On the Kernel of the Symbol Map for Multiple Polylogarithms

RHODES, JOHN,RICHARD

On the Kernel of the Symbol Map for Multiple Polylogarithms

John Richard Rhodes

A Thesis presented for the degree of
Doctor of Philosophy

Pure Mathematics
Department of Mathematical Sciences
Durham University

2012

Abstract

The *symbol map* (of Goncharov) takes multiple polylogarithms, $I_{r_1,\ldots,r_s}(x_1,\ldots,x_s)$, to a tensor product space where calculations are easier, but where important differential and combinatorial properties of the *multiple polylogarithm* are retained. Finding linear combinations of multiple polylogarithms in the kernel of the symbol map is an effective way to attempt finding functional equations. We present and utilise methods for finding new linear combinations of multiple polylogarithms (and specifically *harmonic polylogarithms*) that lie in the kernel of the symbol map.

During this process we introduce a new pictorial construction for calculating the symbol, namely the *hook-arrow tree*, which can be used to easier encode symbol calculations onto a computer.

We also show how the hook-arrow tree can simplify symbol calculations where the depth of a multiple polylogarithm is lower than its weight and give explicit expressions for the symbol of depth 2 and 3 multiple polylogarithms of any weight. Using this we give the full symbol for $I_{2,2,2}(x,y,z)$. Through similar methods we also give the full symbol of *coloured multiple zeta values.*

We provide introductory material including the binary tree (of Goncharov) and the polygon dissection (of Gangl, Goncharov and Levin) methods of finding the symbol of a multiple polylogarithm, and give bijections between (adapted forms of) these methods and the hook-arrow tree.

Declaration

The work in this thesis is based on research carried out in the Pure Mathematics Group at the Department of Mathematical Sciences, Durham University. No part of this thesis has been submitted elsewhere for any other degree or qualification and it is all my own work unless referenced to the contrary in the text.

Acknowledgements

Although many would say it about themselves, I truly believe I am a very particular kind of mathematician; one that might not suit some supervisors. Thank you, Herbert, for communicating with me in a way that best suits my style. I have always very much enjoyed our (often very long) conversations; the mathematical parts were stimulating and the technology parts good fun. I truly am very grateful for your patience and support.

Communication in the mathematics department is something it excels in. I have very much enjoyed the small pure maths postgraduate community that has been kept going by the GandAlF seminar. I also would like to thank Claude for some very interesting talks concerning links between my work and the physics community.

Non-mathematical communication within the department is important to keeping sanity levels reasonable. As many PhD students of Durham will know, coffee club is essential to success. The '10.30 club' is something I have frequently fought for and its most loyal members are garnered with a special medal.

Outside of the mathematics department I have been lucky to have a great set of friends providing me with a tremendous support network. I have always felt like I had people to turn to. The times spent drinking coffee, making dinners and in the pub have been incredibly important to my studies. It is hard to precisely define it in words, but the way in which we discuss trivial matters is something that I never want to be without.

So, for the above reasons and many more, my thanks go to Alex, Ben, Ben, Ben, Caroline, Chris, Dave, Elise, Harriet, Helen, Ian, Jack, James, James, Joey, John, Josh, Kirsty, Luke, Mel, Nathan, Pamela, Rachel, Sarah, Scott, Simon, Steven, and Ric.

Finally, special words of thanks to Mum, Dad, Anne-Marie, and Nan for talking to me on the phone, and of course, everything.

Contents

Chapter 0

Introduction

0.1 Opening remarks

Multiple polylogarithms, a natural generalisation of the logarithm, are an important class of functions which have applications in both mathematics and physics. There exist well-known functional equations between multiple polylogarithms, but current understanding is far from complete. There is much demand for expanding knowledge on a set of generators of polylogarithmic functional equations, which are extremely hard to find. A valuable tool to aid such exploration is the *symbol map* of Goncharov (given first under the name '$\otimes^m$-invariant' in [Gon05]), which takes a linear combination of multiple polylogarithms to an algebraic construction (involving tensor products) which can be manipulated significantly easier than analytic manipulation of the functions themselves. Conjecturally, any functional equation on multiple polylogarithms must lie in the kernel of the symbol map.

Through exploration of past methods and the development of the 'hook-arrow tree' (a new pictorial representation of the symbol) our contribution concerns the kernel of the symbol map with examples of different multiple polylogarithmic elements for several weights. We use the hook-arrow tree to give an explicit description of the symbol of depth 2 and 3 multiple polylogarithms of any weight. We give a full description of the symbol of coloured multiple zeta values, a specialised class of multiple polylogarithms (this appears in our preprint [DGR11] with C Duhr and H Gangl).

Polylogarithms appear in mathematics in a surprisingly large range of areas. These include serving as volume functions for hyperbolic spaces ([ZN85], [Zag86], and [ZG00]) and are connected to Algebraic K-Theory ([Zag91], [Zag90], and [Blo00]) as 'higher regulators' (generalising the natural logarithm's role in the regulator of an algebraic number field, see [BS66]). Algebraic cycles that 'represent' polylogarithms (often referred to as Bloch-Kriz-Totaro cycles) have been extensively studied ([BK95], [GMS99], and [GGL09]). Multiple polylogarithms are strongly linked to periods of moduli spaces ([GM04] and [Bro09]). An important paper of Goncharov introducing the symbol formally is [Gon09]. Further relevant papers concerning symbolic calculations for polylogarithms and their generalisations are [Bro11], [Gon95], [Gan03], [Gan10] and [Zha04].

Functional equations of multiple polylogarithms are not only of considerable interest to mathematicians, but also to the physics community due partly to the need to simplify the calculation of Feynman integrals that can be written in terms of (special classes of) multiple polylogarithms ([MUW02], [VW05], and for a discussion including this link, see [WB09]). We give new functional equations of weight $4, 5, 6, 7$ and 8 for one such class, namely harmonic polylogarithms ([RV00]). The symbol has also, more recently, become prominent in the context of $\mathcal{N} = 4$ Super Yang-Mills theory ([GSVV10]). This is the topic of much research and collaboration between the mathematics and physics communities ([GSVV10], [DDDS10], [HK11], [DGR11], and [Duh12]).

0.2 Definition of multiple polylogarithms

We now define the multiple polylogarithm and give a well-known motivating relation (shown analytically, rather than with the use of symbols).

As mentioned above, a multiple polylogarithm is a natural generalisation of a polylogarithm, which is itself a natural generalisation of a logarithm.

Definition 0.1. The m-th **polylogarithm**, a generalisation of the logarithm is defined to be

$$\mathrm{Li}_m(z) = \sum_{n=1}^{\infty} \frac{z^n}{n^m} \qquad z \in \mathbb{C},\ |z| < 1.$$

Indeed, we see that $\mathrm{Li}_1(z) = -\ln(1-z)$. An excellent, and very readable, article on the dilogarithm, $\mathrm{Li}_2(z)$, by Zagier is [Zag88].

Remark 0.2. We note at this point, by way of a disclaimer, that we do not concentrate in this thesis on the multivaluedness of the polylogarithm (the reader is referred to [Woj02] and [Zha07]). Rather, we are interested in its algebraic properties under the, to be defined, symbol map.

We motivate the definition of a multiple polylogarithm by observing the product of two polylogarithms,

$$\mathrm{Li}_s(x)\mathrm{Li}_t(y) = \sum_{0<m} \frac{x^m}{m^s} \sum_{0<n} \frac{y^n}{n^t}. \tag{1}$$

Each term will take the form

$$\frac{x^i y^j}{i^s j^t}.$$

By splitting into cases where $i < j, i = j$, and $i > j$ we see Equation 1 becomes

$$\mathrm{Li}_s(x)\mathrm{Li}_t(y) = \sum_{0<m<n} \frac{x^m y^n}{m^s n^t} + \sum_{0<m=n} \frac{x^m y^n}{m^s n^t} + \sum_{0<n<m} \frac{x^m y^n}{m^s n^t}.$$

We now relabel in the last summation and, noticing that the second term is a polylogarithm, we get

$$\mathrm{Li}_s(x)\mathrm{Li}_t(y) = \sum_{0<m<n} \frac{x^m y^n}{m^s n^t} + \mathrm{Li}_{s+t}(xy) + \sum_{0<m<n} \frac{y^m x^n}{m^t n^s}.$$

This motivates introducing a multiple polylogarithm, as defined in [Gon97].

Definition 0.3. A weight w **multiple polylogarithm**, $\mathrm{Li}_{r_1,\ldots,r_s}(z_1,\ldots,z_s)$ is defined to be

$$\mathrm{Li}_{r_1,\ldots,r_s}(z_1,\ldots,z_s) := \sum_{0<n_1<\ldots<n_s} \frac{z_1^{n_1}\cdots z_s^{n_s}}{n_1^{r_1}\cdots n_s^{r_s}} \qquad z_i \in \mathbb{C},\ |z_i| < 1$$

where

$$w = \sum_{i=1}^{s} r_i.$$

Equation 1 becomes the aesthetically pleasing and well known relation (it is seen, for $s = t = 1$ in the previously mentioned paper, [Gon97])

$$\mathrm{Li}_s(x)\mathrm{Li}_t(y) = \mathrm{Li}_{s,t}(x, y) + \mathrm{Li}_{s+t}(xy) + \mathrm{Li}_{t,s}(y, x).$$

This is an example of a *stuffle relation* ('shuffle and stuff'), so named in the literature because the variables x and y are 'shuffled' (when $i < j$ and $i > j$), but we also get 'stuff' (when $i = j$). The reader is directed to pages 12-16 of [BBBL01] where a 'stuffle algebra' is discussed (and in particular example Example 5.1 of that paper shows a good example of a stuffle relation on multiple zeta values).

Remark 0.4. Not only does this relation nicely introduce multiple polylogarithms, but it is an example of a polylogarithmic relation with an analytic proof. The proof here is relatively simple. However, analytical methods do not scale well for finding all relations between polylogarithms (which motivates the *symbol map* later).

Remark 0.5. We can still find more relations using the above idea. The product of three polylogarithms will give the following relation.

$$\begin{aligned}\prod_{i=1}^{3} Li_{r_i}(z_i) = \quad & Li_{r_1,r_2,r_3}(z_1, z_2, z_3) + Li_{r_1,r_3,r_2}(z_1, z_3, z_2) + Li_{r_2,r_1,r_3}(z_2, z_1, z_3) \\ + \quad & Li_{r_2,r_3,r_1}(z_2, z_3, z_1) + Li_{r_3,r_1,r_2}(z_3, z_1, z_2) + Li_{r_3,r_2,r_1}(z_3, z_2, z_1) \\ + \quad & Li_{r_1,(r_2+r_3)}(z_1, z_2z_3) + Li_{r_2,(r_1+r_3)}(z_2, z_1z_3) + Li_{r_3,(r_1+r_2)}(z_3, z_1z_2) \\ + \quad & Li_{(r_1+r_2),r_3}(z_1z_2, z_3) + Li_{(r_1+r_3),r_2}(z_1z_3, z_2) + Li_{(r_2+r_3),r_1}(z_2z_3, z_1) \\ + \quad & Li_{r_1+r_2+r_3}(z_1z_2z_3).\end{aligned}$$

0.2.1 Other definitions of multiple polylogarithms

There exist different forms of the definition of multiple polylogarithms, which are given different notation in literature. Definition 0.3 has been given first as it is the one that most intuitively follows from logarithms. However, in this thesis we will use a definition of a multiple polylogarithm defined using an iterated integral. First we define a multiple logarithm.

Definition 0.6. A **multiple logarithm** is defined by an iterated integral, for $x_i \in \mathbb{C}$, to be

$$I(x_0; x_1, \ldots, x_m; x_{m+1}) = \int_{x_0 \leq t_1 \leq \ldots \leq t_m \leq x_{m+1}} \frac{dt_1}{t_1 - x_1} \wedge \ldots \wedge \frac{dt_m}{t_m - x_m}.$$

Definition 0.7. A **weight w multiple polylogarithm**, $I_{r_1,\ldots,r_s}(x_1, \ldots, x_s)$ is defined, in terms of a multiple logarithm, to be

$$I_{r_1,\ldots,r_s}(x_1, \ldots, x_s) := I(0; x_1, \underbrace{0, \ldots, 0}_{r_1-1}, x_2, \underbrace{0, \ldots, 0}_{r_2-1}, \ldots, x_s, \underbrace{0, \ldots, 0}_{r_s-1}; 1).$$

The above definition of a multiple logarithm and multiple polylogarithm appear in [Gon98] on page 5.

Remark 0.8. We note that on page 8 of [Gon01] that a multiple polylogarithm is sometimes known as a hyperlogarithm. We also see from the same paper that since $I(x_0; x_1, \ldots, x_m; x_{m+1})$ is invariant under the affine transformation $x_i \to \alpha x_i + \beta$ we have

$$\begin{aligned} I(x_0; x_1, \ldots, x_m; x_{m+1}) &= I(x_0 - x_0; x_1 - x_0, \ldots, x_m - x_0; x_{m+1} - x_0) \\ &= I(0; a_1, \ldots, a_m; a_{m+1}) \end{aligned}$$

for $a_i = x_i - x_0$.

Another definition of a multiple polylogarithm, which is often the notation used in physics is the function $G(a_1, \ldots, a_n; x)$ (where the 'G' is either a reference to 'Goncharov' or 'generalised'). It is almost identical to the definition of $I(x_0; x_1, \ldots, x_m; x_{m+1})$ and $I_{r_1,\ldots,r_s}(x_1, \ldots, x_s)$ but has reordered variables.

Definition 0.9. We define

$$\begin{aligned} G(a_1, \ldots, a_n; x) &:= I(0; a_n, \ldots, a_1; x) \\ G_{m_1,\ldots,m_k}(t_1, \ldots, t_k) &:= G(\underbrace{0, \ldots, 0}_{m_1-1}, t_1, \ldots, \underbrace{0, \ldots, 0}_{m_k-1}, t_k; 1) \\ &= I_{m_k,\ldots,m_1}(t_k, \ldots, t_1). \end{aligned}$$

The functions G and I are also very similar to Li. We give the following Theorem from pages 9-10 of [Gon01] to establish the link (a similar proof, specifically for $\mathrm{Li}_{1,1}(x,y)$ appears in [Gon97]).

Theorem 0.10.

$$\begin{aligned} Li_{m_1,\ldots,m_k}(x_1,\ldots,x_k) &= (-1)^k\, I_{m_1,\ldots,m_k}\left(\frac{1}{x_1\cdots x_k},\ldots,\frac{1}{x_k}\right) \\ &= (-1)^k\, G_{m_k,\ldots,m_1}\left(\frac{1}{x_k},\ldots,\frac{1}{x_1\cdots x_k}\right) \end{aligned}$$

Proof. (Sketch) We would like to prove, for $w=\sum m_i$ that

$$\begin{aligned} &\mathrm{Li}_{m_1,\ldots,m_k}(x_1,\ldots,x_k) \\ &\quad = (-1)^k \int_{0\le t_1\le\ldots\le t_m\le 1} \frac{dt_1}{t_1-(x_1\cdots x_k)^{-1}} \wedge \underbrace{\frac{dt_2}{t_2}\wedge\cdots\wedge\frac{dt_{m_1}}{t_{m_1}}}_{m_1-1}\wedge\ldots \\ &\qquad\qquad \ldots\wedge\frac{dt_{(w-m_k+1)}}{t_{(w-m_k+1)}-x_k^{-1}}\wedge\underbrace{\frac{dt_{(w-m_k+2)}}{t_{(w-m_k+2)}}\wedge\cdots\wedge\frac{dt_w}{t_w}}_{m_k-1}. \end{aligned}$$

We now use

$$\frac{dt}{t-a} = -\frac{1}{a}\left(\frac{1}{1-\frac{t}{a}}\right)dt = -\frac{1}{a}\sum_{m\ge 0}\left(\frac{t}{a}\right)^m dt = -\sum_{m\ge 1}\left(\frac{t}{a}\right)^m\frac{dt}{t}$$

and integrate the right hand side termwise. For example, for $k=2$, the right hand side,

$$\begin{aligned} (-1)^2\int_{0\le t_1\le\cdots\le t_{m_1+m_2}\le 1} &\frac{dt_1}{t_1-(x_1x_2)^{-1}}\wedge\underbrace{\frac{dt_2}{t_2}\wedge\ldots\wedge\frac{dt_{m_1}}{t_{m_1}}}_{m_1-1} \\ &\wedge\frac{dt_{m_1+1}}{t_{m_1+1}-x_2^{-1}}\wedge\underbrace{\frac{dt_{m_1+2}}{t_{m_1+2}}\wedge\cdots\wedge\frac{dt_{m_1+m_2}}{t_{m_1+m_2}}}_{m_2-1}, \end{aligned}$$

becomes

$$\begin{aligned} (-1)^4\int_{0\le t_1\le\cdots\le t_{m_1+m_2}\le 1} &\sum_{n_1=1}^{\infty}(x_1x_2t_1)^{n_1}\underbrace{\frac{dt_1}{t_1}\wedge\ldots\wedge\frac{dt_{m_1}}{t_{m_1}}}_{m_1-1} \\ &\wedge\sum_{n_2=1}^{\infty}(x_2t_{m_1+1})^{n_2}\underbrace{\frac{dt_{m_1+1}}{t_{m_1+1}}\wedge\cdots\wedge\frac{dt_{m_1+m_2}}{t_{m_1+m_2}}}_{m_2-1}. \end{aligned}$$

After integration termwise this becomes

$$\sum_{n_1,n_2>0} \frac{x_1^{n_1} x_2^{n_1+n_2}}{n_1^{m_1}(n_1+n_2)^{m_2}} = \sum_{r_2>r_1>0} \frac{x_1^{r_1} x_2^{r_2}}{r_1^{m_1} r_2^{m_2}} = \mathrm{Li}_{m_1,m_2}(x_1,x_2).$$

The proof for $k > 2$ works in an entirely similar way. □

Remark 0.11. We will mainly use the function I throughout this thesis and refer to it as a multiple polylogarithm (as justified by Theorem 0.10). We use I for reasons of simplicity that become apparent in Chapter 1 when attaching a labelled polygon to a multiple polylogarithm.

0.2.2 Linear combinations of multiple polylogarithms

The main objects of this thesis are linear combinations of multiple polylogarithms with rational coefficients. We define these to lie in an algebra $\mathcal{I}_\bullet(S)$ for a set S (as Goncharov gives on page 9 of [Gon05]).

Definition 0.12. We define $\mathcal{I}_\bullet(S)$, for a set S, to be the commutative $\mathbb{Q}$-algebra generated by the elements

$$I_{r_1,\ldots,r_s}(x_1,\ldots,x_s) \quad \text{for} \quad \sum_{i=1}^{s} r_i = w \quad \text{and} \quad x_i \in S.$$

We denote by $\mathcal{I}_w(S)$, for a fixed weight w, the vector space generated as follows.

$$\mathcal{I}_w(S) = \left\langle \prod_i I_{r_{i,1},\ldots,r_{i,s}}(x_{i,1},\ldots,x_{i,s}) \;\middle|\; x_{i,j} \in S, \sum_{i,j} r_{i,j} = w \right\rangle.$$

Remark 0.13. In [Gon05], Goncharov also gives a Hopf algebra structure to $\mathcal{I}_\bullet(S)$ and so is equipped with comultiplication. As we will not use this, the interested reader is referred to the paper for more information.

In this thesis we concentrate on linear combinations of multiple polylogarithms with rational coefficients of a fixed weight, w, which lie in $\mathcal{I}_w(S)$. Typically we will take the set S to be rational functions on (normally one or two) complex variables. Therefore, we will be interested in terms of the form

$$\sum_i q_i I_{r_1,\ldots,r_s}(X_1,\ldots,X_s) \in \mathcal{I}_w(S)$$

where $q_i \in \mathbb{Q}$ and $\sum_{k=1}^{s} r_k = w$ and

$$X_i \in S = \mathbb{Q}(x_1, \ldots, x_t), \quad \text{for} \quad x_j \in \mathbb{C}.$$

We will avoid products of multiple polylogarithms, as seen in the stuffle relations above.

Example 0.14. The following are examples from $\mathcal{I}_\bullet(S)$ (where we fix the grading).

- $I_{1,2}(x,y) + I_{1,1,1}(x,y,z) + I_1(x)I_1(y)I_1(z) + I_2(xz)I_1(y) \in \mathcal{I}_3(S)$.
- $I_4(x) + I_{1,2,1}(x,y,z) + I_1(x)I_3(yz) \in \mathcal{I}_4(S)$.

However we will be more interested in elements, as described above, where we do not include products of multiple polylogarithms (while still fixing the grading), i.e., such as

- $I_{2,1}(x,y) + I_{1,1,1}(x,y,z) + I_3(xyz) \in \mathcal{I}_3(S)$,
- $I_4(xy) + I_{1,2,1}(x,y,z) + I_{1,3}(x,yz) \in \mathcal{I}_4(S)$.

0.3 Summary of Chapters

We begin, in Chapter 1, by presenting an overview of the symbol map, both in the form of its first appearance by Goncharov in [Gon05] and in the form given by Gangl, Goncharov and Levin in [GGL09].

We then introduce, in Chapter 2, a new way of viewing the calculation of the symbol of a multiple polylogarithm, namely the *hook-arrow tree*. The hook-arrow tree can be used to calculate the symbol map on a computer (e.g. using the computer package GP/Pari [PAR11]).

Chapter 3 gives bijections between different pictorial methods of finding the symbol; the binary trees from [Gon05], polygon dissections from [GGL09], and hook-arrow trees and ternary trees presented here.

In Chapter 4 we use hook-arrow trees to give a method for simplifying the calculation of the symbol for multiple polylogarithms with a given depth and construct a general formula explicitly for the symbol of all depth 2 and 3 multiple polylogarithms. The full symbol for $I_{2,2,2}(x, y, z)$ is given in Appendix C. Using a similar method, in Chapter 5, we give the general form of the symbol for a specific class of multiple polylogarithms, namely coloured multiple zeta values. This result appears in a preprint of a paper, [DGR11], written with C Duhr and H Gangl.

We are then able to find new linear combinations of multiple polylogarithms in the kernel of the symbol that would otherwise be too lengthy to calculate by hand, by using a method outlined in Appendix A. We give these in Chapters 6 and 7. In Chapter 6 we give linear combinations of *harmonic polylogarithms*, a specific class of polylogarithms particularly useful in particle physics, that are in the kernel of the symbol map. In Chapter 7 we give further linear combinations in the kernel of the symbol, in one and two variables, for general multiple polylogarithms.

The symbol map takes multiple polylogarithms to elements in the 'maximal part' of a bar construction (to be discussed in Section 1.4). Appendix B proposes the equivalent method to hook-arrow tree for finding non-maximal parts of the bar construction.

Chapter 1

The symbol of a multiple polylogarithms via binary trees and polygon dissection

The symbol attached to a multiple polylogarithm is an algebraic object which contains important combinatorial and analytical data about the function. Instead of comparing multiple polylogarithms directly, we instead apply the symbol map and compare their representatives in the image, which is considerably easier, though still far from trivial. Conjecturally, any functional equation between multiple polylogarithms must be in the kernel of the symbol map (a priori, there could exist, for example, functional equations between multiple polylogarithms of different weights, that the symbol map would not see, but these are widely conjectured not to exist).

In this chapter we will give a method for finding the symbol of a multiple polylogarithm by dissecting polygons. The description of polygon dissection in this chapter is adapted mainly from the papers [GGL09] and [DGR11]. However, after briefly introducing some basic tensor calculus, we will first give an outline of an earlier description of the symbol (under the name '$\otimes^m$-invariant') given by Goncharov in [Gon05] via a binary (or plane trivalent rooted) tree. The two descriptions are given in different forms but are nevertheless equivalent up to regrouping of terms. We relate the two in Chapter 3.

We then give an explanation as to why we chose the symbol to represent a multiple polylogarithm, and finally give an example of an element in the kernel of the symbol map.

1.1 Tensor algebra and notation conventions

The symbol will be defined in terms of a formal linear sum of elementary tensors. We overview tensor products now and set some notation conventions that will be used for the rest of the thesis. For readers accustomed with tensor products, the author recommends skipping to Section 1.2.

Definition 1.1. The **tensor product** of R-modules $(V, +_V, \cdot_{RV})$ and $(W, +_W, \cdot_{RW})$ is defined by

$$V \otimes_R W := \frac{\langle v \otimes_R w \mid v \in V,\ w \in W\rangle}{\langle \text{relations}\rangle}$$

where the relations are bilinearity and multiplication by elements of R:

$$\begin{aligned}(a +_V b) \otimes_R c &= a \otimes_R c + a \otimes_R c,\\ a \otimes_R (c +_W d) &= a \otimes_R c + a \otimes_R d,\\ (r \cdot_{RV} a) \otimes_R c &= a \otimes_R (r \cdot_{RV} c) = r \cdot_R V(a \otimes_R c),\end{aligned}$$

where $a, b \in V$, $c, d \in W$ and $r \in R$. We allow formal addition of tensor products.

We will only be concerned with tensor products of $\mathbb{Z}$-modules. We will, for ease of notation, denote $\otimes = \otimes_{\mathbb{Z}}$. In this thesis, where elements of V will be in $\mathbb{C}^\times$, we will take

$$a +_V b = ab.$$

So, for example, the first and second relations above will take the form

$$\begin{aligned}ab \otimes c &= a \otimes c + b \otimes c,\\ a \otimes bc &= a \otimes b + a \otimes c.\end{aligned}$$

We take the operation $\cdot_{\mathbb{Z}}$ on the module for $a, b, \in (V, +_V, \cdot_{\mathbb{Z}V})$ and $r \in \mathbb{Z}$ to be

$$r \cdot_{\mathbb{Z}V} a = \underbrace{a +_V \ldots +_V a}_{r \text{ times}} = a^r.$$

So, the third relation above will take the form

$$a^r \otimes b = a \otimes b^r = r(a \otimes b).$$

Proposition 1.2. *The following hold for $a, b \in \mathbb{Q}$ and $n \in \mathbb{Z}$.*

1. $1 \otimes a = a \otimes 1 = 0$.
2. *For $\mathbb{Z}$-modules, the relation on multiplication by elements of $\mathbb{Z}$ follows from bilinearity.*

Proof. 1. Directly from the definition of a tensor product we see that

$$1 \otimes a = (1 \cdot 1) \otimes a = 1 \otimes a + 1 \otimes a$$

hence $1 \otimes a = 0$. The same method also gives $a \otimes 1 = 0$.

2. For $n > 0$ we see that by repeated applications of the first relation from the definition of the tensor product then

$$a^n \otimes b = \underbrace{(a \otimes b) + \ldots + (a \otimes b)}_{\text{n times}} = n(a \otimes b).$$

By part 1, the case $n = 0$ holds trivially. For $n < 0$ we see that

$$a^n \otimes b + a^{-n} \otimes b = 1 \otimes b = 0$$
$$a^n \otimes b = -a^{-n} \otimes b,$$

so, since $-n > 0$, we have that

$$a^n \otimes b = n(a \otimes b).$$

Again, the proof for $a \otimes b^n = n(a \otimes b)$ works the same.

□

Remark 1.3. As a consequnce of statement 2 of Proposition 1.2, for the rest of this thesis, a tensor product of $\mathbb{Z}$-modules V and W is assumed to be

$$V \otimes W := \frac{\langle v \otimes w \mid v \in V,\ w \in W \rangle}{\langle \text{relations} \rangle}$$

where the relations are

$$ab \otimes c = a \otimes c + a \otimes c,$$
$$a \otimes cd = a \otimes c + a \otimes d,$$

for $a, b \in V$ and $c, d \in W$.

Example 1.4. We give a very basic illustrative example of tensor algebra.

$$\begin{aligned}(4 \otimes 3) + 2(2 \otimes 2) + (\tfrac{1}{4} \otimes 6) &= (4 \otimes 3) + (4 \otimes 2) - (4 \otimes 6) \\ &= (4 \otimes 6) - (4 \otimes 6) \\ &= 0.\end{aligned}$$

Remark 1.5. We note here that tensor algebra defined this way will act much like products of logarithms. We note the similarities (except for tensor products not being commutative) between

$$\begin{aligned} ab \otimes c &= a \otimes c + b \otimes c, \\ a \otimes cd &= a \otimes c + a \otimes d \end{aligned} \quad \text{and} \quad \begin{aligned} \ln(ab)\ln(c) &= \ln(a)\ln(c) + \ln(b)\ln(c), \\ \ln(a)\ln(cd) &= \ln(a)\ln(c) + \ln(a)\ln(d). \end{aligned}$$

The symbol of a multiple polylogarithm will be defined in terms of tensor products; the above similarity between tensor calculus and logarithms is very indicative of the association between symbol and multiple polylogarithm.

Finally, we extend the definition of a tensor product to allow tensor products of more than two $\mathbb{Z}$-modules, so we typically consider elements

$$v_1 \otimes \cdots \otimes v_n \quad \text{for } v_i \in V_i$$

for $\mathbb{Z}$-modules V_i. The definition extends naturally and one imposes multilinearity.

1.1.1 Shuffle product

We will often have situations where we sum $a \otimes b$ and $b \otimes a$. We make the following definition to simplify notation.

Definition 1.6. We define the **shuffle product**, $\sqcup\!\sqcup$, to be

$$a \sqcup\!\sqcup b := a \otimes b + b \otimes a$$

The following two examples demonstrate shuffling tensor products.

Example 1.7.

$$a \sqcup\!\sqcup (b \otimes c) = a \otimes b \otimes c + b \otimes a \otimes c + b \otimes c \otimes a.$$

Example 1.8. A triple shuffle takes the form

$$\begin{aligned} a \sqcup\!\sqcup b \sqcup\!\sqcup c = & \quad a \otimes b \otimes c + a \otimes c \otimes b \\ + & \quad b \otimes a \otimes c + b \otimes c \otimes a \\ + & \quad c \otimes a \otimes b + c \otimes b \otimes a. \end{aligned}$$

1.1.2 2-torsion of tensor products and notation conventions

In this thesis, tensor products are considered equal when they are equivalent up to 2-torsion. By this we mean to say that

$$a_1 \otimes \ldots \otimes (-a_i) \otimes \ldots \otimes a_n \overset{\text{2-torsion}}{=} a_1 \otimes \ldots \otimes a_i \otimes \ldots \otimes a_n$$

because

$$\begin{aligned} 2\big(a_1 \otimes \ldots \otimes (-a_i) \otimes \ldots \otimes a_n\big) =& a_1 \otimes \ldots \otimes (-a_i)^2 \otimes \ldots \otimes a_n \\ =& a_1 \otimes \ldots \otimes a_i^2 \otimes \ldots \otimes a_n \\ =& 2\big(a_1 \otimes \ldots \otimes a_i \otimes \ldots \otimes a_n\big). \end{aligned}$$

We could multiply every tensor product throughout by a factor of 2 or always write '$\overset{\text{2-torsion}}{=}$' during tensor calculus, but for convenience of notation, we simply use '=' with the understanding that the calculations work up to 2-torsion.

We now establish some notation on tensors and shuffle products. We will often meet tensor products of the form

$$(\underbrace{a \otimes ... \otimes a}_{b}),$$

to simplify this we introduce the notation

$$a^{\otimes b} := (\underbrace{a \otimes ... \otimes a}_{b}).$$

Since we will have many nested brackets, we establish the notation convention that a shuffle product is 'stronger' than a tensor product, so we understand

$$a \otimes b \sqcup\!\sqcup c = a \otimes (b \sqcup\!\sqcup c).$$

Finally, to avoid confusion, we take

$$a^{\otimes b} \sqcup\!\sqcup c^{\otimes d} := (\underbrace{a \otimes ... \otimes a}_{b}) \sqcup\!\sqcup (\underbrace{c \otimes ... \otimes c}_{d}).$$

We now define the wedge product, which we will use occasionally; an anticommutative version of the tensor product.

Definition 1.9. The wedge product of $\mathbb{Z}$-module V with itself is defined, in terms of the tensor product, as

$$V \wedge V := \frac{V \otimes V}{< a \wedge b + b \wedge a \quad \text{for } a, b \in V >}.$$

We note that, as a direct consequence, we have that $a \wedge a = -(a \wedge a) = 0$ for all $a \in V$. We will also work modulo 2-torsion for wedge products (in the same way as for tensor products).

1.2 Outline of the symbol from binary trees

We now briefly outline the definition of the symbol of a multiple polylogarithm from [Gon05]. We begin by defining a binary tree.

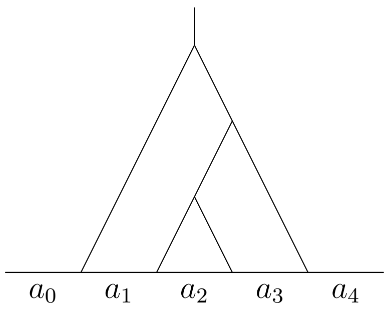
Definition 1.10. We define a **binary tree** to be a trivalent rooted plane tree.

By *trivalent* we mean that every vertex of the tree has either valency 3 (and is an internal vertex) or valency 1 (and is an external vertex). By *rooted* we mean that one external vertex of the tree is distinguished as the root. We picture a binary tree as being embedded in a plane, growing down from its root (which due to being trivalent will be a *planted* tree) to a baseline (a helpful pictorial tool, but strictly speaking not part of the tree). All external vertices of the binary tree lie on the baseline, except the root vertex, and since the tree is embedded in a plane and they are given a strict order (in this thesis this will be from left to right on the baseline).

Definition 1.11. A binary tree, T, with $n+2$ external vertices is said to be **decorated** if the sections of the baseline cut out by the external vertices (except for the root vertex) are labelled anticlockwise, starting with the section after the root, with an ordered set $(a_0, \ldots, a_{n+1})$ for distinct $a_i \in \mathbb{A}^1$, where $\mathbb{A}^1$ is the affine line.

The $n+2$ distinct points (the a_i) will label the sections of the baseline cut out by the external vertices of the binary tree.

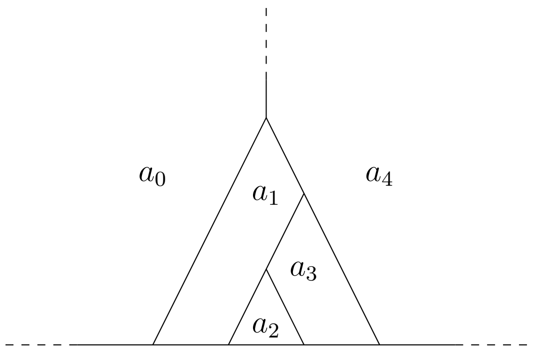
Example 1.12. A binary tree, T, with 5 external vertices, decorated by the ordered set $(a_0, \ldots, a_4)$ can be viewed as the following.



1.2.1 Attaching a symbol to the binary tree

We note that, by viewing the root vertex to be extended to infinity and the baseline to also be extended to infinity (in both directions), the tree cuts out domains of the upper half plane. These domains obtain a unique label from the decoration $(a_0, \ldots, a_{n+1})$. A nice way to consider this is to view it in an upper half plane model of hyperbolic space, where the root vertex is the point at infinity and the baseline is the real line.

Example 1.13. The labelling of domains as described above, gives the following diagram for the tree in Example 1.12.

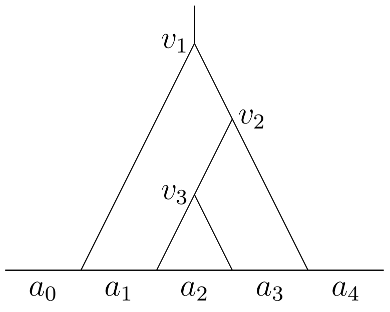


There exists a canonical partial ordering on the internal vertices of the tree dictated by distance from the root. We say that $v_1 \prec v_2$ if and only if a direct path exists from v_2 to the root that passes through v_1. A strict order, $(v_{i_1}, \dots, v_{i_n})$ on the internal vertices $\{v_1, \dots, v_n\}$ is *compatible* with the partial ordering if $v_{i_j} \prec v_{i_k}$ for all $1 \le j < k \le n$.

Each internal vertex, v, of the tree lies at the apex of three domains, which we call $\Delta_1^v, \Delta_2^v, \Delta_3^v$. The order of the labelling of these domains is dictated by a natural anticlockwise direction on the plane. The first domain, Δ_1 lies directly anticlockwise of the edge connecting v to the closest vertex to the root (in the partial ordering). We let $a_{\Delta_1}, a_{\Delta_2}, a_{\Delta_3}$ be the labels corresponding to $\Delta_1^v, \Delta_2^v, \Delta_3^v$ and define

$$f_v^T := \frac{a_{\Delta_3} - a_{\Delta_2}}{a_{\Delta_1} - a_{\Delta_2}} \in \mathbb{Q}(a_0, \dots, a_{n+1}) \quad \text{for } a_i \in \mathbb{A}^1. \tag{1.1}$$

Example 1.14. For the binary tree, T, in Example 1.12 there is only one compatible strict ordering on the 3 internal vertices, (v_1, v_2, v_3), shown on the following diagram:



We have that

$$f^T_{v_1} = \frac{a_4 - a_1}{a_0 - a_1}, \quad f^T_{v_2} = \frac{a_4 - a_3}{a_1 - a_3}, \quad \text{and} \quad f^T_{v_3} = \frac{a_3 - a_2}{a_1 - a_2}.$$

We now define the $\otimes^m$-invariant of a multiple logarithm (which appears on pages 22-23 of [Gon05], though is slightly adapted such that $a_0 = 0$ and $a_{n+1} = 1$). This foreshadows the definition of the symbol, which we will give next.

Definition 1.15. We attach to a weight-m multiple logarithm $I(0; x_1, \dots, x_m; 1)$ the **$\otimes^m$-invariant** given by

$$\sum_T \sum_{\{v_1,\dots,v_m\}} f^T_{v_1} \otimes \cdots \otimes f^T_{v_m}$$

where the first summation runs over all binary trees with $m + 2$ external vertices, decorated with the distinct labels $\{0, x_1, \dots, x_m, 1\}$ and the second summation runs over all strict orders of internal vertices compatible with T.

We now extend this definition to allow for non-distinct arguments of the multiple logarithm. We do this by extending the definition of f^T_v from Equation (1.1) to

$$g^T_v := \begin{cases} \dfrac{a_{\Delta_3} - a_{\Delta_2}}{a_{\Delta_1} - a_{\Delta_2}} & \text{if } a_{\Delta_i} \neq a_{\Delta_j} \text{ for all } i, j \\ 1 & \text{if } a_{\Delta_i} = a_{\Delta_j} \text{ for some } i, j \end{cases} \tag{1.2}$$

where $a_i \in \mathbb{A}^1$. This allows us to define the symbol of a multiple polylogarithm.

Definition 1.16. We attach, to a weight w multiple polylogarithm $I_{r_1,\dots,r_s}(x_1, \dots, x_s)$, the **symbol** given by

$$\mathcal{S}(I_{r_1,\dots,r_s}(x_1, \dots, x_s)) := \sum_T \sum_{\{v_1,\dots,v_w\}} g^T_{v_1} \otimes \cdots \otimes g^T_{v_w}$$

where the first summation runs over all binary trees with $w + 2$ external vertices, decorated with the ordered labels

$$(0, x_1, \underbrace{0, \dots, 0}_{r_1-1}, x_2, \underbrace{0, \dots, 0}_{r_2-1}, \dots, x_s, \underbrace{0, \dots, 0}_{r_s-1}, 1),$$

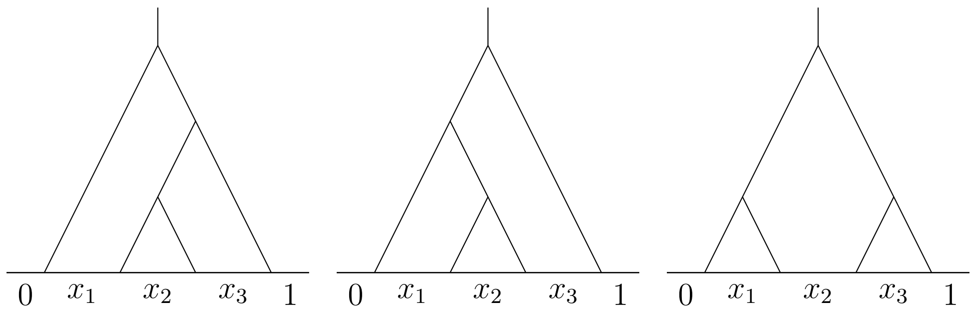
and the second summation runs over all strict orders of internal vertices $(v_1, \dots, v_w)$ compatible with T. The symbol lies in the space

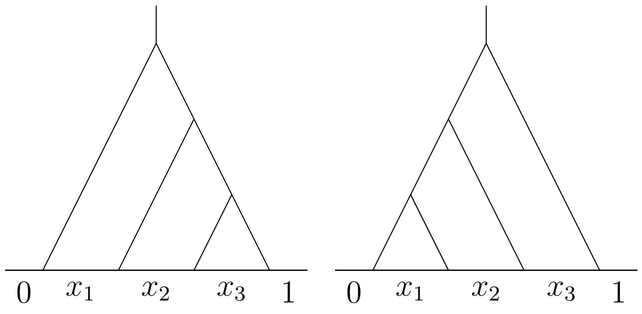
$$\bigotimes_{i=1}^{w} \mathbb{Q}(x_1, \dots, x_s)^*$$

where here $\mathbb{Q}(x_1, \dots, x_s)^*$ denotes the multiplicative group of the invertible elements of $\mathbb{Q}(x_1, \dots, x_s)$.

Remark 1.17. We note that when a binary tree contains a vertex which is at the apex of two domains with the same label, then the respective $g_v^T = 1$, and therefore that term in the symbol, will not contribute due to statement 1 of Proposition 1.2. An equivalent way to view terms not contributing to the symbol due to non-distinct labels (rather than through the definition of $g_v^T = 1$) is to consider these terms to be given a coefficient of zero.

Example 1.18. When finding the symbol of a weight 3 multiple polylogarithm, say $I_{1,1,1}(x_1, x_2, x_3)$, using the above definition, we will sum over all binary trees with 5 external vertices. These are shown below; the first being the tree from Example 1.12.





Giving the terms in the order of the binary trees above, and noting that there are two possible strict orders compatible with the partial order on the internal vertices of the third tree (shown with a shuffle), we find the symbol, $\mathcal{S}(I_{1,1,1}(x_1,x_2,x_3))$ to be

$$\begin{aligned}
= &+ \left(\frac{1-x_1}{-x_1} \otimes \frac{1-x_3}{x_1-x_3} \otimes \frac{x_3-x_2}{x_1-x_2}\right) + \left(\frac{1-x_3}{-x_3} \otimes \frac{x_3-x_1}{-x_1} \otimes \frac{x_3-x_2}{x_1-x_2}\right) \\
&+ \left(\frac{1-x_2}{-x_2} \otimes \frac{x_2-x_1}{-x_1} \sqcup \frac{1-x_3}{x_2-x_3}\right) + \left(\frac{1-x_3}{-x_3} \otimes \frac{x_3-x_2}{-x_2} \otimes \frac{x_2-x_1}{-x_1}\right) \\
&+ \left(\frac{1-x_1}{-x_1} \otimes \frac{1-x_2}{x_1-x_2} \otimes \frac{1-x_3}{x_2-x_3}\right).
\end{aligned}$$

We now, in the next section, define the symbol using polygon dissection. While the definition of the symbol using binary trees and polygon dissection are equivalent, they differ slightly in their approach. As mentioned, the differences are discussed in Chapter 3, and a proof is given that they really do give the same symbol.

1.3 Defining the symbol from polygon dissection

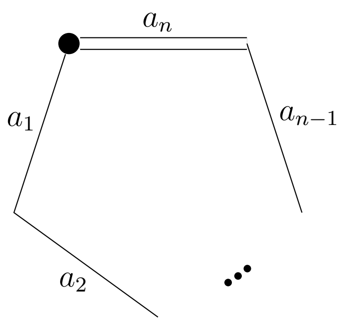
1.3.1 Associating a polygon to a multiple polylogarithm

We begin by defining an algebra of R-decorated polygons (as defined on page 563 of [GGL09]). We will soon associate an n-gon of this form with $n = w + 1$ sides to a weight w multiple polylogarithm.

Definition 1.19. An R**-decorated polygon**, with n sides given by

$$P(a_1, \ldots, a_n)$$

is a polygon where an orientation (always given anticlockwise in this thesis) is given by designating a first vertex (drawn with an enlarged 'dot') and a final side (drawn with a 'double line'). The sides are labelled, beginning with the first side, by $a_1, \ldots, a_n$ from a set R. We do not label the vertices, but due to the designation of a first vertex they are understood to have an order $v_1, \ldots, v_n$.



Definition 1.20. The **graded vector space of polygons** is given by

$$V_\bullet^{\mathbf{pg}}(R) := \bigoplus_{n=0}^{\infty} V_n^{\mathbf{pg}}(R)$$

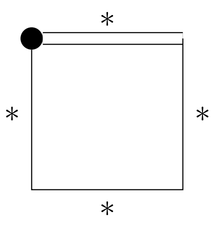
where $V_n^{\mathbf{pg}}(R)$ is the $\mathbb{Q}$-vector space of R-decorated $(n+1)$-gons for $n \geq 1$ and $V_0^{\mathbf{pg}}(R) := \mathbb{Q}$.

Definition 1.21. We call the exterior algebra on the graded vector space of polygons, with decorations in R, the **polygon algebra** and denote it by $\mathcal{P}_\bullet^\bullet = \mathcal{P}_\bullet^\bullet(R)$. The algebra is bigraded. The first grading (denoted with a subscript) comes from summing the number of non-root sides of the polygons in each product. The second grading (denoted with a superscript) comes from the exterior power.

We note that we will sometimes drop the second grading (the exterior power) from the notation. We demonstrate the two gradings of the polygon algebra in the following example.

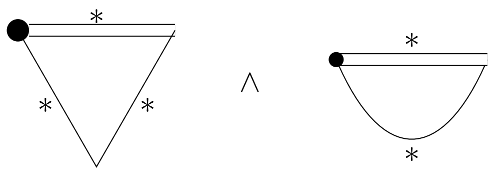
Example 1.22. Some simple examples follow.

1.



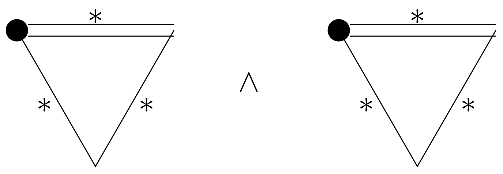
$\in \mathcal{P}_3^1$.

2.



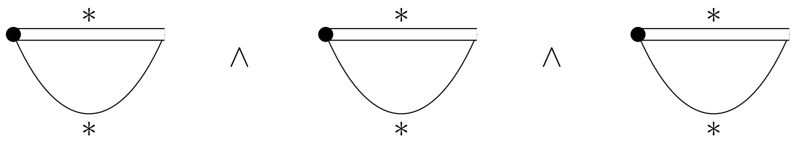
$\in \mathcal{P}_3^2$.

3.



$\in \mathcal{P}_4^2$.

4.



$\in \mathcal{P}_3^3$.

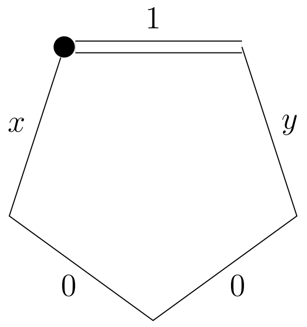
An R-decorated $(w+1)$-gon is associated to a weight w multiple polylogarithm (with arguments in a given set R) by associating

$$I_{m_1,\dots,m_w}(x_1,\dots,x_w) := I(0; x_1, \underbrace{0,\dots,0}_{m_1-1}, x_2, \underbrace{0,\dots,0}_{m_2-1}, \dots, x_w, \underbrace{0,\dots,0}_{m_w-1}; 1)$$

to

$$P(x_1, \underbrace{0,\dots,0}_{m_1-1}, x_2, \underbrace{0,\dots,0}_{m_2-1}, \dots, x_w, \underbrace{0,\dots,0}_{m_w-1}, 1).$$

Example 1.23. The multiple polylogarithm $I_{3,1}(x,y)$ is associated to the polygon $P(x,0,0,y,1)$ given by



1.3.2 Adding dissecting arrows to a polygon

We now describe a process of dissecting decorated rooted n-gons into products of polygons with fewer sides. After fully dissecting we will be left with 2-gons (which cannot be dissected further), of which there will be $n-1$, with a partial order. We will see how there are options as to how to dissect the polygon; the symbol will arise from formally adding these options under an algebraic interpretation.

We now define how to *dissect a polygon.* This is achieved by adding arrows, which we now define.

Definition 1.24. Let $\pi = P(a_1, \dots, a_n)$ be an R-decorated polygon with vertices $v_1, \dots, v_n$ as in Definition 1.19. An **arrow**, α, associates a vertex to a non-adjacent side where

$$v_i \text{ is considered to be adjacent to } \begin{cases} a_1 \text{ and } a_n & \text{for } i = 1, \\ a_{i-1} \text{ and } a_i & \text{for } i = 2, \dots n. \end{cases}$$

An arrow from v_i to the side labelled a_j is said to be **backwards** if $i > j$ and **forwards** if $i < j$.

Definition 1.25. A **dissection of a polygon**, $\pi = P(a_1, \dots, a_n)$ corresponds to an arrow, α, from vertex v_i to side a_j. A polygon is dissected into two polygons given by

$$\pi_1 = P(a_1, \dots, a_{i-1}, a_j, \dots, a_n) \quad \text{and} \quad \pi_2 = P(a_i, \dots, a_j)$$

if α is a forwards arrow and

$$\pi_1 = P(a_1, \dots, a_j, a_i, \dots, a_n) \quad \text{and} \quad \pi_2 = P(a_{i-1}, \dots, a_j)$$

if α is a backwards arrow. It is important to note that, in the case of a backwards arrow, the sides on the polygon $\pi_2 = P(a_{i-1}, a_{i-2}, \dots, a_j)$ are now in a different order to that of the original polygon.

