-fixed two loop,  $\varphi^4$ , 1PI graphs with two external legs weighted by the symmetry factor of the original graph:

```

+ 1/8 * D[G[[1,0],[1,0],[1,1],[2,0],[3,0]],
  [{{2}}, {{0,1}}]]
+ 1/12 * D[G[[1,0],[1,0],[1,0],[2,0],[3,1]],
  [{{0,1}}, {{0,2}}, {{1,2}}]]
;

```

Where the relevant graph sum with edge and vertex labels, of which the coproduct is calculated, is:

$$\frac{1}{8} \left( \begin{array}{c} \text{Diagram 1: Two stacked circles. Top circle has vertex 2 at top and 1 at bottom. Bottom circle has vertex 1 at top and 0 at bottom. A horizontal line connects vertex 2 on the left to vertex 0 on the right. A curved line connects vertex 1 on the left to vertex 3 on the right. Labels: 2 (top), 1 (middle), 0 (bottom), 2 (left), 3 (right), 0 (bottom-left), 1 (bottom-right).} \end{array} \right) + \frac{1}{12} \left( \begin{array}{c} \text{Diagram 2: A circle with vertex 2 at top and 0 at bottom. A horizontal line passes through the center with vertex 0 on the left and vertex 1 on the right. Two horizontal lines extend from vertex 0 to vertex 2 and from vertex 1 to vertex 3. Labels: 2 (top), 0 (middle-left), 1 (middle-right), 0 (bottom), 2 (left), 3 (right).} \end{array} \right)$$

On the other hand,

```

$ ./feyngen 2 -j2 -k4 -pu | ./feyncop -D4 -u

```

will yield the sum of tensor products of unlabeled graphs corresponding to the reduced coproduct applied to the sum of all two loop,  $\varphi^4$ , 1PI graphs with two external legs weighted by their symmetry factor:

```

phi4_j2_h2_unlab_ext_legs_reduced_coproduct_unlabeled :=
+ 1/8 * T[ G[[0,0],[1,0],[2,0]], G[[0,0],[1,0],[2,0]] ]
+ 3/8 * T[ G[[1,0],[1,0],[2,0],[3,0],[4,1],[5,1]],
  G[[0,0],[1,0],[2,0]] ]
;

```

This corresponds to the calculation

$$\tilde{\Delta}_4 \left( \frac{1}{8} \text{Diagram 1} + \frac{1}{12} \text{Diagram 2} \right) = \frac{1}{8} \text{Diagram 3} \otimes \text{Diagram 4} + \frac{3}{8} \text{Diagram 5} \otimes \text{Diagram 6},$$

which can be validated using the identity (3.33) as shown in the example in section 3.3.5.

### 6.4.8 Filtering for primitive graphs

Additionally, **feyncop** has the ability to filter the input graphs for primitive ones. This behaviour is triggered by the **-p** option.

A convenient usage pattern is to pipe the output of **feynngen** of a larger group of graphs into **feyncop** to obtain a set of primitive graphs with the desired properties.

**Example** If all primitive, QED, vertex diagrams with three loops are desired, the call to **feynngen**,

```
$ ./feynngen 3 --qed -b1 -f2 -p
```

to generate the 100 not necessarily primitive QED diagrams is needed.

To filter these diagrams for primitive ones, they can be piped into **feyncop**:

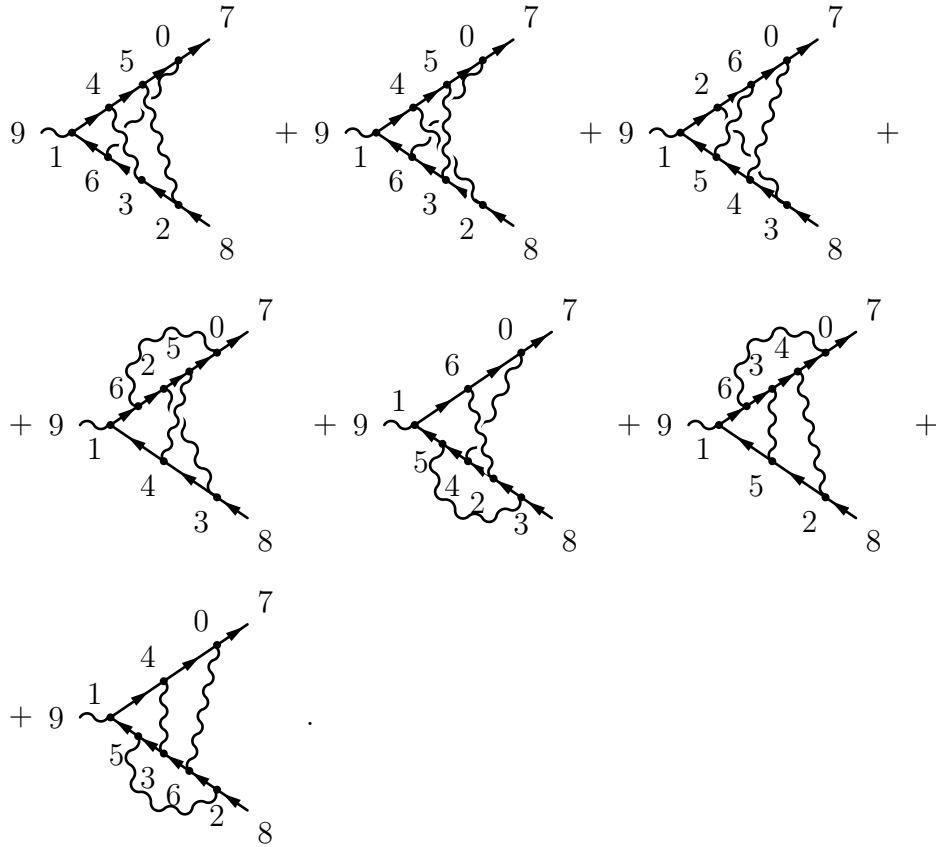
```
$ ./feynngen 3 --qed -b1 -f2 -p | ./feyncop -D4 -p
```

This call will result in the output

```
qed_f2_b1_h3_proj_to_prim :=
+ G[[1,4,1], [2,3,1], [3,6,1], [4,5,1], [5,0,1], [6,1,1], [4,3,2],
  [5,2,2], [6,0,2], [0,7,1], [8,2,1], [9,1,2]]
+ G[[1,4,1], [2,3,1], [3,6,1], [4,5,1], [5,0,1], [6,1,1], [4,2,2],
  [5,3,2], [6,0,2], [0,7,1], [8,2,1], [9,1,2]]
+ G[[1,2,1], [2,6,1], [3,4,1], [4,5,1], [5,1,1], [6,0,1], [3,2,2],
  [4,0,2], [6,5,2], [0,7,1], [8,3,1], [9,1,2]]
+ G[[1,6,1], [2,5,1], [3,4,1], [4,1,1], [5,0,1], [6,2,1], [3,2,2],
  [5,4,2], [6,0,2], [0,7,1], [8,3,1], [9,1,2]]
+ G[[1,6,1], [2,4,1], [3,2,1], [4,5,1], [5,1,1], [6,0,1], [4,0,2],
  [5,3,2], [6,2,2], [0,7,1], [8,3,1], [9,1,2]]
+ G[[1,6,1], [2,5,1], [3,4,1], [4,0,1], [5,1,1], [6,3,1], [4,2,2],
  [5,3,2], [6,0,2], [0,7,1], [8,2,1], [9,1,2]]
+ G[[1,4,1], [2,6,1], [3,5,1], [4,0,1], [5,1,1], [6,3,1], [4,3,2],
  [5,2,2], [6,0,2], [0,7,1], [8,2,1], [9,1,2]]
;
```



corresponding to the sum of the seven primitive, three loop, vertex diagrams in QED:





# Chapter 7

## Conclusion and future prospects

In the scope of this thesis, the program **feynngen** to generate Feynman graphs of large loop orders and the program **feyncop** to calculate the coproduct of given Feynman graphs were developed. These programs can be downloaded from:

<http://people.physik.hu-berlin.de/~borinsky/feyncop/>

In a future work, **feynngen** and **feyncop** could be expanded to handle graphs of further types of quantum field theories as for instance non-abelian Yang-Mills theory or spontaneously broken theories. The performance of **feynngen** to generate graphs with external legs and QED graphs could still be increased significantly by taking the automorphism groups of the graphs into account during the graph generation.