We now define the sign of a polygon dissection.

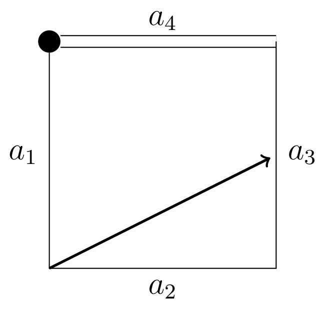
Definition 1.26. Let α be an arrow on a polygon π. We define its **sign** to be

$$\operatorname{sgn}(\alpha) := \begin{cases} (-1)^{\chi(\alpha)} & \text{if } \alpha \text{ is backwards} \\ 1 & \text{otherwise} \end{cases}$$

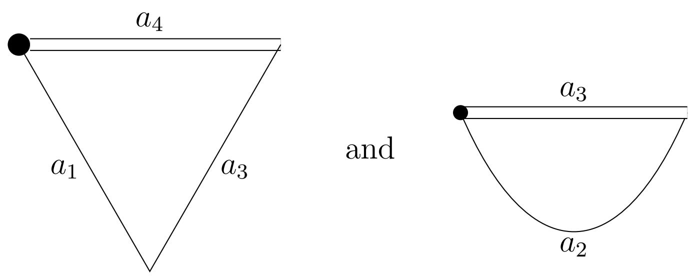
where $\chi(\alpha)$ is defined to be the number of non-root edges of polygon π_2 from Definition 1.25.

The above definitions are best understood via an example.

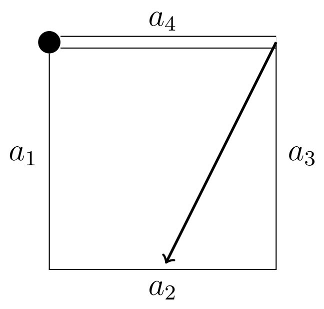
Example 1.27. We start with a 4-gon given by $P(a_1, \ldots, a_4)$ and add a dissecting arrow, α_1, from v_2 to a_3. This will leave us with polygons $P(a_1, a_3, a_4)$ and $P(a_2, a_3)$. If we draw this we see



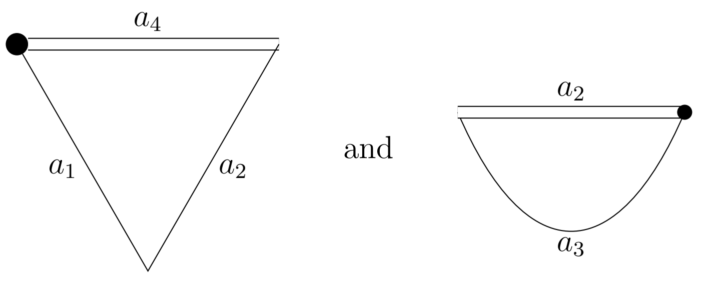
is dissected to



which is an example of a forwards arrow with $\operatorname{sgn}(\alpha_1) = 1$. If instead we dissect $P(a_1, \ldots, a_4)$ by an arrow, α_2, from v_4 to a_2 we are left with polygons $P(a_1, a_3, a_4)$ and $P(a_2, a_3)$. If we draw this we see



is dissected to



which is a backwards arrow with with $\mathrm{sgn}(\alpha_2) = -1$.

Remark 1.28. The best way to see intuitively the inheritance of first vertex and final edge is as follows. The original first vertex or final side of the polygon to be dissected are always retained in any subsequent 'sub-polygons.' If these are not present then the vertex at the base of the arrow or the side at the head of the arrow take the respective roles of first vertex or final side.

1.3.3 Maximal dissections

Definition 1.29. A **maximal dissection**, ρ, of a polygon $P(a_1, \ldots, a_n)$ is a set of $(n-2)$ distinct, non-crossing, dissecting arrows.

For a maximal dissection, ρ, we define an overall sign, which, in a slight abuse of notation, we also denote by $\mathrm{sgn}(\rho)$.

Definition 1.30. Let ρ be a maximal dissection of a polygon. The **sign** of the dissection is defined to be

$$\mathrm{sgn}(\rho) := (-1)^{\#\{\text{backwards arrows}\}}.$$

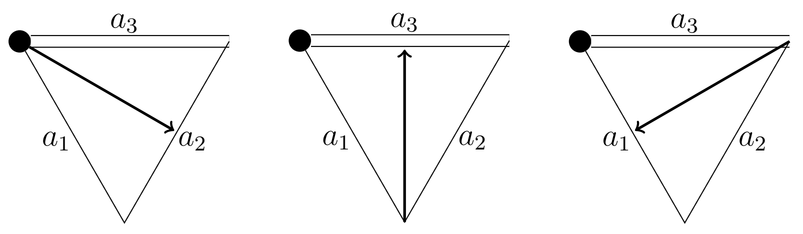
Remark 1.31. This definition of the sign of a maximal dissection comes from the sign of the dissection arising from each individual arrow. In fact we have that

$$\mathrm{sgn}(\rho) = \prod_{\text{backwards} \alpha \in \rho} (-1)^{\chi(\alpha)},$$

where $\chi(\alpha)$ is defined to be the number of non-root edges of polygon π_2 from Definition 1.25 (when ignoring all other arrows in ρ).

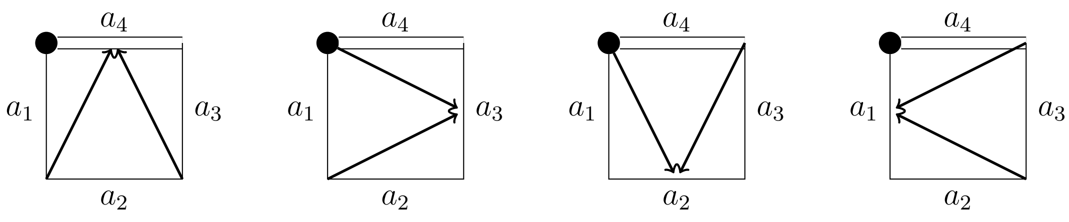
We give maximal dissections of a 3-gon and 4-gon.

Example 1.32. There are 3 possible maximal dissections of a 3-gon, given by the following.

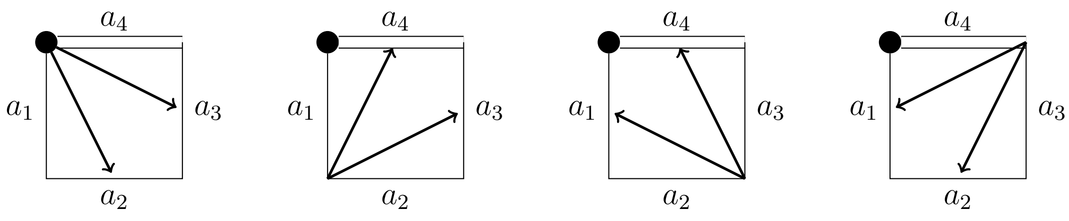


sign = $+1$ sign = $+1$ sign = -1

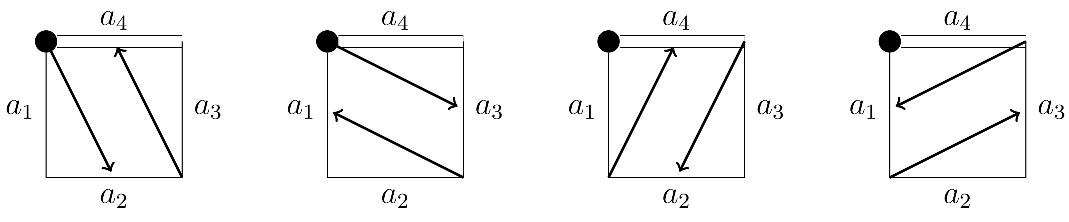
There are 12 possible maximal dissections of a 4-gon, given by the following.



sign = $+1$ sign = $+1$ sign = -1 sign = $+1$



sign = $+1$ sign = $+1$ sign = -1 sign = $+1$

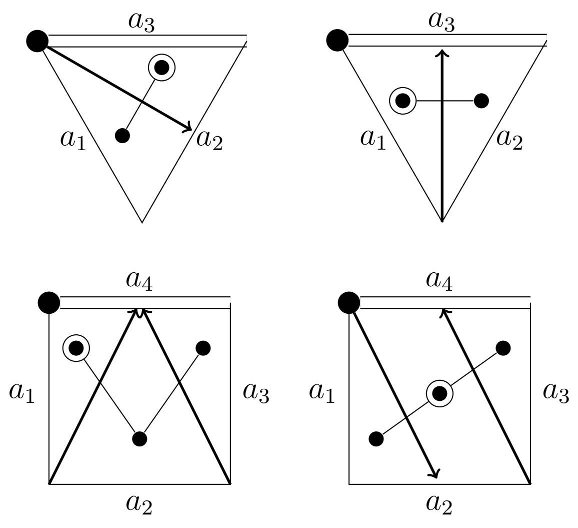


sign = $+1$ sign = -1 sign = -1 sign = -1

A maximal dissection will dissect a polygon into 2-gons. Before we discuss these 2-gons we add a rooted 'dual tree.' This tree will put a partial order on the 2-gons and is formally defined as follows.

Definition 1.33. The **dual tree of a maximal polygon dissection** is a rooted tree with a vertex representing each dissected region (the dissected regions will become 2-gons). Two vertices of the dual tree are connected if they share the same arrow on the boundary of the region. The root of the dual tree is defined to be the region containing both the first vertex and (at least part of) the final side on its boundary, this will be unique by construction.

Example 1.34. We give the dual tree for a few of the maximal dissections from Example 1.32. The root of the tree is shown with an circle around the vertex.



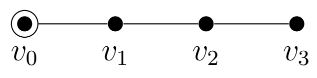
We now describe a partial order on the 2-gons of a maximal dissection, dictated by the dual tree.

Definition 1.35. The **partial order**, $\prec$, on a tree is defined as follows. Let v_0 be the root of the tree. A vertex v_i comes after the root vertex, written $v_0 \prec v_i$ for all $i \neq 0$. Then $v_i \prec v_j$, with $i \neq 0$ if and only if there is a direct path through the tree from v_0 to v_j passing through v_i.

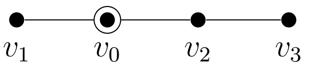
Definition 1.36. A strict order, $v_0 < v_{i_1} < \ldots < v_{i_r}$, on a tree is said to be **compatible** with the partial order, $\prec$, on a tree if

$$v_{i_j} < v_{i_k} \implies v_{i_j} \prec v_{i_k} \quad \text{for all} \quad i_j, i_k.$$

Example 1.37. The tree



has partial order of $v_0 \prec v_1 \prec v_2 \prec v_3$. This has only one compatible strict ordering, namely $v_0 < v_1 < v_2 < v_3$. However, the tree



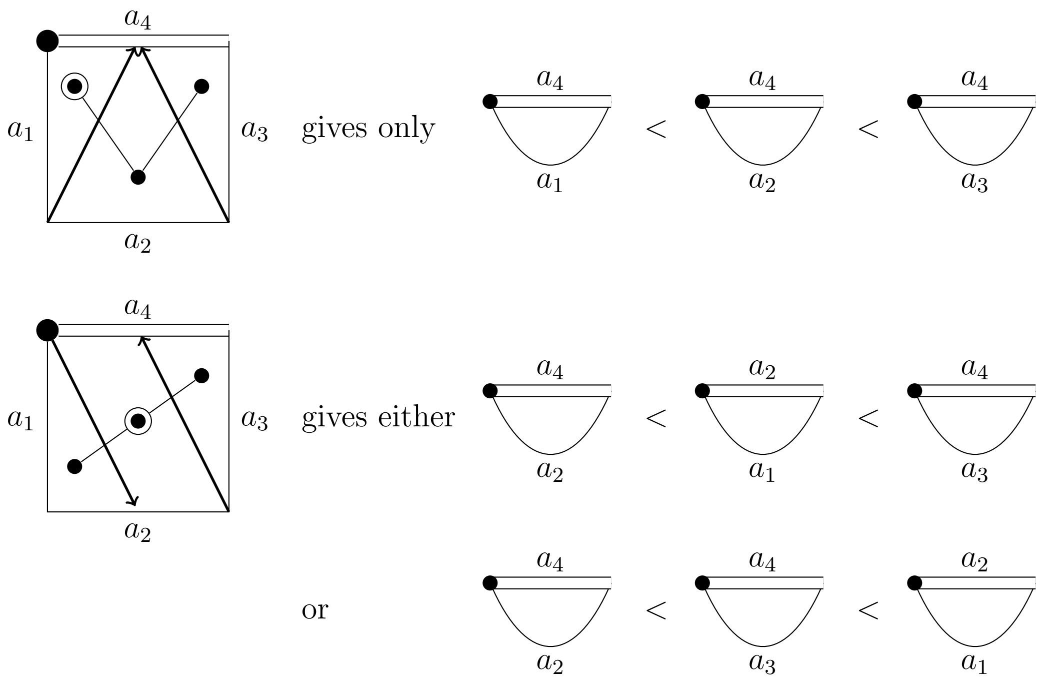
has partial order dictated by $v_0 \prec v_1$ and $v_0 \prec v_2 \prec v_3$. There are three possible strict orderings compatible with this partial order:

$$v_0 < v_1 < v_2 < v_3, \quad v_0 < v_2 < v_1 < v_3 \quad \text{and} \quad v_0 < v_2 < v_3 < v_1.$$

We can think of v_1 as being 'shuffled' through v_2 and v_3.

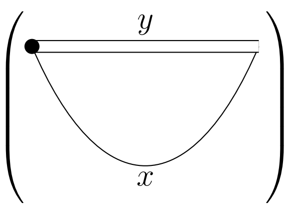
When we 'retract' the arrows in a maximal polygon dissection to form 2-gons we are left with a set of strict orderings (which are compatible with the partial ordering on the dual tree) of the 2-gons.

Example 1.38. The following 4-gons are associated to the following strict orderings on 2-gons.


a_4
a_1
a_3
a_2
gives only
a_4 a_1 $<$ a_4 a_2 $<$ a_4 a_3
a_4
a_1
a_3
a_2
gives either
a_4 a_2 $<$ a_2 a_1 $<$ a_4 a_3
or
a_4 a_2 $<$ a_4 a_3 $<$ a_2 a_1

1.3.4 Definition of the symbol

Definition 1.39. We define a map μ which associates each 2-gon with a rational function on the labels of the original polygon by

μ 

$$:= \begin{cases} 1-\frac{y}{x} & \text{when } x, y \text{ are distinct and non-zero,} \\ y & \text{when } x \equiv 0, \text{ and } y \text{ non-zero} \\ \frac{1}{y} & \text{when } x \equiv y \text{ and are non-zero} \\ 1 & \text{otherwise.} \end{cases}$$

This map will serve a very similar purpose to the function g_v^T defined in Equation (1.2).

Remark 1.40. We note that we will rarely use that $\mu(P(y,y)) = 1/y$ (where $y \not\equiv 0$) as we will mostly use multiple polylogarithms with distinct arguments (where 2-gons of this form do not occur).

We now define the symbol of a multiple polylogarithm (this time found by polygon dissection).

Definition 1.41. The **symbol** of a weight w multiple polylogarithm

$$I_{m_1,\ldots,m_r}(x_1, \ldots x_r),$$

written

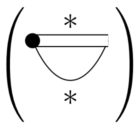
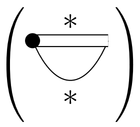
$$\mathcal{S}(I_{m_1,\ldots,m_r}(x_1, \ldots x_r))$$

is found by first associating

$$I_{m_1,\ldots,m_w}(x_1,\ldots,x_w) \quad \text{to} \quad P = P(x_1, \underbrace{0,\ldots,0}_{m_1-1}, x_2, \underbrace{0,\ldots,0}_{m_2-1}, \ldots, x_w, \underbrace{0,\ldots,0}_{m_w-1}, 1)$$

and then

$$\mathcal{S}(I_{m_1,\ldots,m_r}(x_1, \ldots x_r)) :=$$

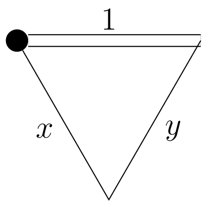
$$\sum_{\rho} \text{sgn}(\rho) \sum_{\substack{\text{Strict orders compatible} \\ \text{with the dual tree of } P}} \mu$$  $\otimes \cdots \otimes \mu$ 

where the first summation runs over all maximal dissections, ρ of P.

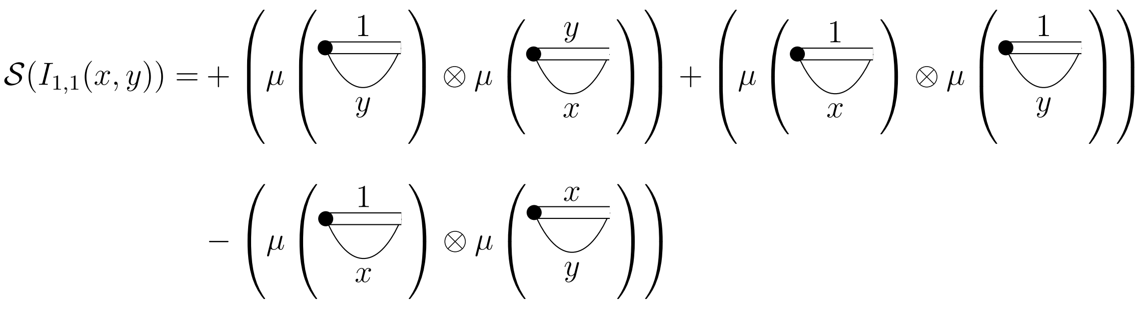
We note that, in a similar way to Remark 1.17, when the map μ takes a 2-gon to the value 1, then the resulting term does not contribute to the symbol (or equivalently is ‘given a coefficient of zero’).

The bijection between this symbol definition and the binary tree version (in Definition 1.16) is given in Chapter 3. We now give an illustrative example.

Example 1.42. We give the symbol of $I_{1,1}(x,y)$. The function is related to the 3-gon



We have already seen all possible dissections of a 3-gon in Example 1.32. The symbol follows:



$$= + \left(\left(1-\tfrac{1}{y}\right)\otimes\left(1-\tfrac{y}{x}\right)\right) + \left(\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{y}\right)\right) - \left(\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{y}\right)\right)$$

We will see further simple examples of symbols in Section 2.5, including observing the similarities between $\mathcal{S}(I_2(x))$ and $\mathcal{S}(I_{1,1}(x,y))$.

Remark 1.43. When there are more than one strict orderings on the dual tree, instead of writing out every possibility we can use the shuffle notation, $a \sqcup\!\sqcup\, b = a\otimes b + b\otimes a$ defined in Section 1.1.

1.3.5 The symbol of products of multiple polylogarithms

The symbol preserves products of multiple polylogarithms. The symbol map takes a product of two multiple polylogarithms to the shuffle of their symbols, i.e.,

$$\begin{aligned}&\mathcal{S}\big(I_{m_1,\dots,m_r}(x_1,\dots x_r)\cdot I_{n_1,\dots,n_s}(y_1,\dots y_s)\big)\\&\qquad\qquad= \mathcal{S}\big(I_{m_1,\dots,m_r}(x_1,\dots x_r)\big)\sqcup\!\sqcup\,\mathcal{S}\big(I_{n_1,\dots,n_s}(y_1,\dots y_s)\big).\end{aligned}$$

We note, however, that this thesis will not concentrate on symbols of products of multiple polylogarithms. We will focus on finding *linear* combinations of multiple polylogarithms in the kernel of the symbol. The interested reader is referred to [DGR11] where products play a more important role.

1.4 Why choose the symbol to represent multiple polylogarithms?

As described in the literature (for example in [Gon01] and [Gon05]), the symbol, originally constructed by Goncharov, has been chosen as a useful representative of multiple polylogarithms as it both holds a great deal of the differential structure of the polylogarithm while being convenient to work with. The tensor calculus that results from comparing symbols, whilst sometimes lengthy, is much easier to work with than comparing the multiple polylogarithms directly (and analytically).

Objects similar to the symbol appear in research on algebraic cycles. In a similar way to attaching the symbol to multiple polylogarithms, a topic of interest is finding algebraic cycles that 'represent' polylogarithms (often referred to as Bloch-Kriz-Totaro cycles). The reader is directed to the paper [BK95]. The 'boundaries' of these cycles (in particular the cycles representing weight 2 multiple polylogarithms) bear a very strong similarity to the symbol. The paper [GGL09] uses the *bar construction on polygons* (see below) to find algebraic cycles for multiple polylogarithms.

In this section we give a 'feel' as to why we would chose the symbol as a representative of a multiple polylogarithm. We outline the bar construction of the polygon algebra and also show how the more direct differential structure of multiple polylogarithms are similar to the symbol map. The author defers to the work of Goncharov for further explanation.

1.4.1 Bar construction of the polygon algebra

The symbol of a multiple polylogarithm can be seen as an algebraic interpretation of the maximal part of a bar construction of the algebra on the polygons. We overview this construction in this section which is adapted from pages 563 to 573 of [GGL09].

1.4.1.1 Differential graded algebra of polygons

Definition 1.44. Let $\mathcal{A}$ be a graded algebra. We can construct a **differential graded algebra** (shortened to DGA) from $\mathcal{A}$ with a map $\partial : A \to A$ of degree $+1$ or -1 such that

- $\partial \circ \partial = 0$
- $\partial(a * b) = (\partial a) * b + (-1)^{\deg(a)} a * (\partial b)$ (which is a graded Leibniz rule).

Here, $\deg(a)$ denotes the degree of a in the grading of the algebra $\mathcal{A}$.

In our case, we will take the algebra on polygons, denoted $\mathcal{P}^{\bullet}_{\bullet}$, from Definition 1.21 and will define a differential $\partial : \mathcal{P}^{\bullet}_{\bullet} \to \mathcal{P}^{\bullet}_{\bullet}$ with degree $+1$. The degree of the differential corresponds to the change in the grading on the exterior algebra. We define ∂ on an R-decorated n-gon $\pi = P(a_1, \ldots, a_n)$ to be

$$\partial\pi := \sum_{\alpha} \operatorname{sgn}(\alpha)\partial_\alpha \pi.$$

The summation runs over all possible arrows, α, as in Definition 1.25. The sign of the dissection, $\operatorname{sgn}(\alpha)$, is from Definition 1.30 and

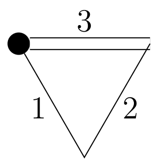
$$\partial_\alpha \pi := \pi_1 \wedge \pi_2$$

where π_1 and π_2 are the polygons resulting from dissecting the polygon π as in Definition 1.25. We note that this is similar to the structure of the symbol, but instead it only sums over single arrow dissections (as opposed to maximal dissections in the symbol) and is anticommutative. We define an augmentation in the usual way to be, for $n_i \in \mathbb{Q}$, the map

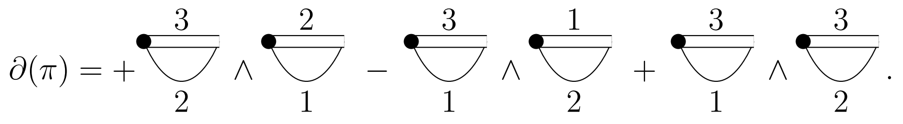
$$\sum_i n_i \pi_i \longmapsto \sum_i n_i.$$

We have now constructed an augmented DGA, $\mathcal{A}_\bullet = \bigoplus_{m\geq 1} \mathcal{A}$ on the polygons (see page 565 of [GGL09] for more details).

Example 1.45. We give an example of the action of ∂ on a polygon. Let $\pi = P(1,2,3)$ be the 3-gon



then



Here we see that

$$\partial : \mathcal{P}_2^1 \to \mathcal{P}_2^2.$$

This demonstrates that the differential has a degree of $+1$ on the exterior algebra of $\partial : \mathcal{P}_\bullet^\bullet$.

1.4.1.2 Defining the bar construction on the polygon algebra

We now define a bar construction. We will denote tensor signs by ‘|’ throughout.

Definition 1.46. The **bar construction**, denoted $B(\mathcal{A})$ for an augmented DGA

$\mathcal{A}_\bullet$ is the tensor coalgebra $\bigoplus_i \mathcal{A}_\bullet^{|i}$ with differential $D_1 + D_2$ given by

$$D_1([a_1|\dots|a_n]) = \sum_{j=1}^{n-1} (-1)^{(\sum_{i\le j} \deg(a_i)-1)} [a_1|\dots|a_j \wedge a_{j+1}|\dots|a_n]$$

$$D_2([a_1|\dots|a_n]) = \sum_{j=1}^{n} (-1)^{(\sum_{i< j} \deg(a_i)-1)} [a_1|\dots|\partial(a_j)|\dots|a_n]$$

for homogeneous $a_1, \dots, a_n \in \mathcal{A}_\bullet$.

It is important to note that, in the calculation of the signs, D_1 has a '$\leq$' in the summation of $\deg(a_i) - 1$ and D_2 has a '$<$'.

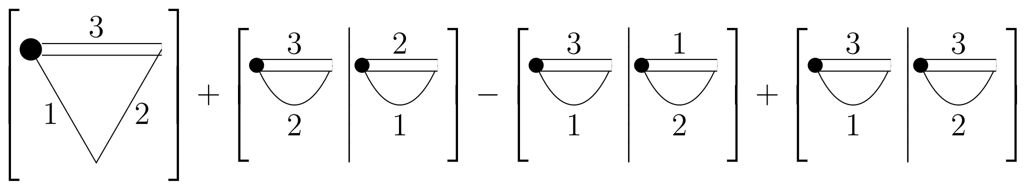
We use the augmented DGA (with differential ∂) on the polygons and create the bar construction $B(\mathcal{P}_\bullet^\bullet)$ (which actually comes equipped with further structure, a coproduct, which will not come into the context of this thesis). We can view $B(\mathcal{P}_\bullet^\bullet)$ as a double complex with respect to D_1 and D_2.

Definition 1.47. The element $B(\pi)$ attached to an n-gon $\pi = P(a_1, \dots, a_n)$, in the **bar construction** of the polygon algebra, denoted $B(\mathcal{P}_\bullet)$, has a component in $\mathcal{P}_\bullet^{|m}$ of

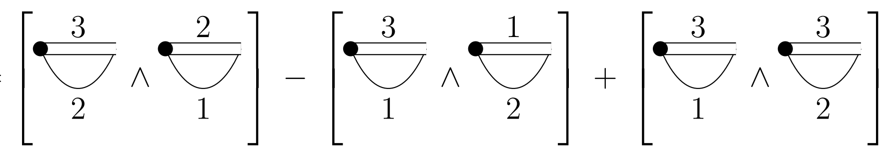
$$\sum_\rho \operatorname{sgn}(\rho) \sum_\lambda [\pi_*|\cdots|\pi_*]$$

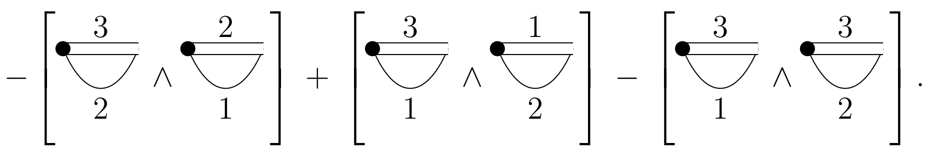
where the first summation runs over all dissections of π consisting of m arrows, the second summation runs over all linear orders, λ, of the subpolygons π_i of the dissection compatible with the partial order, $\prec$, on the dissection from Definition 1.35.

Example 1.48. Let $\pi = P(1, 2, 3)$, then we see that $B(\pi)$ is given by



where

$D_1(B(\pi)) =$  and

$D_2(B(\pi)) =$ 

Note that:

- D_1 sends the $\mathcal{P}_\bullet^{|1}$ component of $B(\pi)$ to 0.
- D_2 sends the $\mathcal{P}_\bullet^{|2}$ component of $B(\pi)$ to 0.

We now see that after cancelling pairwise, we have

$$(D_1 + D_2)(B(\pi)) = 0.$$

Example 1.49. If we let $\pi = P(1,2,3,4)$ we see that

$$B(\pi) = \Pi_3 + \Pi_{1,2} + \Pi_{1,1,1}$$

where Π_3 is the $\mathcal{P}_\bullet$ component of $B(\pi)$, given by

$$[P(1,2,3,4)],$$

$\Pi_{1,2}$ is the $\mathcal{P}_\bullet|\mathcal{P}_\bullet$ component of $B(\pi)$, given by

$$\begin{aligned}&+[P(1,4)|P(2,3,4)] + [P(3,4)|P(1,2,3)] - [P(1,2,4)|P(3,2)] - [P(1,3,4)|P(2,1)]\\ &+[P(1,2,4)|P(3,4)] + [P(1,3,4)|P(2,3)] + [P(2,3,4)|P(1,2)] + [P(1,4)|P(3,2,1)],\end{aligned}$$

and $\Pi_{1,1,1}$ is the $\mathcal{P}_\bullet|\mathcal{P}_\bullet|\mathcal{P}_\bullet$ component of $B(\pi)$, given by

$$\begin{array}{lll}+[P(1,4)|P(2,4)|P(3,4)] & +[P(3,4)|P(1,3)|P(2,3)] & -[P(2,4)|P(1,2) \sqcup P(3,2)]\\ +[P(1,4)|P(3,1)|P(2,1)] & +[P(1,4)|P(3,4)|P(2,3)] & +[P(3,4)|P(2,3)|P(1,2)]\\ +[P(1,4)|P(2,1)|P(3,2)] & -[P(1,4)|P(2,1) \sqcup P(3,4)] & -[P(1,4)|P(2,4)|P(3,2)]\\ -[P(3,4)|P(1,3)|P(2,1)] & +[P(2,4)|P(1,2) \sqcup P(3,4)] & -[P(1,4)|P(3,1)|P(2,3)].\end{array}$$

We note that

$$\Pi_3 \in \mathcal{P}_3, \quad \Pi_{1,2} \in \mathcal{P}_2|\mathcal{P}_1 \oplus \mathcal{P}_1|\mathcal{P}_2, \quad \text{and} \quad \Pi_{1,1,1} \in \mathcal{P}_1|\mathcal{P}_1|\mathcal{P}_1.$$

The symbol of a multiple polylogarithm can be seen as the final (or 'maximal') part of the bar construction of the relevant polygon after taking $P(*,*)$ to $\mu(P(*,*))$. When comparing multiple polylogarithms under the symbol map we can view this as projecting to the final part of the bar construction and working there.

1.4.2 Differential structure of multiple polylogarithms and the symbol

We give an example of how the differential structure of a polylogarithm is reflected in the symbol by examining $dI_{1,1}(x_1, x_2)$.

The statement of Theorem 2.1 in [Gon01] gives us

$$\begin{aligned} &dI(x_0; x_1, \dots, x_m; x_{m+1}) \\ &\quad = \sum_{i=1}^{m} I(x_0; x_1, \dots, \hat{x}_i, \dots, x_m; x_{m+1}) \cdot (d\log(x_{i+1} - x_i) - d\log(x_{i-1} - x_i)). \end{aligned}$$

We recall that

$$I_{1,1}(x_1, x_2) = I(0; x_1, x_2; 1)$$

and so, applying the theorem we get

$$\begin{aligned} dI_{1,1}(x_1, x_2) =& I_1(x_2)(d\log(x_2 - x_1) - d\log(-x_1)) \\ &+ I_1(x_1)\big(d\log(1 - x_2) - d\log(x_1 - x_2)\big) \\ =& I_1(x_2) d\log\left(1 - \frac{x_2}{x_1}\right) \\ &+ I_1(x_1)\big(d\log(1 - x_2) - d\log(-x_2) - d\log(x_1 - x_2) + d\log(-x_2)\big) \\ =& I_1(x_2) d\log\left(1 - \frac{x_2}{x_1}\right) \\ &+ I_1(x_1) d\log\left(1 - \frac{1}{x_2}\right) - I_1(x_1) d\log\left(1 - \frac{x_1}{x_2}\right) \end{aligned}$$

By now applying that $\log(1 - \frac{1}{x}) = -\mathrm{Li}_1\left(\frac{1}{x}\right) = I_1(x)$,

$$\begin{aligned} dI_{1,1}(x_1, x_2) =& I_1(x_1) dI_1(x_2) - I_1(x_1) dI_1\left(\frac{x_2}{x_1}\right) + I_1(x_2) dI_1\left(\frac{x_1}{x_2}\right) \\ =& \log\left(1 - \frac{1}{x_1}\right) d\log\left(1 - \frac{1}{x_2}\right) - \log\left(1 - \frac{1}{x_1}\right) d\log\left(1 - \frac{x_1}{x_2}\right) \\ &+ \log\left(1 - \frac{1}{x_2}\right) d\log\left(1 - \frac{x_2}{x_1}\right). \end{aligned}$$

We recall from Example 1.42 that the symbol for $I_{1,1}(x_1, x_2)$ is

$$\mathcal{S}(I_{1,1}(x_1, x_2)) = +\big(1 - \tfrac{1}{x_1}\big) \otimes \big(1 - \tfrac{1}{x_2}\big) - \big(1 - \tfrac{1}{x_1}\big) \otimes \big(1 - \tfrac{x_1}{x_2}\big) + \big(1 - \tfrac{1}{x_2}\big) \otimes \big(1 - \tfrac{x_2}{x_1}\big),$$

which is clearly very similar to $dI_{1,1}(x_1, x_2)$.

1.5 A simple element in the kernel of the symbol map

As discussed in Chapter 0, we look for a method for finding relations between multiple polylogarithms that is easier than purely analytical methods. As said at the beginning of this chapter, conjecturally, any functional equation between multiple polylogarithms must be in the kernel of the symbol map. We now give an example of a linear combination of multiple polylogarithms in the kernel of the symbol map. The combination arises from Hölder convolution, which we introduce now (it will also be used later in Chapter 5).

1.5.1 Hölder convolution

Hölder convolution (given as Equation 7.1 in [BBBL01]) gives us, in terms of the definition of multiple polylogarithms $G(a_1, \dots, a_n; x)$ in Definition 0.9, that,

$$G(x_w, \dots, x_1; 1) = \sum_{k=0}^{w} (-1)^k G\left(1 - x_1, \dots, 1 - x_w; 1 - \frac{1}{p}\right) G\left(x_{k+1}, \dots, x_w; \frac{1}{p}\right)$$

$\forall p \in \mathbb{C}^*$, and where $x_1 \neq 1$ and $x_w \neq 0$.

We are particularly interested in the limiting case, $p \to \infty$, of Hölder convolution, giving a duality on multiple polylogarithms of

$$G(x_w, \dots, x_1; 1) = (-1)^w G(1 - x_1, \dots, 1 - x_w; 1),$$

which is the same as

$$I_{1,\dots,1}(x_1, \dots, x_w) = (-1)^w I_{1,\dots,1}(1 - x_w, \dots, 1 - x_1).$$

Hölder convolution gives us an entire family of functional equations between multiple polylogarithms. Therefore, conjecturally, we will have

$$\mathcal{S}\left(I_{1,\dots,1}(x_1, \dots, x_w) + (-1)^w I_{1,\dots,1}(1 - x_w, \dots, 1 - x_1)\right) = 0$$

for all $x_1 \neq 1$ and $x_w \neq 0$. We now prove a very simple case of this, and it serves as a nice introduction to linear combinations of multiple polylogarithms in the kernel of the symbol map, albeit one that is as a direct consequence of Hölder convolution.

Example 1.50. We show that

$$I_{1,1}(x,y) - I_{1,1}(1-y,1-x)$$

lies in the kernel of the symbol map. We use the symbol for $I_{1,1}(x,y)$ calculated in Example 1.42 and see that

$$\begin{aligned}
&\mathcal{S}\Big(I_{1,1}(x,y) - I_{1,1}(1-y,1-x)\Big) \in \mathcal{I}_2(S)\\
&\quad = +\Big(\big(1-\tfrac{1}{y}\big)\otimes\big(1-\tfrac{y}{x}\big)\Big) + \Big(\big(1-\tfrac{1}{x}\big)\otimes\big(1-\tfrac{1}{y}\big)\Big) - \Big(\big(1-\tfrac{1}{x}\big)\otimes\big(1-\tfrac{x}{y}\big)\Big)\\
&\qquad + \Big(\big(1-\tfrac{1}{1-x}\big)\otimes\big(1-\tfrac{1-x}{1-y}\big)\Big) + \Big(\big(1-\tfrac{1}{1-y}\big)\otimes\big(1-\tfrac{1}{1-x}\big)\Big)\\
&\qquad - \Big(\big(1-\tfrac{1}{1-y}\big)\otimes\big(1-\tfrac{1-y}{1-x}\big)\Big)\\
&\quad = +\left(\frac{y-1}{y}\otimes\frac{x-y}{x}\right) + \left(\frac{x-1}{x}\otimes\frac{y-1}{y}\right) - \left(\frac{x-1}{x}\otimes\frac{y-x}{y}\right)\\
&\qquad + \left(\frac{x}{x-1}\otimes\frac{y-x}{y-1}\right) + \left(\frac{y}{y-1}\otimes\frac{x}{x-1}\right) - \left(\frac{y}{y-1}\otimes\frac{x-y}{x-1}\right)\\
&\quad = +\left(\frac{x-1}{x}\otimes\left(\frac{y-x}{y-1}\cdot\frac{y-1}{y}\right)\right) - \left(\frac{x-1}{x}\otimes\frac{y-x}{y}\right)\\
&\qquad - \left(\frac{y}{y-1}\otimes\left(\frac{x-y}{x}\cdot\frac{x}{x-1}\right)\right) + \left(\frac{y}{y-1}\otimes\frac{x-y}{x-1}\right) = 0.
\end{aligned}$$

We can see that even for a two term, weight 2, linear combination of multiple polylogarithms, with simple arguments, the symbol calculation is already fairly unwieldy.

The object of main interest in this thesis will be linear combinations of multiple polylogarithms in the kernel of the symbol map.

Chapter 2

Hook-arrow trees

We introduce a new method of finding the symbol of a multiple polylogarithm called the *hook-arrow tree*. The symbol found is equivalent to the polygon and binary tree methods described in Chapter 1 (bijections between them will be given in Chapter 3). The hook-arrow tree has been developed so as to provide an algorithm for finding symbols that can be done on the computer package GP/Pari [PAR11]. Later we also use the hook-arrow tree to find the symbol of coloured multiple zeta values (CMZV) and simplify symbol calculations for given depths of multiple polylogarithms.

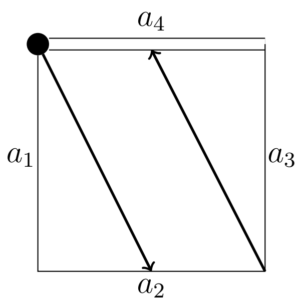
We also introduce a way of viewing the structure of the symbol of a multiple polylogarithm which takes the form of a *ternary tree*.

2.1 Motivation for hook-arrow trees

To find the symbol of a multiple polylogarithm using a computer we face the problem of encoding a visual method into computer code. This involves relatively simple problems such as how to encode an arrow, more difficult problems such as how to check arrows do not cross, and even harder problems like checking all possible arrows are exhausted without duplication.

The arrows in the dissection of a polygon start at a vertex v and end at a non-adjacent edge e. A natural idea would be to encode this as a vector `[v,e]`. We give a possible notation in the following example.

Example 2.1. One possible dissection of the 4-gon $P(a_1, a_2, a_3, a_4)$ of the form



could be encoded as the vector

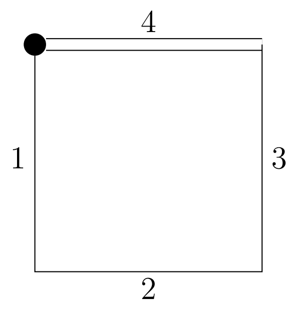
```
[Edge labels, Arrows] = [[a1,a2,a3,a4],[[1,a2],[3,a4]]].
```

However, there are problems with this method of encoding. Firstly, it is quite difficult to generate all possible arrows and check whether they cross. Secondly, to calculate the dissection we need to retract along the arrows which can cause the labels to change order. It also proves very difficult to discern when a dissection is a shuffle.

2.2 Moving from polygons to trees

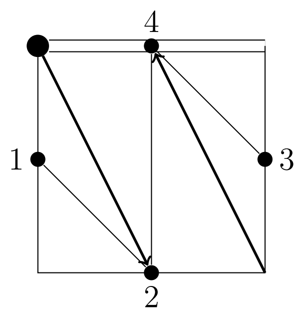
We will define and construct a *hook-arrow tree* from a possible maximal dissection of a polygon. In [GGL09], the authors map trees to polygons. However, the reader should note that the hook-arrow trees we construct here are different (albeit representing similar information) to the trees in [GGL09] and are designed to represent terms in the symbol attached to the polygon.

Remark 2.2. For simplicity, we will sometimes consider all edges of polygons (and then vertices of the trees) to have numerical labelling. The sides will be labelled such that the root side is labelled n on an n-gon. For example, we label a 4-gon $P(1,2,3,4)$ anticlockwise with the final label, 4, being the root side.



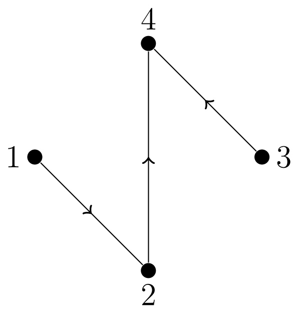
The labels can easily be substituted for algebraic values when terms in the symbol are needed.

Every maximal dissection of a polygon uniquely defines a certain spanning tree τ on the vertices which are the midpoints of the polygon sides, and vice-versa. These vertices, $v_1, ..., v_n$, inherit the label of the side they sit on, and form the vertices of τ. We induce the edges of τ as all possible lines, between the v_i, that do not cross arrows from the dissection. Here is an example of a maximal dissection of a 4-gon with the spanning tree induced:



We also induce a root on τ as the vertex lying on the final side of the polygon.

We will always assume that an orientation of the edges can be induced on a hook-arrow tree as being towards towards the root. For the above example of a dissected 4-gon the rooted spanning tree is:



The edges of the tree will not cross by construction; we define interlacing to reflect this for use in the definition of a hook-arrow tree.

Definition 2.3. A graph with a linear order on its vertices w_j is said to be **interlaced** if there exists a choice of four vertices $w_1 < ... < w_4$ such that both edges $\{w_1, w_3\}$ and $\{w_2, w_4\}$ are contained in the graph.

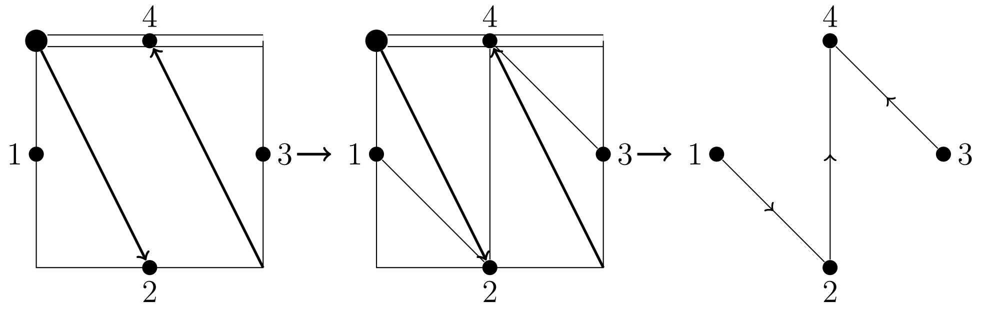
We now give a formal definition of a hook-arrow tree and illustrate the definition with an example.

Definition 2.4. A **hook-arrow tree** is a rooted spanning tree on a set of vertices in a linear order, $(v_1, \ldots, v_n)$, which is not interlaced and has root v_n. The edges are directed towards v_n.

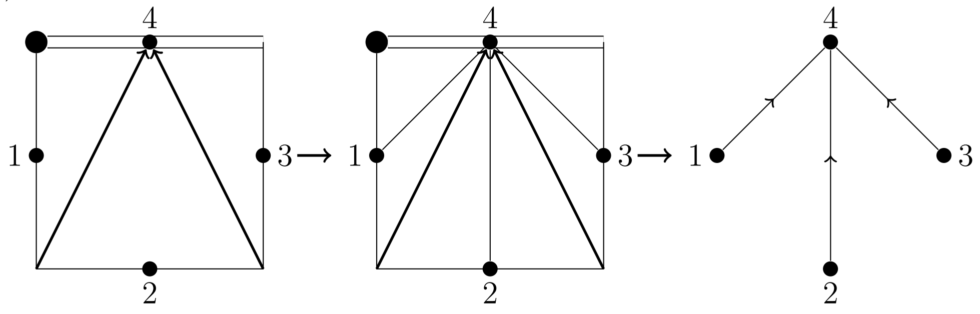
We can think of a hook-arrow tree as being a tree, embedded in a plane, on vertices arranged in a circle.

Example 2.5. The following two examples show the process of moving between a fully dissected 4-gon and a hook-arrow tree.

1)



2)



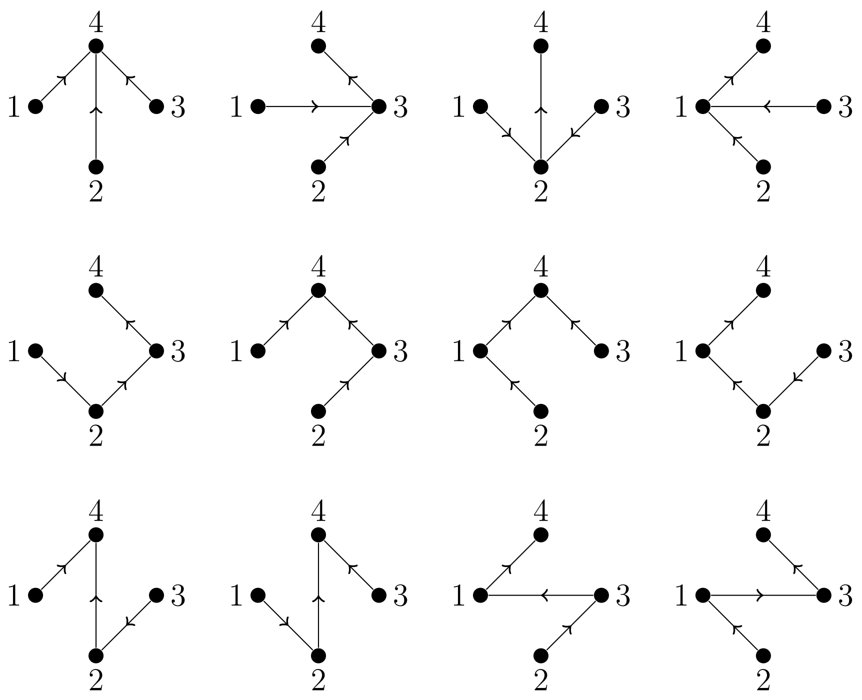
We at once see that each edge of a hook-arrow tree represents a 2-gon resulting from the full dissection of a polygon. The first polygon above represents the term

$$\mu(P(2,4)) \otimes \mu(P(1,2)) \sqcup \mu(P(3,4))$$

and the second represents

$$\mu(P(1,4)) \otimes \mu(P(2,4)) \otimes \mu(P(3,4)).$$

Example 2.6. We give all possible hook-arrow trees on 4 vertices directly by exhaustion of trees that satisfy Definition B.3.



We note that, as with the dissection of a 4-gon in Example 1.32, we have 12 possible hook-arrow trees on 4 vertices.

There is a bijection between the set of all hook-arrow trees on n points in a fixed general position and the set of polygons with maximal dissection. The map of the bijection is outlined in the Definition B.3. However, a bijection between hook-arrow trees and polygon dissections is given in Chapter 3, although it is important to note that the bijection is in fact between a slightly different form of these pictorial representations, namely ones that isolate a single term in the symbol and so do not contain shuffles.

To see fully where hook-arrow trees can be of benefit in finding the symbol attached to a multiple polylogarithm we must describe the analogous method to finding the symbol.

2.3 Obtaining terms in the symbol from a hook-arrow tree

We take a hook-arrow tree, τ, with an ordered vertex set $V_\tau = (v_1^\tau, ..., v_n^\tau)$, a distinguished final vertex v_n^τ (from the strict order on the vertices) and a set of edges $E_\tau = \{e_1^\tau, ..., e_{n-1}^\tau\}$. We must first give a way to write this information without needing to draw the tree.

Notation 2.7. We will use the notation that $\{v_i, v_j\}$ denotes an undirected edge, and so the vertices are given in their original ordering, and that $[v_i, v_j]$ denotes a directed edge from v_i to v_j.

We will understand v_{t+1}^τ to be the vertex directly after v_t^τ in the linear order of vertices of the hook-arrow tree τ.

Definition 2.8. For an hook-arrow tree with n points $V_\tau = (v_1^\tau, ..., v_n^\tau)$ and a set of edges $E_\tau = \{e_1^\tau, ..., e_{n-1}^\tau\}$ then its **description** is

$$D_\tau := \{V_\tau, E_\tau\} = \{(v_1^\tau, ..., v_n^\tau), \{e_1^\tau, ..., e_{n-1}^\tau\}\}$$

The algorithm will involve subtrees of the hook-arrow tree τ. We can display the information for these subtrees in the same way (they themselves will also be hook-arrow trees). We note that although only a subset of the vertices $(v_1^\tau, ..., v_n^\tau)$ will be contained in the subtree, these will be given an order and a final vertex and so still fit into the framework of the description of a full hook-arrow tree.

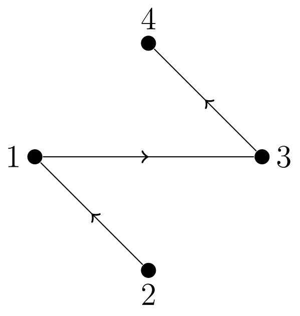
While we have given a linear order on the vertices of a hook-arrow tree, we have not given an order to the edges. The algorithm (to be given below) for extracting terms in the symbol from a hook-arrow tree is effectively a method for choosing a partial order on the edges, but we do not include this in the description D_τ.

The edges in the description are also not given with a direction, but since directing edges towards a particular vertex on a tree is unique, for simplicity, we will not include this in the description. We will display the edges in the description in lexicographic order.

Example 2.9. Two examples of this description of a hook-arrow tree are

$$\{(1,2,3,4), \{\{1,2\}, \{1,3\}, \{3,4\}\}\}$$

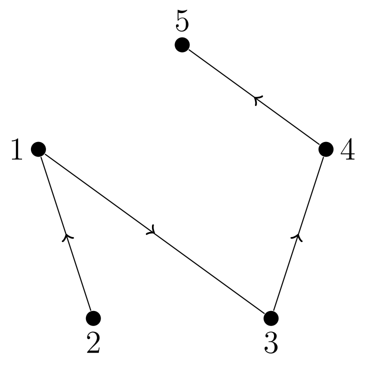
which describes the tree



and

$$\{(1,2,3,4,5), \{\{1,2\}, \{1,3\}, \{3,4\}, \{4,5\}\}\}$$

which describes the tree



We now outline the method of selecting the edges of a hook-arrow tree in a partial order so as to form a tensor product term in the symbol. This involves three steps, and is iterative.

2.3.1 Step 1: Selection of first distinguished edge

We start with a hook-arrow tree, τ, with n vertices and a description $D_\tau = \{V_\tau, E_\tau\}$ as above. We select the unique distinguished edge of τ to be

$$e_d^\tau = [v_a^\tau, v_n^\tau] \in E_\tau$$

where v_n^τ is the final vertex and v_a^τ is the only member of the set

$$\{v_i^\tau \in V_\tau \quad | \quad \{v_i^\tau, v_n^\tau\} \in E_\tau \text{ and } \{\{v_1^\tau, v_n^\tau\}, \ldots, \{v_{(i-1)}^\tau, v_n^\tau\}\} \notin E_\tau\}.$$

We note that the edge e_τ is directed from v_a^τ to v_n^τ.

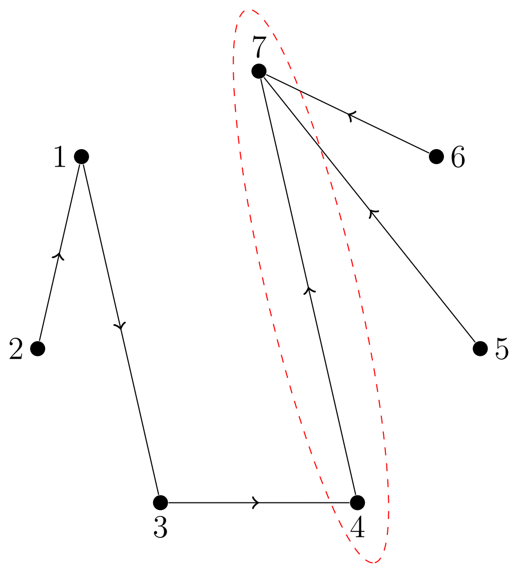
Intuitively, the edge e_d^τ can be thought of as the edge in E_τ connecting v_n^τ with a vertex, v_a^τ, where a is the lowest possible. Or equivalently, if we arrange the vertices in a circle ordered anticlockwise, the first edge hit by moving from outside the circle anticlockwise around the distinguished vertex v_n^τ.

Example 2.10. For a tree τ' with description

$$D_{\tau'} = \{(1,2,3,4,5,6,7), \{\{1,2\},\{1,3\},\{3,4\},\{4,7\},\{5,7\},\{6,7\}\}\}$$

we find that $v_a^{\tau'} = 4$, since $\{4,7\} \in E_{\tau'}$ and $\{\{1,7\},\{2,7\},\{3,7\}\} \notin E_{\tau'}$. We do not choose $\{5,7\}$ to be the distinguished edge because if $v_a^{\tau'} = 5$ then $\{v_{(a-1)}^{\tau'}, v_n^{\tau'}\} = \{4,7\}$ and $\{4,7\} \in E_{\tau'}$. Similarly for $\{6,7\}$.

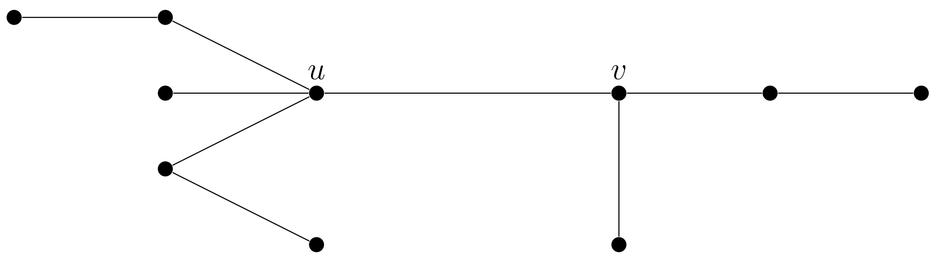
Overall we have that $e_d^\tau = \{4, 7\}$. The following diagram shows the tree τ' where the distinguished edge, e_d^τ, is ringed.



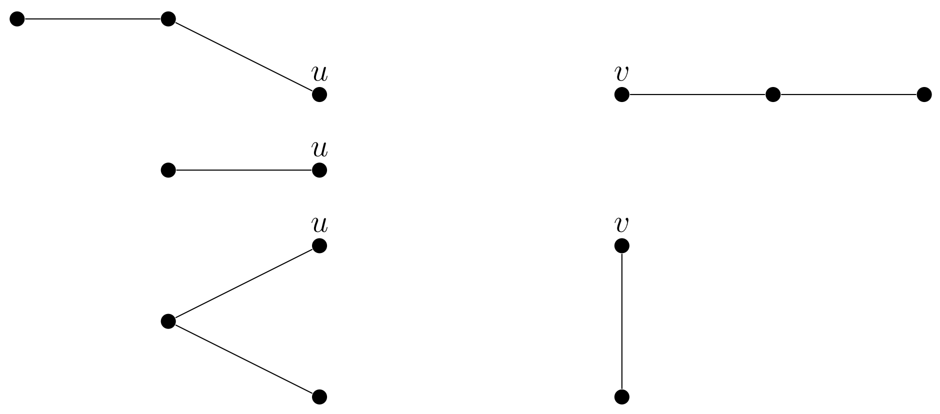
2.3.2 Step 2: Splitting the tree

We will now use our choice of distinguished edge to split the tree into the edge e_d^τ and (up to) three subtrees. First, we give an example of how one might go about splitting a tree into subtrees by removing an edge. However, we will require a slightly different method.

Example 2.11. For a general tree, you can define a set of subtrees by 'removing' an edge. This is done by removing an edge and its bounding vertices and 'breaking' the tree at either end. We then restore the vertices to the created subtrees. For example, for the tree



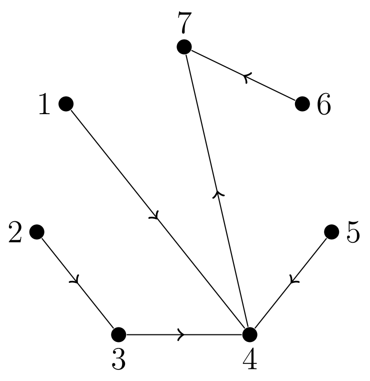
if we remove the edge $\{u, v\}$, break up the edges connected to u and v at either end, and then restore the vertices u and v, we will be left with the subtrees



The splitting we require for hook-arrow trees is slightly different to the above example, although fairly similar.

We will remove the distinguished edge $e_d^\tau = [v_a^\tau, v_n^\tau]$ from a hook arrow tree τ to form subtrees, but we do not wish to entirely split up every edge connected to v_a^τ and v_n^τ. By removing e_d^τ we will create up to three subtrees defined and labelled in the following section. First, however, so as to give the reader an idea of the desired subtrees, we provide an example.

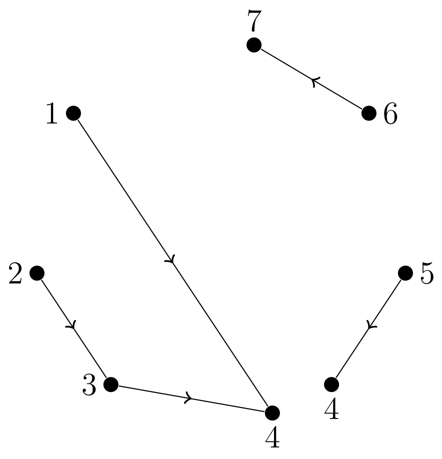
Example 2.12. The hook-arrow tree, τ, of the form



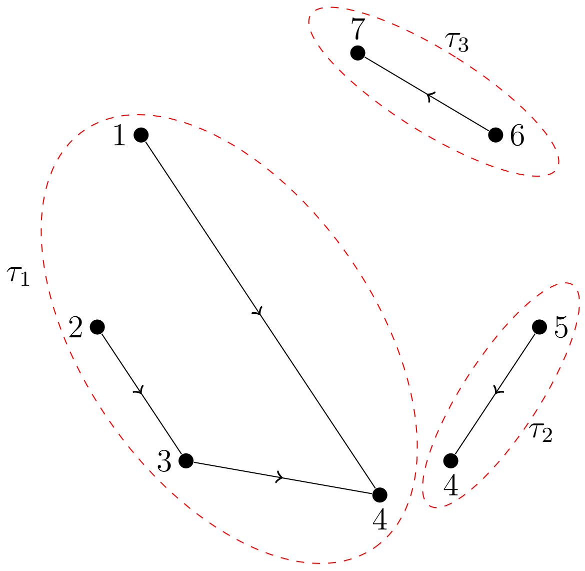
described by

$$\{(1, 2, 3, 4, 5, 6, 7), \{\{1, 4\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{4, 7\}, \{6, 7\}\}\}$$

has distinguished edge $\{4, 7\}$ and by removing this edge we split the tree into three subtrees (in a method to be described below) as follows:



We then label these τ_1, τ_2 and τ_3. These are shown circled in the following diagram.



The difference to the splitting in Example 2.11 is that there τ_1 would have become two subtrees. In a sense we have 'divided' the hook-arrow tree into two halves by cutting out the distinguished edge and then labelled the remaining parts, thus τ_1 remains as one. This will be now formally defined.

2.3.2.1 Formally defining the subtrees τ_1, τ_2 and τ_3

To construct τ_1, τ_2 and τ_3 formally we first isolate a vertex v_x^τ of a hook arrow tree τ with vertices V_τ, edges E_τ and distinguished edge $e_d^\tau = \{v_a^\tau, v_n^\tau\}$. We define x to be the maximum value $i \in \{a, a+1, \ldots, n-2, n-1\}$ such that the direct route from v_i^τ to v_n^τ passes through v_a^τ. Note that if the maximum value of i is equal to a then we set $v_x^\tau = v_a^\tau$.

We can now define the subtrees τ_1, τ_2 and τ_3 by the following descriptions.

$$D_{\tau_1} = \{V_{\tau_1}, E_{\tau_1}\}, \quad D_{\tau_2} = \{V_{\tau_2}, E_{\tau_2}\}, \quad D_{\tau_3} = \{V_{\tau_3}, E_{\tau_3}\},$$

where

$$\begin{aligned} V_{\tau_1} &= (v_1^\tau, v_2^\tau, ..., v_a^\tau), \\ V_{\tau_2} &= (v_x^\tau, v_{(x-1)}^\tau, ..., v_{(a+1)}^\tau, v_a^\tau), \\ V_{\tau_3} &= (v_{(x+1)}^\tau, v_{(x+2)}^\tau, ..., v_{(n-1)}^\tau, v_n^\tau) \end{aligned}$$

and

$$E_{\tau_i} = \{\{v_p^\tau, v_q^\tau\} \in E_\tau \quad | \quad v_p^\tau \in V_{\tau_i} \text{ and } v_q^\tau \in V_{\tau_i}\}.$$

Remark 2.13. It is important to note here that the linear order on the vertices in V_{τ_2} has been reversed from the order of those vertices in V_τ.

Definition 2.14. A subtree of a hook-arrow tree is a **trivial subtree** if it contains only one vertex.

We will, from now on, disregard subtrees that are trivial. We will show the above creation of subtrees in Example 2.16.

Remark 2.15. We note that now we have defined the τ_i there is another way of viewing the definition of the vertex v_x^τ. Note, though, that we could not define the v_x^τ in the following way (as circular logic would occur).

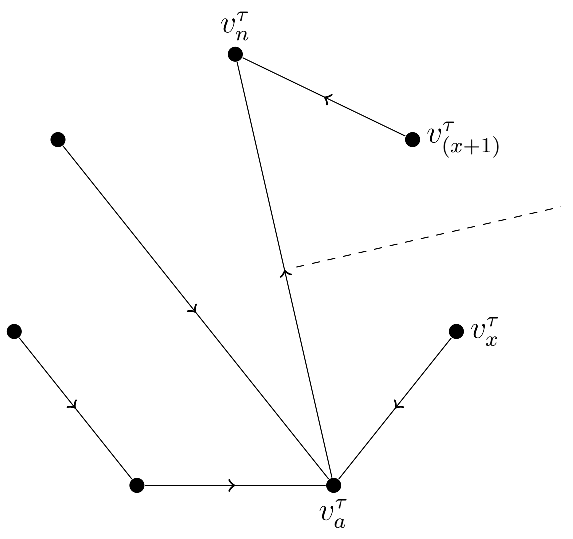
Every point of the original tree τ will be contained in at least one of the subtrees τ_i. The vertices between v_a^τ and v_n^τ in the linear order of V_τ will be the, possibly empty, ordered set

$$S_\tau = \{v^\tau_{(a+1)}, ..., v^\tau_{(n-1)}\}.$$

These points will be, by construction, contained in the, possibly trivial, subtrees τ_2 and τ_3. Because no edges cross, and because every point in S_τ will definitely be contained in one of these two subtrees, there will exist a vertex $v^\tau_x \in S_\tau$, for which

$$\{v^\tau_{(a+1)}, ..., v^\tau_x\} \begin{array}{c} \in \tau_2 \\ \\ \notin \tau_3 \end{array}, \text{ and } \{v^\tau_{(x+1)}, ..., v^\tau_{(n-1)}\} \begin{array}{c} \in \tau_3 \\ \\ \notin \tau_2. \end{array}$$

This can be viewed, in the case of τ in Example 2.12, as

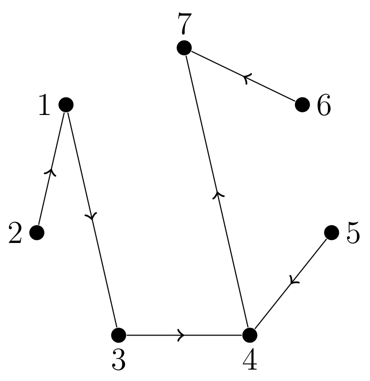


where the dashed line denotes, in a sense, the division between vertices in τ_2 and τ_3.

Example 2.16. We use a hook-arrow tree, τ with description

$$D_\tau = \{V_\tau, E_\tau\} = \{(1,2,3,4,5,6,7), \{\{1,2\},\{1,3\},\{3,4\},\{4,5\},\{4,7\},\{6,7\}\}\}$$

taking the form



The hook-arrow tree has distinguished edge $e_d^\tau = \{4,7\}$. The vertex v_x^τ will be the vertex labelled 5.

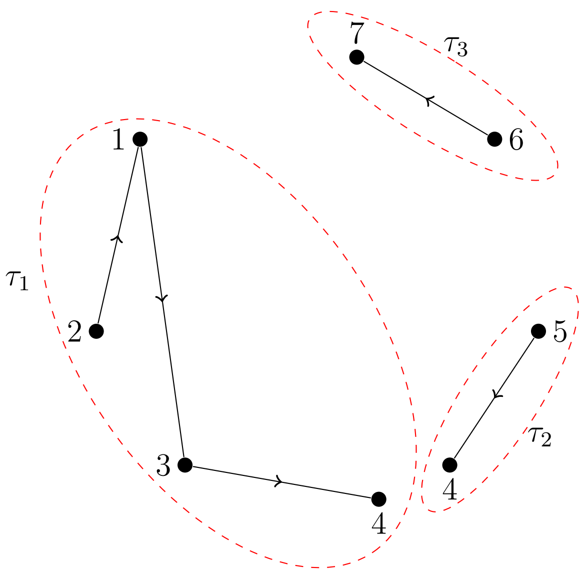
We therefore describe the subtrees τ_1, τ_2 and τ_3 by

$$\begin{aligned} D_{\tau_1} &= \{V_{\tau_1}, E_{\tau_1}\} \\ &= \{(1,2,3,4), \{\{1,2\},\{1,3\},\{3,4\}\}\} \end{aligned}$$

$$\begin{aligned} D_{\tau_2} &= \{V_{\tau_2}, E_{\tau_2}\} \\ &= \{(5,4), \{\{4,5\}\}\} \end{aligned}$$

$$\begin{aligned} D_{\tau_3} &= \{V_{\tau_3}, E_{\tau_3}\} \\ &= \{(6,7), \{\{6,7\}\}\} \end{aligned}$$

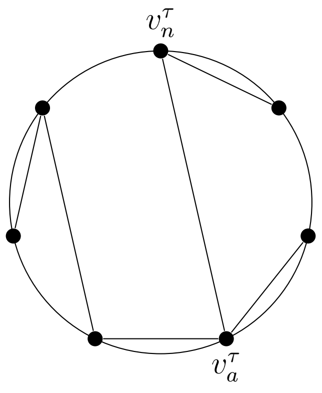
which gives



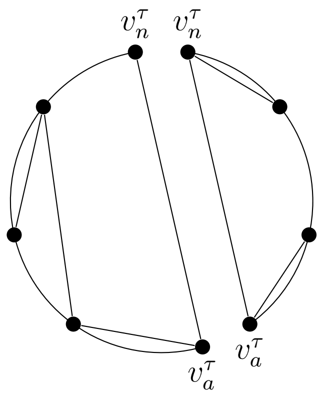
2.3.2.2 A visual method for forming τ_1, τ_2 and τ_3

As an alternative to the above method of splitting a hook-arrow tree into subtrees we give a visual, less formal, method. Each step includes a running example of τ from Example 2.16.

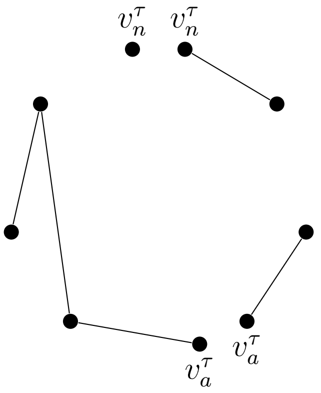
Firstly, introduce a construction circle through the ordered vertices (when arranged in order in a circle).



Next, split the circle into two circle segments along the distinguished edge $e_d^\tau = \{v_a^\tau, v_n^\tau\}$ leaving a copy of e_d^τ on each segment.

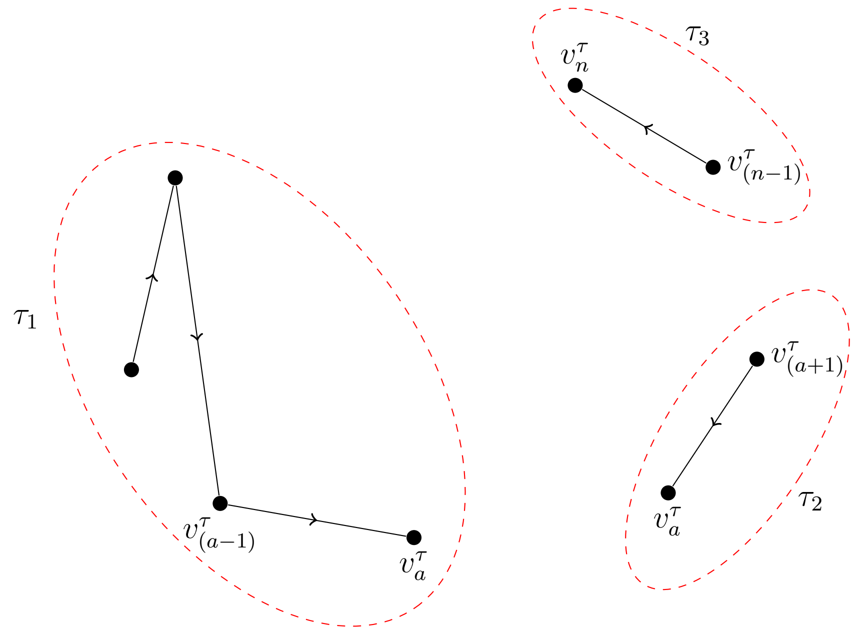


Now remove all remaining parts of the construction circle. Also, remove the distinguished edge e_d^τ (both parts), but not its vertices, $\{v_a^\tau, v_n^\tau\}$.



We remove any trivial subtrees (subtrees with only one vertex). Finally, we label the subtrees (again noting there may be less than three) in the following way.

- τ_1 - The subtree containing at least v_a^τ and $v_{(a-1)}^\tau$.
- τ_2 - The subtree containing at least v_a^τ and $v_{(a+1)}^\tau$.
- τ_3 - The subtree containing at least v_n^τ and $v_{(n-1)}^\tau$.



We are left with the required subtrees.

2.3.3 Step 3: Iterative step

We have created three subtrees, τ_1, τ_2, τ_3, with descriptions, $D_{\tau_1}, D_{\tau_2}, D_{\tau_3}$, of the same form as the original hook-arrow tree, τ. We can therefore iterate the two previous steps (isolate a distinguish edge and then split into more subtrees) above for each of the new subtrees created, until every new subtree is trivial.

Remark 2.17. Each iteration applies steps 1 and 2 to all non-trivial subtrees and puts aside distinguished edges. We will explain how to track (and record) the distinguished edges created in Section 2.3.4.

Proposition 2.18. *The iterative process will terminate for finite hook-arrow trees on n vertices in fewer than n iterations.*

Proof. Because the distinguished edge of a previous iteration is not in any of the resulting subtrees, each new and non-trivial subtree will have a distinguished edge which is different to every other previous distinguished edge. The subtrees inherit a distinguished edge from its supertree, so since the tree is connected, every edge will become a distinguished edge in some iteration. Therefore, for a finite tree there will be an iterative step when every remaining subtree is trivial, and the process ends.

Since at least one edge becomes a distinguished edge in each iteration, and since a hook-arrow tree on n vertices has $n-1$ edges, the iterative process will end in fewer than n iterations. □

Remark 2.19. When the iteration ends, every edge will have gained a direction. The direction on the edges will correspond to the direction on the tree towards the original distinguished vertex, b_τ, as expected.

2.3.4 Recording the results of the algorithm and definition of the symbol

We now explain how to record the results of the algorithm (steps 1, 2 and 3 above) so as to give the symbol attached to the hook-arrow tree. We begin by giving the algebraic expression attached to a directed edge (directed towards v_n^τ) of a hook-arrow tree, τ. Since an edge of a hook-arrow tree represents the same information as a 2-gon, we merely extend Definition 1.39.

Definition 2.20. For a directed edge of a hook-arrow tree, $[a,b]$ we define

$$\mu([a,b]) := \mu(P(a,b)),$$

where the map μ is defined as in Definition 1.39.

Remark 2.21. Due to the direction on each edge being inherent, we understand that whenever μ is applied to an edge, the direction of the edge is obtained from the direction towards v_n^τ on a hook-arrow tree, τ.

We start with a hook-arrow tree τ on n vertices. We apply steps 1 and 2 of the above algorithm, but record the distinguished edge of τ and the subtrees created as

$$T = \{e_d^\tau = [v_a^\tau, v_n^\tau], \tau_1, \tau_2, \tau_3\}.$$

As specified in Step 3 we iterate the algorithm on the subtrees τ_1, τ_2, and τ_3. We record in the same way and obtain T_1 for τ_1, T_2 for τ_2 and T_3 for τ_3. These will take the form

- $T_1 = \{e_d^{\tau_1}, \tau_{11}, \tau_{12}, \tau_{13}\}$,
- $T_2 = \{e_d^{\tau_2}, \tau_{21}, \tau_{22}, \tau_{23}\}$,
- $T_3 = \{e_d^{\tau_3}, \tau_{31}, \tau_{32}, \tau_{33}\}$.

We continue and apply steps 1 and 2 on all the τ_{ij} which are non-trivial (obtaining $T_{11}, T_{12}, \ldots$). We continue until all subtrees are trivial (when the iterative process terminates).

To obtain the required symbol from the $T_{i_1,\ldots,i_k}$ we first write

$$\mu([v_a^\tau, v_b^\tau]) \quad \otimes \quad T_1 \quad ⧢ \quad T_2 \quad ⧢ \quad T_3$$

and then systematically replace each $T_{i_1,\ldots,i_k}$ with

$$\left(\mu([v_a^{\tau_{i_1,\ldots,i_k}}, v_b^{\tau_{i_1,\ldots,i_k}}]) \quad \otimes \quad T_{i_1,\ldots,i_k,1} \quad ⧢ \quad T_{i_1,\ldots,i_k,2} \quad ⧢ \quad T_{i_1,\ldots,i_k,3}\right)$$

until we have covered all $T_{i_1,\ldots,i_r}$.

We label the resulting tensor product $\mathrm{Alg}(\tau)$.

2.3.5 The sign of a hook-arrow tree

We now give a way to determine the sign of each hook-arrow tree, as in the polygon dissection. This is done by counting the number of 'backwards' edges (again, similar to backwards arrows in a polygon dissection). The sign is then determined by taking -1 to the power of this number.

Definition 2.22. Let τ be a hook-arrow tree with description

$$D_\tau = \{V_\tau = (v_1^\tau, \dots, v_n^\tau), E_\tau = \{e_1^\tau, \dots, e_{n-1}^\tau\}\}.$$

An edge $e_i^\tau = [v_s^\tau, v_t^\tau]$, directed from v_s^τ to v_t^τ towards v_n^τ, is said to be a **backwards edge** if $s > t$.

Definition 2.23. The **sign of a hook-arrow tree**, written $\mathrm{sgn}(\tau)$, is defined to be

$$\mathrm{sgn}(\tau) = (-1)^m \quad \text{where} \quad m = \#\{e \in E_\tau \mid e \text{ backwards}\}.$$

2.3.6 The definition of the symbol via hook-arrow trees

We can now define the symbol of a multiple polylogarithm, as obtained through hook-arrow trees.

Definition 2.24. The **symbol** of a multiple polylogarithm, $I_{r_1,\dots,r_s}(x_1, \dots, x_s)$ is defined to be

$$\mathcal{S}(I_{r_1,\dots,r_s}(x_1, \dots, x_s)) = \sum_\tau \mathrm{sgn}(\tau)\mathrm{Alg}(\tau),$$

where the summation runs over all possible hook-arrow trees on the vertices

$$\big(x_1, \underbrace{0, ..., 0}_{r_1-1}, x_2, \underbrace{0, ..., 0}_{r_2-1}, \quad \dots, \quad x_s, \underbrace{0, ..., 0}_{r_s-1}, 1\big)$$

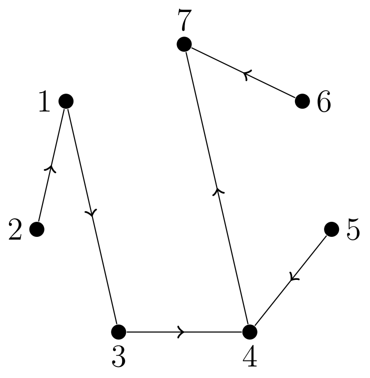
and $\mathrm{Alg}(\tau)$ is as in Section 2.3.4.

The reader is reminded that Chapter 3 explores the bijections between this definition and the definitions of the symbol via binary trees and polygon dissections.

2.3.7 A worked example

The following example gives a full working of $\mathrm{Alg}(\tau)$ of a hook-arrow tree, τ.

Example 2.25. We give an example of the algorithm described. We will again use the tree from Example 2.16,

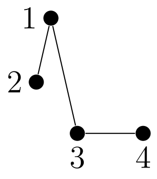


and it is described by

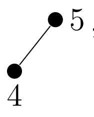
$$D_\tau = \{(1,2,3,4,5,6,7), \{\{1,2\},\{1,3\},\{3,4\},\{4,5\},\{4,7\},\{6,7\}\}\}.$$

We now follow the algorithm described to find $T, T_1, T_2, ...$

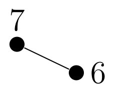
$T = \Big\{ [4,7], \quad \tau_1 =$



$, \quad \tau_2 =$

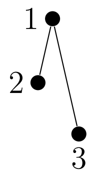


$, \quad \tau_3 =$



$\Big\}$

$T_1 = \Big\{ [3,4], \quad \tau_{11} =$

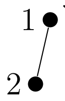


$\Big\}$

$$T_2 = \{[5,4]\}$$

$$T_3 = \{[6,7]\}$$

$T_{11} = \Big\{ [1,3], \quad \tau_{112} =$



$\Big\}$

$$T_{112} = \{[2,1]\}$$

We now combine the $T_{i_1,\ldots,i_k}$.

1. $[4,7] \quad | \quad T_1 \quad ⧢ \quad T_2 \quad ⧢ \quad T_3.$
2. $[4,7] \quad | \quad ([3,4] \quad | \quad T_{11}) \quad ⧢ \quad [5,4] \quad ⧢ \quad [6,7].$
3. $[4,7] \quad | \quad ([3,4] \quad | \quad ([1,3] \quad | \quad T_{112})) \quad ⧢ \quad [5,4] \quad ⧢ \quad [6,7].$
4. $[4,7] \quad | \quad ([3,4] \quad | \quad ([1,3] \quad | \quad [2,1])) \quad ⧢ \quad [5,4] \quad ⧢ \quad [6,7].$

After removing unnecessary parentheses we obtain

$$[4,7] \,|\, ([3,4] \,|\, [1,3] \,|\, [2,1]) \;⧢\; [5,4] \;⧢\; [6,7].$$

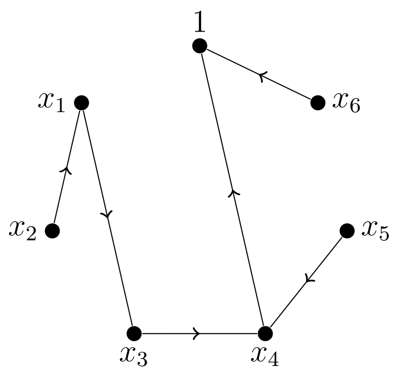
There are two backwards edges in the hook-arrow tree: the edges $[2,1]$ and $[5,4]$. The sign of the term is therefore $(-1)^2$, giving

$$+[4,7] \mid ([3,4] \mid [1,3] \mid [2,1]) \sqcup [5,4] \sqcup [6,7].$$

When finding the symbol of a multiple polylogarithm the labels of a hook-arrow tree representing terms in its symbol will have algebraic labels. Suppose that we relabel the vertices of the above hook-arrow tree from

$$(1,2,3,4,5,6,7) \quad \text{to} \quad (x_1,x_2,x_3,x_4,x_5,x_6,1)$$

to give the hook-arrow tree



Applying the algebraic association to edges gives us

$$\mu([x_4,x_7]) \otimes \Big(\mu([x_3,x_4]) \otimes \mu([x_1,x_3]) \otimes \mu([x_2,x_1])\Big) \sqcup \mu([x_5,x_4]) \sqcup \mu([x_6,x_7])$$
$$= +\left(1-\tfrac{1}{x_4}\right) \otimes \left(\left(1-\tfrac{x_4}{x_3}\right) \otimes \left(1-\tfrac{x_3}{x_1}\right) \otimes \left(1-\tfrac{x_1}{x_2}\right)\right) \sqcup \left(1-\tfrac{x_7}{x_6}\right) \sqcup \left(1-\tfrac{x_4}{x_5}\right)$$

which is the term in a symbol which this particular hook-arrow tree represents. Note that there are two backwards edges and so we have a positive sign.

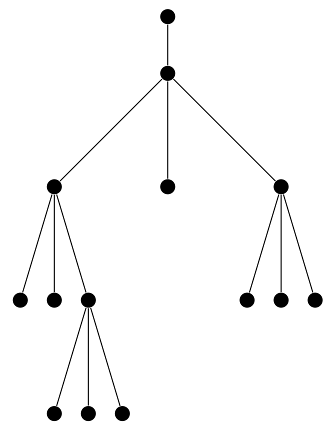
Repeating this for every possible tree on these vertices and formally adding all tensor products will give us the symbol attached to the polygon $P(x_1,x_2,x_3,x_4,x_5,x_6,1)$, which will be the symbol of the multiple polylogarithm $I_{1,1,1,1,1,1}(x_1,x_2,x_3,x_4,x_5,x_6)$.

2.4 Viewing the algorithm as a ternary/4-valent tree

We now define a *ternary tree* which a good way to view the structure of a hook-arrow tree and the symbol, although cannot be easily used to find the symbol itself.

The algorithm described in the previous section may at first appear complicated and unwieldy. However, after applying it a few times the author hopes that the reader will start to see its structure come through. The ternary tree should help with this.

Loosely speaking a ternary tree is a planted tree where every internal vertex has three 'children'. The following diagram shows a typical ternary tree.



We can view our algorithm in this way by considering distinguished edges of an iteration to be internal vertices of the ternary tree. The 'children' of each internal vertex will either be another internal vertex representing the distinguished edge of a subtree (from the next iteration), or an external vertex representing a trivial subtree. There will be exactly the same number of internal vertices in the related ternary tree as there are edges on the hook-arrow tree. The above example would therefore represent a hook-arrow tree with 4 edges, which would arise from the dissection of a pentagon.

We now formally define a ternary tree and construct it from a hook-arrow tree. The definition and isomorphism between ternary trees is adapted to our needs from a paper by Klarner [Kla70].

Remark 2.26. Ternary trees appear in literature in different forms, we will define them here to be 4-valent planted plane trees.

2.4.1 The definition of a ternary tree

Definition 2.27. A **planted plane tree** is a tree given by (V, v_0, E, R), with

1. V - A set of vertices
2. v_0 - A root vertex with $v_0 \in V$.
3. E - a set of edges of the form $\{w_1, w_2\}$ with $w_1, w_2 \in V$.
4. R - A linear order relation on V possessing the following two properties, given a function $\rho(x)$, defined to be the length of the path from v_0 to $x \in V$:
   (a) If $x, y \in V$ and $\rho(x) < \rho(y)$ then $x < y$ in R.
   (b) If $\{r, s\}, \{x, y\} \in E$ with $\rho(r) = \rho(x) = \rho(s) - 1 = \rho(y) - 1$ and $r < x$ in R, then $s < y$ in R.

Definition 2.28. A planted plane tree is said to be k**-valent** if every vertex has either degree 1 or k.

Definition 2.29. A **ternary tree** is a 4-valent planted plane tree.

2.4.2 An isomorphism on planted plane trees

The linear order relation R and its two conditions give us a well-defined order on vertices with the same distance from the route. It also allows us to define an isomorphism on planted plane trees.

Two planted plane trees on the same set of vertices V, given by (V, v, E, R) and (V, w, F, S), are said to be isomorphic if there exists a permutation π of V such that

1. $\pi v = w$.

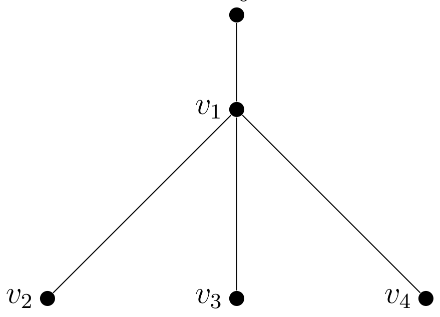
2. $\{\pi x, \pi y\} \in F$ for all $\{x, y\} \in E$.

3. $\pi x < \pi y$ in S for all x, y in R.

This allows us to see that the tree

$$\big(\{v_0, ..., v_4\}, v_0, \{\{v_0, v_1\}, \{v_1, v_2\}, \{v_1, v_3\}, \{v_1, v_4\}\}, v_0 < v_1 < v_2 < v_3 < v_4\big),$$

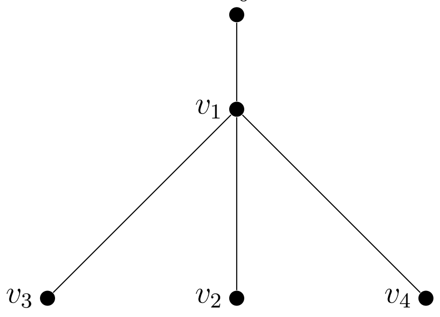
which can be viewed as



is isomorphic to

$$\big(\{v_0, ..., v_4\}, v_0, \{\{v_0, v_1\}, \{v_1, v_2\}, \{v_1, v_3\}, \{v_1, v_4\}\}, v_0 < v_1 < v_3 < v_2 < v_4\big)$$

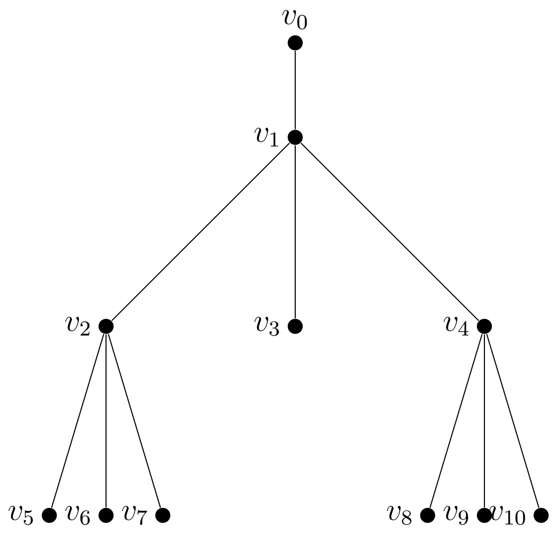
which can be viewed as



with π being the identity except for permuting v_2 and v_3. However the tree

$$\left(\{v_0, ..., v_{10}\}, v_0, \left\{\begin{array}{c} \{v_0, v_1\}, \{v_1, v_2\}, \{v_1, v_3\}, \{v_1, v_4\}, \{v_2, v_5\}, \\ \{v_2, v_6\}, \{v_2, v_7\}, \{v_4, v_8\}, \{v_4, v_9\}, \{v_4, v_{10}\} \end{array}\right\}, v_0 < ... < v_{10}\right)$$

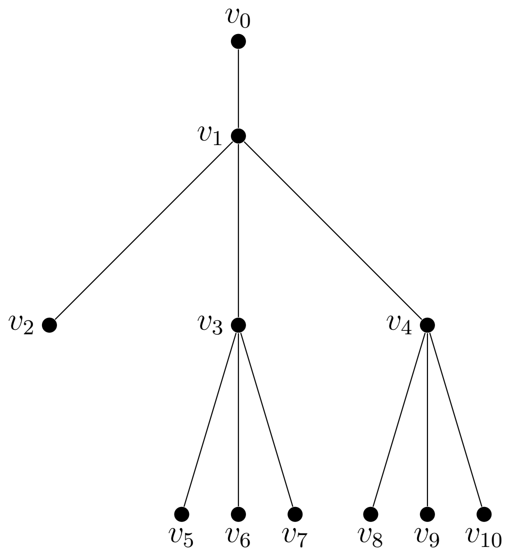
which can be viewed as



is not isomorphic to the tree

$$\left(\{v_0,...,v_{10}\}, v_0, \left\{\begin{array}{c}\{v_0,v_1\},\{v_1,v_2\},\{v_1,v_3\},\{v_1,v_4\},\{v_3,v_5\},\\ \{v_3,v_6\},\{v_3,v_7\},\{v_4,v_8\},\{v_4,v_9\},\{v_4,v_{10}\}\end{array}\right\}, v_0 < ... < v_{10}\right)$$

which can be viewed as



They are not isomorphic as no permutation on the vertices $v_0, ..., v_{10}$ exists that satisfy the required properties. Note that by permuting v_2 and v_3 in either tree, it not only changes the order in R but also the edges attaching the vertices v_5, v_6 and v_7, which does not give the other tree.

2.4.3 Forming a ternary tree from the algorithm on a hook-arrow tree

Remark 2.30. In Chapter 3 we will give a more direct (and possibly simpler) recipe for forming the ternary tree. The method given now is chosen as it gives an indication as to the structure of the symbol map.

We can construct a ternary tree from a non-trivial hook-arrow tree with n vertices by following the algorithm described in Section 2.3 for finding the term in the symbol attached to a hook-arrow tree.

We denote, but largely overlook, a root vertex by r, and attach it to a first internal vertex w_0; we let this vertex represent the initial distinguished edge $[v_a^\tau, v_n^\tau]$ of τ, a hook-arrow tree described in the normal way. The vertex w_0 is then connected by three edges to vertices w_1, w_2, w_3 (which we now create).

In the algorithm we form up to three subtrees, τ_1, τ_2 and τ_3, of the hook-arrow tree after selecting the initial distinguished edge. We note that an order on these subtrees can be inherited from the order on the vertices of the hook-arrow tree (which we have shown in an anti-clockwise direction). For the first subtrees τ_i this will be the order of τ_1, τ_2 and τ_3.

If the subtree τ_i is trivial, we let w_i represent it, and it is an external vertex of the ternary tree. If τ_i is not trivial then w_i is an internal vertex representing the distinguished edge of τ_i and we attach three more vertices, $w_{i,1}, w_{i,2}$, and $w_{i,3}$.

If the subtrees τ_i contain more than one edge then we continue. At this point we must note an important fact. In the algorithm we reversed the order on the vertices in the subtree τ_2 (see Remark 2.13). This will correspond to the order of the tensor components in each term of the symbol. However, for reasons which are made clear in Chapter 3, we do not reflect this in the ordering of the vertices in the ternary tree. Instead we associate the internal vertices of the ternary tree to the edges of the hook-arrow tree based on their lexicographic order.

We again let the vertices $w_{i,j}$ represent either the distinguished edge of a subtree, or it is an external vertex and nothing if the subtree is trivial.

We continue to attach more sets of three vertices to any vertices representing non-trivial subtrees of the hook-arrow tree until all non-trivial subtrees are exhausted.

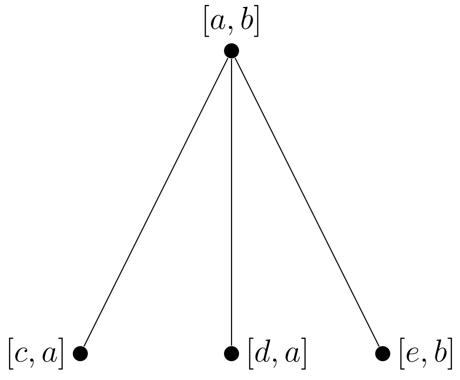
The linear relation R is given by dictating that vertices are ordered in sets by level and then by lexicographic order on the indices within each group. For example

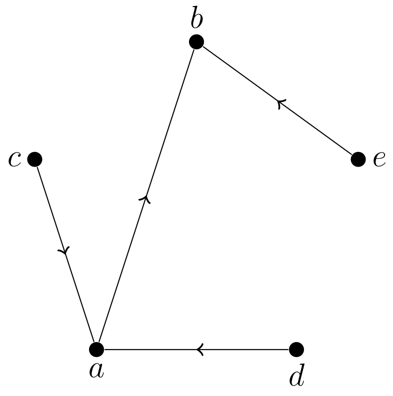
$$r < w_0 < w_1 < w_2 < w_3 < w_{1,1} < w_{1,2} < w_{1,3} < ...$$

Remark 2.31. A term in a symbol given by,

$$\mu([a,b]) \otimes \mu([c,a]) \sqcup\!\sqcup \mu([d,a]) \sqcup\!\sqcup \mu([e,b])$$

is equivalent to the following hook-arrow tree and section of a ternary tree (with external vertices omitted).

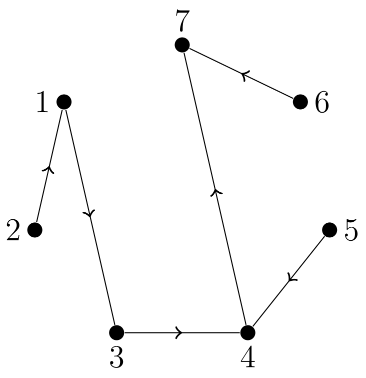




Note that this is a very intuitive demonstration of what is meant by a shuffle. We have an ‘option’ at each vertex as to which of the three ‘children’ to go to next.

The systematic replacing of $T_{i_1,\ldots,i_k}$ described in Section 2.3.4 can be seen to correlate to retracting the edges of the ternary tree from the bottom, disregarding trivial, external vertices.

Example 2.32. We now find the ternary tree for the hook-arrow tree in Example 2.25.



We set the root vertex to be r and then attach a vertex w_0 which represents the initial distinguished edge $[4, 7]$. All of the three subtrees τ_1, τ_2 and τ_3 are nontrivial and so we create the vertices

- w_1 - representing the distinguished edge of τ_1, which is $[3, 4]$.
- w_2 - representing the distinguished edge of τ_2, which is $[5, 4]$.
- w_3 - representing the distinguished edge of τ_3, which is $[6, 7]$.

We continue through the algorithm creating vertices.

- $w_{1,1}$ - represents the distinguished edge of τ_{11}, which is $[1, 3]$.
- $w_{1,1,2}$ - representing the distinguished edge of τ_{112}, which is $[2, 1]$.
- $w_{1,2}, w_{1,3}, w_{2,1}, w_{2,2}, w_{2,3}, w_{3,1}, w_{3,2}, w_{3,3}, w_{1,1,1}, w_{1,1,3}, w_{1,1,2,1}, w_{1,1,2,2}, w_{1,1,2,3}$ - are all external vertices representing trivial subtrees.

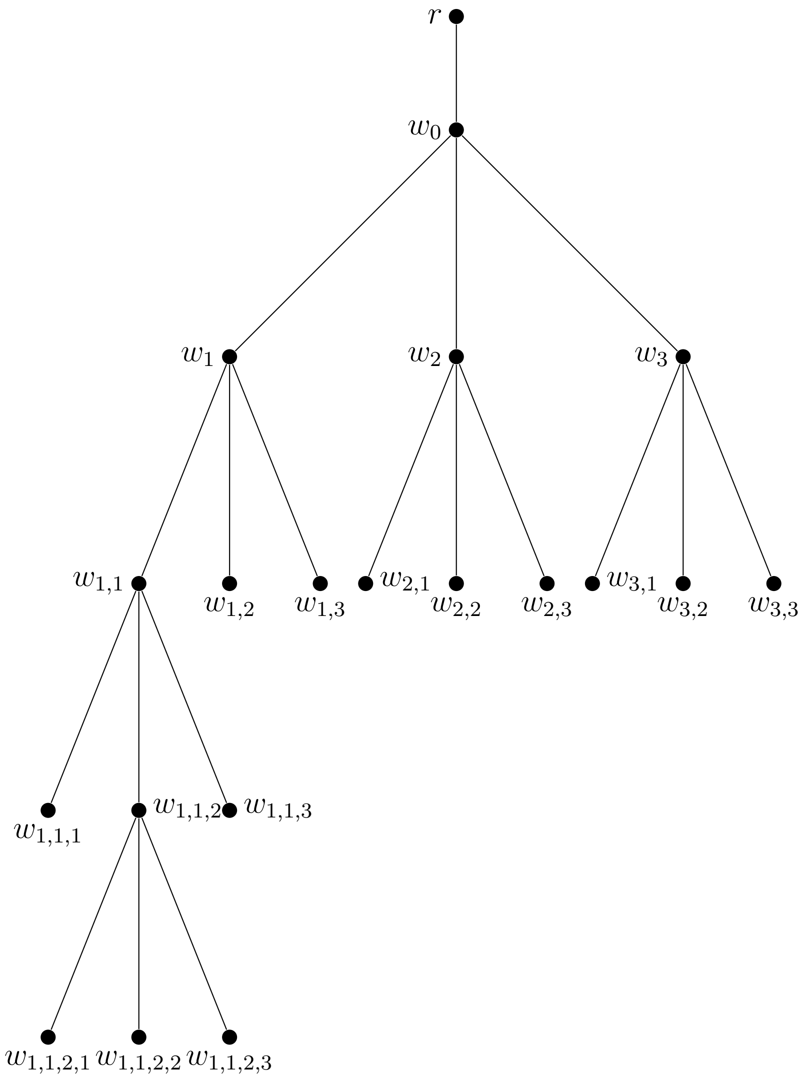
The full description of the ternary tree is therefore given by (V, v_0, E, R) where

$$V = \left\{ \begin{array}{c} r, w_0, w_1, w_2, w_3, w_{1,1}, w_{1,2}, w_{1,3}, w_{2,1}, w_{2,2}, w_{2,3}, w_{3,1}, w_{3,2}, w_{3,3}, \\ w_{1,1,1}, w_{1,1,2}, w_{1,1,3}, w_{1,1,2,1}, w_{1,1,2,2}, w_{1,1,2,3} \end{array} \right\}$$

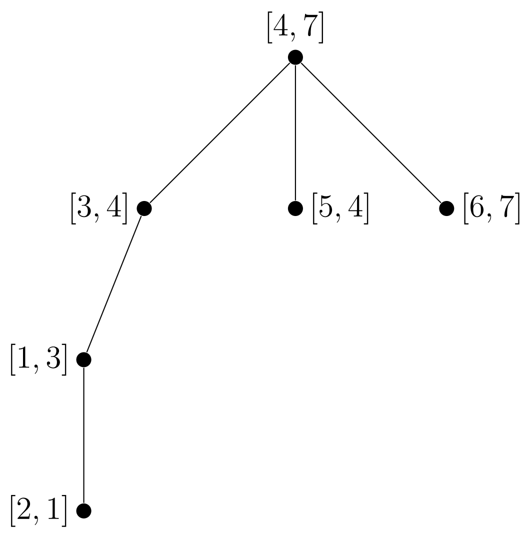
$$E = \left\{ \begin{array}{c} \{r, w_0\}, \big\{\{w_0, w_i\} \text{ for } i = 1,2,3\big\}, \\ \big\{\{w_i, w_{i,j}\} \text{ for } i = 1,2,3,\ j = 1,2,3\big\}, \\ \big\{\{w_{1,1,}, w_{1,1,i}\} \text{ for } i = 1,2,3\big\}, \big\{\{w_{1,1,2,i}, w_{1,1,i}\} \text{ for } i = 1,2,3\big\} \end{array} \right\}$$

$$R = \begin{array}{l} r < w_0 < w_1 < w_2 < w_3 < w_{1,1} < w_{1,2} < w_{1,3} < w_{2,1} < w_{2,2} < w_{2,3} < w_{3,1} \\ < w_{3,2} < w_{3,3} < w_{1,1,1} < w_{1,1,2} < w_{1,1,3} < w_{1,1,2,1} < w_{1,1,2,2} < w_{1,1,2,3} \end{array}$$

and takes the following form:



If we remove the external nodes and substitute the labels on the vertices with the distinguished edges they represent we can see the structure of the term in the symbol even more clearly.