From a larger perspective, the next step will be to implement a program that evaluates the renormalized amplitudes of 1PI graphs by parametric integration techniques. First steps in this endeavour are pursued in [12].

I would like to thank Dirk Kreimer for his great supervision and for his colorful way of teaching that I could enjoy in the last years. I wish to express my grateful thanks to Oliver Schnetz for his great advice and steady assistance. For numerous illuminating discussions about physics, life and everything else, I would like to thank Olaf Krüger, Bettina Grauel, Marko Berghoff, Eric Panzer, Lutz Klaczynski, Matthias Sars and everyone else from our group who made the writing of this thesis such an enjoyable task.



# Bibliography

- [1] B. Bollobás. *Modern Graph Theory*. Graduate Texts in Mathematics. Springer-Verlag GmbH, 1998.
- [2] Francis Brown and Dirk Kreimer. Angles, scales and parametric renormalization. *Letters in Mathematical Physics*, 103(9):933–1007, 2013.
- [3] Alain Connes and Dirk Kreimer. Renormalization in quantum field theory and the riemann–hilbert problem i: The hopf algebra structure of graphs and the main theorem. *Communications in Mathematical Physics*, 210(1):249–273, 2000.
- [4] Thomas H. Cormen, Clifford Stein, Ronald L. Rivest, and Charles E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001.
- [5] Matthias Sars Dirk Kreimer and Walter Suijlekom. Quantization of gauge fields, graph polynomials and graph cohomology. *arXiv.org: hep-th/1208.6477*, 2012.
- [6] Kurusch Ebrahimi-Fard and Dirk Kreimer. The hopf algebra approach to feynman diagram calculations. *Journal of Physics A: Mathematical and General*, 38(50):R385, 2005.
- [7] P. Flajolet and R. Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.
- [8] Claude Itzykson and Jean-Bernard Zuber. *Quantum field theory*. International series in pure and applied physics. McGraw-Hill, New York, NY, 1980. Also a reprint ed.: Mineola, Dover, 2005.
- [9] Dominique Manchon. Hopf algebras, from basics to applications to renormalisation, rencontres mathématiques de glanon. *arXiv.org: math/0408405v2*, 2003.

- [10] Brendan D. McKay. Practical graph isomorphism. In *10th. Manitoba Conference on Numerical Mathematics and Computing; Congressus Numerantium*, 30, pages 45–87. Department of Computer Science, Vanderbilt University, 1981.
- [11] Brendan D McKay. Isomorph-free exhaustive generation. *Journal of Algorithms*, 26(2):306–324, 1998.
- [12] Erik Panzer. On the analytic computation of massless propagators in dimensional regularization. *arXiv.org: hep-th/1305.2161*, 2013.
- [13] WalterD. Suijlekom. Renormalization of gauge fields: A hopf algebra approach. *Communications in Mathematical Physics*, 276(3):773–798, 2007.
- [14] H.S. Wilf. *Generating Functionology*. Ak Peters Series. A K Peters, 2006.

# Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Abschlussarbeit selbständig und nur mit den angegebenen Hilfsmitteln verfasst habe.

Ich erkläre ausdrücklich, dass ich sämtliche in der Arbeit verwendeten fremden Quellen, auch aus dem Internet, als solche kenntlich gemacht habe. Insbesondere bestätige ich, dass ich ausnahmslos sowohl bei wörtlich übernommenen Aussagen bzw. unverändert übernommenen Tabellen, Grafiken u. Ä. (Zitaten) als auch bei in eigenen Worten wiedergegebenen Aussagen bzw. von mir abgewandelten Tabellen, Grafiken u. Ä. anderer Autorinnen und Autoren (indirektes Zitieren) die Quelle angegeben habe.

Mir ist bewusst, dass Verstöße gegen die Grundsätze der Selbstständigkeit als Täuschung betrachtet und entsprechend der Prüfungsordnung und/oder der Allgemeinen Satzung für Studien- und Prüfungsangelegenheiten der HU (ASSP) geahndet werden.

Die Arbeit wurde in gleicher oder ähnlicher Form bisher bei keiner anderen Institution eingereicht.

---

Ort, den (Datum)

Unterschrift