2.4.4 Enumeration of hook-arrow trees

The association between hook-arrow trees and ternary tree allows us to count how many possible hook-arrow trees there are on a given number of vertices.

Proposition 2.33. *The number of distinct hook-arrow trees, representing a weight w multiple polylogarithm, on $n = w + 1$ vertices is equal to*

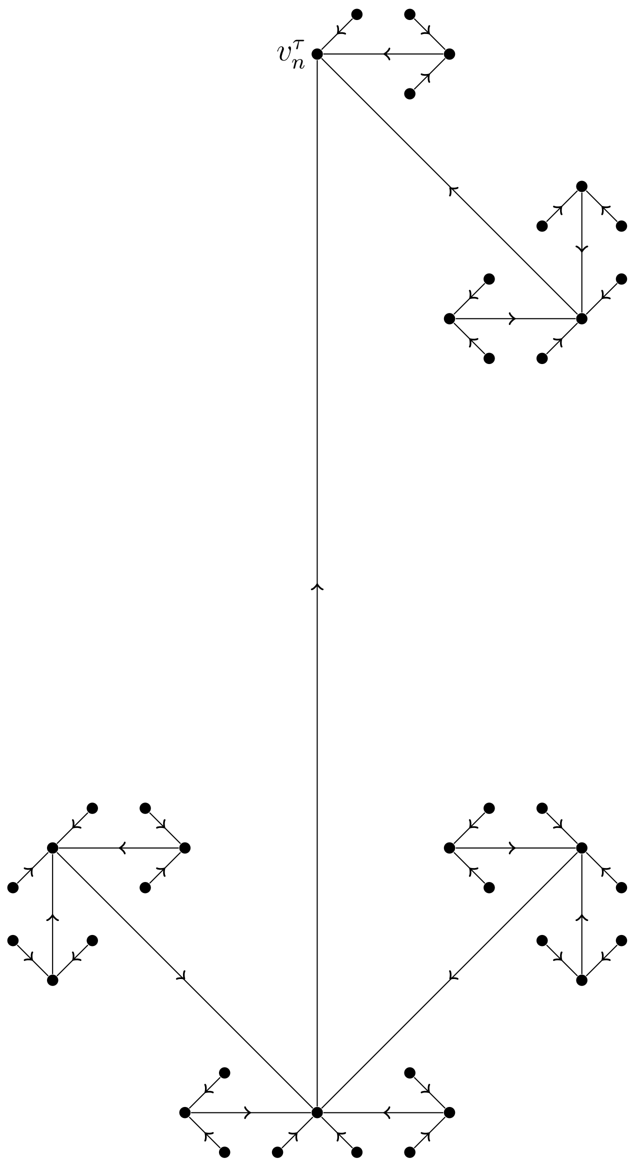
$$\frac{1}{2w+1}\binom{3w}{w}.$$

Proof. This result follows from the enumeration of 4-valent planted plane trees in [Kla70] and we do not explicitly give this here. □

Remark 2.34. As well as in [Kla70], discussion on enumerating ternary trees and their relation to generalised Catalan numbers also appears in [HP91]. We also note that the On-Line Encyclopedia of Integer Sequences ([OEI12]) is an invaluable resource here.

2.4.5 Schematic picture of a hook-arrow tree

The name 'hook-arrow tree' comes from the resemblance of each τ and its three subtrees τ_1, τ_2 and τ_3 to a right hook or left hook arrow. The following diagram shows this. The reader is invited to see why, if we take the longest edge to be the first distinguished edge, then every five vertex, four edge tree containing the longest edge can be displayed on this diagram by selecting four connected edges. Note that, if we decide to add another edge to the tree at a vertex, the longest edge (in the picture) attached to that vertex must be chosen first.



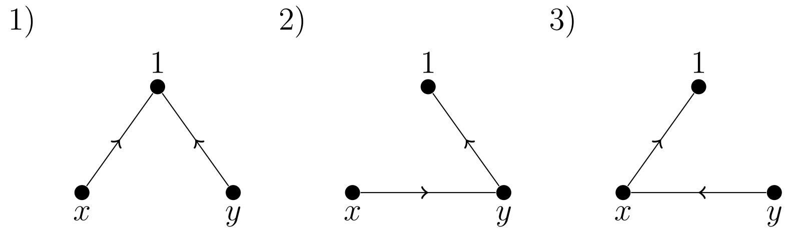
We can see that the dual of this tree (and in this case we mean turning edges into vertices and vice-versa as with a normal dual) is the ternary tree, albeit after removing the the vertices r and w_0, and the edge $\{r, w_0\}$.

2.5 Simple examples of finding the symbol using hook-arrow trees

We now give two examples of symbol calculations using hook-arrow trees.

2.5.1 Symbol for $I_{1,1}(x, y)$

There are three possible hook-arrow trees on the vertices $(x, y, 1)$ representing the multiple polylogarithm $I_{1,1}(x, y)$. These are as follows.



This gives us the following symbol, where the terms are given in the order of the trees they correspond to above.

$$\begin{aligned}&\mathcal{S}(I_{1,1}(x, y)) \\ &\quad = \left(1 - \frac{1}{x}\right) \otimes \left(1 - \frac{1}{y}\right) + \left(1 - \frac{1}{y}\right) \otimes \left(1 - \frac{y}{x}\right) - \left(1 - \frac{1}{x}\right) \otimes \left(1 - \frac{x}{y}\right).\end{aligned}$$

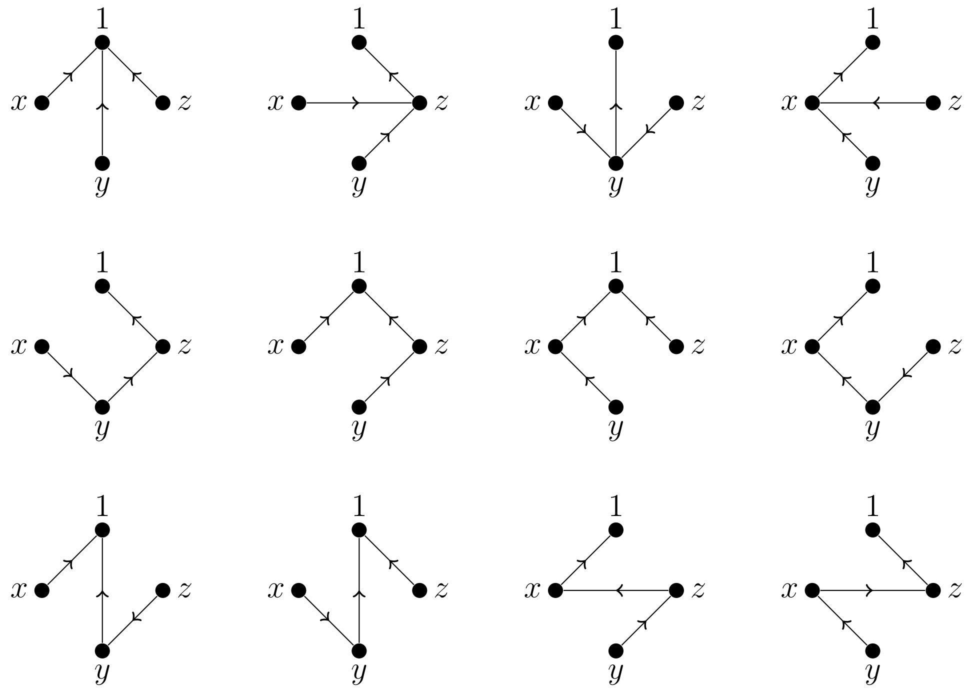
We note that we can easily obtain the symbol for $I_2(x)$ from the above. $I_2(x)$ essentially comes from setting $y = 0$ (see Definition 0.7), as we will be looking for hook-arrow trees on the vertices labelled $(x, 0, 1)$. As described in Definition 2.20 we will therefore see that the terms above including $(1 - \frac{1}{y})$ or $(1 - \frac{y}{x})$ will be given a coefficient of zero, and that $(1-\frac{x}{y})$ will become x. Therefore

$$\mathcal{S}(I_2(x)) = -\left(1-\frac{1}{x}\right)\otimes x = -(1-x)\otimes x.$$

A similar procedure also applies to general classical polylogarithms, $I_m(x)$, as we will see in Section 2.5.3.

2.5.2 Symbol for $I_{1,1,1}(x,y,z)$

There are twelve possible hook-arrow trees on the vertices $(x,y,z,1)$ representing the multiple polylogarithm $I_{1,1}(x,y,z)$. These are as follows:



This gives us the following symbol, where the terms are given in the order of the trees they correspond to above.

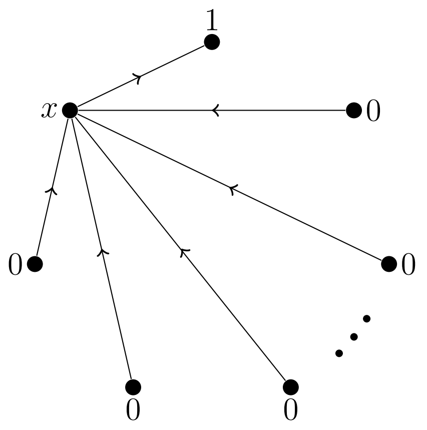
$$\begin{aligned}
\mathcal{S}(I_{1,1,1}(x,y,z)) = \; & +\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{y}\right)\otimes\left(1-\tfrac{1}{z}\right) && +\left(1-\tfrac{1}{z}\right)\otimes\left(1-\tfrac{z}{x}\right)\otimes\left(1-\tfrac{z}{y}\right)\\
& -\left(1-\tfrac{1}{y}\right)\otimes\left(1-\tfrac{y}{x}\right)\sqcup\!\sqcup\left(1-\tfrac{y}{z}\right) && +\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{z}\right)\otimes\left(1-\tfrac{x}{y}\right)\\
& +\left(1-\tfrac{1}{z}\right)\otimes\left(1-\tfrac{z}{y}\right)\otimes\left(1-\tfrac{y}{x}\right) && +\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{z}\right)\otimes\left(1-\tfrac{z}{y}\right)\\
& -\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{z}\right)\sqcup\!\sqcup\left(1-\tfrac{x}{y}\right) && +\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{y}\right)\otimes\left(1-\tfrac{y}{z}\right)\\
& -\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{y}\right)\otimes\left(1-\tfrac{y}{z}\right) && +\left(1-\tfrac{1}{y}\right)\otimes\left(1-\tfrac{y}{x}\right)\sqcup\!\sqcup\left(1-\tfrac{1}{z}\right)\\
& -\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{z}\right)\otimes\left(1-\tfrac{z}{y}\right) && -\left(1-\tfrac{1}{z}\right)\otimes\left(1-\tfrac{z}{x}\right)\otimes\left(1-\tfrac{x}{y}\right).
\end{aligned}$$

2.5.3 Symbol for $I_m(x)$

The symbol of the classical polylogarithm is very simple and well-known. $I_m(x)$ will correspond to hook-arrow trees on the vertices

$$(x, \underbrace{0, \ldots, 0}_{m-1}, 1).$$

After a little consideration, and bearing mind that any edge of a hook arrow tree directed towards a vertex labelled 0, we see that the only one hook-arrow tree that will not be given a zero coefficient is



We therefore see that

$$\mathcal{S}(I_m(x)) = (-1)^m \left(\left(1 - \frac{1}{x}\right) \otimes x^{\otimes(m-1)} \right) = (-1)^m \Big((1-x) \otimes x^{\otimes(m-1)} \Big).$$

The hook-arrow tree pictorial presentation of symbol calculations, while it does function in a very similar way, has some useful features that are easier to see than polygon dissections. In particular, it is easier to do symbol calculations on a computer. We are then able to do fast calculations with symbols and compare the symbols of relatively high weight multiple polylogarithms. Before we do this, and after relating the different pictorial forms of the symbol, we give some other applications of the hook-arrow tree. Firstly, it provides a nice way to simplify calculating the symbol of a multiple polylogarithm with a given depth. For the maximum depth, fixed weight, multiple polylogarithm, $I_{1,\ldots,1}(x_1, \ldots, x_m)$, every hook-arrow tree contributes to the symbol, whereas all but one hook-arrow tree contributes to the symbol for the minimum depth multiple polylogarithm $I_m(x)$. We will see in Chapter 4 how we can simplify the procedure for intermediate depths. In Chapter 5 we will see another application of the hook-arrow tree: in the symbol calculation of coloured multiple zeta values.

Chapter 3

Relating different pictorial representations of the symbol

There are several ways of viewing the symbol in a pictorial form. Goncharov gave a binary tree (or 3-valent planted plane tree) in [Gon05], a dissected polygon is given by Gangl, Goncharov and Levin in [GGL09], and we introduced the hook-arrow tree and ternary tree (or 4-valent planted plane tree) in Chapter 2. This chapter discusses and shows the relationships between these forms of the same data. The full, and same (to be proved) symbol can be found from each of these representations. However, it is important to note that, for a given weight, the different representations group the terms in the symbol in different ways. In the following section we explore this and add extra data to the pictorial representations to isolate 'single terms' (to be defined) in the symbol.

3.1 Isolating single terms in the symbol

It is useful to have an indexing set of terms in the symbol; one that is, in a sense, the most 'broken-up.' For polygon dissections and hook-arrow trees this will be when the symbol terms are extracted and we have expanded shuffles. The indexing set we will use are ‘single terms.’

Definition 3.1. We define a **single term** of a weight w symbol to be one of the form

$$c\left(1-\frac{x_{a_{1,1}}}{x_{a_{1,2}}}\right)\otimes\left(1-\frac{x_{a_{2,1}}}{x_{a_{2,2}}}\right)\otimes\cdots\otimes\left(1-\frac{x_{a_{w,1}}}{x_{a_{w,2}}}\right)$$

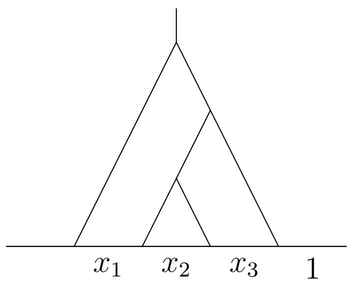
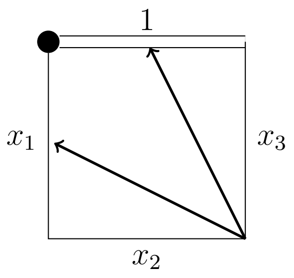
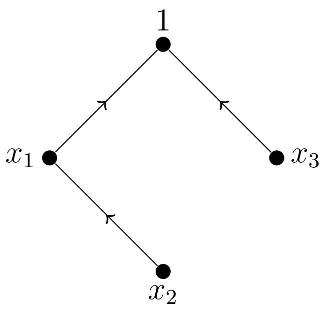
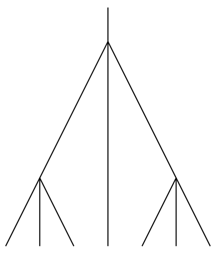
for some $x_{a_{i,j}}$ and constant c.

Remark 3.2. We note that for a term arising from a binary tree it is always possible to break it up into a sum of single terms by simply noting that

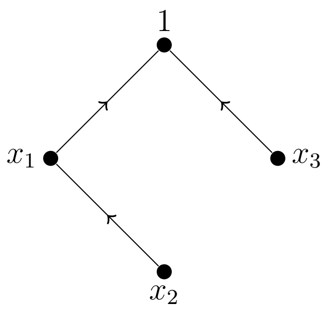
$$\frac{a-b}{a-c}=\left(1-\frac{b}{a}\right)\cdot\left(1-\frac{c}{a}\right)^{-1}$$

and using normal tensor product operations.

Example 3.3. We look at the symbol for the weight 3 multiple polylogarithm $I_{1,1,1}(x_1,x_2,x_3)$. The last column of the following diagram shows how many of each representation of weight 3 exists and gives the general formula for that enumeration for other weights (i.e. for the symbol of $I_{1,\ldots,1}(x_1,\ldots x_n)$).

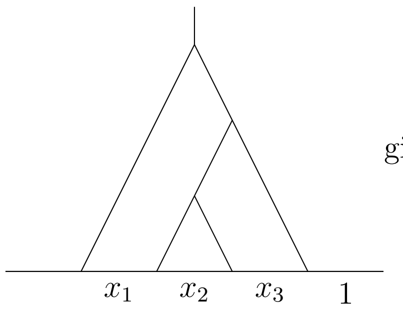
Representation	Example	Enumeration
Binary trees		$C(3) = 5$ where $C(n) = \frac{1}{n+1}\binom{2n}{n}$ (Catalan numbers)
Polygon dissections		$f(3) = 12$ where $f(n) = \frac{1}{2n+1}\binom{3n}{n}$
Hook-arrow trees		$f(3) = 12$ where $f(n) = \frac{1}{2n+1}\binom{3n}{n}$
Ternary trees		$f(3) = 12$ where $f(n) = \frac{1}{2n+1}\binom{3n}{n}$
Single symbol term	$-\left(1-\frac{1}{x_1}\right)\otimes\left(1-\frac{1}{x_3}\right)\otimes\left(1-\frac{x_1}{x_2}\right)$	$g(3) = 15$ where $g(n) = (2n-1)!!$

The symbol of $I_{1,1,1}(x_1, x_2, x_3)$ has 15 single terms. However, the other representations all have less than 15 forms. In the case of polygon dissections, hook-arrow trees and ternary trees, some give rise to a shuffle. In the table above the example symbol single term given is actually only one of two possible terms that could come from the examples of polygon dissection, hook-arrow tree and ternary tree. In fact



gives rise to $-\left(1-\frac{1}{x_1}\right) \otimes \left(1-\frac{x_1}{x_2}\right) ⧢ \left(1-\frac{1}{x_3}\right).$

On the other hand, there are only 5 possible binary trees, the symbol term being one of 4 possible terms from the binary tree given. In fact



gives rise to $\left(\frac{x_1-1}{x_1}\right) \otimes \left(\frac{x_3-1}{x_3-x_1}\right) \otimes \left(\frac{x_2-x_3}{x_2-x_1}\right),$

and

$$\begin{aligned}
\left(\frac{x_1-1}{x_1}\right) \otimes \left(\frac{x_3-1}{x_3-x_1}\right) \otimes \left(\frac{x_2-x_3}{x_2-x_1}\right) = \quad & \left(1-\frac{1}{x_1}\right) \otimes \left(\frac{1-\frac{1}{x_3}}{1-\frac{x_1}{x_3}}\right) \otimes \left(\frac{1-\frac{x_3}{x_2}}{1-\frac{x_1}{x_2}}\right) \\
= \quad & \left(1-\frac{1}{x_1}\right) \otimes \left(1-\frac{1}{x_3}\right) \otimes \left(1-\frac{x_3}{x_2}\right) \\
- \quad & \left(1-\frac{1}{x_1}\right) \otimes \left(1-\frac{x_1}{x_3}\right) \otimes \left(1-\frac{x_3}{x_2}\right) \\
- \quad & \left(1-\frac{1}{x_1}\right) \otimes \left(1-\frac{1}{x_3}\right) \otimes \left(1-\frac{x_1}{x_2}\right) \\
+ \quad & \left(1-\frac{1}{x_1}\right) \otimes \left(1-\frac{x_1}{x_3}\right) \otimes \left(1-\frac{x_1}{x_2}\right).
\end{aligned}$$

It is important to note here that the other term in the shuffle from the hook-arrow tree does not appear as one of the terms from this binary tree, and is a good example of different groupings of the 15 terms in the symbol from different pictorial representations.

This motivates us to add extra data to each representation to isolate a single term of the symbol.

3.1.1 Isolating a single term on a binary tree

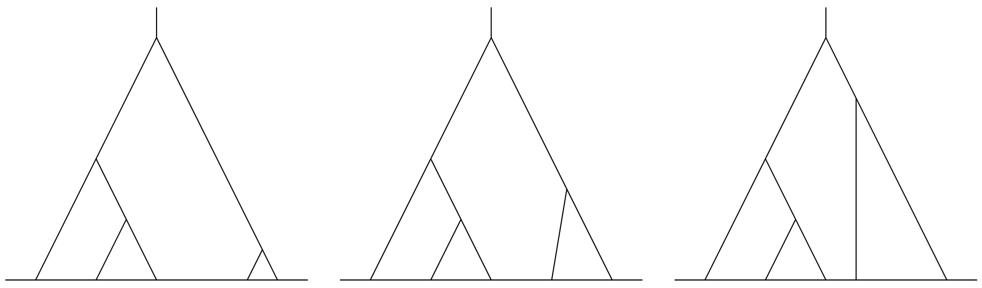
We define a 'level binary tree with connected regions' to be the version of the binary tree symbol representation which represents a single term in the symbol. Firstly we define a 'level binary tree'.

Definition 3.4. A **level binary tree** is a binary tree (as in Definition 1.10) with a strict ordering on the height of the internal vertices.

A binary trees will have one or more level binary trees associated to it. These are found by running through all possible strict height orders compatible with the partial ordering inherent from a binary tree being embedded in a plane.

Example 3.5. We will use the following weight $w = 4$ Goncharov tree as a running example.

Note that there are three possible level binary trees associated to this, which are



As before, we consider all lines away from the root to be extended to a base line (which is not part of the tree). The root vertex is considered to be at infinity (so as to separate the regions to left and right of the tree; we will refer to these regions as being 'outside' the tree). Every region cut out by the Goncharov tree will touch the base line, we label these regions $a_0, ..., a_{w+1}$ by moving from left to right along the base line.

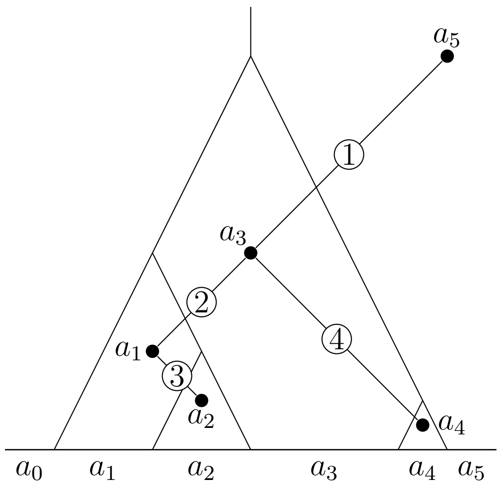
We then place vertices in each region (excluding a_0). The height of these vertices must respect the height ordering on the internal vertex at the top of each region, with the vertex in a_{w+1} being above the first internal vertex. The vertices inherit the label of the region they inhabit. A single term in the symbol is then represented by the binary tree with another tree on these region vertices where only adjacent regions may be connected.

Definition 3.6. A **level binary tree with connected regions** is a level binary tree with a dual tree, τ on the regions $a_1, \ldots a_{w+1}$ that satisfy:

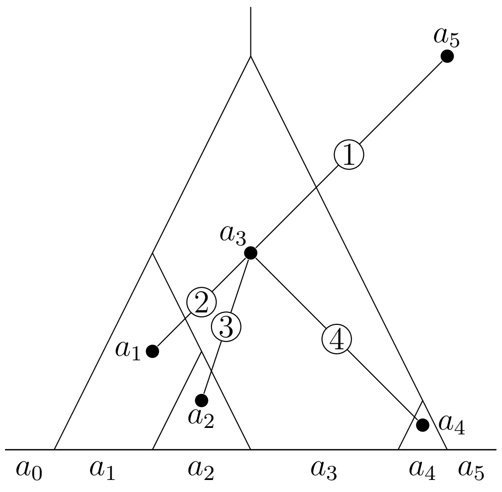
1. The vertices of the dual tree have labels inherited from the regions, $a_1, \ldots, a_{w+1}$.
2. The vertices of the dual tree have a strict height ordering
$$a_{i_1} < \ldots < a_{i_w} < a_{w+1}$$
dictated by the height ordering on the internal vertices (of the binary tree) at the top of each region and a_{w+1} being the greatest.
3. Let $S_{a_i} = \{a_j | a_i < a_j\}$. There must only be one edge in the tree of the form $\{a_i, a_k\}$ for $a_k \in S_{a_i}$, and for all i. In other words, each vertex may be connected to only one of the vertices that lies above it.
4. The edges are given a strict order, starting with an edge attached to a_{w+1}, ending with an edge attached to the lowest region vertex, and dictated by the position of the lower vertex of each edge in the strict height ordering of the region vertices.

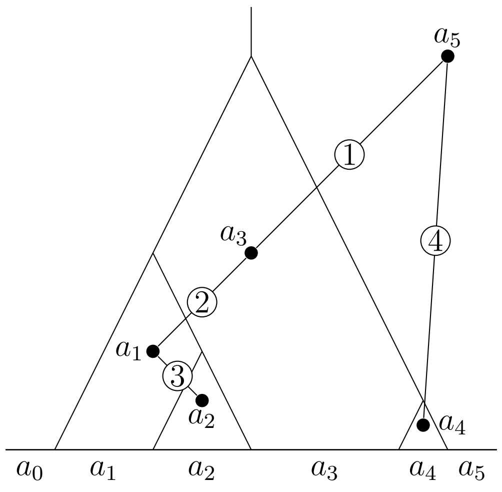
The definition is demonstrated in the following example.

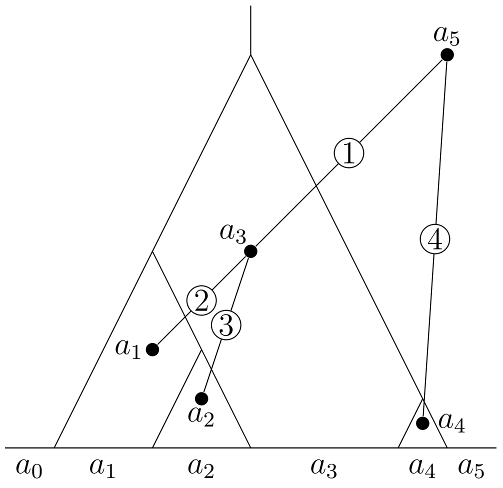
Example 3.7. The following represents a possible single term of the symbol obtained from the level binary tree in Example 3.5.



We run through all other possible ways of connecting the a_i while obeying the rule that each a_i, for $i = 1, ..., 4$, which must be attached to exactly one higher vertex, can only be attached to a higher vertex a_j if they are in adjacent regions. As a consequence we see that a_1 must be attached to a_3 and a_3 must be attached to a_5. However, a_2 can either be attached to a_1 or a_3, and likewise a_4 to either a_3 or a_5; resulting in four possibilities. One is above, the other three are now given.





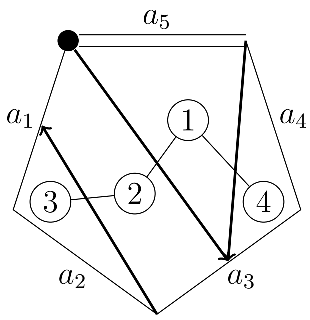


3.1.2 Isolating a single term on a polygon dissection

To isolate a single term on a polygon dissection we must pick out one particular term that arises from the expansion of any shuffles. Shuffles arise on a polygon due to the dual tree only having a partial order and not a strict order. For example, the region of the dissection containing both the final edge of the polygon and the first vertex will always map to the first vertex (or root) on the dual tree, but if there are more than one other vertices attached to the root of the dual tree, we must shuffle these vertices. We can therefore isolate a single term by choosing an order on the vertices of the dual tree (albeit one that is compatible with the partial order).

Definition 3.8. An **ordered polygon dissection** is a polygon dissection with a dictated strict order, compatible with the partial order, on its dual tree.

Example 3.9. The following shows the dissection of a 5-gon where a strict order on the dual tree is shown by a numbering on its vertices.



The resulting single term is

$$+\left(1-\frac{a_5}{a_3}\right)\otimes\left(1-\frac{a_3}{a_1}\right)\otimes\left(1-\frac{a_1}{a_2}\right)\otimes\left(1-\frac{a_3}{a_4}\right),$$

as opposed to all terms in the symbol arising from this dissection, given by

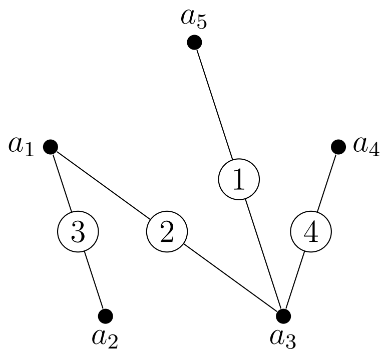
$$+\left(1-\frac{a_5}{a_3}\right)\otimes\left(\left(1-\frac{a_3}{a_1}\right)\otimes\left(1-\frac{a_1}{a_2}\right)\right)ш\left(1-\frac{a_3}{a_4}\right).$$

3.1.3 Isolating a single term on a hook-arrow tree

Isolating a single term on a hook-arrow tree is a very similar procedure to that for a polygon dissection.

Definition 3.10. An **ordered hook-arrow tree** is one that has a strict order chosen on its edges compatible with the partial order on its dual tree.

Example 3.11. The hook-arrow tree of the single term given by the polygon dissection in Example 3.9, with added strict ordering on the edges, is given by the following.



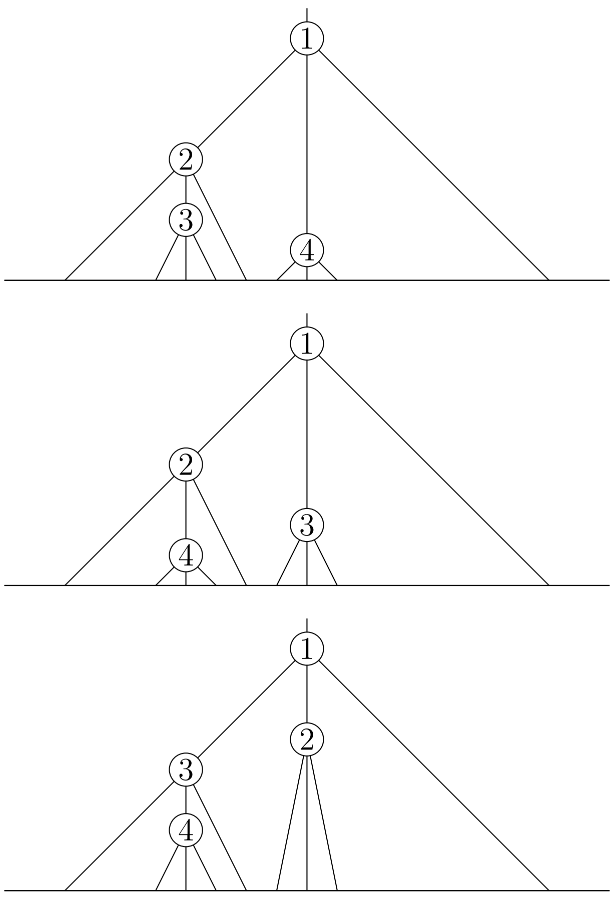
3.1.4 Isolating a single term on a ternary tree

The internal vertices of a ternary tree are, by construction, directly related to the dual tree of a hook-arrow tree.

Definition 3.12. A **level ternary tree**, representing a single term in the symbol, is a ternary tree with a strict ordering on the internal vertices.

We denote the strict ordering on the internal vertices by giving them a height order, from root to base line, within the ternary tree. We demonstrate this definition through the following example.

Example 3.13. What follows are three copies of the same ternary tree but three different level ternary trees (each representing a different single term of the symbol).



3.2 Bijections between pictorial representations of the symbol

We will now give selected direct maps between the different pictorial representations, and overall they will form bijections between all. Figure 3.1 outlines the maps we will give and they are labelled with the sections in which they appear. The map from hook-arrow trees to ternary trees is unnecessary for the bijection, but is still useful.

$$\begin{tikzcd}
 & \text{Hook-arrow tree} \arrow[dr, "3.2.1"] \arrow[dd, "3.2.5"] & \\
\text{Goncharov tree} \arrow[ur, "3.2.4"] & & \text{Polygon dissection} \arrow[dl, "3.2.2"] \\
 & \text{Ternary tree} \arrow[ul, "3.2.3"] &
\end{tikzcd}$$

Figure 3.1: Commutative diagram of maps between pictorial representations labelled with section numbers.

3.2.1 Hook-arrow trees to polygon dissection

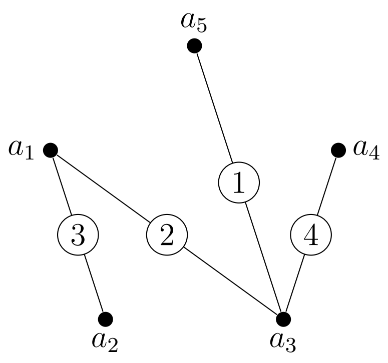
Remark 3.14. We note that hook-arrow trees were defined from polygon dissections in Chapter 2. We require the map in the reverse direction here so as to show bijections in Figure 3.1.

We will use a weight 4 hook-arrow tree representing a single term in the symbol as a running example.

Example 3.15. A possible weight 4 labelled hook-arrow tree, τ is given by

$$D_\tau = \big\{(1,2,3,4,5), (\{3,5\},\{1,3\},\{2,1\},\{4,3\})\big\}.$$

This takes the following form.



We start with a weight w hook-arrow tree with vertices in a linear order labelled $(a_1, ..., a_{w+1})$.

Step 1 Create the polygon $P(a_1, ..., a_{w+1})$ and add the hook-arrow tree such that the vertices of the hook-arrow tree lie on the centre of the respective edge of the polygon.

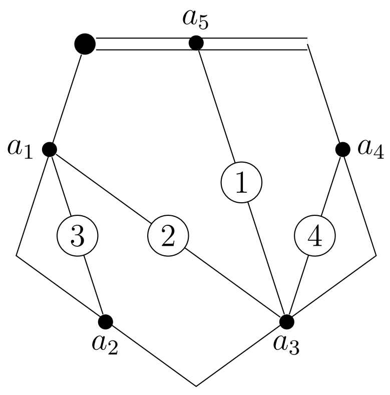
Step 2 We add all possible arrows that satisfy the following conditions:

- The arrow starts at a vertex of the polygon and ends at a vertex of the hook-arrow tree which is not on an adjacent side.
- The arrow will bisect the vertices of the hook-arrow tree into two sets with one common vertex. For the arrow to be allowed, the hook-arrow tree must only contain edges that connect two vertices in the same set, i.e., the arrows must not 'cross' an edge of the hook-arrow tree.

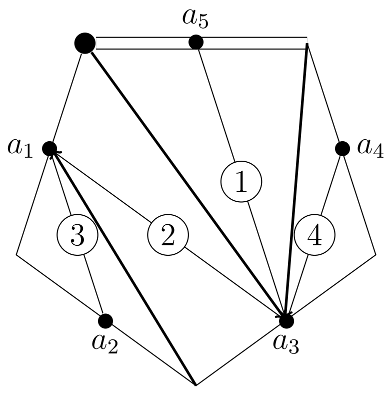
Step 3 We remove the edges and vertices of the hook-arrow tree, but allow the labels on the centre of hook-arrow tree to become vertices inhabiting the region of the dissected polygon they lie in. We then form the dual tree of the polygon dissection by connecting these numbered vertices if they lie in adjacent regions.

We are then left with the required polygon dissection representing a single term of the symbol.

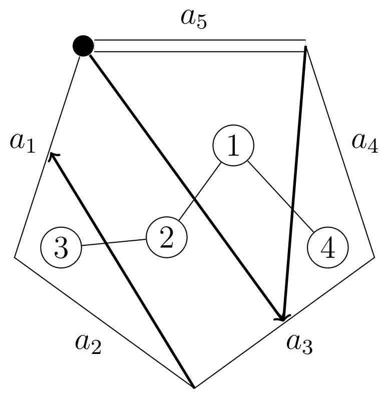
Example 3.16. We apply steps 1,2 and 3 to the hook-arrow tree in Example 3.15.



After applying Step 1.



After applying Step 2.



After applying Step 3.

3.2.2 Polygon dissections to ternary trees

We start with a term in the symbol of a multiple polylogarithm arising from a given linear order on the dual tree of a maximal dissection of a polygon. Our goal is to map this to a level binary tree with connected regions. We note that the sides of the polygon are 'paired-up' by the bigons cut out by a maximal dissection.

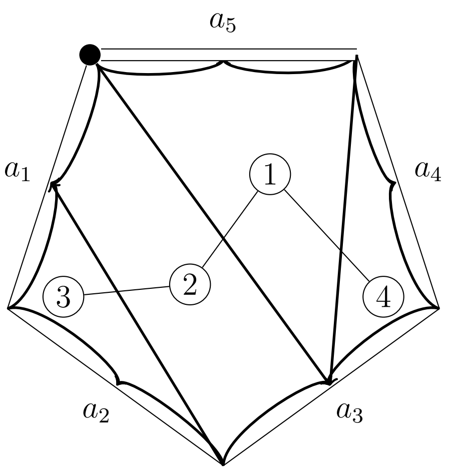
Example 3.17. We use the same weight 4 polygon maximal dissection as in Examples 3.9 and 3.16 where the sides of the polygon are paired

$$\big\{\{a_1, a_2\}, \{a_1, a_3\}, \{a_3, a_4\}, \{a_3, a_5\}\big\}.$$

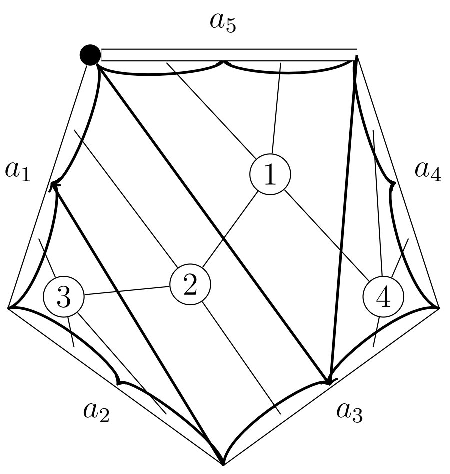
Step 1 We start the transition to a ternary tree by adding all possible trivial arrows to the dissection, i.e., all possible arrows that end on adjacent sides. We note that this is a natural relaxation of the stipulation that arrows must not be attached to adjacent edges in a dissection.

Step 2 Next, we 'complete' the dual tree of a maximal polygon dissection (including all the trivial arrows) in a natural way to a ternary tree. The new edges correspond to (the parts cut-off by) the trivial arrows.

Example 3.18. For the polygon dissection in Example 3.17 these two steps appear as follows.



Step 1

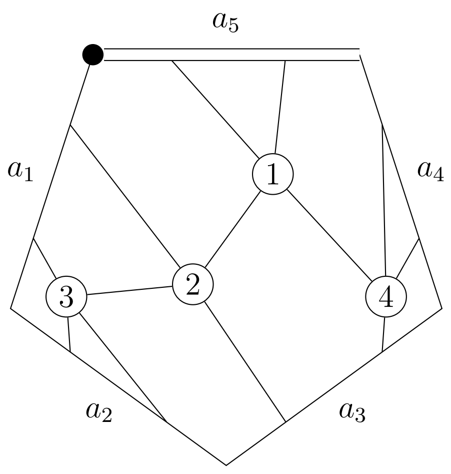


Step 2

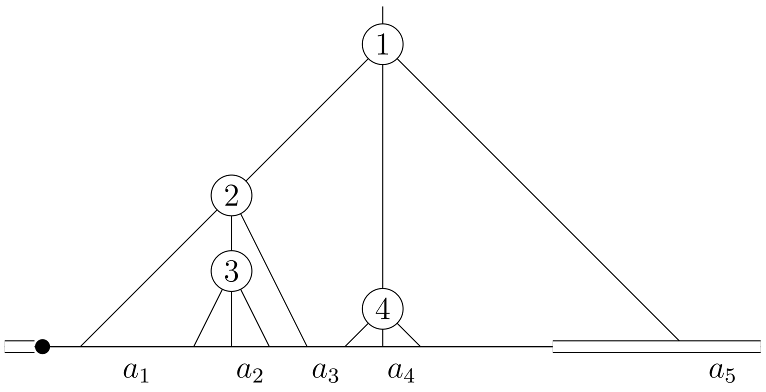
Step 3 We then remove the arrows (and trivial arrows) and extend the leaves of the dual tree to the edges of the polygon.

Step 4 The root vertex of the ternary tree is inherited from the root edge of the polygon (and is on the leaf closest to first vertex of the polygon on the root edge). We 'break' the polygon at the root vertex of the ternary tree and 'roll-out'. The edges of the polygon will then form a base line, which is not part of the ternary tree. It is crucial to note that we induce a height ordering on the internal vertices of the ternary tree inherited from the linear order on the dual tree.

Example 3.19. Steps 3 and 4 for our running example appear as follows.

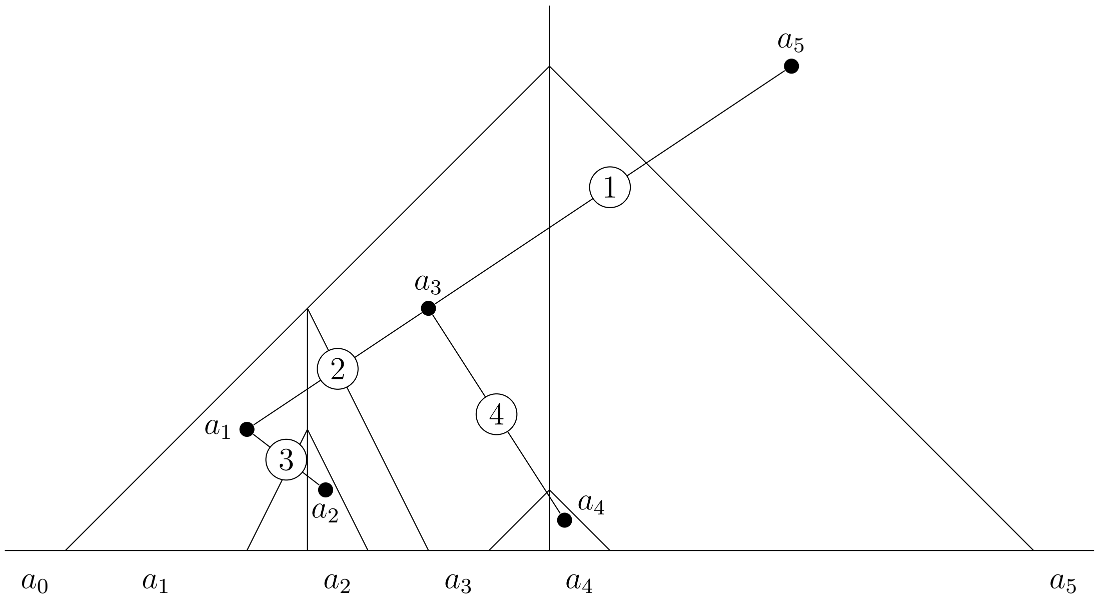


Step 3



Step 4

Remark 3.20. We note that, by retaining the labels from the edges of the polygon and setting them to represent only the middle portion of each original polygon side (divided by the leaves of the ternary tree), we have now inherited alternating labels on the regions of the ternary tree. We could, at this point, label these regions as such, and connect by the pairings of edges from the polygon dissection. We put a linear ordering on these edges based on the height ordering of the lower vertex on each edge. In the case of the running example, noting the pairings from Example 3.17, we get the following.



This combination of region vertices and ordered edges will be exactly the hook-arrow tree for this term in the symbol.

3.2.3 Ternary trees to binary trees

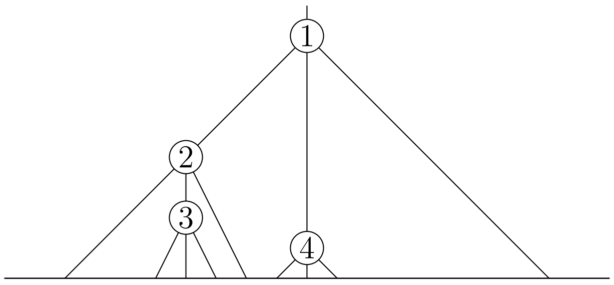
We start with a level ternary tree with a base line.

Step 1 We apply an alternating labelling (alternating between an a_i and an empty label) of the partitions of the base line (skipping the first partition and labelling it a_0). So, we label the partitions, from left to right,

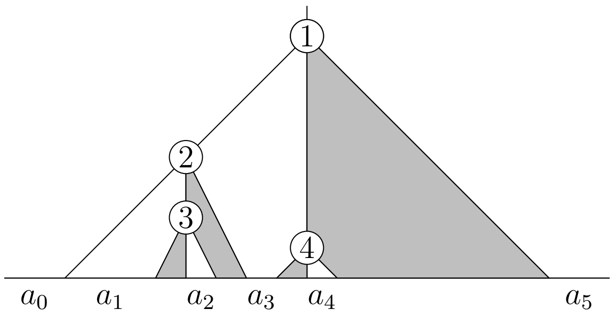
$$a_0, a_1, \emptyset, a_2, \emptyset, ..., a_i, \emptyset, ..., a_w, \emptyset, a_{w+1}.$$

For clarity we will shade the regions with an empty label.

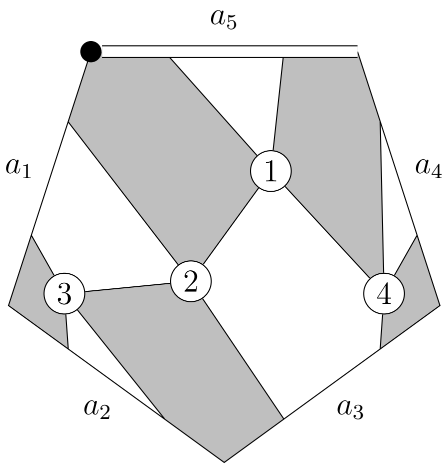
Example 3.21. We use the following level ternary tree. The height order on the internal vertices has been labelled for clarity.



After labelling partitions of the base line and shading regions with an empty label on the ternary tree in Example 3.19 we obtain the following.



Remark 3.22. The alternating shading can be seen nicely if we recall the 'roll-out' as explained in Section 3.2.2. If we shade the regions cut-out by the ternary tree that include a corner of the original polygon, for Example 3.19 we obtain the following.

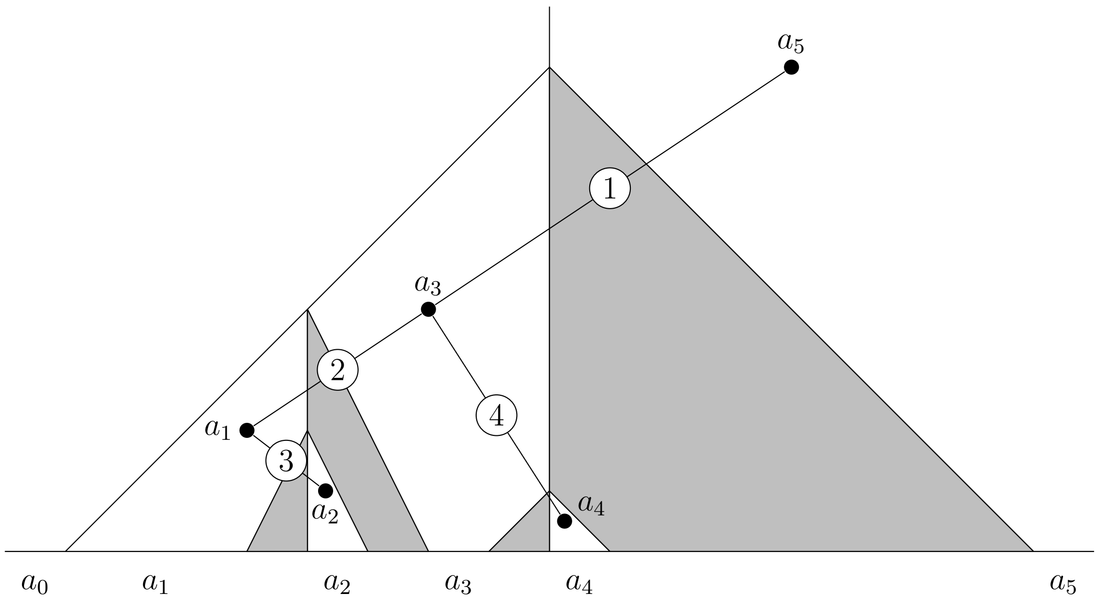


'Rolling-out' (by breaking the polygon at the root vertex of the ternary tree) we obtain the required shading (as well as an explanation for the labelling of the regions).

Step 2 On a shaded ternary tree we add a vertex in the centre of the non-shaded regions (except a_0) and allow the vertices to inherit its label. It is important that these vertices also obey the strict height ordering on the internal vertices at the top of each region. We now connect the vertices $a_1, ..., a_{w+1}$ if the regions they inhabit share the same internal vertex.

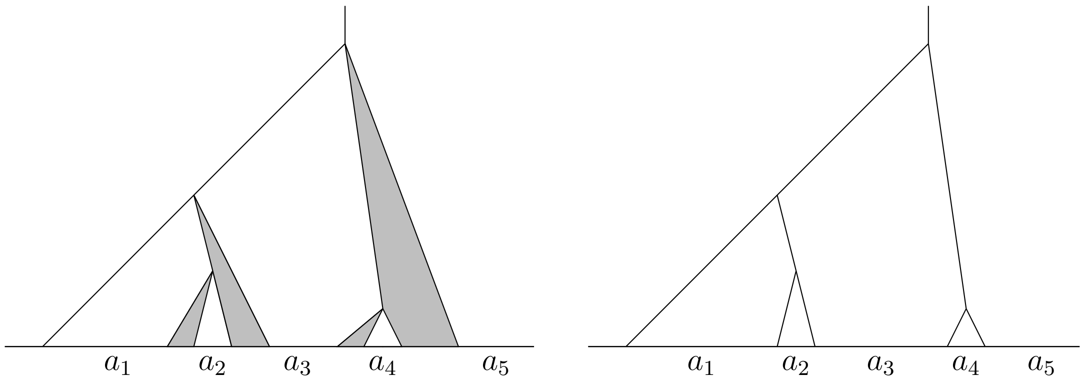
We order the edges connecting the a_i in the following way. Each edge $\{a_i, a_j\}$ will have a unique lower vertex. We apply an order on the edges by the position (from top to bottom) of their lower vertices in the height order.

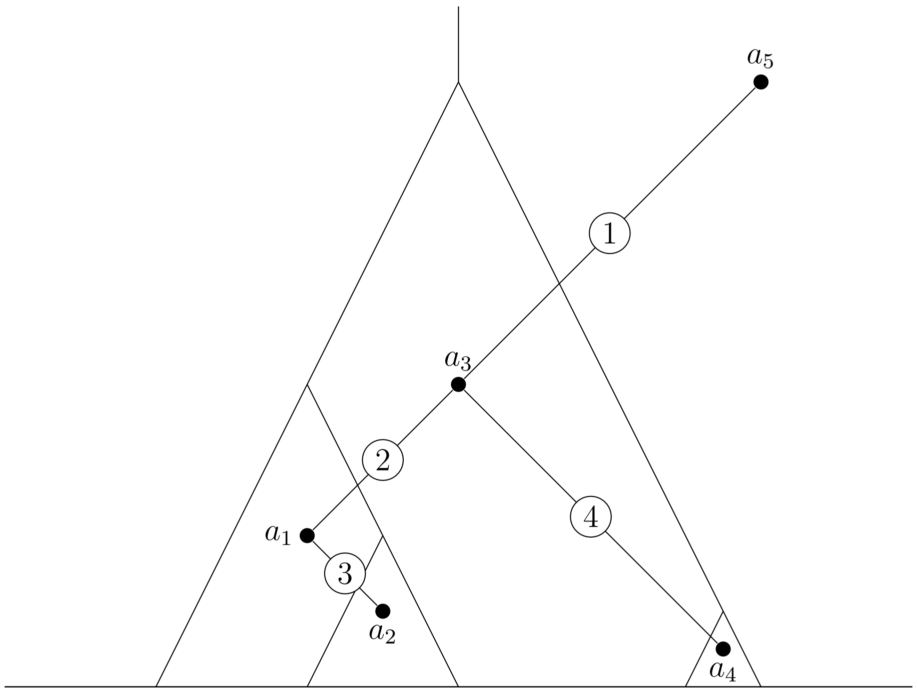
Example 3.23. Adding vertices to regions and their numbered connecting edges to our running example gives the following.



Step 3 The final step involves a horizontal contraction of the shaded regions. Within each shaded region we identify all points of the same height. Due to the alternating shading, each internal vertex will be reduced in valency by exactly one. The result of this procedure will be a 3-valent tree, the level binary tree required, complete with decorations designating a single term in the symbol.

Example 3.24. We apply horizontal contraction to our example, temporarily removing the region vertices and edges for clarity. We show steps part of the way through the horizontal contraction and the resulting binary tree (after replacing the region vertices and edges).





3.2.4 Binary trees to a hook-arrow trees

By taking a decorated level binary tree we will have a single term in the symbol. We then simply extract the region vertices and their connecting edges. We use the order on the vertices given by the order the regions appear along the base line of the binary tree from left to right, to obtain the ordered hook-arrow trees representing a single symbol term. This also comes with an ordering on the edges (so it really does represent one term in the symbol).

As an example, compare the first decorated level binary tree in Example 3.7 and the ordered hook-arrow tree in Example 3.11.

3.2.5 Hook-arrow trees to ternary trees

We now give an alternative to the recipe for forming a ternary tree (or 4-valent planted plane tree) to the one given in Section 2.4.3. This method is more direct and pictorial, and takes a labelled hook-arrow tree to a level ternary tree. We will use the hook-arrow tree from Example 3.15 as a running example.

Step 1 Begin with a hook-arrow tree τ with description

$$D_\tau = \{(v_1^\tau, \dots, v_{w+1}^\tau), \{e_1^\tau, \dots e_w^\tau\}\}.$$

We form a boundary construction circle which passes through the vertices of the hook-arrow tree. We consider the boundary circle and the region outside it to be 'out of bounds'.

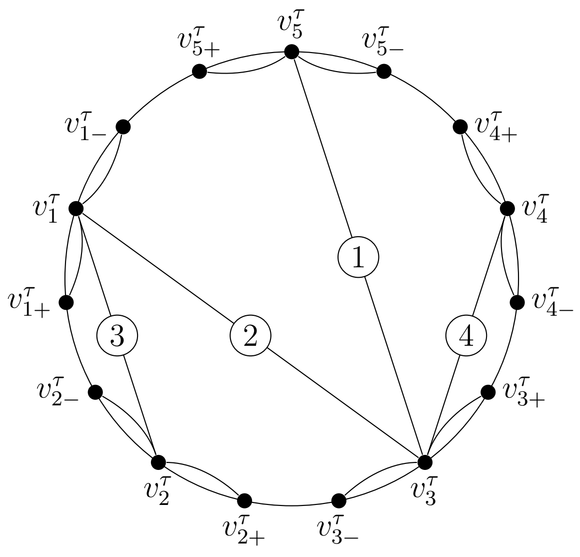
We now add $2w+2$ extra vertices on the boundary circle, one on each side of each v_i^τ. We label the new vertex directly after a v_i^τ with v_{i+}^τ and the vertex directly before with v_{i-}^τ. We now add edges $\{v_i^\tau, v_{i+}^\tau\}$ and $\{v_i^\tau, v_{i-}^\tau\}$.

We have created a tree, $\overline{\tau}$, with a description (as for a hook-arrow tree) of

$$\begin{aligned} D_{\overline{\tau}} &= \{V_{\overline{\tau}}, E_{\overline{\tau}}\} \\ &= \{(v_{1-}^\tau, v_1^\tau, v_{1+}^\tau, \dots, v_{(w+1)-}^\tau, v_{w+1}^\tau, v_{(w+1)+}^\tau), \\ &\qquad \{e_1^\tau, \dots e_w^\tau, \{v_1^\tau, v_{1+}^\tau\}, \{v_1^\tau, v_{1-}^\tau\}, \dots, \{v_{w+1}^\tau, v_{(w+1)+}^\tau\}, \{v_{w+1}^\tau, v_{(w+1)-}^\tau\}\}\}. \end{aligned}$$

On a weight w hook-arrow tree, this will bring the total number of vertices up to $3w+3$ and the number of edges to $3w+2$ (up from $w+1$ vertices and w edges).

Example 3.25. We add the boundary circle, extra vertices and edges to the hook-arrow tree, τ, from Example 3.15. We switch to labelling the vertices of the hook-arrow tree with $v_1, \ldots, v_5$ rather than $1, \ldots, 5$.



Step 2 We place new vertices in the centre of each of the new edges; a vertex labelled w_{i+} in the middle of $\{v_i^\tau, v_{i+}^\tau\}$ and a vertex labelled w_{i-} in the middle of $\{v_i^\tau, v_{i-}^\tau\}$. We also now place new vertices, u_i in the centre of the original hook-arrow tree edges (note that in a labelled hook-arrow tree where we have labelled the centres of these edges, these labels can serve as these new vertices).

We form a set C containing the w_{i-}, w_{i+} and u_i. We also let $w_{(w+1)+} = r$ (it will become the root vertex of the ternary tree).

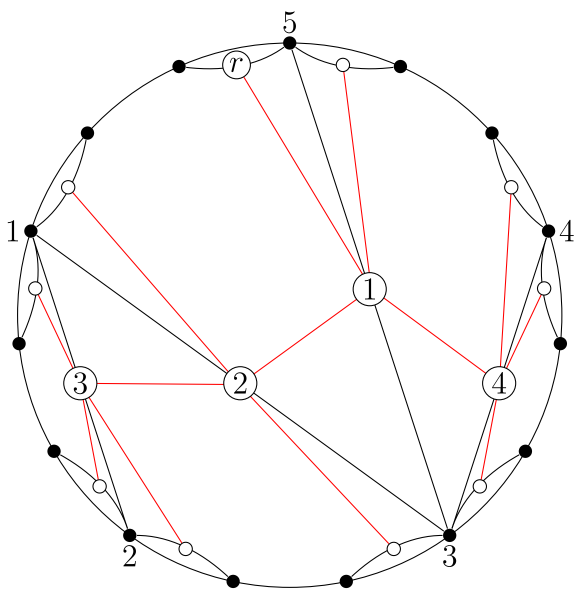
We now connect the vertices of C with edges. This is done in the following way.

The vertex $c_1 \in C$ on edge $e_1 \in E_{\overline{\tau}}$ is connected to $c_2 \in C$ on edge $e_2 \in E_{\overline{\tau}}$ if both the following hold:

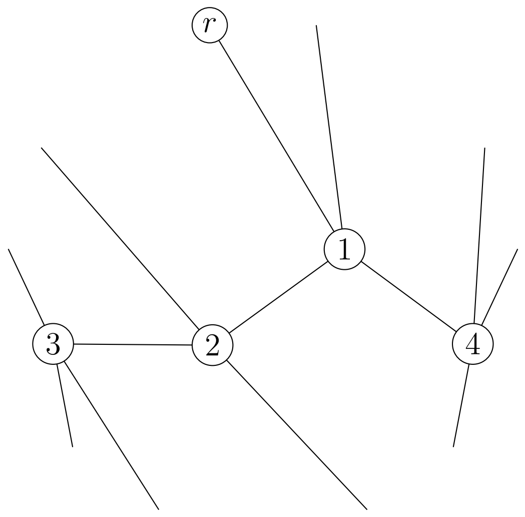
- c_1 and c_2 lie on the boundary of the same region inside the boundary circle.
- e_1 and e_2 share a vertex of the original hook-arrow tree.

We then discard everything except the vertices in C and their connecting edges.

Example 3.26. We add centre vertices to the hook-arrow tree from Example 3.25 and connect them if they satisfy the conditions.

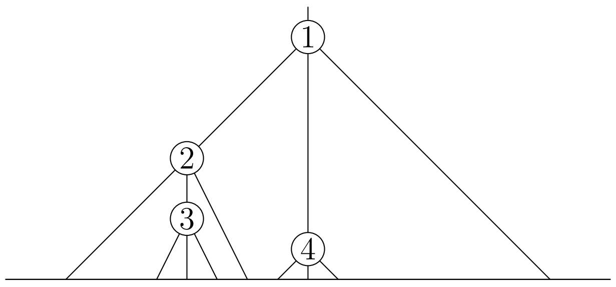


We then discard unwanted vertices and lines.



Step 3 The vertex labelled r is the root of the level ternary tree. The numbers on the vertices u_i from the original labelled hook-arrow tree dictate the height of the internal vertices of the ternary tree. Finally, the anticlockwise order inherited from the linear order on $V_{\overline{T}}$ gives us the required linear order on vertices which are of the same distance from the root.

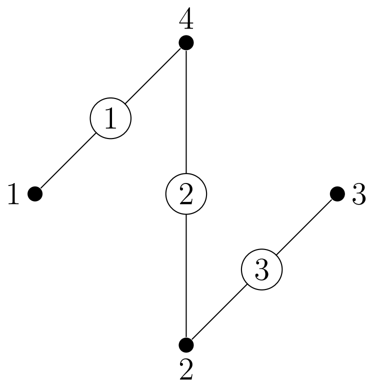
Example 3.27. We again look at the hook-arrow tree from Example 3.25. We now take into account the inherited order of the vertices from the boundary circle and extend the edges of the ternary tree to a base line to give the following:



We note that this step is very similar to the previously seen concept of 'rolling-out'.

3.2.6 A specific example of moving between all pictorial representations

Since the above maps had the same running example we provide diagrams for another single term. We start with a weight 3 hook-arrow tree, H, given by the following:

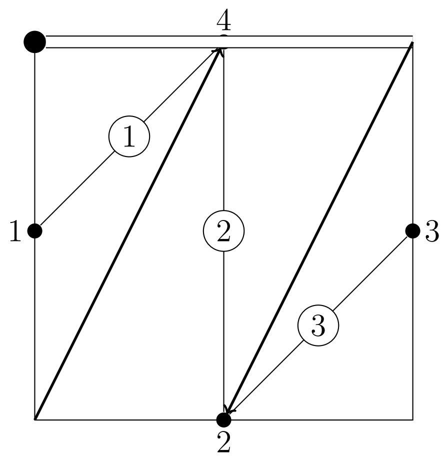


Hook-arrow tree H

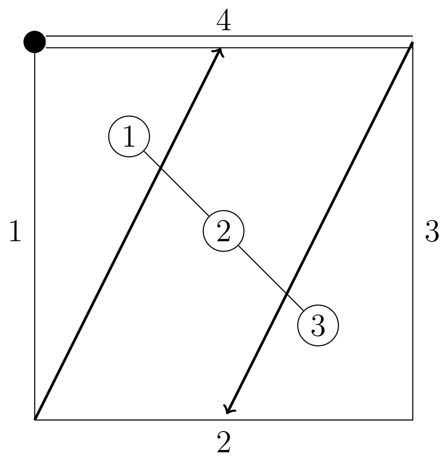
We now move around the diagram in Figure 3.1 in the following way:

$$H \xrightarrow{3.2.1} P \xrightarrow{3.2.2} T \xrightarrow{3.2.3} B \xrightarrow{3.2.4} H \xrightarrow{3.2.5} T.$$

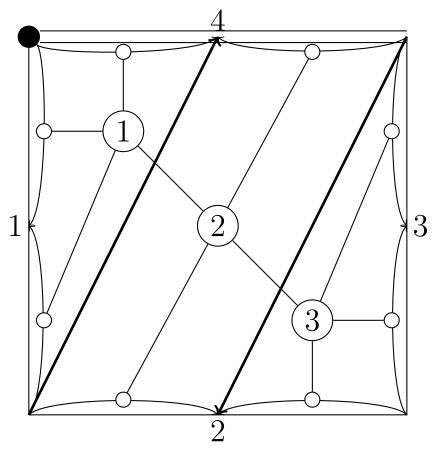
We give pictures for various steps along the process. The reader is invited to look at the relevant sections for explanations.



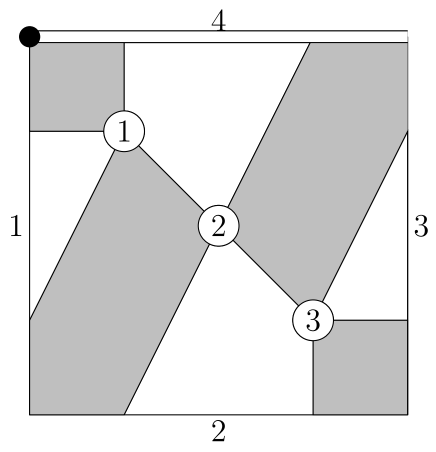
3.2.1 Step 1 & 2.



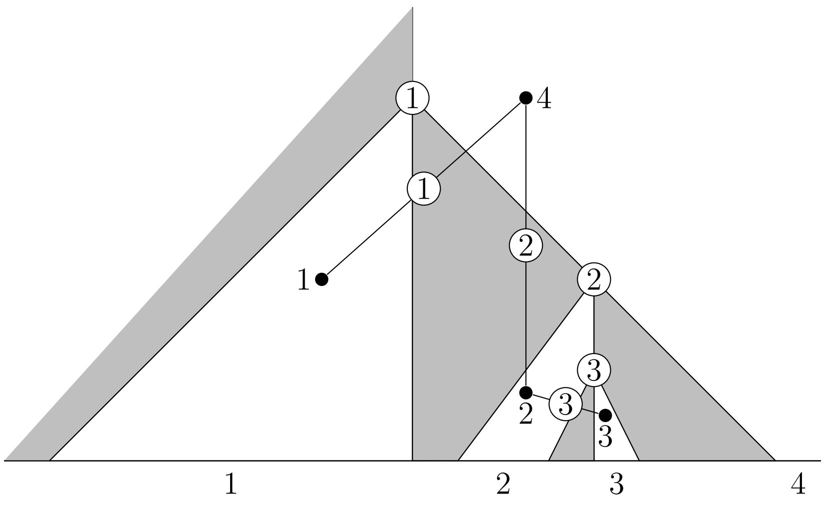
Polygon dissection P after 3.2.1 Step 3.



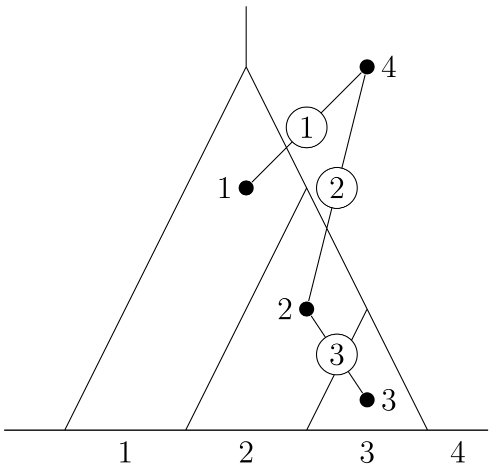
3.2.2 Steps 1 & 2.



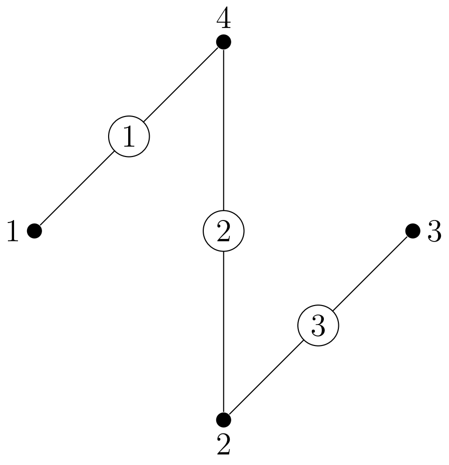
3.2.2 Step 3 (with shading as in Remark 3.22).



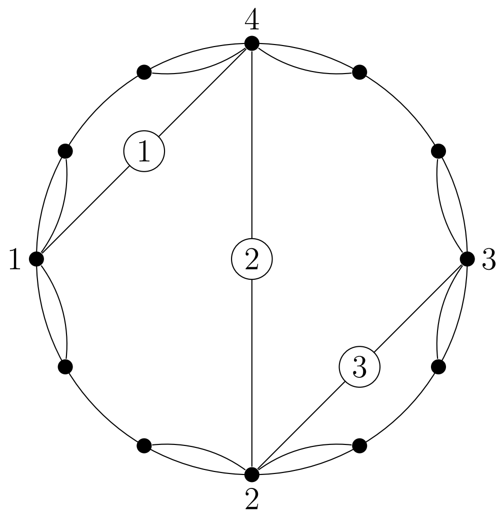
Ternary tree T after 3.2.2 Step 4 and 3.2.3 Steps 1 & 2.



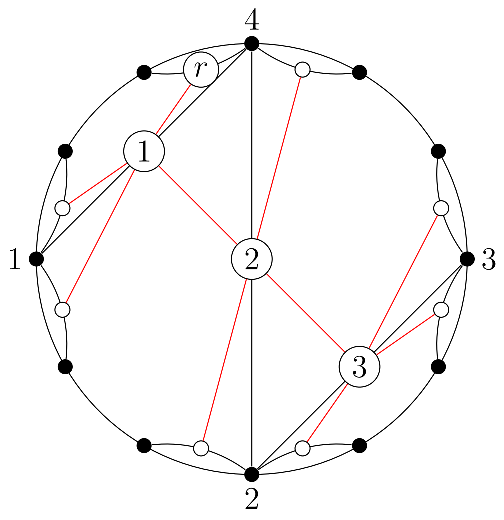
Binary tree B after 3.2.3 Step 3.



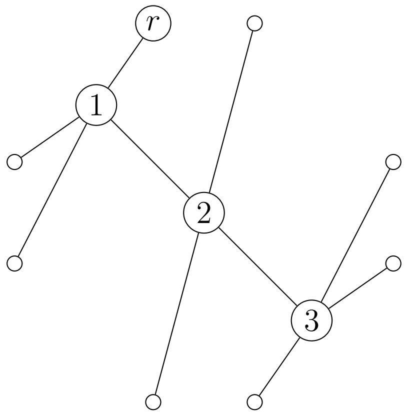
Hook-arrow tree H after 3.2.4.



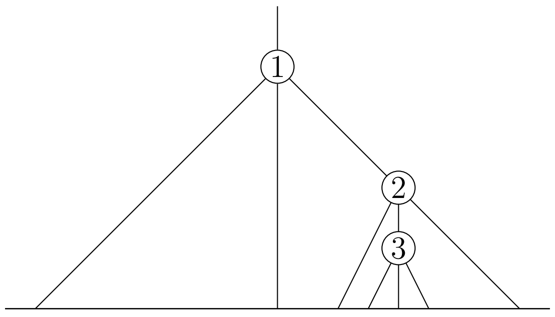
3.2.5 Step 1.



3.2.5 Step 2.



3.2.5 Step 2 (continued).



Ternary tree T after 3.2.5 Step 3.

Chapter 4

Symbols of multiple polylogarithms of a given depth

To find the symbol of a multiple polylogarithm for a given weight, we must currently consider all hook-arrow trees (or polygon dissections or binary trees). In this chapter we propose a more efficient method if the multiple polylogarithms is of a given depth. There is motivation for this from the physics community's interest in the symbol of multiple polylogarithms. We begin by noting a conjecture of Goncharov (as noted in [DGR11] on page 23, the conjecture was learned from Goncharov by word of mouth by H Gangl).

Conjecture 4.1. *Any multiple polylogarithm $I_{m_1,\ldots,m_k}(x_1,\ldots,x_m)$ with $m_j = 1$ for some j can be expressed in terms of multiple polylogarithms where no index is equal to* 1*.*

There is therefore motivation to examine multiple polylogarithms where the depth is lower than the weight, and so have a non-zero *co-depth*.

Definition 4.2. The **co-depth** of a depth d, weight w, multiple polylogarithm is equal to $w - d$.

Although there has been considerable interest in weight 4 polylogarithms for some time through their part in 2-loop Wilson loops (as in [GSVV10]), it is assumed that

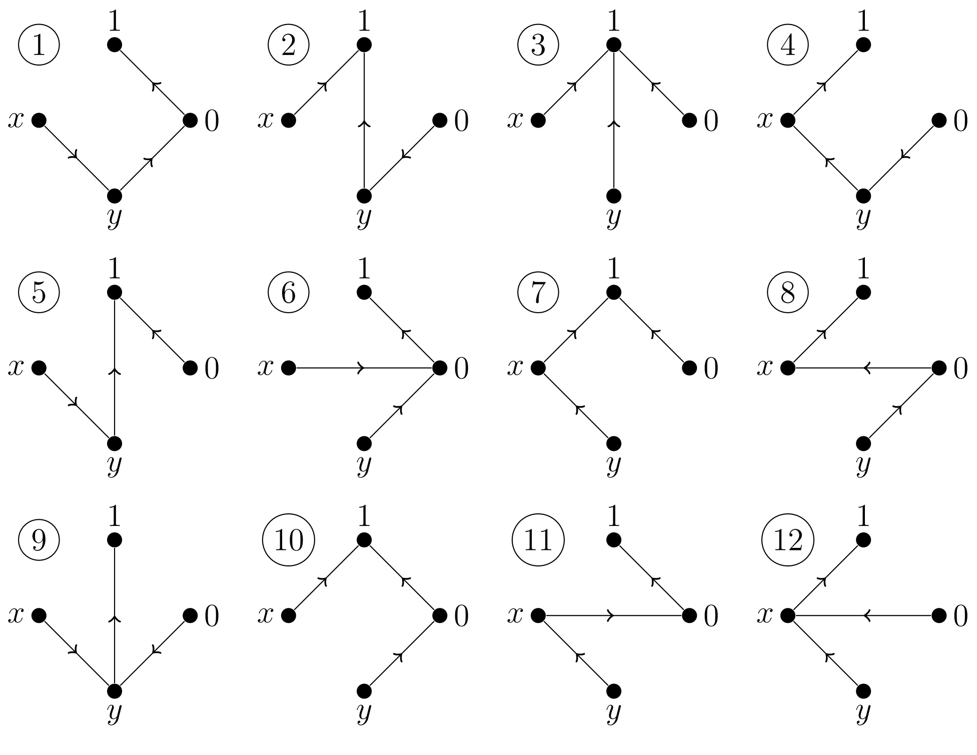
3-loop Wilson loops will consist of weight 6 polylogarithms (see [HK11]). By only considering multiple polylogarithms with indices greater than 1, the highest depth multiple polylogarithms of weight 4 are of depth 2 (indeed, there is only $I_{2,2}(x_1, x_2)$). If we move to weight 6 we consider $I_{2,2,2}(x_1, x_2, x_3)$. In this chapter we provide a method for finding $\mathcal{S}(I_{p,q,r}(x_1, x_2, x_3))$ efficiently for any $p, q, r \in \mathbb{N}$ and we provide the full symbol of $I_{2,2,2}(x_1, x_2, x_3)$ in Appendix C.

We begin by exploring depth 2 multiple polylogarithms, and specifically take the symbol of $I_{1,2}(x, y)$ as an example.

Example 4.3. The 4-gon representing $I_{1,2}(x, y)$ is

$$P(x, y, 0, 1).$$

As in the algorithm previously described we now form every hook-arrow tree with vertices $x, y, 0$ and 1. We get the following 12 trees.



We discard (give a zero coefficient in the symbol) trees 1, 6, 8, 10 and 11 as they have an arrow ending at the vertex 0, as described in Definition 2.20. We then discard trees 3, 5 and 7 for containing the edge [0, 1] (again due to Definition 2.20). The only trees contributing to the symbol are 2, 4, 9 and 12.

We see that the only way to form a hook-arrow tree that contributes to the symbol for $I_{1,2}(x, y)$ is when the vertex 0 is only attached to either vertex x or vertex y, and necessarily forms a directed edge $[0, x]$ or $[0, y]$.

Proposition 4.4. *Let τ be a hook-arrow tree on labelled vertices $v_1^\tau, \dots, v_n^\tau$. For τ to represent a term in the symbol with a non-zero coefficient, any vertices $v_i^\tau = 0$ (i.e. labelled with a 0) must only be contained in one edge and it must of the form $[0, t]$, where $t \not\equiv 0, 1$.*

Proof. We prove that no other possible edge can exist.

First, we prove that the vertex 0 can only have one other vertex directly connected to it. We want to show that if this is not the case then there must be a edge in the tree of the form $[a, 0]$, for some a, and so the tree would represent a term with a zero coefficient in the symbol. The algorithm dictates that the direction of the edges correspond to the direction towards the final vertex, v_n^τ, which is unique on a tree. Since the vertex 0 must be connected to the final edge by a unique sequence of edges then there will be an edge containing and directed away from it; we call this edge $[0, b]$.

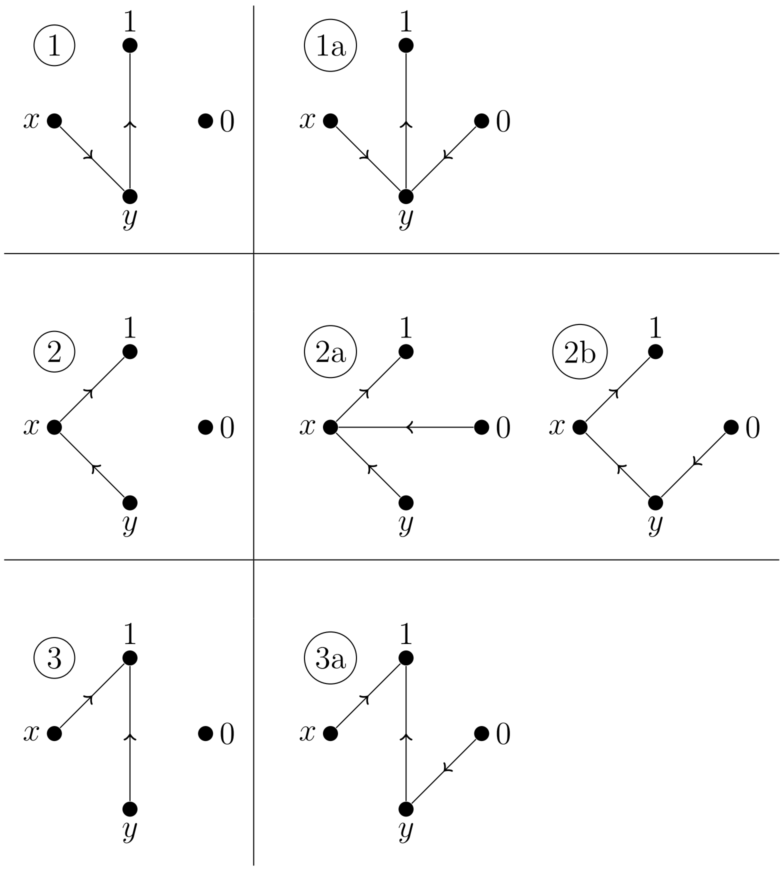
We also see that for any other vertex, $c_i \neq b$, directly connected to the vertex 0, then the unique path from c_i to the final vertex, v_n^τ, must contain $[0, b]$.

The edges containing each vertex c_i and 0 must therefore be directed $[c_i, 0]$ and therefore the tree will not contribute to the symbol (because $\mu([c_i, 0]) = 1$ from Definition 2.20). We conclude that for the tree to have a non-zero coefficient in the symbol, then the only edge connected to the vertex 0 must be $[0, b]$ for some b.

Finally we note that b must not equal 0 or 1 by Definition 2.20. $\square$

A vertex of a tree representing a weight $w = \sum r_s$, depth s, hook-arrow tree representing $I_{r_1,\dots,r_s}(x_1, ..., x_s)$ will either be a 0-vertex, an argument x_i, or the solitary, final vertex, 1. We put aside the 0-vertices and form every possible tree formed from the vertices labelled with the arguments x_i and the vertex labelled 1. In other words we find all possible hook-arrow trees of the vertices $(x_1, ..., x_s, 1)$. This will be the same as finding all possible hook-arrow trees for $I_{1,...,1}(x_1, ..., x_s)$. Since we know that each 0-vertex can only be added to this tree by attaching it to one of the x_i, we can find all possible hook-arrow trees with a non-zero coefficient in the symbol by exhausting possible additions of $w - s$ edges of the form $[0, x_i]$. There is some restriction where these are added as the edges of a hook-arrow tree must not interleave by definition. We now give some examples.

Example 4.5. We found all possible trees for $I_{1,2}(x, y)$ in Example 4.3. We corroborate the above method by finding these again.

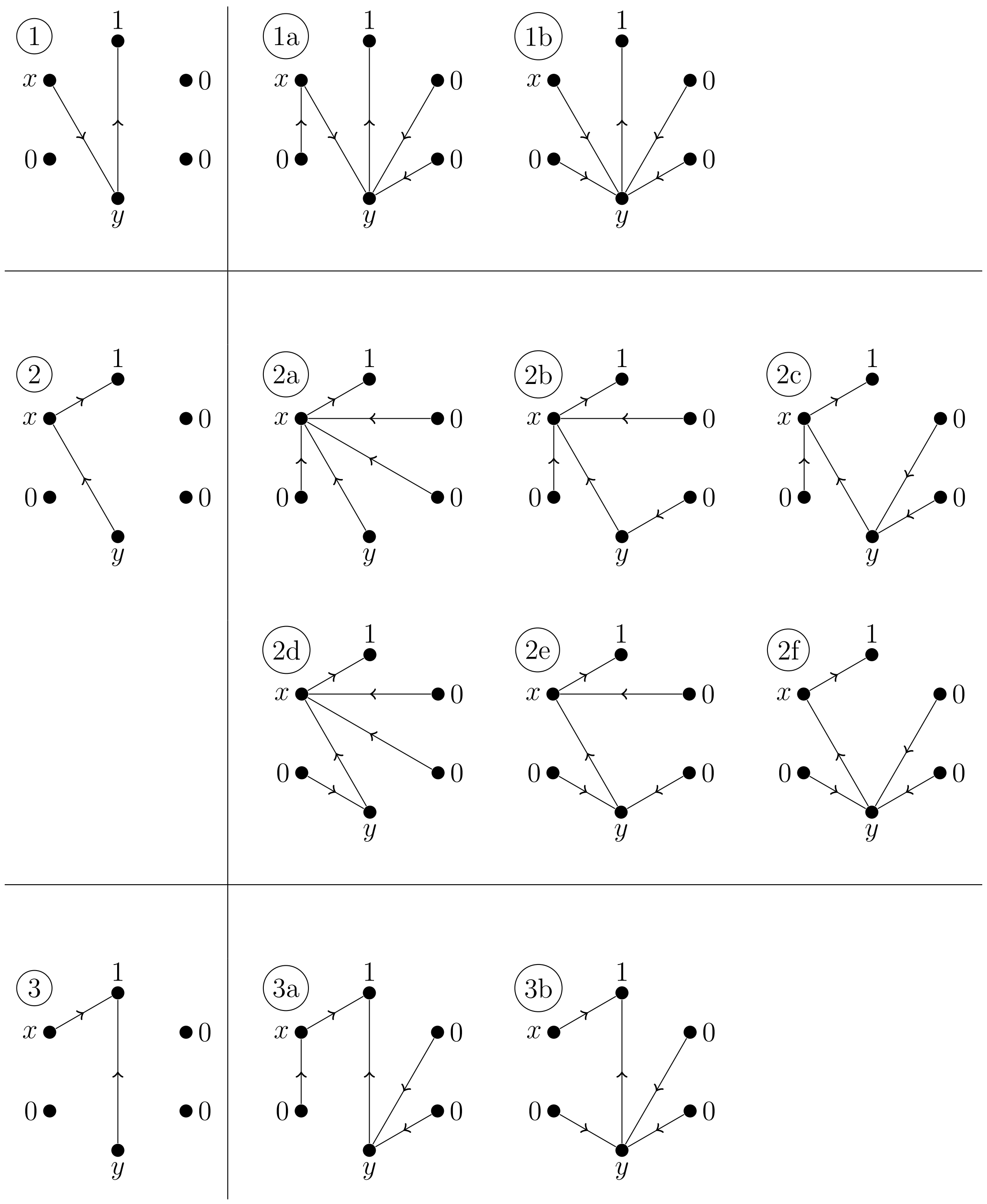


We see that in 1 and 3 there is only one possible place to add the edge containing the vertex 0, both to the vertex y. However in 2 we can attach the vertex 0 to either the vertex x or y, creating two terms of the symbol. The four trees $1a, 2a, 2b$ and $3a$ exactly match trees $9, 12, 4$ and 2 respectively from example 4.3.

Example 4.6. We now explore an example with more 0-vertices. The polygon representing $I_{2,3}(x,y)$ is

$$P(x,0,y,0,0,1).$$

We again find the three trees for the vertices $(x,y,1)$ and add all possible 0-vertices.



We can see that, because $r_1 = 2$, there is $r_1 - 1 = 1$ vertex labelled 0 that can be attached to either vertices x or y in trees $1, 2$ and 3. Because $r_2 = 3$ there are $r_2 - 1 = 2$ vertices labelled 0 that can only be attached to vertex y in trees 1 and 3, but to either vertices x or y in tree 2. Because the cyclic order of the vertices is fixed, and because arrows must not interleave, there are now three different arrangements for the $r_2 - 1$ vertices labelled 0 in tree 2. These will be to include the edges

$$\{[0, x], [0, x]\}, \quad \{[0, x], [0, y]\} \quad \text{or} \quad \{[0, y], [0, y]\}.$$

We have 10 hook-arrow trees on the vertices $(x, 0, y, 0, 0, 1)$ with a non-zero coefficient and it is an exhaustive list. If we were to write out all trees representing dissections of a 6-gon and then remove trees with a zero coefficient, we would have sorted through 273 trees to find the 10 required.

4.1 The symbol of $I_{r_1,r_2}(x_1, x_2)$

Examples 4.5 and 4.6 motivate an attempt to generalise finding the symbol for any depth 2 multiple polylogarithm. By using the 3 possible hook-arrow trees for vertices $(x, y, 1)$ and simply adding 0-vertices either between the vertices labelled x and y or between y and 1, and considering all combinations of attaching these to the hook-arrow tree, we can group all the terms in the symbol into three summations.

Proposition 4.7. *A multiple polylogarithm $I_{r_1,r_2}(x_1, x_2)$, represented by the polygon*

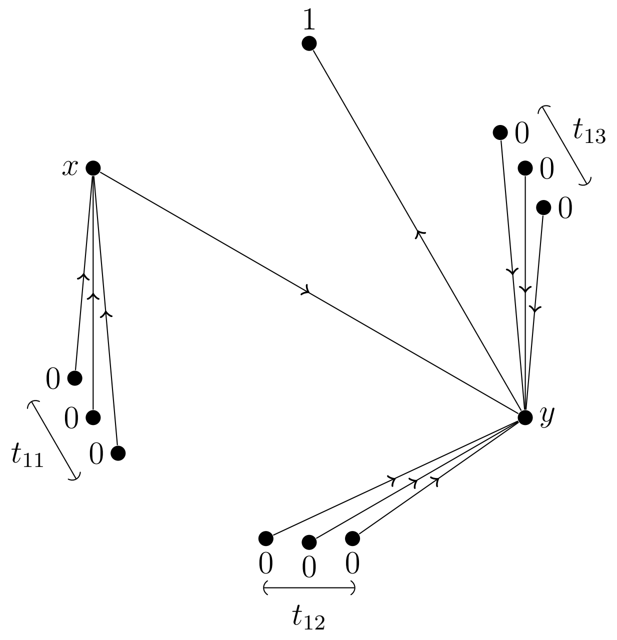
$$P(x_1, \underbrace{0, ..., 0}_{r_1-1}, x_2, \underbrace{0, ..., 0}_{r_2-1}, 1),$$

has $r_1r_2 + 2r_1$ terms in the symbol with non-zero coefficients.

Proof. As in Examples 4.5 and 4.6, we first ignore the 0-vertices and consider the three hook-arrow trees on the vertices $(x, y, 1)$. We now add $r_1 - 1$ vertices labelled 0 between vertices x and y and $r_2 - 1$ vertices labelled 0 between vertices y and 1. We run through every combination of attaching the 0-vertices.

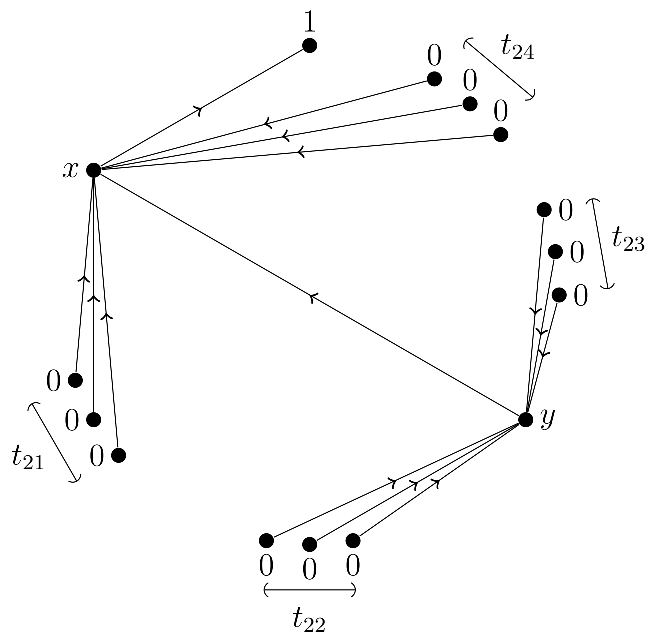
Firstly we have r_1 trees of the following form, where

$$t_{11} + t_{12} = r_1 - 1 \quad \text{and} \quad t_{13} = r_2 - 1.$$



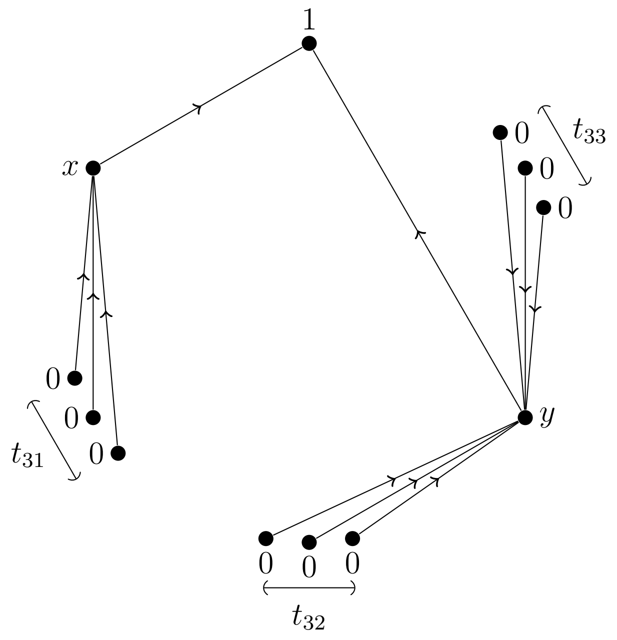
Then we have $r_1 r_2$ trees of the following form, where

$$t_{21} + t_{22} = r_1 - 1 \quad \text{and} \quad t_{23} + t_{24} = r_2 - 1.$$



Finally we have r_1 trees of the following form, where

$$t_{31} + t_{32} = r_1 - 1 \quad \text{and} \quad t_{33} = r_2 - 1.$$



In total we have $r_1r_2 + 2r_1$ hook-arrow trees with a non-zero coefficient. □

We can now extract the symbol using the algorithm and write the terms in the symbol. We first require a basic fact about shuffling copies of the same variable.

Proposition 4.8. *Given any a and $b, c \in \mathbb{Z}$ then*

$$a^{\otimes b} \sqcup\!\sqcup\, a^{\otimes c} = \binom{b+c}{c} a^{\otimes(b+c)} = \binom{b+c}{b} a^{\otimes(b+c)}$$

Proof. The result follows from a standard result that there are $\binom{b+c}{c}$ ways to arrange b objects into $c+1$ boxes given that order is retained. □

Theorem 4.9. *The symbol for the multiple polylogarithm $I_{r_1,r_2}(x_1,x_2)$ is the following formal sum of tensor products.*

$$\sum_{t_1+t_2=r_1-1} (-1)^{t_1+r_2-1}\left[\left(1-\frac{1}{x_2}\right)\otimes x_2^{\otimes(r_2-1)} \sqcup\!\sqcup \left(\left(1-\frac{x_2}{x_1}\right)\otimes x_1^{\otimes t_1} \sqcup\!\sqcup\, x_2^{\otimes t_2}\right)\right]$$

$$+ \sum_{\substack{t_1+t_2=r_1-1\\ t_3+t_4=r_2-1}} (-1)^{t_1+r_2}\binom{t_2+t_4}{t_2}\left[\left(1-\frac{1}{x_1}\right)\otimes x_1^{\otimes t_3}\otimes\left(1-\frac{x_1}{x_2}\right)\otimes x_2^{\otimes t_2+t_4} \sqcup\!\sqcup\, x_1^{\otimes t_1}\right]$$

$$+ \sum_{t_1+t_2=r_1-1} (-1)^{t_1+r_2+1}\binom{t_2+r_2-1}{t_2}\left[\left(1-\frac{1}{x_1}\right)\otimes x_1^{\otimes t_1}\right.$$

$$\left.\sqcup\!\sqcup\left(\left(1-\frac{1}{x_2}\right)\otimes x_2^{\otimes(t_2+r_2-1)}\right)\right].$$

Proof. This is done by simply working out the tensor element for each tree in the proof of Proposition 4.7. The first summation covers trees of the type 'tree 1' in that proof, the second summation of type 'tree 2' and the third summation of type 'tree 3'. The sign for each term is calculated as in the algorithm previously described. Simplifying using Notation 1.1, and Proposition 4.8 we obtain the expression in the statement. □

4.2 The symbol of $I_{r_1,r_2,r_3}(x_1,x_2,x_3)$

The obvious next step is to consider depth 3 multiple polylogarithms. The polygon representing $I_{r_1,r_2,r_3}(x_1,x_2,x_3)$ will be

$$P(x_1,\underbrace{0,...,0}_{r_1-1},x_2,\underbrace{0,...,0}_{r_2-1},x_3,\underbrace{0,...,0}_{r_3-1},1).$$

We look at possible hook-arrow trees on vertices with these labels and again temporarily ignore the 0-vertices to leave the 4 vertices $(x_1,x_2,x_3,1)$. As seen in Example 2.6, there are 12 possible hook-arrow trees on 4 general vertices. We continue as before.

Proposition 4.10. *A multiple polylogarithm $I_{r_1,r_2,r_3}(x_1,x_2,x_3)$, represented by the polygon*

$$P(x_1,\underbrace{0,...,0}_{r_1-1},x_2,\underbrace{0,...,0}_{r_2-1},x_3,\underbrace{0,...,0}_{r_3-1},1),$$

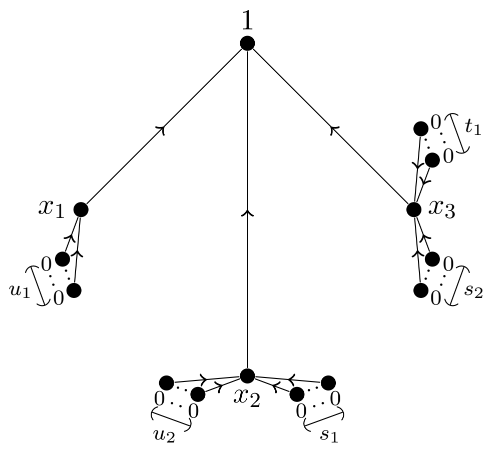
has

$$\frac{1}{2}r_1r_2(r_3+2)(r_1+r_2+r_3+5)$$

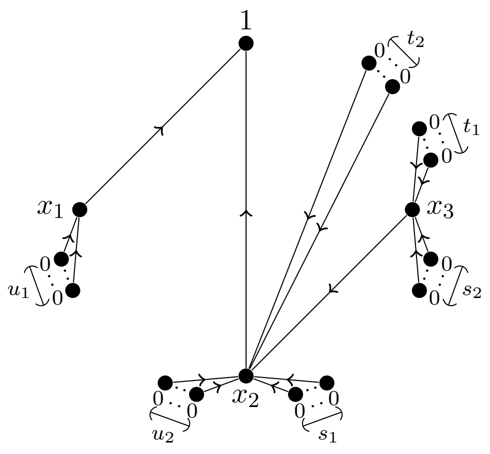
terms in the symbol with non-zero coefficients.

Proof. We consider the 12 possible hook-arrow trees with vertices $(x_1,x_2,x_3,1)$. We add r_1-1 vertices labelled 0 between vertices x_1 and x_2, we add r_2-1 vertices labelled 0 between vertices x_2 and x_3, and r_3-1 vertices labelled 0 between vertices x_3 and 1. We then run through every combination of possible ways of attaching the 0-vertices.

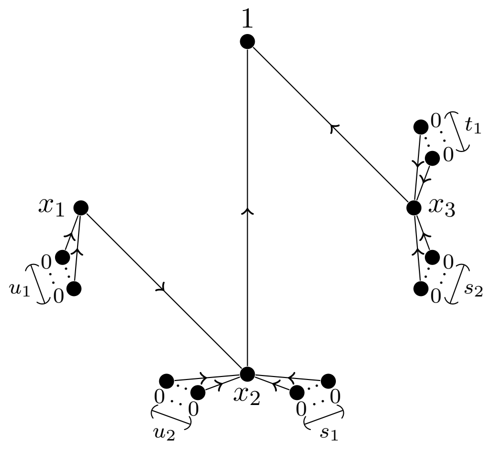
1.


1
x_1
x_2
x_3
u_1
u_2
s_1
s_2
t_1
0

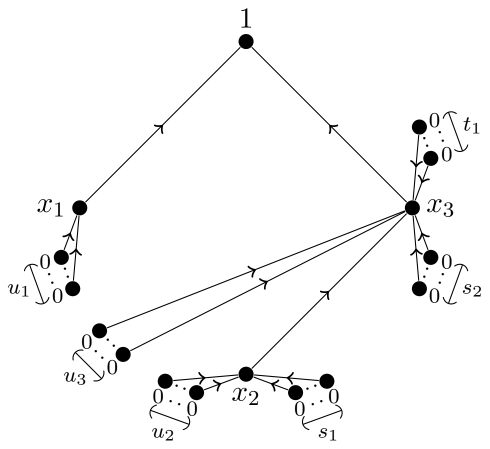
2.


1
x_1
x_2
x_3
u_1
u_2
s_1
s_2
t_1
t_2
0

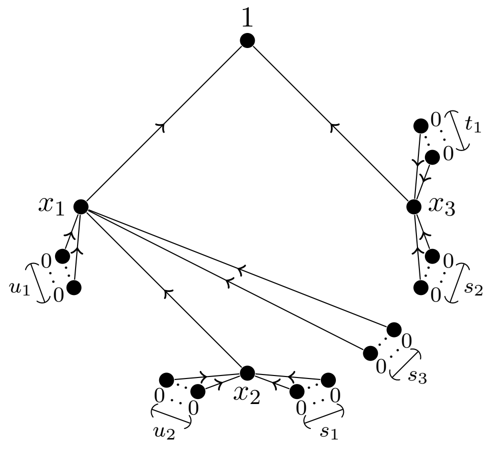
3.


1
x_1
x_2
x_3
u_1
u_2
s_1
s_2
t_1
0

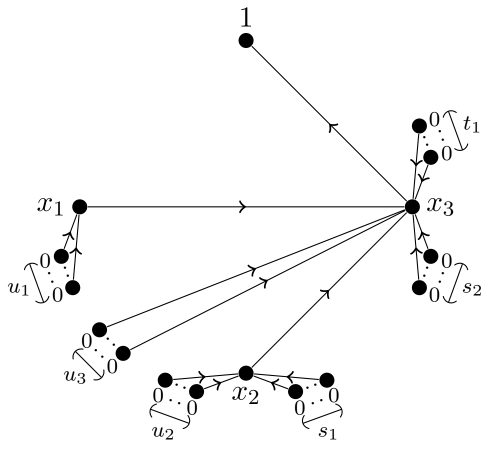
4.


1
x_1
x_2
x_3
u_1
u_2
u_3
s_1
s_2
t_1
0

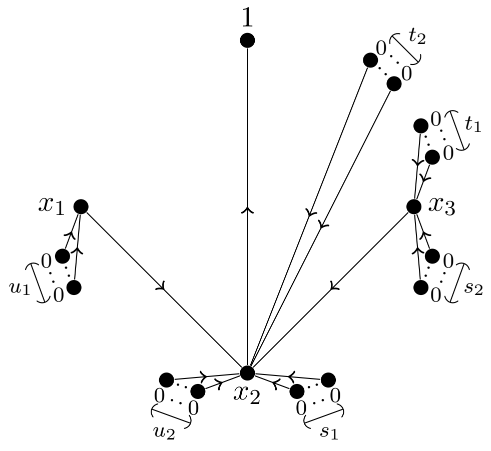
5.


1
x_1
x_2
x_3
u_1
u_2
s_1
s_2
s_3
t_1
0

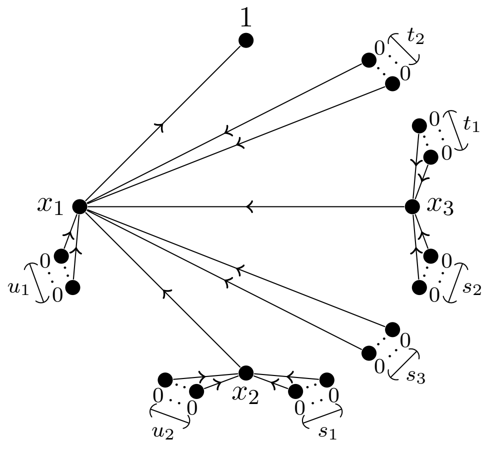
6.


1
x_1
x_2
x_3
u_1
u_2
u_3
s_1
s_2
t_1
0

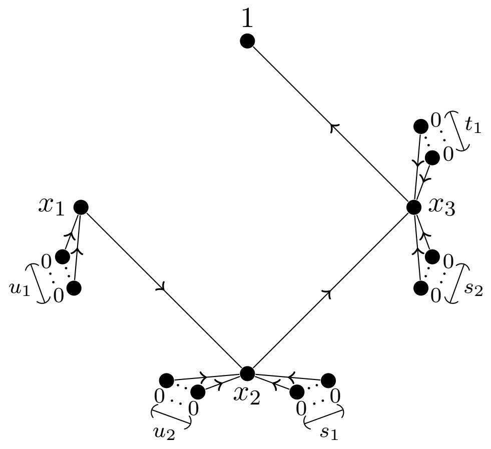
7.


1
t_2
t_1
x_1
x_3
u_1
s_2
x_2
u_2
s_1
0

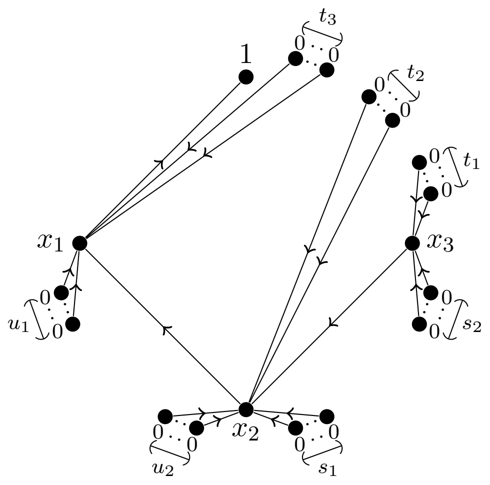
8.


1
t_2
t_1
x_1
x_3
u_1
s_2
s_3
x_2
u_2
s_1
0

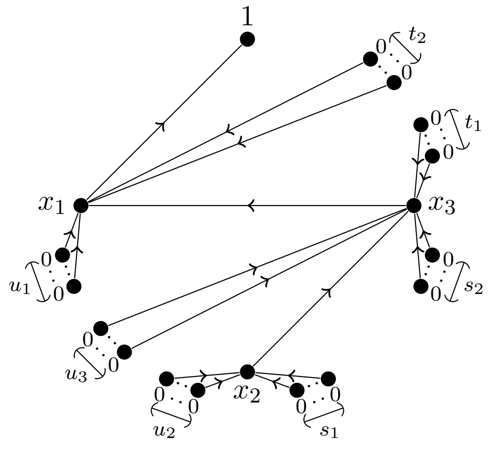
9.


1
t_1
x_1
x_3
u_1
s_2
x_2
u_2
s_1
0

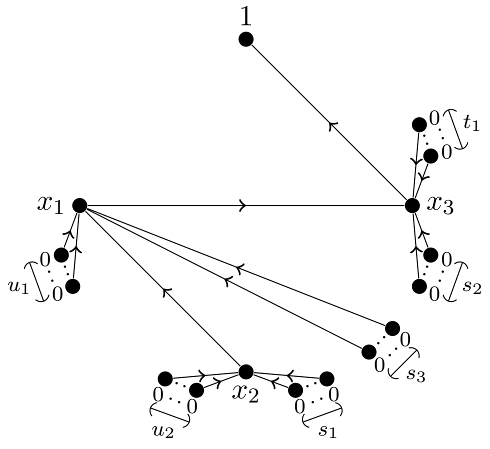
10.


t_3
1
t_2
t_1
x_1
x_3
u_1
s_2
x_2
u_2
s_1
0

11.


1
t_2
t_1
x_1
x_3
u_1
s_2
u_3
x_2
u_2
s_1
0

12.


1
t_1
x_1
x_3
u_1
s_2
s_3
x_2
u_2
s_1
0

We now enumerate the number of hook-arrow trees represented by each diagram.

Diagram	#{Hook-arrow trees}
1	r_1r_2
2	$r_1r_2r_3$
3	r_1r_2
4	$\binom{r_1+1}{2}r_2$
5	$r_1\binom{r_2+1}{2}$
6	$\binom{r_1+1}{2}r_2$
7	$r_1r_2r_3$
8	$r_1\binom{r_2+1}{2}r_3$
9	r_1r_2
10	$r_1r_2\binom{r_3+1}{2}$
11	$\binom{r_1+1}{2}r_2r_3$
12	$r_1\binom{r_2+1}{2}$
Total	$\frac{1}{2}r_1r_2(r_3+2)(r_1+r_2+r_3+5)$

□

Notation 4.11. Before we explicitly give the symbol for a depth 3 multiple polylogarithm, for ease of notation we write

$$\sum_{*} := \sum_{\substack{\sum u_i = r_1-1 \\ \sum s_i = r_2-1 \\ \sum t_i = r_3-1}} .$$

Note that this notation only applies for the following Theorem.

Theorem 4.12. *The symbol for the multiple polylogarithm $I_{r_1,r_2,r_3}(x_1,x_2,x_3)$ can be written in tensor form as the sum of* 12 *summations. The sum of products now follows.*

$$\sum_{*} (-1)^{u_1+s_1+r_3-1}\binom{s_1+u_2}{s_1}\binom{s_2+r_3-1}{s_2}\left[\left(1-\frac{1}{x_1}\right)\otimes x_1^{\otimes u_1} ⧢ \left(\left(1-\frac{1}{x_2}\right)\otimes x_2^{\otimes(s_1+u_2)} ⧢ \left(\left(1-\frac{1}{x_3}\right)\otimes x_3^{\otimes(s_2+r_3-1)}\right)\right)\right]$$

$$+ \sum_{*} (-1)^{u_1+s_1+r_3}\binom{s_2+t_1}{s_2}\left[\left(1-\frac{1}{x_1}\right)\otimes x_1^{\otimes u_1} ⧢ \left(\left(1-\frac{1}{x_2}\right)\otimes x_2^{\otimes u_2} ⧢ \left(x_2^{\otimes t_2}\otimes\left(1-\frac{x_2}{x_3}\right)\otimes x_3^{\otimes(s_2+t_1)} ⧢ x_2^{\otimes s_1}\right)\right)\right]$$

$$+ \sum_{*} (-1)^{u_1+s_1+r_3-1}\binom{s_2+r_3-1}{s_2}\left[\left(1-\frac{1}{x_2}\right)\otimes x_2^{\otimes s_1} ⧢ \left(\left(1-\frac{1}{x_3}\right)\otimes x_3^{\otimes(s_2+r_3-1)}\right) ⧢ \left(\left(1-\frac{x_2}{x_1}\right)\otimes x_1^{\otimes u_1} ⧢ x_2^{\otimes u_2}\right)\right]$$

$$+ \sum_{*} (-1)^{u_1+s_1+r_3-1}\binom{s_1+u_2}{s_1}\left[\left(1-\frac{1}{x_1}\right)\otimes x_1^{\otimes u_1} ⧢ \left(\left(1-\frac{1}{x_3}\right)\otimes x_3^{\otimes(r_3-1)} ⧢ \left(x_3^{\otimes u_3}\otimes\left(1-\frac{x_3}{x_2}\right)\otimes x_2^{\otimes(s_1+u_2)} ⧢ x_3^{\otimes s_2}\right)\right)\right]$$

$$+ \sum_{*} (-1)^{u_1+s_1+s_2+r_3}\binom{s_1+u_2}{s_1}\binom{s_3+r_3-1}{s_3}\left[\left(\left(1-\frac{1}{x_1}\right)\otimes x_1^{\otimes s_2}\otimes\left(1-\frac{x_1}{x_2}\right)\otimes x_2^{\otimes(s_1+u_2)} ⧢ x_1^{\otimes u_1}\right) ⧢ \left(\left(1-\frac{1}{x_3}\right)\otimes x_3^{\otimes(s_3+r_3-1)}\right)\right]$$

$$+ \sum_{*} (-1)^{u_1+s_1+r_3-1}\binom{s_1+u_2}{s_1}\left[\left(1-\frac{1}{x_3}\right)\otimes x_3^{\otimes(r_3-1)} ⧢ \left(\left(1-\frac{x_3}{x_1}\right)\otimes x_1^{\otimes u_1} ⧢ \left(x_3^{\otimes u_3}\otimes\left(1-\frac{x_3}{x_2}\right)\otimes x_2^{\otimes(s_1+u_2)} ⧢ x_3^{\otimes s_2}\right)\right)\right]$$

$$+\sum_{*}(-1)^{u_1+s_1+r_3}\binom{s_2+t_1}{s_2}\Big[\Big(1-\frac{1}{x_2}\Big)\otimes\Big(\Big(1-\frac{x_2}{x_1}\Big)\otimes x_2^{\otimes u_2}\sqcup x_1^{\otimes u_1}\Big)\Big)\sqcup\Big(x_2^{\otimes t_2}\otimes\Big(1-\frac{x_2}{x_3}\Big)\otimes x_2^{\otimes s_1}\sqcup x_3^{\otimes(s_2+t_1)}\Big)\Big]$$

$$+\sum_{*}(-1)^{u_1+s_1+s_3+r_3-1}\binom{s_2+t_1}{s_2}\binom{s_1+u_2}{s_1}\Big[\Big(1-\frac{1}{x_1}\Big)\otimes x_1^{\otimes t_2}\otimes\Big(1-\frac{x_1}{x_3}\Big)\otimes x_3^{\otimes(s_2+t_1)}\sqcup\Big(x_1^{\otimes s_3}\otimes\Big(1-\frac{x_1}{x_2}\Big)\otimes x_2^{\otimes(s_1+u_2)}\sqcup x_1^{\otimes u_1}\Big)\Big]$$

$$+\sum_{*}(-1)^{u_1+s_1+r_3-1}\Big[\Big(1-\frac{1}{x_3}\Big)\otimes\Big(x_3^{\otimes(r_3-1)}\sqcup\Big(\Big(1-\frac{x_3}{x_2}\Big)\otimes x_3^{\otimes s_2}\sqcup x_2^{\otimes s_1}\sqcup\Big(\Big(1-\frac{x_2}{x_1}\Big)\otimes x_1^{\otimes u_1}\sqcup x_2^{\otimes u_2}\Big)\Big)\Big)\Big)\Big]$$

$$+\sum_{*}(-1)^{u_1+s_1+r_3-1}\binom{s_2+t_1}{s_2}\Big[\Big(1-\frac{1}{x_1}\Big)\otimes x_1^{\otimes t_3}\otimes\Big(1-\frac{x_1}{x_2}\Big)\otimes x_1^{\otimes u_1}\sqcup x_2^{\otimes u_2}\sqcup\Big(x_2^{\otimes t_2}\otimes\Big(1-\frac{x_2}{x_3}\Big)\otimes x_2^{\otimes s_1}\sqcup x_3^{\otimes(s_2+t_1)}\Big)\Big]$$

$$+\sum_{*}(-1)^{u_1+s_1+r_3}\binom{s_1+u_2}{s_1}\Big[\Big(1-\frac{1}{x_1}\Big)\otimes x_1^{\otimes t_2}\otimes\Big(1-\frac{x_1}{x_3}\Big)\otimes x_1^{\otimes u_1}\sqcup x_3^{\otimes t_1}\sqcup\Big(x_3^{\otimes u_3}\otimes\Big(1-\frac{x_3}{x_2}\Big)\otimes x_3^{\otimes s_2}\sqcup x_2^{\otimes(s_1+u_2)}\Big)\Big]$$

$$+\sum_{*}(-1)^{u_1+s_1+s_3+r_3}\binom{s_1+u_2}{s_1}\Big[\Big(1-\frac{1}{x_3}\Big)\otimes x_3^{\otimes(r_3-1)}\sqcup\Big(\Big(1-\frac{x_3}{x_1}\Big)\otimes x_3^{\otimes s_2}\sqcup\Big(x_1^{\otimes s_3}\otimes\Big(1-\frac{x_1}{x_2}\Big)\otimes x_1^{\otimes u_1}\sqcup x_2^{\otimes(s_1+u_2)}\Big)\Big)\Big].$$

Proof. As in the case for $I_{r_1,r_2}(x_1, x_2)$ we examine each tree in the proof of proposition 4.10 and apply the algorithm for finding the symbol from a hook-arrow tree. We then simplify as before. $\square$

In Appendix C we give the symbol of a depth 3, weight 6 multiple polylogarithm, $I_{2,2,2}(x, y, z)$, which the above method allows us to find relatively easily.

4.3 Higher depths

The above process can of course be generalised to higher depths and can provide benefits to calculating symbols for multiple polylogarithms with a non-zero co-depth. The process amounts to effectively reducing the complexity of the calculation of the symbol by the co-depth.

A general multiple polylogarithm $I_{r_1,...,r_k}(x_1, ..., x_k)$ of depth k and weight

$$n = r_1 + ... + r_k$$

will correspond to a polygon

$$P(x_1, \underbrace{0, ..., 0}_{r_1-1}, x_2, \underbrace{0, ..., 0}_{r_2-1}, ..., x_k, \underbrace{0, ..., 0}_{r_k-1}, 1).$$

As before we first ignore the sides labelled and, switching to hook-arrow trees, look at possible hook-arrow trees on the vertices labelled $(x_1, ..., x_k, 1)$ of which there are

$$\frac{1}{2k+1}\binom{3k}{k},$$

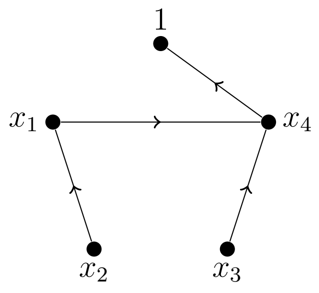
as in Proposition 2.33. The process of enumerating and evaluating the number of ways to add the 0-vertices gets more complicated as the weight and co-depth increases. However, we now further justify the advantages of this process rather than calculating all hook-arrow trees on n vertices and disregarding the terms with coefficient zero. We give a brief discussion of the symbol of depth 4 multiple polylogarithms.

4.3.1 Discussion on the symbol of a general depth 4 multiple polylogarithm

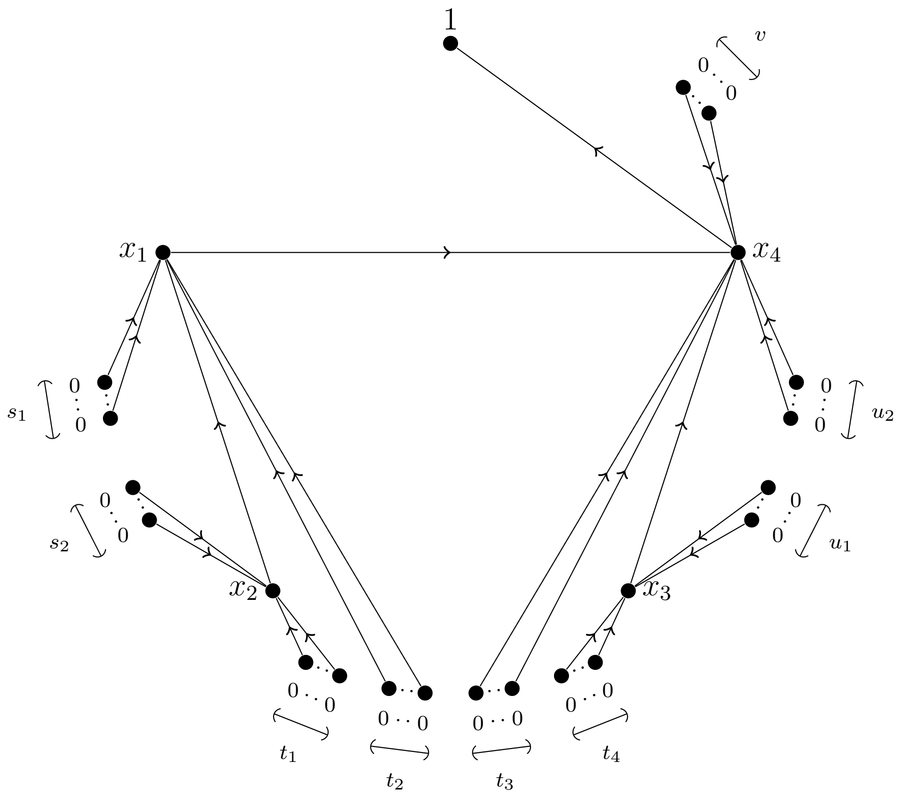
For a general depth 4 multiple polylogarithm we can give the symbol as the sum of 55 summations. This is because we have

$$55 = \frac{1}{2 \cdot 4 + 1} \binom{3 \cdot 4}{4}$$

possible hook-arrow trees on the vertices $(x_1, x_2, x_3, x_4, 1)$. One possible hook-arrow tree on these 5 vertices is



which, if we now add all zero vertices representing the vertices of a hook-arrow tree for a general depth 4 multiple polylogarithm $I_{r_1,\ldots,r_4}(x_1, \ldots, x_4)$ will give us



where

$$s_1 + s_2 = r_1 - 1, \quad t_1 + t_2 + t_3 + t_4 = r_2 - 1, \quad u_1 + u_2 = r_3 - 1, \quad v = r_4 - 1.$$

When we apply the algorithm the hook-arrow trees represented by the above diagram corresponds to the following terms in the symbol.

$$\sum_{\substack{\sum u_i = r_1 - 1 \\ \sum s_i = r_2 - 1 \\ \sum t_i = r_3 - 1}} \kappa_{s,t,u,v} \left[\left(1 - \frac{1}{x_4}\right) \otimes x_3^{\otimes(v)} ⧢ \left(\left(1 - \frac{x_4}{x_1}\right) \otimes \Upsilon^1_{s,t,u,v} ⧢ \Upsilon^2_{s,t,u,v} \right) \right]$$

where

$$\begin{aligned}
\kappa_{s,t,u,v} &= (-1)^{1+s_1+t_1+t_2+u_1+v_1} \binom{s_2+t_1}{s_2} \binom{t_4+u_1}{t_4}, \\
\Upsilon^1_{s,t,u,v} &= x_1^{\otimes t_2} \otimes \left(1 - \frac{x_1}{x_2}\right) \otimes x_1^{\otimes s_1} ⧢ x_2^{\otimes(s_2+t_1)}, \\
\Upsilon^2_{s,t,u,v} &= x_4^{\otimes t_4} \otimes \left(1 - \frac{x_4}{x_3}\right) \otimes x_4^{\otimes u_2} ⧢ x_2^{\otimes(u_1+t_4)}.
\end{aligned}$$

To find the full general form of the symbol of $I_{r_1,\dots,r_4}(x_1,\dots,x_4)$ we would add this to the 54 other summations, found in a similar way. This is fairly unwieldy. The following example should demonstrate why, even for a very simple depth 4 multiple polylogarithm, it is still vastly easier than computing all dissections on n vertices.

Example 4.13. We consider the multiple polylogarithm $I_{3,2,3,2}(x_1, x_2, x_3, x_4)$ which has weight $n = 10$ and co-depth 6. If we try to find the full dissection head on we must first find all hook-arrow trees on 11 vertices, or possible dissections of a polygon with 11 sides, of which there will be

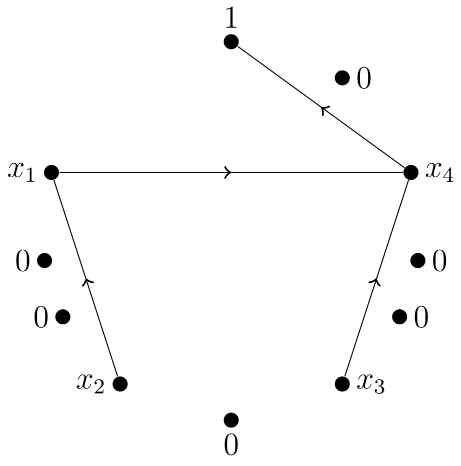
$$\frac{1}{2 \cdot 11 - 1} \binom{3 \cdot (11 - 1)}{11 - 1} = 1430715.$$

Each of these will then need to be decorated with

$$(x_1, 0, 0, x_2, 0, x_3, 0, 0, x_4, 0, 1),$$

examined to see if they have coefficient zero (if there is a disregarded edge/2-gon), and if not, have the corresponding tensor product calculated. This will be a lengthy process even for a computer.

On the other hand, by first considering the hook-arrow trees on the vertices labelled $(x_1, x_2, x_3, x_4, 1)$ and attaching zeros we need only consider, essentially, 55 trees. For $I_{3,2,3,2}(x_1, x_2, x_3, x_4)$ the above example of a dissection will be trees of the form



where we run through all possible ways of attaching the 0-vertices.

We can then use the summation we have already established by setting $r_1 = 2, r_2 = 3, r_3 = 2$ and $r_4 = 3$, to find all tensor expansions. The above tree represents 36 possible hook-arrow trees. This can then be repeated, still in a fairly lengthy way, for the other 54 forms.

Remark 4.14. It is important to note that if the general 55 summation signs for depth 4 are formulated once then they can be used repeatedly for the symbol of any depth 4 multiple polylogarithm.

The above process will, however, require considerably less work than running through all $C(11) = 1430715$ (where C(n) is the Catalan numbers) possible hook-arrow trees on 11 vertices and then discarding trees that do not contribute to the symbol.

Chapter 5

The symbol of coloured multiple zeta values

In this chapter we give an application of polygons and hook-arrow trees, namely finding the symbol for coloured multiple zeta values (defined below in Definition 5.1).

The main theorem of this chapter is included in a joint paper with Claude Duhr and Herbert Gangl [DGR11]. The paper concerns an attempt to find a systematic approach to 'integrating' a symbol, i.e., how to construct a function corresponding to a given symbol. Having a good grasp of the generators of the kernel of the symbol map is important during this procedure. Coloured multiple zeta values are specialised multiple polylogarithms for which we can fully formulate the symbol in all weights.

Definition 5.1. A **coloured multiple zeta value** is defined to be

$$\zeta(m_1, ..., m_k; \varepsilon_1, ..., \varepsilon_k) := \sum_{0<n_1<...<n_k} \frac{\varepsilon_1^{n_1}\varepsilon_2^{n_2}\cdots\varepsilon_k^{n_k}}{n_1^{m_1}n_2^{m_2}\cdots n_k^{m_k}}.$$

with $m_i \in \mathbb{N}$ and $\varepsilon_i \in \{\pm 1\}$.

We note that coloured multiple zeta values are clearly a special class of multiple polylogarithms via the identity

$$\zeta(m_1, ..., m_k; \varepsilon_1, ..., \varepsilon_k) = (-1)^k I_{m_1,...,m_k}(\hat{\varepsilon_1}, ..., \hat{\varepsilon_k}) \quad \text{where} \quad \hat{\varepsilon_i} = \prod_{j=i}^{k} \varepsilon_j. \tag{5.1}$$

This follows directly from Theorem 0.10. We now formulate the symbol for coloured multiple zeta values.

Theorem 5.2. *For $\varepsilon_i \in \{-1, 1\}$ and $k \geq 1$,*

1. *When $m_i = 1$ for all i,*

$$\mathcal{S}\left(\zeta(\underbrace{1, \dots, 1}_{k\ times}; \varepsilon_1, \dots, \varepsilon_{k-1}, -1)\right) = (-1)^k 2^{\otimes k}.$$

2. *If at least one of the m_i is different from 1, then*

$$\mathcal{S}(\zeta(m_1, \dots, m_k; \varepsilon_1, \dots, \varepsilon_k)) = 0.$$

We prove Theorem 5.2 in the following section.

Remark 5.3. The theorem restricts to cases where $\varepsilon_k = -1$. This corresponds to coloured multiple zeta values which converge. We will, in Proposition 5.4, find the symbol of coloured multiple zeta values with this restriction lifted; where the coefficient of $2^{\otimes k}$ when the $m_i = 1$ is calculated using a binomial.

Using the correspondence between coloured multiple zeta values and multiple polylogarithms (from Equation 5.1) we can associate to $\zeta(m_1, ..., m_k; \varepsilon_1, ..., \varepsilon_k)$ the polygon

$$P(\hat{\varepsilon}_1, \underbrace{0, \dots, 0}_{m_1-1 \text{ times}}, \hat{\varepsilon}_2, \underbrace{0, \dots, 0}_{m_2-1 \text{ times}}, \dots, \hat{\varepsilon}_k \underbrace{0, \dots, 0}_{m_k-1 \text{ times}}, 1) \quad \text{where} \quad \hat{\varepsilon_i} = \prod_{j=i}^{k} \varepsilon_j.$$

We note that the factor $(-1)^k$ of Equation 5.1 must be kept in mind.

5.1 Proof of Theorem 5.2

In this section we prove Propositions 5.4 and 5.7. These are equivalent to statements 1 and 2 of Theorem 5.2, respectively, but are given in terms of the symbol attached to polygons. We will discuss the exact correspondence to Theorem 5.2 at the end of the section.

Proposition 5.4. *The symbol corresponding to the decorated polygon $P(x_1, ..., x_n, 1)$, for some $x_i \in \{-1, 1\}$, is equal to $\lambda_{a,n}(2^{\otimes n})$ for*

$$\lambda_{a,n} = (-1)^a \binom{n-1}{a} \quad \text{and} \quad a = n - \max\{i \mid x_i = -1\}.$$

We will prove Proposition 5.4 after noting the benefits of applying the Hölder convolution and proving a proposition involving generating functions.

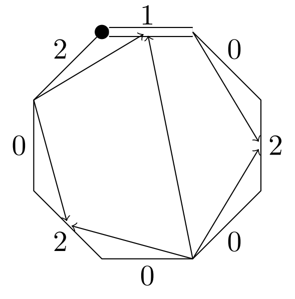
By the Hölder convolution introduced in Section 1.5.1 it follows that $P(x_1, ..., x_n, 1)$ has the same symbol as the polygon $P(1 - x_n, ..., 1 - x_1, 1)$ times a factor of $(-1)^n$. So, without loss of generality, when $x_i = \pm 1$, we consider the polygon

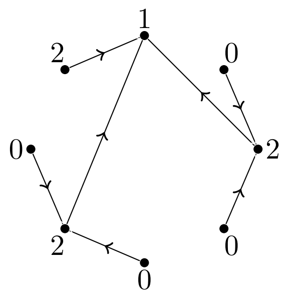
$$P(\underbrace{0, ..., 0}_{t_0}, 2, \underbrace{0, ..., 0}_{t_1}, 2, 0, ..., 0, 2, \underbrace{0, ..., 0}_{t_m}, 1),$$

find its symbol, and reintroduce a factor of $(-1)^n$ from Hölder convolution at the end. The move from sides labelled 1 and -1 to sides labelled 0 and 2 increases the number of dissections that do not contribute to the symbol. The combinatorics of the dissections of polygons of this type is best captured by hook-arrow trees due to the convenience of how to enumerate dissections with edges/vertices labelled 0. The enumeration of these 0-vertices is similar to the situation outlined in Chapter 4.

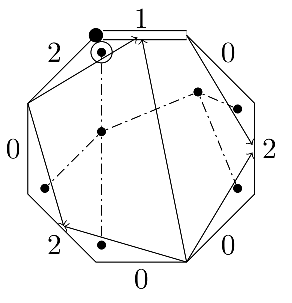
We first explore one possible polygon which represents a coloured multiple zeta value under the Hölder convolution.

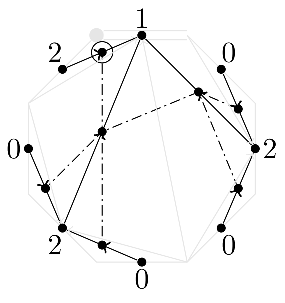
Example 5.5. For the polygon $P(2, 0, 2, 0, 0, 2, 0, 1)$ we have a possible dissection of



with hook-arrow tree


We now reintroduce the dual tree view of a dissection from Definition 1.33, beneficial in finding the symbol attached to a polygon. As with the dissection of a polygon using arrows, the dual tree can easily be seen in the hook-arrow tree view. For clarity we give the dual tree a dash-dotted line.





Polygon dissection and dual tree. Hook-arrow tree and dual tree.

For the proof of Proposition 5.4, we also require the following proposition which is proved using generating functions (and using methods outlined in [Wil94]).

Proposition 5.6. *If $c, n \in \mathbb{Z}^{\geq 0}$ then*

$$\sum_{i=0}^{n}(-1)^i\binom{n-i+c}{n-i}\binom{n+c+1}{i} = (-1)^n.$$

Proof. Let $r = n-i$ and view both sides as coefficients of generating functions. To prove the identity we therefore need to show

$$\rho = \sum_{n=0}^{\infty} x^n \sum_{r=0}^{n}(-1)^{n-r}\binom{r+c}{r}\binom{n+c+1}{n-r}$$

is equivalent to $\sum_{n=0}^{\infty}(-x)^n$. Firstly since for $r > n$ we have $\binom{n+c+1}{n-r} = 0$ then we can change the summation of r to run over all positive integers.

$$\rho = \sum_{n=0}^{\infty} x^n \sum_{r=0}^{\infty}(-1)^{n-r}\binom{r+c}{r}\binom{n+c+1}{n-r}.$$

We then re-order the summation signs, assuming small x. Also, since $\binom{n+c+1}{n-r} = \binom{n+c+1}{r+c+1}$, we have that

$$\begin{aligned}\rho &= \sum_{r=0}^{\infty}\binom{r+c}{r}\sum_{n=0}^{\infty}(-1)^{n-r}\binom{n+c+1}{r+c+1}x^n \\ &= \sum_{r=0}^{\infty}\frac{(-1)^{r-c-1}}{x^{c+1}}\binom{r+c}{r}\sum_{n=0}^{\infty}\binom{n+c+1}{r+c+1}(-x)^{n+c+1}.\end{aligned}$$

We let $s = n + c + 1$ and relabel. We can sum $\sum_{s=0}^{\infty}$ as opposed to $\sum_{s=c+1}^{\infty}$ because if $0 \leq s < c+1$ we have $s = n + c + 1 < r + c + 1$ and so $\binom{n+c+1}{r+c+1} = 0$.

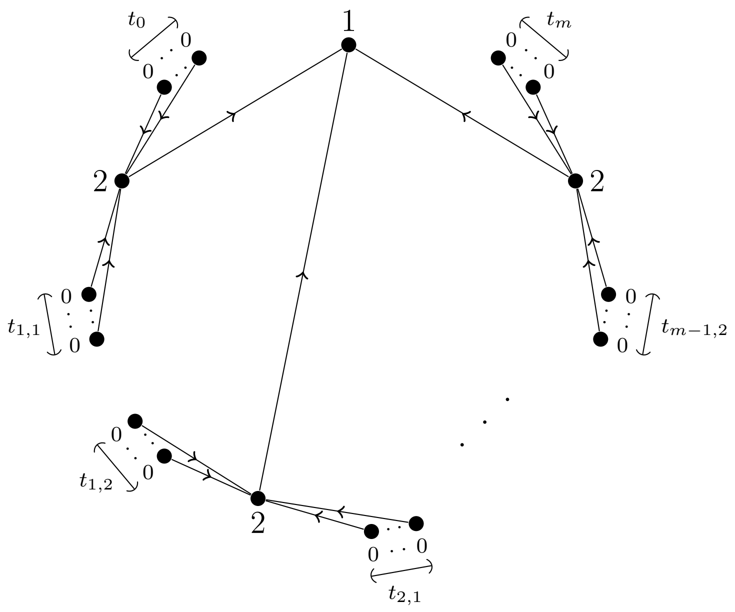
$$\begin{aligned}
\rho &= \sum_{r=0}^{\infty} \frac{(-1)^{r-c-1}}{x^{c+1}} \binom{r+c}{r} \sum_{s=0}^{\infty} \binom{s}{r+c+1} (-x)^s \\
&= \sum_{r=0}^{\infty} \binom{r+c}{r} \frac{x^r}{(1+x)^{r+c+2}} \\
&= \frac{1}{(1+x)^{c+2}} \sum_{r=0}^{\infty} \binom{r+c}{r} \left(\frac{x}{1+x}\right)^r \\
&= \frac{1}{(1+x)^{c+2}\left(1 - \frac{x}{1+x}\right)^{c+1}} \\
&= \frac{1}{1+x} \\
&= \sum_{n=0}^{\infty} (-x)^n.
\end{aligned}$$

. □

Proof. (of Proposition 5.4) After applying the Hölder convolution, and without loss of generality, we attempt to find all hook-arrow trees relating to the polygon

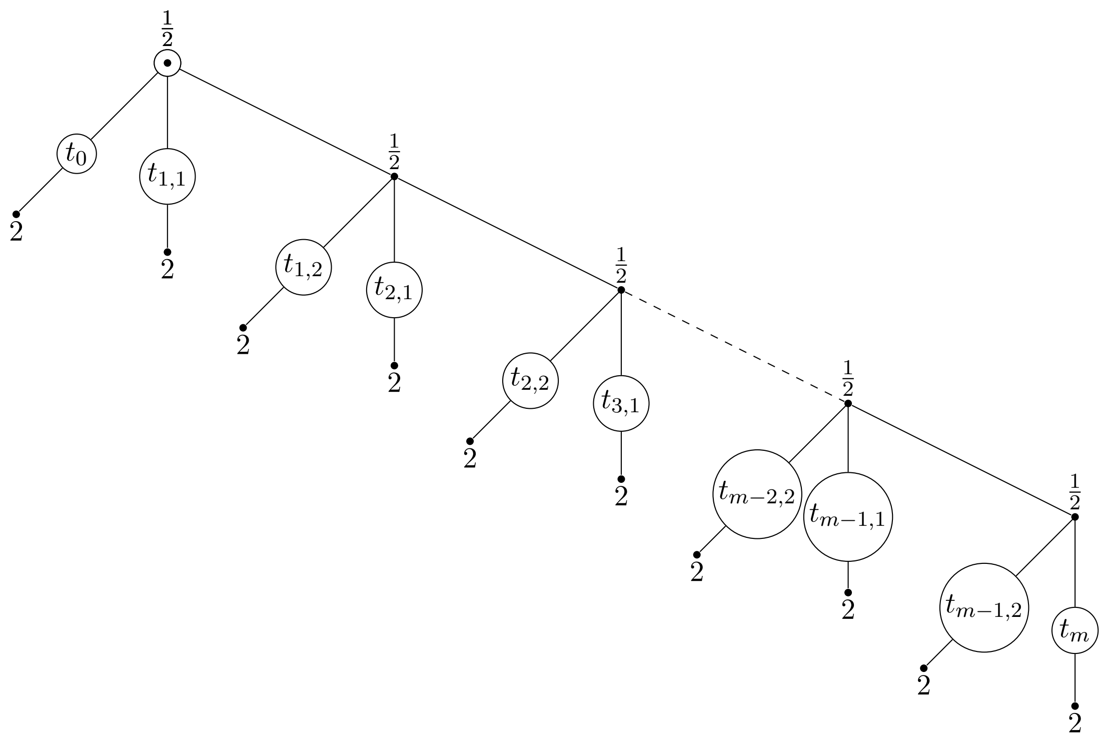
$$P(\underbrace{0, ..., 0}_{t_0}, 2, \underbrace{0, ..., 0}_{t_1}, 2, 0, \quad ... \quad , 0, 2, \underbrace{0, ..., 0}_{t_m}, 1)$$

which do not represent terms with coefficient 0 in the symbol. After some consideration and because of Proposition 4.4 we see that these hook-arrow trees must take the following form.

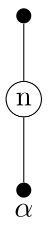


Each $t_{i,1}$ and $t_{i,2}$, for $i = 1, ..., m-1$ are chosen integers $0 \leq t_{i,1}, t_{i,2} \leq ti$ such that $t_{i,1} + t_{i,2} = t_i$. The choice of the $t_{i,j}$ arises from the fact that we can choose where to partition each group of t_i vertices labelled 0, for $i = 1, ..., m-1$, and attach them to the vertices labelled 2, remembering that the vertices must not cross. In the case of the function $P(2, 0, 2, 0, 0, 2, 0, 1)$ from Example 5.5, where $m = 3, t_0 = 0, t_1 = 1, t_2 = 2$ and $t_3 = 1$, we have 6 possible valid dissections, arising from two choices of the $t_{i,j}$ in $t_{1,1} + t_{1,2} = 1$ and three choices from $t_{2,1} + t_{2,2} = 2$. We note that example 5.5 explored the particular dissection where $t_{1,1} = 0, t_{1,2} = 1, t_{2,1} = 1$ and $t_{2,2} = 1$.

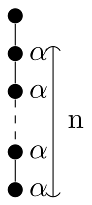
We will now show how it is possible to simplify this tree by, in effect, removing the edges joining vertices labelled 2 and 1 and replacing them with edges connecting vertices labelled 0 and 2. For this we return to the dual tree notation. The dual tree of the hook-arrow tree above is



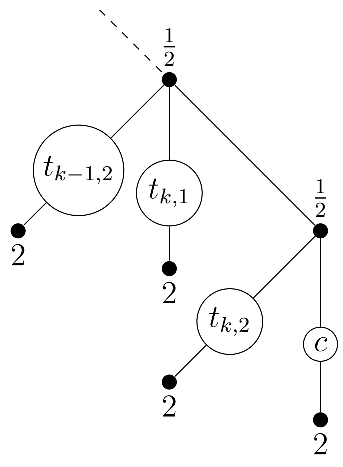
where we define



to be



We claim that

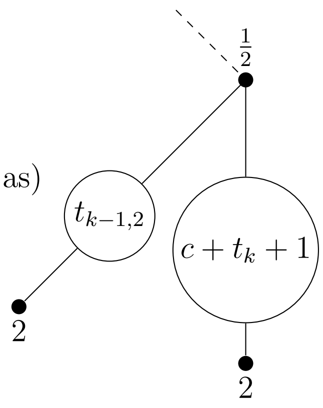


can be simplified to

(i.e., gives the same symbol term as)

$(-1)^{t_{k+1}+1}$

times the tree



We will now write the tensor product of the symbol of the left hand dual tree part in the above claim. The left hand dual tree part in the claim will have the symbol

$$\sum_{t_{k,1}=0}^{t_k} (-1)^{t_{k,1}} \left(\frac{1}{2} \otimes 2^{\otimes t_{k-1,2}} ⧢ 2^{\otimes t_{k,1}} ⧢ \left(\frac{1}{2} \otimes 2^{\otimes t_{k,2}} ⧢ 2^{\otimes c}\right)\right)$$

$$= - \sum_{t_{m-1,1}=0}^{t_{m-1}} (-1)^{t_{m-1,1}} \binom{t_{m-1} - t_{m-1,1} + t_m}{t_{m-1} - t_{m-1,1}} \cdot \binom{t_{m-1} + t_m + 1}{t_{m-1,1}} \cdot \left(\frac{1}{2} \otimes 2^{\otimes t_{k-1,2}} ⧢ 2^{\otimes (t_{m-1}+t_m+1)}\right)$$

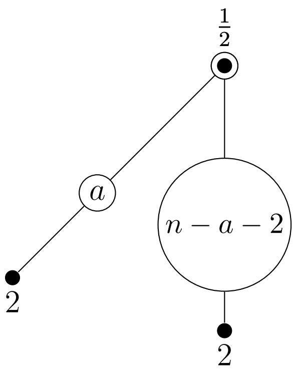
$$= (-1)^{t_{k+1}+1} \left(\frac{1}{2} \otimes 2^{\otimes t_{k-1,2}} ⧢ 2^{\otimes (c+t_k+1)}\right).$$

which is exactly the symbol for the tree on the right side. Note that we used Proposition 5.6 in the last line of the calculation.

By repeated application of this simplification, starting with $c = t_m$ and $k = m-1$, we will arrive at a much simplified tree. By noting that

$$n - a - 1 = \sum_{i=1}^{m} (t_i + 1)$$

and recalling that $t_0 = a$ we see that this tree is



times a factor of $(-1)^{(n-a-1)}$. This represents the symbol

$$(-1)^{n-a-1} \left(\frac{1}{2} \otimes 2^{\otimes a} ⧢ 2^{\otimes (n-a-2)}\right) = (-1)^{n-a} \binom{n-1}{a} 2^{\otimes n}.$$

Finally, by applying the factor of $(-1)^n$ from the application of the Hölder involution we find

$$\lambda_{a,n} = (-1)^a \binom{n-1}{a}.$$

□

Proposition 5.7. *The polygon*

$$P(x_1, \underbrace{0, ..., 0}_{m_1-1}, \ldots, x_k, \underbrace{0, ..., 0}_{m_k-1}, 1)$$

for $x_i \in \{-1, 1\}$ *and at least one of the* $m_i \neq 1$*, has a symbol with coefficient* 0*.*

Proof. (Sketch) After applying the Hölder involution we try to find possible hook-arrow trees which do not correspond to terms with coefficient 0 in the symbol. The vertices of the hook-arrow tree will be labelled corresponding to the sides of the polygon

$$P(\gamma_{1,1}, ..., \gamma_{t_0,1}, 2, \gamma_{1,2}, ..., \gamma_{t_1,2}, 2, ..., 2, \gamma_{1,m}, ..., \gamma_{t_m,m}, 1).$$

where all the $\gamma_{i,j}$ are equal to either 0 or 1. As in the proof of Proposition 5.4, the vertices labelled 2 must connect directly to the final 1 and the vertices labelled 0 must connect to a vertex labelled 2. However, there is no way to to connect the vertices labelled 1 to any other vertex without setting the coefficient of the term to 0. There is therefore no term that has a non-zero coefficient. □

5.2 Correspondence between Propositions 5.4 and 5.7 and Theorem 5.2

The labels on the polygon corresponding to a coloured multiple zeta value are, because of Equation 5.1, found from successive products of the arguments. As discussed in Remark 5.3, the convergence of a coloured multiple zeta value,

$$\zeta(1, \ldots, 1; \varepsilon_1, \ldots, \varepsilon_k)$$

requires $\varepsilon_k = -1$. The symbol of this will be $(-1)^k$ (from Equation 5.1) times the symbol attached to the polygon

$$P(\hat{\varepsilon}_1, \ldots, \hat{\varepsilon}_{k-1}, -1, 1) \quad \text{where} \quad \hat{\varepsilon}_i = \prod_{j=i}^{k} \varepsilon_j.$$

If we now apply the result of Proposition 5.4 (noting that, for this polygon, $a = 0$, and that therefore $\lambda_{1,n} = 1$), the symbol attached to the above polygon is simply $2^{\otimes n}$. Therefore,

$$\mathcal{S}\left(\zeta(\underbrace{1,\ldots,1}_{k \text{ times}};\varepsilon_1,\ldots,\varepsilon_{k-1},-1)\right) = (-1)^k 2^{\otimes k}.$$

Since Proposition 5.7 can be applied directly: this completes the proof of Theorem 5.2.

Chapter 6

Relations on harmonic polylogarithms up to weight 8

In this chapter we examine a specific class of polylogarithm, the so-called harmonic polylogarithms [RV00], and find the first non-trivial (to the author's knowledge) linear combinations of them in the kernel of the symbol in weight 8.

Harmonic polylogarithms have been shown (e.g. in [RV00]) to play a part in evaluating Feynman integrals (something which does not appear within the scope of this thesis). Therefore, finding relations between them is of great relevance in the physics community.

We define harmonic polylogarithms in terms of Goncharov polylogarithms (from Definition 0.9).

Definition 6.1. A weight w harmonic polylogarithm $H(\mathbf{a}, x)$ is defined, for a vector

$$\mathbf{a} = (a_1, \ldots, a_w) \quad \text{with } a_i \in \{-1, 0, 1\} \quad \forall i,$$

to be

$$H(\mathbf{a}, x) := (-1)^k G(a_1, \ldots, a_w; x),$$

where

$$k = \# \{i \mid a_i = 1\}.$$

6.1 The symbol of a harmonic polylogarithm

The symbol of a harmonic polylogarithm has been well studied (see [DGR11]) and is easy to find. We will recall it briefly here. Firstly we note the following relationship

$$H(\mathbf{a}, x) \equiv (-1)^k I_{1,\dots,1}\left(\frac{a_w}{x}, \dots, \frac{a_1}{x}\right)$$

where again $k = \#\{i \mid a_i = 1\}$. Therefore $(-1)^k H(\mathbf{a}, x)$ relates to a hook-arrow tree, after scaling by x, with vertices labelled

$$(a_w, \dots, a_1, x),$$

Every edge of any possible hook-arrow trees on these vertices will therefore take one of two forms. An edge of the form

$$\begin{tikzcd} x \\ a_i \arrow[u] \end{tikzcd} \qquad \text{relating to, in the symbol,} \begin{cases} 1 - x & \text{for } a_i = 1 \\ 1 + x & \text{for } a_i = -1 \\ x & \text{for } a_i = 0, \end{cases}$$

or an edge of the form

$$\begin{tikzcd} a_j \\ a_i \arrow[u] \end{tikzcd} \quad \text{relating to, in the symbol,} \begin{cases} \text{a term of coefficient zero} & \text{for } a_i = a_j \\ \text{a term of coefficient zero} & \text{for } a_i = 0 \text{ or } a_j = 0 \\ 2 & \text{for } a_i = -a_j \neq 0. \end{cases}$$

We can therefore see that every term in the symbol of a harmonic polylogarithm will be formed from the tensor products of $x, 1 \pm x$ and 2.

We will now specialise to harmonic polylogarithms $H(\mathbf{a}, x)$ where the vector $\mathbf{a}$ has $a_i \in \{0, 1\}$ for all i. The symbol of this class of harmonic polylogarithms is, in fact, very simple. The Proposition 6.2 appears in Example 6.1 of [DGR11] and is also well known.

Proposition 6.2. *Let $H(\mathbf{a}, x)$ be a harmonic polylogarithm where the vector $\mathbf{a}$ has $a_i \in \{0, 1\}$ for all i, then*

$$\mathcal{S}(H(\mathbf{a}, x)) = (-1)^k \left((a_w - x) \otimes \cdots \otimes (a_1 - x)\right)$$

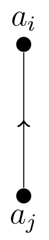
where

$$k = \# \{i \mid a_i = 1\}.$$

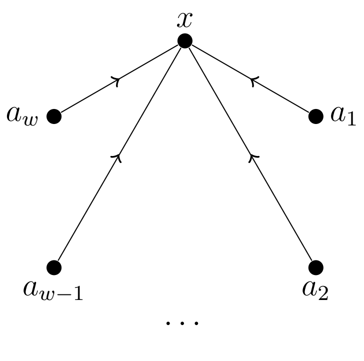
Proof. $(-1)^k H(\mathbf{a}, x)$ will relate to a hook-arrow tree with vertices labelled

$$(a_w, \ldots, a_1, x).$$

We now observe that since $a_i \in \{0, 1\}$ for all i, every hook arrow tree containing an edge of the form



will have a coefficient of zero in the symbol, the only possible hook-arrow tree with a non-zero coefficient will therefore be



which has symbol, up to torsion, of

$$(a_w - x) \otimes \cdots \otimes (a_1 - x).$$

Reintroducing the factor of $(-1)^k$ we have the required result. □

6.2 Finding a relation between harmonic polylogarithms

Because of the simple nature of the symbol of a harmonic polylogarithm it is easier to find terms in the kernel of the symbol. A numerical evaluation is kindly provided by C Duhr.

Proposition 6.3. *The following linear combination of harmonic polylogarithms, which we will call Υ_5, lies in the kernel of the symbol map:*

$$\begin{aligned}
&+H(0,0,0,1,1;x)-H(0,0,0,1,1;1-x)-H\left(0,0,0,1,1;\tfrac{1}{1-x}\right)+H\left(0,0,0,1,1;\tfrac{x}{x-1}\right)\\
&+H(0,0,1,0,1;x)-H(0,0,1,0,1;1-x)-H\left(0,0,1,0,1;\tfrac{1}{1-x}\right)+H\left(0,0,1,0,1;\tfrac{x}{x-1}\right)\\
&+H(0,1,0,0,1;x)-H(0,1,0,0,1;1-x)-H\left(0,1,0,0,1;\tfrac{1}{1-x}\right)+H\left(0,1,0,0,1;\tfrac{x}{x-1}\right)\\
&+H(1,0,0,0,1;x)-H(1,0,0,0,1;1-x)-H\left(1,0,0,0,1;\tfrac{1}{1-x}\right)+H\left(1,0,0,0,1;\tfrac{x}{x-1}\right)\\
&+2\Bigg(H(0,0,0,0,1;x)-H(0,0,0,0,1;1-x)-H\left(0,0,0,0,1;\tfrac{1}{1-x}\right)+H\left(0,0,0,0,1;\tfrac{x}{x-1}\right)\Bigg)
\end{aligned}$$

and in fact, it can be numerically shown, with a certain choice of branch cut, that

$$\begin{aligned}
720\Upsilon_5 =& -(2\pi i)^4\ln(x(1-x))-15(2\pi i)^3(\ln(x(1-x)))^2\\
&-60(2\pi i)^2\big(\ln(x(1-x))\big)^3-60(2\pi i)\big(\ln(x(1-x))\big)^4\\
&+120(2\pi i)(\ln(1-x))^3\ln(x)+120(2\pi i)\ln(1-x)(\ln(x))^3\\
&+60(2\pi i)^2(\ln(1-x))^2\ln(x)+60(2\pi i)^2\ln(1-x)(\ln(x))^2\\
&-240(2\pi i)^2\zeta(3)-720(2\pi i)\ln(1-x)\zeta(3)-720(2\pi i)\ln(x)\zeta(3)\\
&+(2\pi i)^5.
\end{aligned}$$

The symbol calculation was originally done (when finding the relation), using a GP/Pari script and the method described in Appendix A. However, we can give a full proof that the above combination of harmonic polylogarithms have a zero symbol fairly elegantly.

Notation 6.4. Since many of the terms above are very similar, and 1s and 0s can be seen to be shuffled in the arguments of the harmonic polylogarithms, we introduce notation to shorten the relation. Firstly, let

$$\mathcal{V}^{(a,b)} := \left\{ (v_1, \ldots, v_{a+b}) \,\middle|\, v_i \in \{0,1\},\ v_{a+b} = 1,\ \sum_i^{a+b} v_i = a \right\},$$

in other words, the set of vectors of length $a+b$, with entries all either 1 or 0, ending in 1 with a values equalling 1 and b values equalling 0. Note that this can be viewed, albeit with a small abuse of notation, as all vectors arising from shuffling

$$(\underbrace{\{1, \ldots, 1\}}_{a-1} ш \underbrace{\{0, \ldots, 0\}}_{b}, 1).$$

We now let

$$\mathcal{H}^{(a,b)}(x) := \sum_{\mathbf{v} \in \mathcal{V}^{(a,b)}} H(\mathbf{v}; x).$$

So, for example,

$$\mathcal{H}^{(2,3)}(x) = H(1,0,0,0,1;x) + H(0,1,0,0,1;x) + H(0,0,1,0,1;x) + H(0,0,0,1,1;x).$$

Applying this notation to our relation we can now write

$$\begin{aligned}\Upsilon_5 =& \mathcal{H}^{(2,3)}(x) - \mathcal{H}^{(2,3)}(1-x) - \mathcal{H}^{(2,3)}(\tfrac{1}{1-x}) + \mathcal{H}^{(2,3)}(\tfrac{x}{x-1}) \\ &+ 2\Big(\mathcal{H}^{(1,4)}(x) - \mathcal{H}^{(1,4)}(1-x) - \mathcal{H}^{(1,4)}(\tfrac{1}{1-x}) + \mathcal{H}^{(1,4)}(\tfrac{x}{x-1}) \Big).\end{aligned}$$

Proof. As seen above we have that $\mathcal{S}(H(\mathbf{a}, x)) = (-1)^k \left((a_w - x) \otimes \cdots \otimes (a_1 - x) \right)$. We note that every part of every tensor arising from the harmonic polylogarithms in our relation will be of the form

$$x^{b_0}(1-x)^{c_0} \otimes x^{b_1}(1-x)^{c_1} \otimes x^{b_2}(1-x)^{c_2} \otimes x^{b_3}(1-x)^{c_3} \otimes x^{b_4}(1-x)^{c_4}$$

for some b_i and c_i. This can then be expanded using tensor calculus into a linear combination of tensor products of the form

$$f_0 \otimes f_1 \otimes f_2 \otimes f_3 \otimes f_4$$

with each $f_i \equiv x$ or $1-x$. We now index the 32 tensor products using binary notation as follows. First let

$$b(f_i(x)) := \begin{cases} 0 & \text{if} \quad f_i \equiv x \\ 1 & \text{if} \quad f_i \equiv 1-x \end{cases}$$

then define a map

$$B\big(f_0 \otimes \cdots \otimes f_n\big) := \left[\sum_{i=0}^{n} 2^i b(f_i)\right].$$

We have, in effect, attached a unique integer to each tensor product formed from only $1-x$ and x via a binary number. For example

$$B\big((1-x)\otimes x \otimes x \otimes (1-x) \otimes x\big) = [9],$$
$$B\big(x \otimes (1-x) \otimes x \otimes (1-x) \otimes (1-x)\big) = [26].$$

We formally add linear combinations of $[n]$ in exactly the same way as we would the tensor products they represent (and it gives us a more compact way to display the lengthy tensor calculation that follows). We also note that this method is very similar to the more general approach described in Appendix A.

We therefore have

$$\mathcal{S}(\Upsilon_5) = P_1^5 - P_2^5 - P_3^5 + P_4^5 + 2\big(Q_1^5 - Q_2^5 - Q_3^5 + Q_4^5\big)$$

where

$$\begin{aligned}
P_1^5 &= S\big(\mathcal{H}^{(2,3)}(x)\big) && = (1-x)\otimes(1-x) \sqcup\!\sqcup \big(x\otimes x\otimes x\big)\\
P_2^5 &= S\big(\mathcal{H}^{(2,3)}(1-x)\big) && = x\otimes x \sqcup\!\sqcup \big((1-x)\otimes(1-x)\otimes(1-x)\big)\\
P_3^5 &= S\big(\mathcal{H}^{(2,3)}(\tfrac{1}{1-x})\big) && = -\big(\tfrac{x}{x-1}\otimes\tfrac{x}{x-1} \sqcup\!\sqcup \big((1-x)\otimes(1-x)\otimes(1-x)\big)\big)\\
P_4^5 &= S\big(\mathcal{H}^{(2,3)}(\tfrac{x}{x-1})\big) && = (1-x)\otimes(1-x) \sqcup\!\sqcup \big(\tfrac{x}{x-1}\otimes\tfrac{x}{x-1}\otimes\tfrac{x}{x-1}\big)\\
Q_1^5 &= S\big(\mathcal{H}^{(1,4)}(x)\big) && = -\big((1-x)\otimes x\otimes x\otimes x\otimes x\big)\\
Q_2^5 &= S\big(\mathcal{H}^{(1,4)}(1-x)\big) && = -\big(x\otimes(1-x)\otimes(1-x)\otimes(1-x)\otimes(1-x)\big)\\
Q_3^5 &= S\big(\mathcal{H}^{(1,4)}(\tfrac{1}{1-x})\big) && = -\big(\tfrac{x}{x-1}\otimes(1-x)\otimes(1-x)\otimes(1-x)\otimes(1-x)\big)\\
Q_4^5 &= S\big(\mathcal{H}^{(1,4)}(\tfrac{x}{x-1})\big) && = (1-x)\otimes\tfrac{x}{x-1}\otimes\tfrac{x}{x-1}\otimes\tfrac{x}{x-1}\otimes\tfrac{x}{x-1}.
\end{aligned}$$

We now apply the map B. We first give $B(P_1^5)$ explicitly.

$$\begin{aligned}
B(P_1^5) &= B\big((1-x)\otimes(1-x)\otimes x\otimes x\otimes x\big) + B\big((1-x)\otimes x\otimes(1-x)\otimes x\otimes x\big)\\
&= B\big((1-x)\otimes x\otimes x\otimes(1-x)\otimes x\big) + B\big((1-x)\otimes x\otimes x\otimes x\otimes(1-x)\big)\\
&= [3]+[5]+[9]+[17]
\end{aligned}$$

and then give the remaining terms in table form, where each column gives how many of each $[n]$ in the term contains.

$[n]$	$B(P_1^5)$	$B(P_2^5)$	$B(P_3^5)$	$B(P_4^5)$	$B(Q_1^5)$	$B(Q_2^5)$	$B(Q_3^5)$	$B(Q_4^5)$
0								
1					-1			1
2								
3	1			1				-1
4								
5	1			1				-1
6								
7				-2				1
8								
9	1			1				-1
10								
11				-2				1
12								
13				-2				1
14		1	-1					
15			1	3				-1
16								
17	1			1				-1
18								
19				-2				1
20								
21				-2				1
22		1	-1					
23			1	3				-1
24								
25				-2				1
26		1	-1					
27			1	3				-1
28		1	-1					
29			1	3				-1
30			4			-1	-1	
31			-4	-4			1	1

By looking across each row we can see that

$$B\Big(s_1 - s_2 - s_3 + s_4 + 2\big(t_1 - t_2 - t_3 + t_4\big)\Big) = 0$$

and so we have shown

$$\mathcal{S}(\Upsilon_5) = 0.$$

The numerical evaluation shown was calculated by C Duhr. □

The weight 5 harmonic polylogarithm relation in the previous section has a noticeable structure, in that it features harmonic polylogarithms of the form

$$\mathcal{H}^{2,3}(f(x)) = H(\{1\} \sqcup \{0,0,0\}, 1; f(x)) \quad \text{and} \quad \mathcal{H}^{1,4}(f(x)) = H(0,0,0,0,1; f(x))$$

for $f(x) \in \{x, 1-x, \frac{1}{1-x}, \frac{x}{x-1}\}$. For other weights, w, we are motivated to examine terms of the form

$$\mathcal{H}^{a,w-a}(f(x)) \quad \text{for} \quad f(x) \in \{x, 1-x, \tfrac{1}{1-x}, \tfrac{x}{x-1}\} \quad \text{and} \quad 1 \le a \le \frac{w}{2}.$$

We begin by examining weight 4, and so consider

$$\begin{aligned}
P_1^4 = &\; \mathcal{S}\big(\mathcal{H}^{(2,2)}(x)\big), & P_2^4 = &\; \mathcal{S}\big(\mathcal{H}^{(2,2)}(1-x)\big),\\
P_3^4 = &\; \mathcal{S}\big(\mathcal{H}^{(2,2)}(\tfrac{1}{1-x})\big), & P_4^4 = &\; \mathcal{S}\big(\mathcal{H}^{(2,2)}(\tfrac{x}{x-1})\big),\\
Q_1^4 = &\; \mathcal{S}\big(\mathcal{H}^{(1,3)}(x)\big), & Q_2^4 = &\; \mathcal{S}\big(\mathcal{H}^{(1,3)}(1-x)\big),\\
Q_3^4 = &\; \mathcal{S}\big(\mathcal{H}^{(1,3)}(\tfrac{1}{1-x})\big), & Q_4^4 = &\; \mathcal{S}\big(\mathcal{H}^{(1,3)}(\tfrac{x}{x-1})\big)
\end{aligned}$$

and, after calculations similar to the previous section we obtain the following table.

$[n]$	$B(P_1^4)$	$B(P_2^4)$	$B(P_3^4)$	$B(P_4^4)$	$B(Q_1^4)$	$B(Q_2^4)$	$B(Q_3^4)$	$B(Q_4^4)$
0								
1					-1			1
2								
3	1			1				-1
4								
5	1			1				-1
6		1	1					
7			-1	-2				1
8								
9	1			1				-1
10		1	1					
11			-1	-2				1
12		1	1					
13			-1	-2				1
14			-3			-1	1	
15			3	3			-1	-1

We see that, for weight 4 we can form a linear combination of harmonic polylogarithms in the kernel of the symbol map without using P_2^4 or P_3^4. We have

$$\begin{aligned}\Upsilon_4 =& + \mathcal{H}^{(2,2)}(x) + \mathcal{H}^{(2,2)}(\tfrac{x}{x-1})\\ &+ 2\mathcal{H}^{(1,3)}(x) + \mathcal{H}^{(1,3)}(1-x) + \mathcal{H}^{(1,3)}(\tfrac{1}{1-x}) + 2\mathcal{H}^{(1,3)}(\tfrac{x}{x-1})\\ =& + H(0,0,1,1;x) + H(0,1,0,1;x) + H(1,0,0,1;x)\\ &+ H\left(0,0,1,1;\tfrac{x}{x-1}\right) + H\left(0,1,0,1;\tfrac{x}{x-1}\right) + H\left(1,0,0,1;\tfrac{x}{x-1}\right)\\ &+ 2H(0,0,0,1;x) + H(0,0,0,1;1-x) + H\left(0,0,0,1;\tfrac{1}{1-x}\right)\\ &+ 2H\left(0,0,0,1;\tfrac{x}{x-1}\right),\end{aligned}$$

with $\mathcal{S}(\Upsilon_4) = 0$.

6.3 Extending to other weights

By similar methodology, we can find analogous linear combinations of harmonic polylogarithms in the kernel of the symbol map for weights 6, 7 and 8, and possibly further. We give these, and repeat the linear combinations for weights 4 and 5 for comparison.

Theorem 6.5. *Let $\Upsilon_4, \Upsilon_5, \Upsilon_6, \Upsilon_7$ and Υ_8 be the following linear combinations of harmonic polylogarithms.*

$$\begin{array}{rllll}
\Upsilon_4 = & +\ \mathcal{H}^{(2,2)}(x) & & & +\ \mathcal{H}^{(2,2)}(\frac{x}{x-1}) \\
& +2\mathcal{H}^{(1,3)}(x) & +\ \mathcal{H}^{(1,3)}(1-x) & +\ \mathcal{H}^{(1,3)}(\frac{1}{1-x}) & +2\mathcal{H}^{(1,3)}(\frac{x}{x-1}) \\[1em]
\Upsilon_5 = & +\ \mathcal{H}^{(2,3)}(x) & -\ \mathcal{H}^{(2,3)}(1-x) & -\ \mathcal{H}^{(2,3)}(\frac{1}{1-x}) & +\ \mathcal{H}^{(2,3)}(\frac{x}{x-1}) \\
& +2\mathcal{H}^{(1,4)}(x) & -2\mathcal{H}^{(1,4)}(1-x) & -2\mathcal{H}^{(1,4)}(\frac{1}{1-x}) & +2\mathcal{H}^{(1,4)}(\frac{x}{x-1}) \\[1em]
\Upsilon_6 = & +\ \mathcal{H}^{(3,3)}(x) & -2\mathcal{H}^{(3,3)}(1-x) & -2\mathcal{H}^{(3,3)}(\frac{1}{1-x}) & +\ \mathcal{H}^{(3,3)}(\frac{x}{x-1}) \\
& +\ \mathcal{H}^{(2,4)}(x) & -4\mathcal{H}^{(2,4)}(1-x) & -4\mathcal{H}^{(2,4)}(\frac{1}{1-x}) & +\ \mathcal{H}^{(2,4)}(\frac{x}{x-1}) \\
& +2\mathcal{H}^{(1,5)}(x) & -7\mathcal{H}^{(1,5)}(1-x) & -7\mathcal{H}^{(1,5)}(\frac{1}{1-x}) & +2\mathcal{H}^{(1,5)}(\frac{x}{x-1}) \\[1em]
\Upsilon_7 = & +\mathcal{H}^{(3,4)}(x) & -2\mathcal{H}^{(3,4)}(1-x) & +2\mathcal{H}^{(3,4)}(\frac{1}{1-x}) & -\mathcal{H}^{(3,4)}(\frac{x}{x-1}) \\
& +\mathcal{H}^{(2,5)}(x) & -5\mathcal{H}^{(2,5)}(1-x) & +5\mathcal{H}^{(2,5)}(\frac{1}{1-x}) & -\mathcal{H}^{(2,5)}(\frac{x}{x-1}) \\
& & -9\mathcal{H}^{(1,6)}(1-x) & +9\mathcal{H}^{(1,6)}(\frac{1}{1-x}) & \\[1em]
\Upsilon_8 = & +\ \mathcal{H}^{(4,4)}(x) & & & -\ \mathcal{H}^{(4,4)}(\frac{x}{x-1}) \\
& +2\mathcal{H}^{(3,5)}(x) & & & -2\mathcal{H}^{(3,5)}(\frac{x}{x-1}) \\
& +2\mathcal{H}^{(2,6)}(x) & -2\mathcal{H}^{(2,6)}(1-x) & +2\mathcal{H}^{(2,6)}(\frac{1}{1-x}) & -2\mathcal{H}^{(2,6)}(\frac{x}{x-1}) \\
& & -7\mathcal{H}^{(1,7)}(1-x) & +7\mathcal{H}^{(1,7)}(\frac{1}{1-x}) &
\end{array}$$

then $\mathcal{S}(\Upsilon_i) = 0$ for $i = 4, 5, 6, 7, 8$.

Remark 6.6. The above linear combinations can be written even more concisely if you combine $\mathcal{H}^{(i,j)}(x)$ and $\mathcal{H}^{(i,j)}(\frac{x}{x-1})$ and also combine $\mathcal{H}^{(i,j)}(1-x)$ and $\mathcal{H}^{(i,j)}(\frac{1}{1-x})$. We would add these in the first three cases and subtract one from the other in the other two.

Proof. The proof that the combinations of weights 4 and 5 are provided above. The higher weights can be proven in a similar way (which we do not provide here for brevity). □

Chapter 7

Linear combinations of multiple polylogarithms with a zero symbol and conclusion

In Chapter 6 we found linear combinations of harmonic polylogarithms, we now find linear combinations of general multiple polylogarithms (whose symbols can have many more terms) that have a symbol of zero. The combinations we find will be of multiple polylogarithms in 2 variables, of depth ≤ 2 and in weights 4, 5 and 6.

There is a scarcity of functional equations for polylogarithms of higher weights in the literature, even for classical polylogarithms, $\mathrm{Li}_m(x)$. The author is not aware of any prior combinations of *multiple* polylogarithms in higher weights, apart from shuffle/stuffle relations and from a weight 4 functional equation in [Dan11].

We can use the algorithm outlined in Chapter 2 to encode hook-arrow trees into GP/Pari. We do not explicitly provide the code in this thesis, but it closely follows the algorithm. We can therefore calculate the symbol for multiple polylogarithms that would otherwise be too lengthy to do by hand. We are, of course, still constrained by computing power. Appendix A outlines the method of finding the linear combinations using GP/Pari (as well as giving further combinations).

7.1 Elements of $\mathcal{I}_w$ in $\ker \mathcal{S}$ for $w = 4, 5$ and 6

Theorem 7.1. *The following linear combinations of multiple polylogarithms, $\Psi_4 \in \mathcal{I}_4, \Psi_5, \Phi_5 \in \mathcal{I}_5$ and $\Psi_6, \Phi_6 \in \mathcal{I}_6$ have a symbol of zero.*

Weight 4

$$\begin{aligned}\Psi_4(x,y) =& 2\begin{pmatrix} I_{2,2}\left(\frac{1}{xy},\frac{1}{x}\right)+I_{2,2}\left(\frac{1}{xy},\frac{1}{y}\right)+I_{2,2}\left(\frac{y}{x},\frac{1}{x}\right)+I_{2,2}\left(\frac{y}{x},y\right) \\ +I_{2,2}\left(xy,x\right)+I_{2,2}\left(xy,y\right)+I_{2,2}\left(\frac{x}{y},x\right)+I_{2,2}\left(\frac{x}{y},\frac{1}{y}\right)\end{pmatrix}\\ &+2\left(I_4\left(\frac{1}{x}\right)+I_4\left(\frac{1}{y}\right)+I_4\left(x\right)+I_4\left(y\right)\right)\\ &-3\left(I_4\left(\frac{1}{xy}\right)+I_4\left(\frac{y}{x}\right)+I_4\left(\frac{x}{y}\right)+I_4\left(xy\right)\right)\end{aligned}$$

Weight 5

$$\begin{aligned}\Psi_5(x,y) =& -I_{2,3}\left(\frac{1}{x},\frac{y}{x}\right)-I_{2,3}\left(\frac{1}{x},y\right)-I_{2,3}\left(\frac{1}{y},\frac{1}{x}\right)-I_{2,3}\left(\frac{1}{y},\frac{x}{y}\right)-I_{2,3}\left(\frac{1}{y},x\right)\\ &+I_{2,3}\left(x,\frac{x}{y}\right)+I_{2,3}\left(x,\frac{1}{y}\right)+I_{2,3}\left(y,x\right)+I_{2,3}\left(y,\frac{y}{x}\right)+I_{2,3}\left(y,\frac{1}{x}\right)\\ &-I_{3,2}\left(\frac{1}{x},\frac{y}{x}\right)-I_{3,2}\left(\frac{1}{x},y\right)-I_{3,2}\left(\frac{1}{y},\frac{1}{x}\right)-I_{3,2}\left(\frac{1}{y},\frac{x}{y}\right)-I_{3,2}\left(\frac{1}{y},x\right)\\ &+I_{3,2}\left(x,\frac{x}{y}\right)+I_{3,2}\left(x,\frac{1}{y}\right)+I_{3,2}\left(y,x\right)+I_{3,2}\left(y,\frac{y}{x}\right)+I_{3,2}\left(y,\frac{1}{x}\right)\\ &-2I_{2,3}\left(\frac{1}{xy},\frac{1}{x}\right)-I_{2,3}\left(\frac{1}{xy},\frac{1}{y}\right)+I_{2,3}\left(xy,y\right)+2I_{2,3}\left(xy,x\right)\\ &-I_{3,2}\left(\frac{1}{xy},\frac{1}{x}\right)-2I_{3,2}\left(\frac{1}{xy},\frac{1}{y}\right)+2I_{3,2}\left(xy,y\right)+I_{3,2}\left(xy,x\right)\\ &+I_{3,2}\left(\frac{y}{x},\frac{1}{x}\right)-I_{3,2}\left(\frac{y}{x},y\right)+I_{3,2}\left(\frac{x}{y},\frac{1}{y}\right)-I_{3,2}\left(\frac{x}{y},x\right)\\ &+4I_5\left(\frac{1}{xy}\right)-4I_5\left(xy\right)\end{aligned}$$

$$\begin{aligned}\Phi_5(x,y) =& +I_{2,3}(x,y)-I_{2,3}(y,x)+I_{2,3}(xy,y)-I_{2,3}(xy,x)\\ &+I_{3,2}(x,y)-I_{3,2}(y,x)-I_{3,2}(xy,y)+I_{3,2}(xy,x)\end{aligned}$$

Weight 6

$$\begin{aligned}\Psi_6(x,y) = &+ I_{2,4}(xy,y) + 2I_{2,4}(xy,x) - I_{2,4}(x,y) + I_{2,4}(y,x)\\ &- 2I_{3,3}(xy,y) - 2I_{3,3}(xy,x) - 2I_{3,3}(x,y)\\ &+ 2I_{4,2}(xy,y) + I_{4,2}(xy,x) - 2I_{4,2}(x,y) - I_{4,2}(y,x)\\ &- I_6(xy)\end{aligned}$$

$$\begin{aligned}\Phi_6(x,y) = &+ I_{2,4}(xy,y) - I_{2,4}(xy,x) + 2I_{2,4}(x,y) - 2I_{2,4}(y,x)\\ &\qquad\qquad\qquad\qquad + 2I_{3,3}(x,y) - 2I_{3,3}(y,x)\\ &- I_{4,2}(xy,y) + I_{4,2}(xy,x) + I_{4,2}(x,y) - I_{4,2}(y,x)\end{aligned}$$

Proof. The relations were found using GP/Pari with the procedure outlined in Appendix A. □

7.2 Remarks on the Theorem

Firstly we note that these linear combinations are given only on multiple polylogarithms $I_{r_1,\dots,r_s}(x_1,\dots,x_s)$ where $r_i > 1$ for all i. In light of Conjecture 4.1 we are more interested in terms in the kernel of the symbol of this form.

7.2.1 Examining $\Psi_4(x,y)$

We note that $\Psi_4(x,y)$ is symmetric under inverting each argument independently,

$$\begin{aligned}\Psi_4(x,y) = \Psi_4(y,x) = \Psi_4\left(\frac{1}{x},y\right) = \Psi_4\left(x,\frac{1}{y}\right) = \Psi_4\left(\frac{1}{x},\frac{1}{y}\right)\\ = \Psi_4\left(y,\frac{1}{x}\right) = \Psi_4\left(\frac{1}{y},x\right) = \Psi_4\left(\frac{1}{y},\frac{1}{x}\right).\end{aligned}$$

We now recall the stuffle relation from Section 0.2. The iterated integrals form of a two variable stuffle relation is, for $r,s \in \mathbb{Z}$,

$$I_r(x)I_s(y) = I_{r,s}(xy,y) - I_{r+s}(xy) + I_{s,r}(xy,x).$$

By applying this relation to $\Psi_4(x,y)$, we see that

$$\begin{aligned}\Psi_4(x,y) =& 2\left(I_2\left(\frac{1}{xy}\right)I_2\left(\frac{1}{x}\right)+I_2\left(\frac{y}{x}\right)I_2\left(\frac{1}{x}\right)+I_2(xy)\,I_2(x)+I_2\left(\frac{x}{y}\right)I_2(x)\right)\\ &+2\left(I_4\left(\frac{1}{x}\right)+I_4\left(\frac{1}{y}\right)+I_4(x)+I_4(y)\right)\\ &-\left(I_4\left(\frac{1}{xy}\right)+I_4\left(\frac{y}{x}\right)+I_4\left(\frac{x}{y}\right)+I_4(xy)\right).\end{aligned}$$

Notation 7.2. We introduce notation for the pure weight parts of the 2-variable stuffle relation above. We let

$$\mathrm{Stu}_{r,s}(x,y) := I_{r,s}(xy,y) - I_{r+s}(xy) + I_{s,r}(xy,x).$$

Next, recall the well-known inversion relation on multiple polylogarithms, stating that

$$I_r(x) + (-1)^r I_r\left(\frac{1}{x}\right)$$

can be written in terms of 'lower weight objects' (see, among others, [Zag90], for more details). We therefore use the notation

$$\mathrm{Inv}_r(x) = I_r(x) + (-1)^r I_r\left(\frac{1}{x}\right)$$

We can now write

$$\begin{aligned}\Psi_4(x,y) =& +2\mathrm{Stu}_{2,2}(x,y)+2\mathrm{Stu}_{2,2}\left(x,\frac{1}{y}\right)+2\mathrm{Stu}_{2,2}\left(\frac{1}{x},y\right)+2\mathrm{Stu}_{2,2}\left(\frac{1}{x},\frac{1}{y}\right)\\ &+2\mathrm{Inv}_4(x)+2\mathrm{Inv}_4(y)-\mathrm{Inv}_4(xy)-\mathrm{Inv}_4\left(\frac{x}{y}\right).\end{aligned}$$

We have shown that $\Psi_4(x,y)$ is merely a combination of other functional equations, however there is still worth to the combination. $\mathrm{Stu}_{2,2}(x,y)$ and $\mathrm{Inv}_4(x)$ do not in fact lie in the kernel of the symbol, whereas $\Psi_4(x,y)$ does. $\mathrm{Stu}_{2,2}(x,y)$ and $\mathrm{Inv}_4(x)$ lie in the kernel of a 'restricted' symbol map which allows for anticommutativity, i.e., instead of taking tensor products in the definition of the symbol, we use the wedge product introduced in Definition 1.9.

Example 7.3. Since the symbol for $\mathrm{Stu}_{2,2}(x,y)$ is fairly long, we use $\mathrm{Stu}_{1,1}(x,y)$ as an explanatory example. We see that

$$\begin{aligned}\mathcal{S}(\mathrm{Stu}_{1,1}(x,y)) =& + \left(1-\tfrac{1}{y}\right)\otimes\left(1-\tfrac{1}{x}\right) + \left(1-\tfrac{1}{xy}\right)\otimes\left(1-\tfrac{1}{y}\right) - \left(1-\tfrac{1}{xy}\right)\otimes(1-x)\\ &+ \left(1-\tfrac{1}{xy}\right)\otimes xy\\ &+ \left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{y}\right) + \left(1-\tfrac{1}{xy}\right)\otimes\left(1-\tfrac{1}{x}\right) - \left(1-\tfrac{1}{xy}\right)\otimes(1-y)\\ =& + \left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{y}\right) + \left(1-\tfrac{1}{y}\right)\otimes\left(1-\tfrac{1}{x}\right)\\ \neq& 0.\end{aligned}$$

However, if we look at its image when passing to the wedge product, the above calculation still holds (since all properties of a tensor product hold for wedge products) but then,

$$\left(1-\tfrac{1}{x}\right)\wedge\left(1-\tfrac{1}{y}\right) + \left(1-\tfrac{1}{y}\right)\wedge\left(1-\tfrac{1}{x}\right) = \left(1-\tfrac{1}{x}\right)\wedge\left(1-\tfrac{1}{y}\right) - \left(1-\tfrac{1}{x}\right)\wedge\left(1-\tfrac{1}{y}\right) = 0.$$

Similar (though more lengthy) calculations hold for general $\mathrm{Stu}_{r,s}(x,y)$.

$$\mathcal{S}(\mathrm{Inv}_r(x)) = \left(1-\tfrac{1}{x}\right)\otimes x^{\otimes(r-1)} + (-1)^r(1-x)\otimes\left(\frac{1}{x}\right)^{\otimes(r-1)} = x^{\otimes r}\neq 0$$

but again with wedge products, $x\wedge\cdots\wedge x = -(x\wedge\cdots\wedge x) = 0$.

For $\Psi_4(x,y)$ we have that

$$\begin{aligned}&\mathcal{S}\left(2\mathrm{Stu}_{2,2}(x,y) + 2\mathrm{Stu}_{2,2}\left(x,\frac{1}{y}\right) + 2\mathrm{Stu}_{2,2}\left(\frac{1}{x},y\right) + 2\mathrm{Stu}_{2,2}\left(\frac{1}{x},\frac{1}{y}\right)\right)\\ &= \mathcal{S}\left(-2\mathrm{Inv}_4(x) - 2\mathrm{Inv}_4(y) + \mathrm{Inv}_4(xy) + \mathrm{Inv}_4\left(\frac{x}{y}\right)\right)\\ &= 2(x)^{\otimes 4} + 2(y)^{\otimes 4} + 2(xy)^{\otimes 4} + 2\left(\frac{x}{y}\right)^{\otimes 4}.\end{aligned}$$

7.2.2 Examining $\Psi_5(x,y)$ and $\Phi_5(x,y)$

Unlike for $\Psi_4(x,y)$, we are not (as far as the author can see) able to reduce $\Psi_5(x,y)$ to a combination of stuffle relations. However, it does relate strongly to $\Phi_5(x,y)$.

By symmetrising $\Psi_5(x,y)$ and summing we see that

$$\begin{aligned}\Psi_5(x,y)-\Psi_5(y,x)=&-I_{2,3}\left(\frac{1}{y},\frac{1}{x}\right)+I_{2,3}\left(\frac{1}{x},\frac{1}{y}\right)+I_{2,3}\left(y,x\right)-I_{2,3}\left(x,y\right)\\&-I_{3,2}\left(\frac{1}{y},\frac{1}{x}\right)+I_{3,2}\left(\frac{1}{x},\frac{1}{y}\right)+I_{3,2}\left(y,x\right)-I_{3,2}\left(x,y\right)\\&-I_{2,3}\left(\frac{1}{xy},\frac{1}{x}\right)+I_{2,3}\left(\frac{1}{xy},\frac{1}{y}\right)-I_{2,3}\left(xy,y\right)+I_{2,3}\left(xy,x\right)\\&+I_{3,2}\left(\frac{1}{xy},\frac{1}{x}\right)-I_{3,2}\left(\frac{1}{xy},\frac{1}{y}\right)+I_{3,2}\left(xy,y\right)-I_{3,2}\left(xy,x\right)\\=&-\Phi_5(x,y)-\Phi_5\left(\frac{1}{x},\frac{1}{y}\right).\end{aligned}$$

We now look simply at $\Phi_5(x,y)$, which has similarities to stuffle relations $\mathrm{Stu}_{2,3}(x,y)$ and $\mathrm{Stu}_{2,3}(y,x)$ and we see that

$$\begin{aligned}\Phi_5(x,y)=&+I_{2,3}(x,y)-I_{2,3}(y,x)+I_{3,2}(x,y)-I_{3,2}(y,x)\\&+\mathrm{Stu}_{2,3}(y,x)-\mathrm{Stu}_{2,3}(x,y).\end{aligned}$$

but also that $\Phi_5(x,y)$ cannot be reduced completely into stuffle relations.

In summary, we have found distinct $\Psi_5,\Phi_5\in\mathcal{I}_5(S)$ that are individually in the kernel of the symbol and are related by

$$\Psi_5(x,y)-\Psi_5(y,x)=-\Phi_5(x,y)-\Phi_5\left(\frac{1}{x},\frac{1}{y}\right).$$

7.2.3 Examining $\Psi_6(x,y)$ and $\Phi_6(x,y)$

We can reduce the expression $\Psi_6(x,y)$ by stuffle relations to

$$\begin{aligned}\Psi_6(x,y)=&\mathrm{Stu}_{2,4}(x,y)+2\mathrm{Stu}_{2,4}(y,x)-2\mathrm{Stu}_{3,3}(x,y)-2I_6(xy)\\&+I_{2,4}(y,x)-I_{2,4}(x,y)-2I_{3,3}(x,y)-I_{4,2}(y,x)-2I_{4,2}(x,y)\end{aligned}$$

but, as with Φ_5, we cannot reduce further by stuffle relations.

We also see that Ψ_6 are Φ_6 are strongly related by

$$\Psi_6(x,y)-\Psi_6(y,x)=-\Phi_6(x,y).$$

Further linear combinations of multiple polylogarithms in the kernel of the symbol to those featured in this chapter are provided in Section A.3.

7.3 Conclusion

In this thesis we develop techniques to explore the symbol map of Goncharov and find new linear combinations of multiple polylogarithms in its kernel.

In the literature the symbol has been defined pictorially by Goncharov from binary trees in [Gon05] and from polygon dissections by Gangl, Goncharov and Levin in [GGL09]. We have shown that these two definitions agree and have given a pictorial way to move between them.

We have also introduced a new method of calculating the symbol (the hook-arrow tree) and again shown that it agrees with binary trees and polygons. The hook-arrow tree provides an algorithm for symbol calculation that can be computed with GP/Pari. It also allowed us to simplify symbol calculations where the depth of the multiple polylogarithm is given. We have given explicit formulation of the symbol of depth 2 and 3 multiple polylogarithms of any weight.

Using the hook-arrow tree we have presented the explicit formulation of the symbol of coloured multiple zeta values (CMZV). This takes the form $\lambda 2^{\otimes w}$ for a weight w CMZV, where λ is explicitly formulated.

We examined the symbol of harmonic polylogarithms and have given the first non-trivial functional equations relating different harmonic polylogarithms in weight 8.

Finally, in this chapter, we used GP/Pari to find linear combinations of multiple polylogarithms in weights 4, 5 and 6 in two variables and in depth $\leq$2.

Bibliography

[BBBL01] J.M. Borwein, D.M. Bradley, D.J. Broadhurst, and P. Lisonek, *Special values of multiple polylogarithms*, Trans. Amer. Math. Soc. **353** [arXiv:math/9910045] (2001), 907–941.

[BK95] S. Bloch and I Kriz, *Mixed Tate motives*, Annals of Math., 140 (1995), 557–605.

[Blo00] S. Bloch, *Higher regulators, algebraic K- theory and zeta functions of elliptic curves*, CRM Monograph Series, 11. American Mathematical Society, Providence, RI (better known as Irvine Lecture Notes, 1978) (2000).

[Bro09] F.C.S. Brown, *Multiple zeta values and periods of moduli spaces* $\mathfrak{M}_{0,n}$, Annales scientifiques de l'ENS 42, fascicule 3 (2009), 371–489.

[Bro11] ______, *Mixed Tate motives over* $\mathbb{Z}$, arXiv:1102.1312v1 [math.AG] (2011).

[BS66] Z.I. Borevich and I.R. Shafarevich, *Number theory*, Academic Press, 1966.

[Dan11] N. Dan, *Sur la conjecture de Zagier pour* $n = 4$*. II*, arXiv:1101.1557v1 [math.KT] (2011).

[DDDS10] V. Del Duca, C. Duhr, and V. Smirnov, *An analytic result for the two-loop hexagon Wilson loop in* $\mathcal{N} = 4$ *SYM*, J. High Energy Phys. no. 3, 099, (2010).

[DGR11] C. Duhr, H. Gangl, and J. Rhodes, *From polygons and symbols to polylogarithmic functions (preprint)*, arXiv:1110.0458v1 [math-ph] (2011).

[Duh12] C. Duhr, *Hopf algebras, coproducts and symbols: an application to Higgs boson amplitudes*, arXiv:1203.0454v1 [hep-ph] (2012).

[Gan03] H. Gangl, *Functional equations of higher logarithms*, Selecta Math. (N.S.) 9, no. 3. (2003), 361–377.

[Gan10] ______, *Functional equations and ladders for polylogarithms*, preprint (2010).

[GGL07] H. Gangl, A.B. Goncharov, and A. Levin, *Multiple logarithms, algebraic cycles and trees*, Frontiers in Number Theory, Physics and Geometry II: On Conformal Field Theories, Discrete Groups and Renormalization (2007), 759–774.

[GGL09] ______, *Multiple polylogarithms, polygons, trees and algebraic cycles*, Proc. of Summer Institute in Algebraic Geometry, Seattle 2005, Proc. Symp. Pure Math. 80 (2009), 547–594.

[GM04] A. Goncharov and Y. Manin, *Multiple ζ-motives and moduli spaces $\mathcal{M}_{0,n}$*, Compos. Math. 140, no. 1 (2004), 1–14.

[GMS99] H. Gangl and S. Mueller-Stach, *Polylogarithmic identities in cubical higher Chow groups*, Proc. of Symp. in Pure Math., volume 67; AMS, Providence, (1999), 25–40.

[Gon95] A. Goncharov, *Geometry of configurations, polylogarithms and motivic cohomology*, Advances in Mathematics, Vol. 114, No. 2, (1995).

[Gon97] ______, *The double logarithm and Manin's complex for modular curves*, Mathematical Research Letters, vol. 4, No 1 (1997), 1–20.

[Gon98] ______, *Multiple polylogarithms, cyclotomy and modular complexes*, Mathematical Research Letters, vol. 5, No 3, (1998), 497–516.

[Gon01] ______, *Multiple polylogarithms and mixed Tate motives*, arXiv: math.AG/0103059 (2001).

[Gon05] ______, *Galois symmetries of fundamental groupoids and noncommutative geometry*, Duke Math. J. 128, no. 2 (2005), 209–284.

[Gon09] ______, *A simple construction of Grassmannian polylogarithms*, arXiv:0908.2238v3 [math.AG] (2009).

[GSVV10] A.B. Goncharov, M. Spradlin, C. Vergu, and A. Volovich, *Classical polylogarithms for amplitudes and Wilson loops*, Phys. Rev. Lett. 105 151605 [arXiv:1006.5703 [hep-th]] (2010).

[HK11] P. Heslop and V.V. Khoze, *Wilson loops @ 3-loops in special kinematics*, arXiv:1109.0058v2 [hep-th] (2011).

[HP91] P. Hilton and J. Pederson, *Catalan numbers, their generalization, and their uses*, The Mathematical Intelligencer Vol 13, No 2 (1991), 64–75.

[Kla70] D. Klarner, *Correspondences between plane trees and binary sequences*, Journal of Combinatorial Theory, 9 (1970), 401–411.

[MUW02] S. Moch, P. Uwer, and S. Weinzierl, *Two-loop amplitudes with nested sums: Fermionic contributions to $e+e- \to q\bar{q}g$*, Phys.Rev.D66:114001 (2002).

[OEI12] OEIS Foundation Inc., *The on-line encyclopedia of integer sequences*, 2012, http://oeis.org.

[PAR11] PARI Group, Bordeaux, *GP/Pari, version* `2.5.0`, 2011, available from http://pari.math.u-bordeaux.fr/.

[RV00] E. Remiddi and J. A. M. Vermaseren, *Harmonic polylogarithms*, Int. J. Mod. Phys. A 15 [arXiv:hep-ph/9905237] (2000), 725–754.

[VW05] J. Vollinga and S. Weinzierl, *Numerical evaluation of multiple polylogarithms*, Comput. Phys. Commun. 167 (2005), 177–194.

[WB09] S. Weinzierl and C. Bogner, *Feynman graphs in perturbative quantum field theory*, arXiv:0912.4364v1 [math-ph] (2009).

[Wil94] H.S. Wilf, *Generatingfunctionology*, Academic Press, 1994.

[Woj02] Z. Wojtkowiak, *Mixed Hodge structures and iterated integrals. I.*, Motives, Polylogarithms and Hodge Theory (Part I: Motives and Polylogarithms), F. Bogomolov, L. Katzarkov (Eds.), Int. Press Lect. Ser., 3, I, Somerville, MA (2002), 121–208.

[Zag86] D. Zagier, *Hyperbolic manifolds and special values of Dedekind zeta-functions*, Inventiones Math. **83** (1986), 285 – 301.

[Zag88] ______, *The remarkable dilogarithm*, J. Math and Phys. Soc. **22** (1988), 131–145.

[Zag90] ______, *The Bloch-Wigner-Ramakrishnan polylogarithm function*, Math-Annalen **286** (1990), 612–624.

[Zag91] ______, *Polylogarithms, Dedekind zeta functions, and the algebraic K-theory of fields*, Arithmetic Algebraic Geometry, Prog. in Math. **89** (1991), 391–430.

[ZG00] D. Zagier and H. Gangl, *Classical and elliptic polylogarithms and special values of L-series*, The arithmetic and geometry of algebraic cycles (Banff, AB, 1998) NATO Sci. Ser. C Math. Phys. Sci., Kluwer Acad. Publ. (2000), 561–615.

[Zha04] J. Zhao, *Variations of mixed Hodge structures of multiple polylogarithms*, Canad. J. Math. 56, no. 6. (2004), 1308–1338.

[Zha07] J. Zhao, *Analytic continuation of multiple polylogarithms*, Analysis Mathematica, 33 (2007), 301–323.

[Zha10] ______, *Standard relations of multiple polylogarithm values at roots of unity*, Documenta Mathematica 15 (2010), 1–34.

[ZN85] D. Zagier and W.D. Neumann, *Volumes of hyperbolic 3-manifolds*, Topology **24** (1985), 307–332.

Appendix A

Using GP/Pari to find elements in the kernel of the symbol

The hook-arrow tree is harder to define than the polygons from [GGL09]. However, the author hopes that the reader can see some benefits when trying to make a computer calculate the symbol of a multiple polylogarithm. In particular, the shuffle relations appear by construction and so there is no need to program a check for whether a polygon has a shuffle, something which proved difficult in initial attempts to encode finding a symbol. Section 2.3 is deliberately set out in algorithm form so as to ease transition to GP/Pari.

Remark A.1. A computer program has been written by the author to calculate the symbol but we do not give it here. It was used for calculating symbols too cumbersome to do by hand throughout this thesis.

In this appendix we outline a method of finding linear combinations of multiple polylogarithms in the kernel of the symbol, given a computer program that can calculate the symbol. The method we use is based on standard linear algebraic techniques. We start with a motivational example.

Example A.2. The Hölder convolution from Definition 1.5.1 tells us to expect

$$\mathcal{S}\big(I_2(x) - I_{1,1}(1, 1-x)\big) = 0$$

We calculate this explicitly as

$$\begin{aligned}
\mathcal{S}\big(I_2(x) - I_{1,1}(1,1-x)\big) &= \mathcal{S}\big(I_2(x)\big) - \mathcal{S}\big(I_{1,1}(1,1-x)\big) \\
&= \left(\frac{x}{x-1} \otimes x\right) + \left(\frac{x-1}{x} \otimes x\right) \\
&= \left(\frac{x}{x-1} \otimes x\right) - \left(\frac{x}{x-1} \otimes x\right) \\
&= 0.
\end{aligned}$$

We see that the symbol of $I_2(x)$ and the symbol of $I_{1,1}(1,1-x)$ are not identical; a small amount of tensor calculus is required to make them cancel. This is possible to do by hand because the symbols in question are simple, however, the symbols of higher weight multiple polylogarithms can become very large. This motivates us to find a good way to compare symbols, i.e. in this case, a way to do tensor calculus methodically with a computer.

We now outline one possible way to go about this. It is by no means perfect and is, to a certain extent, a 'sledgehammer' approach. However, given that computers with relatively high power are readily available, it has proved to give some good results. The method was told to the author by H Gangl.

A.1 Attaching a vector to a tensor product

We observe that arguments in the relation of Example A.2 and the arguments of the tensor product in the symbol are all equal to

$$x^a(1-x)^b$$

for some $a, b \in \mathbb{Z}$. Due to linearity of tensor products we therefore see that

$$x^a(1-x)^b \otimes x^c(1-x)^d = ac\big(x \otimes x\big) + ad\big(x \otimes 1-x\big) + bc\big(1-x \otimes x\big) + bd\big(1-x \otimes 1-x\big).$$

We can attach to $p\big(x^a(1-x)^b \otimes x^c(1-x)^d\big)$, with $p \in \mathbb{Q}$, the vector

$$[pac, pad, pbc, pbd].$$

For the above linear combination the calculation to show it has zero symbol now takes the form

$$\begin{aligned}
\mathcal{S}\big(I_2(x) - I_{1,1}(1, 1-x)\big) &= \mathcal{S}\big(I_2(x)\big) - \mathcal{S}\big(I_{1,1}(1, 1-x)\big) \\
&= \left(\frac{x}{x-1} \otimes x\right) + \left(\frac{x-1}{x} \otimes x\right) \\
&\to [1, 0, -1, 0] + [-1, 0, 1, 0] \\
&= 0.
\end{aligned}$$

The addition of vectors is considerably easier, and faster, for a computer than tensor calculus. We are motivated to test the symbols of many multiple polylogarithms and therefore reduce the search for elements in the kernel of the symbol to a linear algebra problem. It is important to remember that this method does require every component of every tensor in a symbol to be formed as the product of elements from a selected set of functions. We will discuss this further after formally defining the procedure of attaching a vector to a tensor product.

Definition A.3. Given a set of base functions $F = \{f_1, ..., f_m\}$ and a tensor product

$$T_f = p \bigotimes_{s=1}^{n} \prod_{t=1}^{m} f_s^{a_{s,t}},$$

with $p \in \mathbb{Z}$, then we define a vector V_F of length m^n with i-th component

$$p \prod_{s=1}^{n} a_{s,t_s}$$

where t_s is the s-th component of a vector of the form

$$(t_1, ..., t_n) \quad \text{with} \quad t_s \in \{1, ..., m\}.$$

We define i to be position of the chosen vector $(t_1, ..., t_n)$ in the lexicographic order of all possible vectors of that form.

We now give another worked example.

Example A.4. We take $F = \{x, 1-x\}$ and examine the weight 3 functions $I_3, I_{1,2}(x, x), I_{1,2}\big(\frac{1}{x}, \frac{1}{x}\big)$ and $I_{2,1}\big(\frac{1}{1-x}, \frac{1}{1-x}\big)$.

Firstly we have

$$\mathcal{S}(I_3) = \frac{x-1}{x} \otimes x \otimes x$$

and so is associated to the vector

$$[-1,0,0,0,1,0,0,0]$$

which is equivalent to

$$\begin{pmatrix} -1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} x & \otimes & x & \otimes & x \\ x & \otimes & x & \otimes & 1-x \\ x & \otimes & 1-x & \otimes & x \\ x & \otimes & 1-x & \otimes & 1-x \\ 1-x & \otimes & x & \otimes & x \\ 1-x & \otimes & x & \otimes & 1-x \\ 1-x & \otimes & 1-x & \otimes & x \\ 1-x & \otimes & 1-x & \otimes & 1-x \end{pmatrix}.$$

The other functions are associated to vectors as follows.

$I_{1,2}(x,x)$	associates to	$[-1,0,1,0,1,0,-1,0]$
$I_{1,2}\left(\frac{1}{x},\frac{1}{x}\right)$	associates to	$[0,0,0,0,0,0,1,0]$
$I_{2,1}\left(\frac{1}{1-x},\frac{1}{1-x}\right)$	associates to	$[0,0,1,0,0,0,0,0]$

We can see that

$$\mathcal{S}\left(I_3 - I_{1,2}(x,x) - I_{1,2}\left(\frac{1}{x},\frac{1}{x}\right) + I_{2,1}\left(\frac{1}{1-x},\frac{1}{1-x}\right)\right)$$

is equivalent to summing the vectors

$$[-1,0,0,0,1,0,0,0]+[1,0,-1,0,-1,0,1,0]+[0,0,0,0,0,0,-1,0]+[0,0,1,0,0,0,0,0],$$

which is the zero vector and tells us that the suggested symbol is indeed zero.

Remark A.5. It is important to note that setting the arguments of a multiple polylogarithm to be formed as a product of elements from a set of functions does not guarantee the tensor products in its symbol will have elements that are a product of elements in the set. For example, given $F = \{x, 1-x\}$, then

$$\mathcal{S}(I_3(x(1-x))) = \frac{-x^2+x-1}{x(1-x)} \otimes x(1-x) \otimes x(1-x).$$

There obviously does not exist $a, b \in \mathbb{Z}$ such that

$$\frac{-x^2+x-1}{x(1-x)} = x^a(1-x)^b.$$

for all x. This stresses the importance of the choice of the set F. We can expand the function set to $\{x, 1-x, 1-x+x^2\}$ and then

$$\frac{-x^2+x-1}{x(1-x)} = -x^{-1}(1-x)^{-1}(1-x+x^2).$$

So, we can associate the vector

$$V_F = [-1,-1,0,-1,-1,0,0,0,0,-1,-1,0,-1,-1,0,0,0,0,1,1,0,1,1,0,0,0,0]$$

to $I_3(x(1-x))$. Balancing the scope of the variables used and the size of F will play an important role when we try to maximise the results. Increasing the size of F increases the size of V_F exponentially.

A.2 An overview of finding elements in the kernel of the symbol with GP/Pari

We now use both our GP/Pari script for finding the symbol of a multiple polylogarithm and our proposed method of converting a symbol into a vector to linear combinations of multiple polylogarithms in the kernel of the symbol. The outline of a script to do this follows:

1. Choose a set of functions $F = \{f_1, ..., f_m\}$.
2. Choose a range $[d_1, d_2]$ for some $d_1, d_2 \in \mathbb{Z}$.
3. Choose a selection of multiple polylogarithms of a fixed weight, possibly all of them.

4. Construct a list, L, of possible arguments

$$\prod_{i=1}^{m} f_i^{b_i}$$

for all $b_i \in [d_1, d_2] \subset \mathbb{N}$.

5. For all possible selections of arguments from list L find the symbol of the multiple polylogarithms chosen.

6. Determine if each symbol can be associated to a vector using basis $F = \{f_1, ..., f_m\}$ and create a list of successful multiple polylogarithms with their vectors.

7. Perform linear algebraic methods to find linear dependences within the set of vectors.

This is the method used in Chapter 7 and in the following section for finding the linear combinations of multiple polylogarithms in the kernel of the symbol map.

A.3 Further linear combinations of multiple polylogarithms in the kernel of the symbol

The following list of elements,

$$\Xi_{w,n} \in \mathcal{I}_w(S), \quad \text{with} \quad \mathcal{S}(\Xi_{w,n}) = 0$$

are provided ‘as is’ and have not been fully explored.

Note that here that the calculations were made using the convention that

$$\mu(P(a,a)) = 1 \quad \text{rather than} \quad \mu(P(a,a)) = \frac{1}{a}.$$

As a consequence terms in the symbol arising from a polygon dissection with a 2-gon of the form $P(a,a)$ do not contribute.

Weight 2

$$\Xi_{2,1} = I_{1,1}(x,y) + I_{1,1}(x,1-y) + I_{1,1}(1-x,y) + I_{1,1}(1-x,1-y)$$

Weight 3

$$\begin{aligned}\Xi_{3,1} =& - I_{1,2}\left(\frac{1}{x},\frac{1}{x}\right) - I_{1,2}(x,x) + I_{2,1}\left(\frac{1}{1-x},\frac{1}{1-x}\right) + I_3(x)\\ \Xi_{3,2} =& - I_{1,2}\left(\frac{T}{x},\frac{T}{x}\right) + I_{1,2}\left(\frac{T}{x},T\right) + I_{1,2}\left(\frac{T}{x},x\right) + I_{1,2}\left(T,\frac{T}{x}\right)\\ &- I_{1,2}\left(T,T\right) + I_{1,2}\left(T,1-x\right)\\ &\text{where } T = 1-x+x^2.\end{aligned}$$

$$\Xi_{3,3} = I_3(xy) + I_{1,2}(x,y) + I_{1,2}(y,x) - I_{1,2}(xy,x) + I_{2,1}(x,y) - I_{2,1}(xy,y)$$

$$\begin{aligned}\Xi_{3,4} =& I_{1,2}(x,y) + I_{1,2}(y,x) + I_{1,2}(x,1-y) + I_{1,2}(1-y,x)\\ &+ I_{2,1}(x,y) + I_{2,1}(x,1-y)\end{aligned}$$

$$\begin{aligned}\Xi_{3,5} =& I_{1,2}(x,y) + I_{1,2}(y,x) - I_{1,2}(1-x,1-y) - I_{1,2}(1-y,1-x)\\ &+ I_{2,1}(x,y) + I_{2,1}(x,1-y) - I_{2,1}(1-y,1-x) - I_{2,1}(1-y,x)\end{aligned}$$

Weight 4

$$\begin{aligned}\Xi_{4,1} =& 2I_{1,3}(x,x) + I_{1,3}(x(1-x),x) + I_{2,2}(x,x) + I_{3,1}(x,x)\\ &- I_{3,1}(x(1-x),1-x) - I_4(x(1-x))\end{aligned}$$

$$\begin{aligned}\Xi_{4,2} =& I_{1,3}(y(1-x),y) + I_{1,3}(xy,y) + I_{3,1}(y(1-x),1-x)\\ &+ I_{3,1}(xy,x) - I_4(y(1-x)) - I_4(xy)\end{aligned}$$

$$\begin{aligned}\Xi_{4,3} =& I_{1,3}(x,y) + I_{1,3}(y,x) - I_{1,3}(xy,x) + I_{2,2}(x,y)\\ &+ I_{3,1}(x,y) - I_{3,1}(xy,y) + I_4(xy)\end{aligned}$$

Weight 5

$$\begin{aligned}\Xi_{5,1} =& - I_{2,2,1}(y,y,x) + I_{2,2,1}(y,x,y) - I_{2,2,1}(y,xy,y) + I_{2,2,1}(y,xy,x)\\ &+ I_{2,1,2}(y,y,x) - I_{2,1,2}(x,y,x) + I_{2,1,2}(y,xy,y) - I_{2,1,2}(y,xy,x)\\ &+ I_{1,2,2}(y,x,x) - I_{1,2,2}(y,x,y)\\ &+ I_{2,3}(x,y) - I_{2,3}(y,x)\end{aligned}$$

$$\begin{aligned}\Xi_{5,2} =& - I_{2,2,1}(y,y,x) + I_{2,2,1}(x,y,y) + I_{2,2,1}(y,xy,x) - I_{2,2,1}(x,xy,y)\\ &- I_{2,2,1}(xy,y,y) + I_{2,2,1}(xy,x,y)\\ &+ I_{2,1,2}(y,xy,y) - I_{2,1,2}(x,xy,x)\\ &- I_{1,2,2}(xy,x,y) + I_{1,2,2}(xy,x,x)\\ &- I_{2,3}(y,xy) + I_{2,3}(x,xy) + I_{2,3}(xy,y) - I_{2,3}(xy,x)\end{aligned}$$

$$\begin{aligned}\Xi_{5,3} =& - I_{1,2,2}(x,y,x) + I_{1,2,2}(x,y,y) - I_{1,2,2}(y,x,x) + I_{1,2,2}(y,x,y)\\ &+ I_{1,2,2}(xy,y,x) - I_{1,2,2}(xy,y,y)\\ &- I_{2,1,2}(x,x,y) - I_{2,1,2}(x,y,x) + I_{2,1,2}(y,x,y) + I_{2,1,2}(y,y,x)\\ &+ I_{2,2,1}(xy,x,x) - I_{2,2,1}(xy,y,x) - I_{2,2,1}(y,x,x) + I_{2,2,1}(y,y,x)\\ &+ I_{3,2}(xy,y) - I_{3,2}(xy,x)\end{aligned}$$

Weight 6

$$\begin{aligned}\Xi_{6,1} =& I_{2,4}(xy,y) - I_{2,4}(xy,x) - 2I_{2,4}(y,x) + 2I_{2,4}(x,y)\\ &- 2I_{3,3}(y,x) + 2I_{3,3}(x,y)\\ &- I_{4,2}(xy,y) + I_{4,2}(xy,x) - I_{4,2}(y,x) + I_{4,2}(x,y)\end{aligned}$$

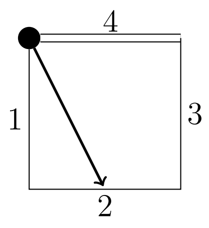
Appendix B

Non-maximal dissection in the language of hook-arrow trees

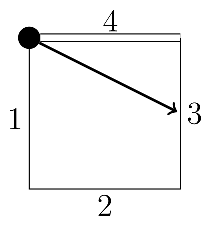
B.1 Motivation

The hook-arrow tree construction outlined in Chapter 2 is only an equivalent representation of maximal dissections of polygons. However, the full bar construction from [GGL09] of a polygon also includes non-maximal dissections (a dissection of an n-gon with less than $n-2$ arrows).

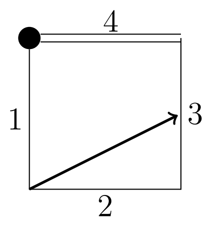
Example B.1. As seen on page 572 of [GGL09] there are 8 possible placements of adding a single arrow to the 4-gon $P(1,2,3,4)$. The relevant interpretation of these non-maximal dissections is also included.



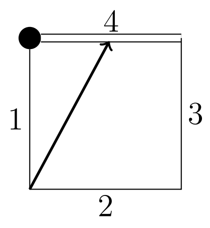
$+\ 234 \mid 12$



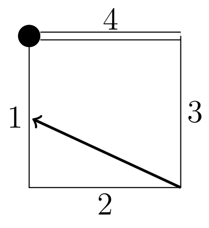
$+\ 34 \mid 123$



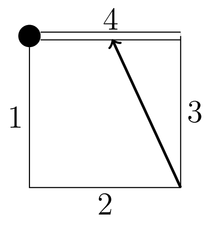
$+\ 134 \mid 23$



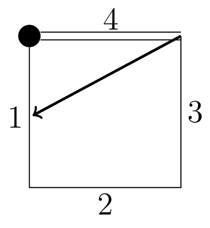
$+\ 14 \mid 234$



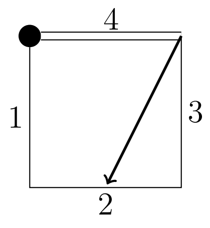
$-\ 134 \mid 21$



$+\ 124 \mid 34$



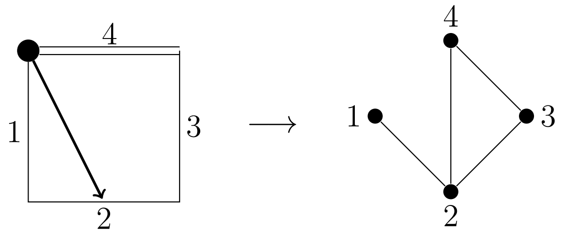
$+\ 14 \mid 321$



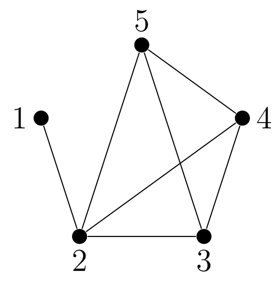
$-\ 124 \mid 32$

B.2 Hook-arrow bulbs

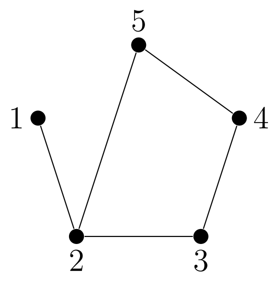
We are motivated to find the equivalent method for the hook-arrow tree view of dissections. To do this, we follow a very similar procedure as described in Section 2.2 and add vertices to the centre of each side of a non-maximal dissected polygon and allow them to inherit the label from the edge. We then join every possible vertex with a straight line that will not cross an arrow. We remove the polygon and arrows and are left with a connected graph. We note that due to the dissection of the polygon not being maximal we will have loops in the resulting graph. For example, the first non-maximally dissected 4-gon in the above example gives a graph in the following way.



These loops will appear in sets of vertices that are all connected to each other (representing parts of the polygon that could have had more arrows added). We consider each of these sets to be a face and disregard the diagonals. We don't actually consider the faces as 2-dimensional: instead as a collection of vertices and edges bounding a region, and will denote the faces by their set of bounding vertices. As an example



will have diagonals disregarded and become



creating faces $f_1 = \{1, 2\}$ and $f_2 = \{2, 3, 4, 5\}$.

Notation B.2. We will call these graphs hook-arrow bulbs, in that they represent a non-maximal dissection and are thus partially formed hook-arrow trees (and slightly resemble a sprouting bulb). The faces of a hook-arrow bulb are a generalisation of the edges of hook-arrow trees. By construction, a face will meet any other face at a maximum of one vertex and so will never share an edge.

We now formally define a hook-arrow bulb.

Definition B.3. A **hook-arrow bulb**, β is a rooted spanning graph on a set of vertices in a linear order, $(v_1^\beta, \dots, v_n^\beta)$, for which no edges are interlaced (see Definition 2.3) and has root v_n^β.

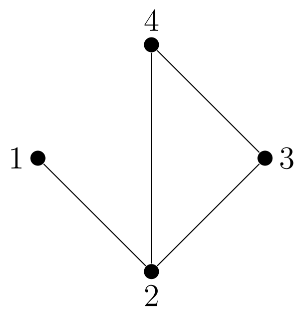
Remark B.4. Due to disregarding the diagonals on faces, the definition of interlacing here applies suitably and hook-arrow bulbs are seen as a natural extension of the definition of hook-arrow trees that now allow loops.

A hook-arrow bulb, β, on n vertices $V_\beta = (v_1^\beta, ..., v_n^\beta)$ has a set of edges $F_\beta = \{f_1^\beta, ..., f_m^\beta\}$. The number of faces can be seen to be $m = t+1$ where t is the number of dissecting arrows in the polygon dissection.

Definition B.5. A hook-arrow bulb has a **description** of

$$C_\beta = \{V_\beta, F_\beta\} = \{(v_1^\beta, ..., v_n^\beta), \{f_1^\beta, ..., f_m^\beta\}\}.$$

Example B.6. The hook-arrow bulb, β',



would have a description of

$$C_{\beta'} = \{(1,2,3,4), \{\{1,2\},\{2,3,4\}\}\}.$$

B.2.1 Obtaining the bar construction element of a hook-arrow bulb

We now explain how to direct a hook-arrow bulb and extract its element in the bar construction. The method is a generalisation of the procedure of directing and extracting the symbol from a hook-arrow tree.

B.2.1.1 Step 1 - Selecting the first face

The first face is selected in the same way as the first distinguished edge on a hook-arrow tree. The first distinguished face will again intuitively be the first face hit by moving around the distinguished final vertex anticlockwise. More precisely, we start with a hook-arrow bulb β with description C_β. We select a unique first and distinguished face of β, denoted f_d^β as

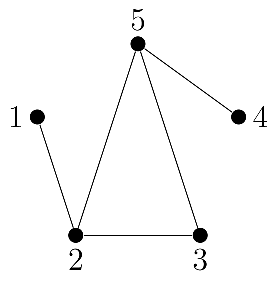
$$f_d^\beta = \{f \in F_\beta \,|v_i^\beta, v_n^\beta \in f \text{ and } v_1^\beta, \dots, v_{(i-1)}^\beta \notin f\}.$$

Again, this is a natural extension of the idea of a distinguished edge for hook-arrow trees.

This chosen distinguished face is then given a linear order on its vertices, obeying the linear order in V_β and ending with v_n^β. As with hook-arrow trees we will use the notation $[v_\bullet^\beta, \ldots, v_\bullet^\beta]$ to denote a face where the linear order is designated.

Example B.7. The distinguished face of the hook-arrow bulb described by

$$C_\beta = \{(1,2,3,4,5), \{\{1,2\}, \{2,3,5\}, \{4,5\}\}\},$$

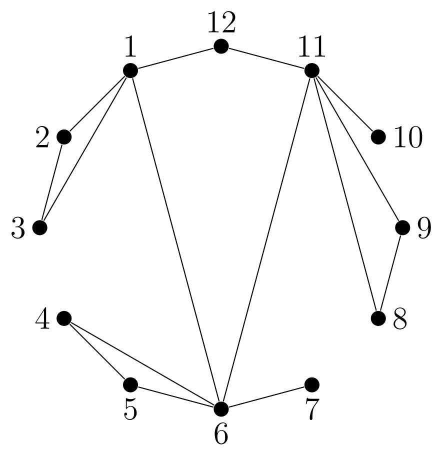


is $\{2,3,5\}$ and is given a linear order of $[2,3,5]$.

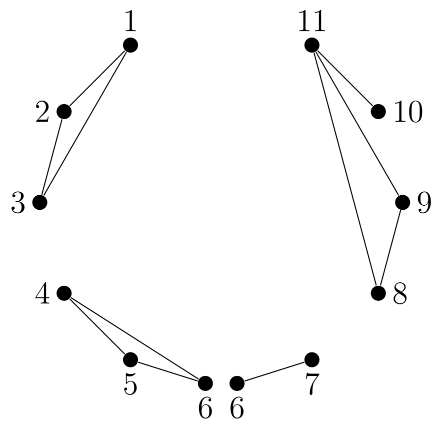
B.2.1.2 Step 2 - Splitting the hook-arrow bulb

We again generalise the algorithm for the hook-arrow trees. We remove the edges of the first distinguished face and form 'sub-bulbs'. Sub-bulbs containing only one vertex are trivial. Sub-bulbs must also only contain vertices that are including and immediately after, or, including and immediately before, a vertex. This method is analogous to that in step 2 of the hook-arrow tree algorithm and a suitably complicated example follows to demonstrate this (instead of explicitly explaining). The sub-bulbs are labelled β_i in the direction of the linear order of V_β.

Example B.8. We examine the hook-arrow bulb, β, given by



The first distinguished face will be $[1, 6, 11, 12]$. After removing this face we are left with 4 sub-bulbs.



We orientate a sub-bulb, β_i and give its vertices:

- the same linear order as that of V_β if it is formed from a vertex of f_d^β and vertices immediately before in the linear order of V_β, i.e.,

$$v_i^\beta, v_{(i-1)}^\beta \in \beta_i \text{ where } v_i^\beta \in f_d^\beta.$$

- the opposite orientation to the linear order of V_β if it is formed from a vertex of f_d^β and vertices immediately after in the linear order of V_β, i.e.,

$$v_i^\beta, v_{(i+1)}^\beta \in \beta_i \text{ where } v_i^\beta \in f_d^\beta.$$

The vertices of each sub-bulbs now have a linear order and we take the final vertex of each sub-bulb to be the vertex it includes from f_d^β. We can therefore give each sub-bulb a description, C_{β_i} and it is itself a hook-arrow bulb.

B.2.1.3 Step 3 - Iterate the process

We then repeat the steps 1 and 2 on each sub-bulb and record the results in a similar way to that of the hook-arrow trees, shuffling sub-bulbs which are at the same depth.

Example B.9. We continue to apply the algorithm to the hook-arrow bulb from example B.8.

We have the distinguished face $f_d^\beta = [1, 6, 11, 12]$ with sub-bulbs with descriptions as follows:

- β_1 given by $C_{\beta_1} = \{(3, 2, 1), \{\{1, 2, 3\}\}\}$.
- β_2 given by $C_{\beta_2} = \{(4, 5, 6), \{\{4, 5, 6\}\}\}$.
- β_3 given by $C_{\beta_3} = \{(7, 6), \{\{6, 7\}\}\}$.
- β_4 given by $C_{\beta_4} = \{(8, 9, 10, 11), \{\{8, 9, 11\}, \{10, 11\}\}\}$.

We note that β_1 and β_3 have a different linear order to f_d^β.

The sub-bulbs β_1, β_2 and β_3 only have one face which therefore becomes the distinguished face of that sub-tree. We record these faces as

$$f_d^{\beta_1} = [3, 2, 1],\ f_d^{\beta_2} = [4, 5, 6] \text{ and } f_d^{\beta_3} = [7, 6].$$

The distinguished face of β_4 will be $[8, 9, 11]$. We obtain one sub-bulb of β_4, namely $\beta_{4,1}$, which has description

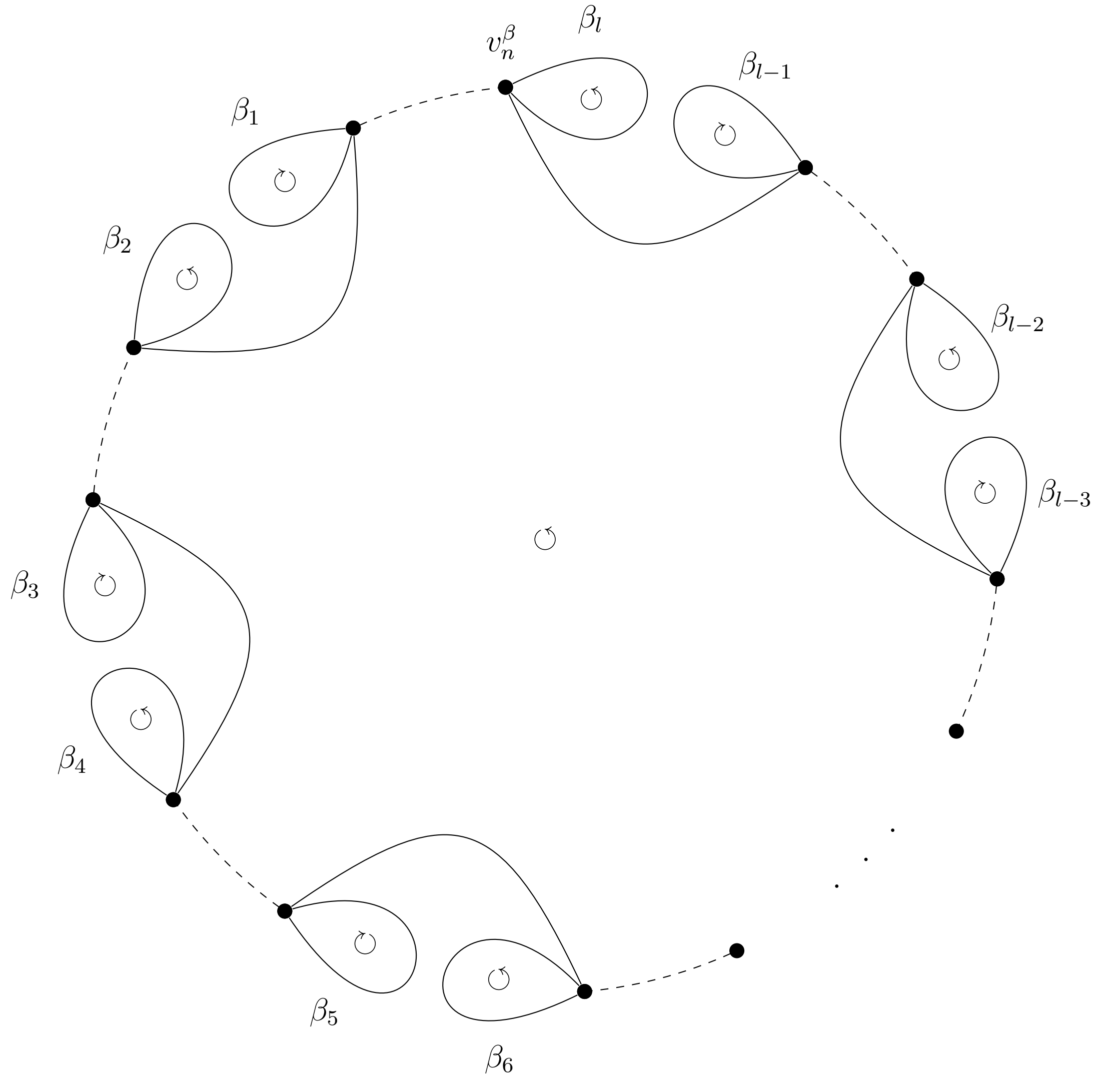
$$C_{\beta_{4,1}} = \{(10, 11), \{\{10, 11\}\}\}.$$

Overall the bar element represented by this hook-arrow bulb is

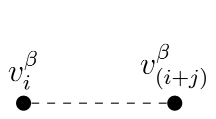
$$[1, 6, 11, 12] \,\big|\, \big([3, 2, 1] \sqcup\!\sqcup [4, 5, 6] \sqcup\!\sqcup [7, 6] \sqcup\!\sqcup ([8, 9, 11] \,|\, [10, 11])\big).$$

B.3 General picture

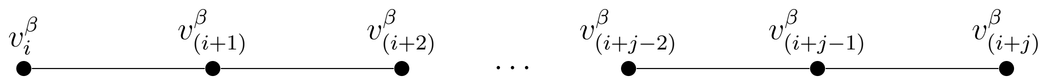
The analogous general picture of a hook-arrow bulb to that of the hook-arrow tree in Section 2.4.5 is as follows:



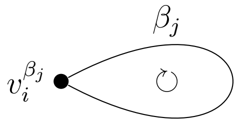
where



represents $j+1$ points connected to each other in a line in the form



and



represents a sub-bulb with the orientation indicated.

Note that, as with the subtrees formed in the algorithm for hook-arrow trees, some (or all) of the sub-bulbs, β_i, may be trivial (i.e., consist only of a single vertex). We also note that edges between different $v_i^{\beta_j}$ are only displayed curved for ease of the schematic drawing.

B.4 Further thoughts on hook-arrow bulbs

Hook-arrow bulbs are offered here as an alternative to the non-maximal dissections of polygons that make up the full bar construction outlined in Section 1.4.1. Further research into extending the symbol to cover all parts of the bar construction and comparing multiple polylogarithms may well lead to seeing extra structure. An important aspect of the symbol is that it simplifies the comparison of, while still holding a lot of important information about, multiple polylogarithms. However, if reasonable algebraic manipulation of non-maximal dissections can be constructed, particularly if a computer could calculate them quickly, interesting results could arise. It is hoped that the systematic structure of the algorithm in this appendix could be encoded in a program such as GP/Pari.

Appendix C

The symbol for $I_{2,2,2}(x, y, z)$

C.1 Hook-arrow trees attached to $I_{2,2,2}(x, y, z)$

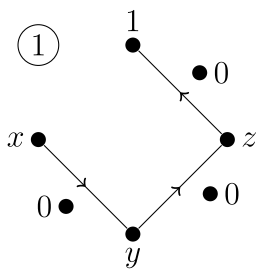
The polygon representing $I_{2,2,2}(x, y, z)$,

$$P(x, 0, y, 0, z, 0, 1)$$

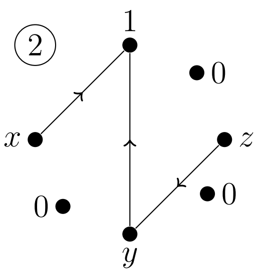
will have, as seen from Proposition 4.10,

$$\frac{1}{2}r_1r_2(r_3 + 2)(r_1 + r_2 + r_3 + 5) = \frac{1}{2} \cdot 2 \cdot 2 \cdot 4 \cdot 11 = 88$$

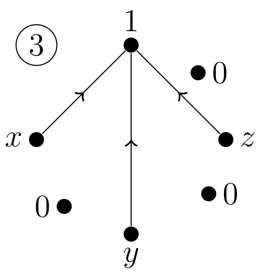
possible dissections. However, by the proof of Proposition 4.10 we know that these can be divided into 12 groups (represented by each summation sign in Theorem 4.12). Each group being represented by a hook-arrow tree of a dissection of the polygon $P(x, y, z, 1)$. We then find every possible addition of the three vertices labelled 0. These tree groupings are now shown.



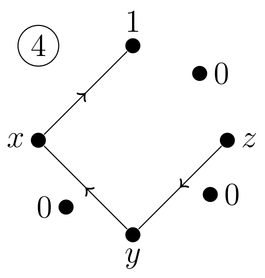
Trees of this type = 4



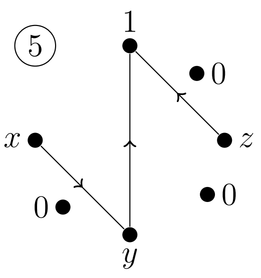
Trees of this type = 8



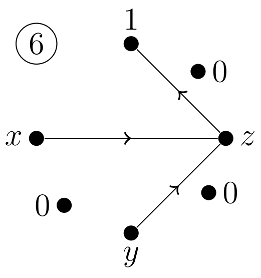
Trees of this type = 4



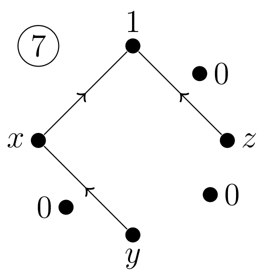
Trees of this type = 12



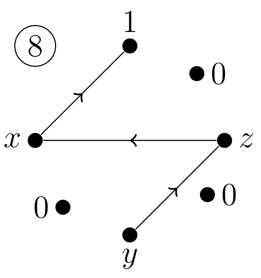
Trees of this type = 4



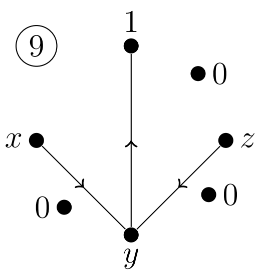
Trees of this type = 6



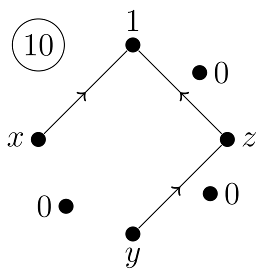
Trees of this type = 6



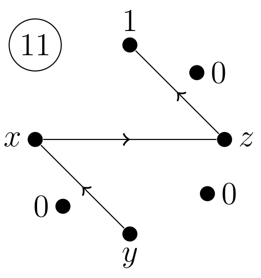
Trees of this type = 12



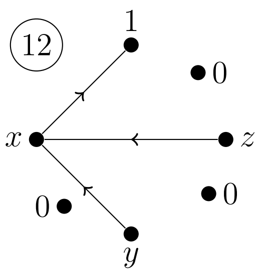
Trees of this type = 8



Trees of this type = 6



Trees of this type = 6



Trees of this type = 12

We observe that there are 88 possible hook-arrow trees, as expected.

C.2 The symbol $\mathcal{S}(I_{2,2,2}(x,y,z))$

We now use Theorem 4.12 to give that full symbol for $I_{2,2,2}(x,y,z)$. We group the terms by the number of the hook-arrow tree from which they originate above.

$\mathcal{S}(I_{2,2,2}(x,y,z)) =$

$$\begin{aligned}
&-\left(1-\tfrac{1}{z}\right)\otimes z \sqcup\!\sqcup \Big(\left(1-\tfrac{z}{y}\right)\otimes y \sqcup\!\sqcup \left(\left(1-\tfrac{y}{x}\right)\otimes x\right)\Big) && \text{(Tree 1)}\\
&+\left(1-\tfrac{1}{z}\right)\otimes z \sqcup\!\sqcup \Big(\left(1-\tfrac{z}{y}\right)\otimes z \sqcup\!\sqcup \left(\left(1-\tfrac{y}{x}\right)\otimes x\right)\Big)\\
&+\left(1-\tfrac{1}{z}\right)\otimes z \sqcup\!\sqcup \Big(\left(1-\tfrac{z}{y}\right)\otimes y \sqcup\!\sqcup \left(\left(1-\tfrac{y}{x}\right)\otimes y\right)\Big)\\
&-\left(1-\tfrac{1}{z}\right)\otimes z \sqcup\!\sqcup \Big(\left(1-\tfrac{z}{y}\right)\otimes z \sqcup\!\sqcup \left(\left(1-\tfrac{y}{x}\right)\otimes y\right)\Big)
\end{aligned}$$

$$\begin{aligned}
&+\left(1-\tfrac{1}{x}\right)\otimes x \sqcup\!\sqcup \Big(\left(1-\tfrac{1}{y}\right)\otimes y\otimes\left(1-\tfrac{y}{z}\right)\otimes y\Big) && \text{(Tree 2)}\\
&+\left(1-\tfrac{1}{x}\right)\otimes x \sqcup\!\sqcup \Big(\left(1-\tfrac{1}{y}\right)\otimes\left(1-\tfrac{y}{z}\right)\otimes y \sqcup\!\sqcup z\Big)\\
&-\left(1-\tfrac{1}{x}\right)\otimes x \sqcup\!\sqcup \Big(\left(1-\tfrac{1}{y}\right)\otimes y\otimes\left(1-\tfrac{y}{z}\right)\otimes z\Big)\\
&-\left(1-\tfrac{1}{x}\right)\otimes x \sqcup\!\sqcup \Big(\left(1-\tfrac{1}{y}\right)\otimes\left(1-\tfrac{y}{z}\right)\otimes z \sqcup\!\sqcup z\Big)\\
&-\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{y}\right)\otimes y \sqcup\!\sqcup \left(y\otimes\left(1-\tfrac{y}{z}\right)\otimes y\right)\\
&-\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{y}\right)\otimes y \sqcup\!\sqcup \left(\left(1-\tfrac{y}{z}\right)\otimes y \sqcup\!\sqcup z\right)\\
&+\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{y}\right)\otimes y \sqcup\!\sqcup \left(y\otimes\left(1-\tfrac{y}{z}\right)\otimes z\right)\\
&+\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{y}\right)\otimes y \sqcup\!\sqcup \left(\left(1-\tfrac{y}{z}\right)\otimes z \sqcup\!\sqcup z\right)
\end{aligned}$$

$$\begin{aligned}
&-\left(1-\tfrac{1}{x}\right)\otimes x \sqcup\!\sqcup \Big(\left(1-\tfrac{1}{y}\right)\otimes y \sqcup\!\sqcup \left(\left(1-\tfrac{1}{z}\right)\otimes z\right)\Big) && \text{(Tree 3)}\\
&+\left(1-\tfrac{1}{x}\right)\otimes x \sqcup\!\sqcup \Big(\left(1-\tfrac{1}{y}\right)\otimes\left(1-\tfrac{1}{z}\right)\otimes z \sqcup\!\sqcup z\Big)\\
&-\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{y}\right)\otimes y \sqcup\!\sqcup y \sqcup\!\sqcup \left(\left(1-\tfrac{1}{z}\right)\otimes z\right)\\
&+\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{1}{y}\right)\otimes y \sqcup\!\sqcup \left(\left(1-\tfrac{1}{z}\right)\otimes z \sqcup\!\sqcup z\right)
\end{aligned}$$

$$-\left(1-\tfrac{1}{x}\right)\otimes x\otimes\left(1-\tfrac{x}{y}\right)\otimes x\sqcup\left(\left(1-\tfrac{y}{z}\right)\otimes y\right) \qquad \text{(Tree 4)}$$
$$-\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{y}\right)\otimes x\sqcup\left(y\otimes\left(1-\tfrac{y}{z}\right)\otimes y\right)$$
$$-\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{y}\right)\otimes x\sqcup\left(\left(1-\tfrac{y}{z}\right)\otimes y\sqcup z\right)$$
$$+\left(1-\tfrac{1}{x}\right)\otimes x\otimes\left(1-\tfrac{x}{y}\right)\otimes x\sqcup\left(\left(1-\tfrac{y}{z}\right)\otimes z\right)$$
$$+\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{y}\right)\otimes x\sqcup\left(y\otimes\left(1-\tfrac{y}{z}\right)\otimes z\right)$$
$$+\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{y}\right)\otimes x\sqcup\left(\left(1-\tfrac{y}{z}\right)\otimes z\sqcup z\right)$$
$$+\left(1-\tfrac{1}{x}\right)\otimes x\otimes\left(1-\tfrac{x}{y}\right)\otimes y\sqcup\left(\left(1-\tfrac{y}{z}\right)\otimes y\right)$$
$$+\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{y}\right)\otimes y\sqcup\left(y\otimes\left(1-\tfrac{y}{z}\right)\otimes y\right)$$
$$+\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{y}\right)\otimes y\sqcup\left(\left(1-\tfrac{y}{z}\right)\otimes y\sqcup z\right)$$
$$-\left(1-\tfrac{1}{x}\right)\otimes x\otimes\left(1-\tfrac{x}{y}\right)\otimes y\sqcup\left(\left(1-\tfrac{y}{z}\right)\otimes z\right)$$
$$-\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{y}\right)\otimes y\sqcup\left(y\otimes\left(1-\tfrac{y}{z}\right)\otimes z\right)$$
$$-\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{y}\right)\otimes y\sqcup\left(\left(1-\tfrac{y}{z}\right)\otimes z\sqcup z\right)$$

$$-\left(1-\tfrac{1}{y}\right)\otimes\left(\left(1-\tfrac{y}{x}\right)\otimes x\right)\sqcup y\sqcup\left(\left(1-\tfrac{1}{z}\right)\otimes z\right) \qquad \text{(Tree 5)}$$
$$+\left(1-\tfrac{1}{y}\right)\otimes\left(\left(1-\tfrac{y}{x}\right)\otimes x\right)\sqcup\left(\left(1-\tfrac{1}{z}\right)\otimes z\sqcup z\right)$$
$$+\left(1-\tfrac{1}{y}\right)\otimes\left(\left(1-\tfrac{y}{x}\right)\otimes y\right)\sqcup y\sqcup\left(\left(1-\tfrac{1}{z}\right)\otimes z\right)$$
$$-\left(1-\tfrac{1}{y}\right)\otimes\left(\left(1-\tfrac{y}{x}\right)\otimes y\right)\sqcup\left(\left(1-\tfrac{1}{z}\right)\otimes z\sqcup z\right)$$

$$-\left(1-\tfrac{1}{z}\right)\otimes z\sqcup\Big(\left(1-\tfrac{z}{x}\right)\otimes x\sqcup\Big(\left(1-\tfrac{z}{y}\right)\otimes y\Big)\Big) \qquad \text{(Tree 6)}$$
$$+\left(1-\tfrac{1}{z}\right)\otimes z\sqcup\Big(\left(1-\tfrac{z}{x}\right)\otimes x\sqcup\Big(\left(1-\tfrac{z}{y}\right)\otimes z\Big)\Big)$$
$$+\left(1-\tfrac{1}{z}\right)\otimes z\sqcup\Big(\left(1-\tfrac{z}{x}\right)\otimes z\otimes\left(1-\tfrac{z}{y}\right)\otimes y\Big)$$
$$-\left(1-\tfrac{1}{z}\right)\otimes z\sqcup\Big(\left(1-\tfrac{z}{x}\right)\otimes z\otimes\left(1-\tfrac{z}{y}\right)\otimes z\Big)$$
$$+\left(1-\tfrac{1}{z}\right)\otimes z\sqcup\Big(\left(1-\tfrac{z}{x}\right)\otimes\left(1-\tfrac{z}{y}\right)\otimes y\sqcup y\Big)$$
$$-\left(1-\tfrac{1}{z}\right)\otimes z\sqcup\Big(\left(1-\tfrac{z}{x}\right)\otimes\left(1-\tfrac{z}{y}\right)\otimes y\sqcup z\Big)$$

$$\begin{aligned}
&+\left(1-\tfrac{1}{x}\right)\otimes\left(\left(1-\tfrac{1}{z}\right)\otimes z\right)\sqcup\!\sqcup\left(x\otimes\left(1-\tfrac{x}{y}\right)\otimes x\right) && \text{(Tree 7)}\\
&+\left(1-\tfrac{1}{x}\right)\otimes\left(\left(1-\tfrac{1}{z}\right)\otimes z\right)\sqcup\!\sqcup\left(\left(1-\tfrac{x}{y}\right)\otimes x\sqcup\!\sqcup y\right)\\
&-\left(1-\tfrac{1}{x}\right)\otimes\left(\left(1-\tfrac{1}{z}\right)\otimes z\sqcup\!\sqcup z\right)\sqcup\!\sqcup\left(\left(1-\tfrac{x}{y}\right)\otimes x\right)\\
&-\left(1-\tfrac{1}{x}\right)\otimes\left(\left(1-\tfrac{1}{z}\right)\otimes z\right)\sqcup\!\sqcup\left(x\otimes\left(1-\tfrac{x}{y}\right)\otimes y\right)\\
&-\left(1-\tfrac{1}{x}\right)\otimes\left(\left(1-\tfrac{1}{z}\right)\otimes z\right)\sqcup\!\sqcup\left(\left(1-\tfrac{x}{y}\right)\otimes y\sqcup\!\sqcup y\right)\\
&+\left(1-\tfrac{1}{x}\right)\otimes\left(\left(1-\tfrac{1}{z}\right)\otimes z\sqcup\!\sqcup z\right)\sqcup\!\sqcup\left(\left(1-\tfrac{x}{y}\right)\otimes y\right)
\end{aligned}$$

$$\begin{aligned}
&+\left(1-\tfrac{1}{x}\right)\otimes x\otimes\left(1-\tfrac{x}{z}\right)\otimes x\sqcup\!\sqcup\left(\left(1-\tfrac{z}{y}\right)\otimes y\right) && \text{(Tree 8)}\\
&-\left(1-\tfrac{1}{x}\right)\otimes x\otimes\left(1-\tfrac{x}{z}\right)\otimes z\otimes\left(1-\tfrac{z}{y}\right)\otimes y\\
&-\left(1-\tfrac{1}{x}\right)\otimes x\otimes\left(1-\tfrac{x}{z}\right)\otimes\left(1-\tfrac{z}{y}\right)\otimes y\sqcup\!\sqcup y\\
&+\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{z}\right)\otimes x\sqcup\!\sqcup z\sqcup\!\sqcup\left(\left(1-\tfrac{z}{y}\right)\otimes y\right)\\
&-\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{z}\right)\otimes z\sqcup\!\sqcup\left(z\otimes\left(1-\tfrac{z}{y}\right)\otimes y\right)\\
&-\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{z}\right)\otimes z\sqcup\!\sqcup\left(\left(1-\tfrac{z}{y}\right)\otimes y\sqcup\!\sqcup y\right)\\
&-\left(1-\tfrac{1}{x}\right)\otimes x\otimes\left(1-\tfrac{x}{z}\right)\otimes x\sqcup\!\sqcup\left(\left(1-\tfrac{z}{y}\right)\otimes z\right)\\
&+\left(1-\tfrac{1}{x}\right)\otimes x\otimes\left(1-\tfrac{x}{z}\right)\otimes z\otimes\left(1-\tfrac{z}{y}\right)\otimes z\\
&+\left(1-\tfrac{1}{x}\right)\otimes x\otimes\left(1-\tfrac{x}{z}\right)\otimes\left(1-\tfrac{z}{y}\right)\otimes z\sqcup\!\sqcup y\\
&-\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{z}\right)\otimes x\sqcup\!\sqcup z\sqcup\!\sqcup\left(\left(1-\tfrac{z}{y}\right)\otimes z\right)\\
&+\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{z}\right)\otimes z\sqcup\!\sqcup\left(z\otimes\left(1-\tfrac{z}{y}\right)\otimes z\right)\\
&+\left(1-\tfrac{1}{x}\right)\otimes\left(1-\tfrac{x}{z}\right)\otimes z\sqcup\!\sqcup\left(\left(1-\tfrac{z}{y}\right)\otimes y\sqcup\!\sqcup z\right)
\end{aligned}$$

$$\begin{aligned}
&+\left(1-\tfrac{1}{y}\right)\otimes\left(\left(1-\tfrac{y}{x}\right)\otimes x\right)\sqcup\!\sqcup\left(y\otimes\left(1-\tfrac{y}{z}\right)\otimes y\right) && \text{(Tree 9)}\\
&+\left(1-\tfrac{1}{y}\right)\otimes\left(\left(1-\tfrac{y}{x}\right)\otimes x\right)\sqcup\!\sqcup\left(\left(1-\tfrac{y}{z}\right)\otimes z\sqcup\!\sqcup y\right)\\
&-\left(1-\tfrac{1}{y}\right)\otimes\left(\left(1-\tfrac{y}{x}\right)\otimes x\right)\sqcup\!\sqcup\left(y\otimes\left(1-\tfrac{y}{z}\right)\otimes z\right)\\
&-\left(1-\tfrac{1}{y}\right)\otimes\left(\left(1-\tfrac{y}{x}\right)\otimes x\right)\sqcup\!\sqcup\left(\left(1-\tfrac{y}{z}\right)\otimes z\sqcup\!\sqcup z\right)\\
&-\left(1-\tfrac{1}{y}\right)\otimes\left(\left(1-\tfrac{y}{x}\right)\otimes y\right)\sqcup\!\sqcup\left(y\otimes\left(1-\tfrac{y}{z}\right)\otimes y\right)\\
&-\left(1-\tfrac{1}{y}\right)\otimes\left(\left(1-\tfrac{y}{x}\right)\otimes y\right)\sqcup\!\sqcup\left(\left(1-\tfrac{y}{z}\right)\otimes z\sqcup\!\sqcup y\right)\\
&+\left(1-\tfrac{1}{y}\right)\otimes\left(\left(1-\tfrac{y}{x}\right)\otimes y\right)\sqcup\!\sqcup\left(y\otimes\left(1-\tfrac{y}{z}\right)\otimes z\right)\\
&+\left(1-\tfrac{1}{y}\right)\otimes\left(\left(1-\tfrac{y}{x}\right)\otimes y\right)\sqcup\!\sqcup\left(\left(1-\tfrac{y}{z}\right)\otimes z\sqcup\!\sqcup z\right)
\end{aligned}$$

$$
\begin{aligned}
&-\big(1-\tfrac{1}{x}\big)\otimes x \sqcup\!\sqcup \Big(\big(1-\tfrac{1}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{z}{y}\big)\otimes y\Big)\Big) && \text{(Tree 10)}\\
&+\big(1-\tfrac{1}{x}\big)\otimes\big(1-\tfrac{1}{z}\big)\otimes z \sqcup\!\sqcup \Big(z\otimes\big(1-\tfrac{z}{y}\big)\otimes y\Big)\\
&+\big(1-\tfrac{1}{x}\big)\otimes\big(1-\tfrac{1}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{z}{y}\big)\otimes y \sqcup\!\sqcup y\Big)\\
&+\big(1-\tfrac{1}{x}\big)\otimes x \sqcup\!\sqcup \Big(\big(1-\tfrac{1}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{z}{y}\big)\otimes z\Big)\Big)\\
&-\big(1-\tfrac{1}{x}\big)\otimes\big(1-\tfrac{1}{z}\big)\otimes z \sqcup\!\sqcup \Big(z\otimes\big(1-\tfrac{z}{y}\big)\otimes z\Big)\\
&-\big(1-\tfrac{1}{x}\big)\otimes\big(1-\tfrac{1}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{z}{y}\big)\otimes y \sqcup\!\sqcup z\Big)
\end{aligned}
$$

$$
\begin{aligned}
&+\big(1-\tfrac{1}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{z}{x}\big)\otimes\big(1-\tfrac{x}{y}\big)\otimes x \sqcup\!\sqcup y\Big) && \text{(Tree 11)}\\
&+\big(1-\tfrac{1}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{z}{x}\big)\otimes x\otimes\big(1-\tfrac{x}{y}\big)\otimes x\Big)\\
&-\big(1-\tfrac{1}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{z}{x}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{x}{y}\big)\otimes x\Big)\Big)\\
&-\big(1-\tfrac{1}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{z}{x}\big)\otimes\big(1-\tfrac{x}{y}\big)\otimes y \sqcup\!\sqcup y\Big)\\
&-\big(1-\tfrac{1}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{z}{x}\big)\otimes x\otimes\big(1-\tfrac{x}{y}\big)\otimes y\Big)\\
&+\big(1-\tfrac{1}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{z}{x}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{x}{y}\big)\otimes y\Big)\Big)
\end{aligned}
$$

$$
\begin{aligned}
&-\big(1-\tfrac{1}{x}\big)\otimes x\otimes\big(1-\tfrac{x}{z}\big)\otimes\big(1-\tfrac{x}{y}\big)\otimes y \sqcup\!\sqcup x && \text{(Tree 12)}\\
&-\big(1-\tfrac{1}{x}\big)\otimes x\otimes\big(1-\tfrac{x}{z}\big)\otimes x\otimes\big(1-\tfrac{x}{y}\big)\otimes x\\
&+\big(1-\tfrac{1}{x}\big)\otimes x\otimes\big(1-\tfrac{x}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{x}{y}\big)\otimes x\Big)\\
&-\big(1-\tfrac{1}{x}\big)\otimes\big(1-\tfrac{x}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{x}{y}\big)\otimes y \sqcup\!\sqcup x\Big)\\
&+\big(1-\tfrac{1}{x}\big)\otimes\big(1-\tfrac{x}{z}\big)\otimes z \sqcup\!\sqcup \Big(x\otimes\big(1-\tfrac{x}{y}\big) \sqcup\!\sqcup x\Big)\\
&+\big(1-\tfrac{1}{x}\big)\otimes\big(1-\tfrac{x}{z}\big)\otimes z \sqcup\!\sqcup z \sqcup\!\sqcup \Big(\big(1-\tfrac{x}{y}\big) \sqcup\!\sqcup x\Big)\\
&+\big(1-\tfrac{1}{x}\big)\otimes x\otimes\big(1-\tfrac{x}{z}\big)\otimes\big(1-\tfrac{x}{y}\big)\otimes y \sqcup\!\sqcup y\\
&+\big(1-\tfrac{1}{x}\big)\otimes x\otimes\big(1-\tfrac{x}{z}\big)\otimes x\otimes\big(1-\tfrac{x}{y}\big)\otimes y\\
&-\big(1-\tfrac{1}{x}\big)\otimes x\otimes\big(1-\tfrac{x}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{x}{y}\big)\otimes y\Big)\\
&+\big(1-\tfrac{1}{x}\big)\otimes\big(1-\tfrac{x}{z}\big)\otimes z \sqcup\!\sqcup \Big(\big(1-\tfrac{x}{y}\big)\otimes y \sqcup\!\sqcup y\Big)\\
&-\big(1-\tfrac{1}{x}\big)\otimes\big(1-\tfrac{x}{z}\big)\otimes z \sqcup\!\sqcup \Big(x\otimes\big(1-\tfrac{x}{y}\big) \sqcup\!\sqcup y\Big)\\
&-\big(1-\tfrac{1}{x}\big)\otimes\big(1-\tfrac{x}{z}\big)\otimes z \sqcup\!\sqcup z \sqcup\!\sqcup \Big(\big(1-\tfrac{x}{y}\big) \sqcup\!\sqcup y\Big).
\end{aligned}
$$