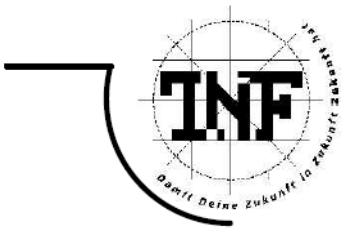
JOHANNES KEPLER
UNIVERSITÄT LINZ
Netzwerk für Forschung, Lehre und Praxis



Contributions to MacMahon's Partition Analysis

MASTERARBEIT

zur Erlangung des akademischen Grades

DIPLOM-INGENIEURIN

im Masterstudium

COMPUTERMATHEMATIK

Angefertigt am *Institut für Symbolisches Rechnen (RISC)*

Betreuung:

Univ.-Prof. Dr. Peter Paule

Eingereicht von:

Manuela Wiesinger

Linz, Jänner 2009

Support notice: This thesis was supported by SFB grant F1305 of the Austrian FWF.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Masterarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Linz, ... Jänner 2009 .....

Unterschrift

Acknowledgements

I foremost want to thank Prof. Peter Paule who awakened my interest in combinatorics and supervised this thesis. With his cheerful nature he always managed to encourage me. He never got tired of answering my questions.

I also want to thank the lecturers at JKU who tought me so much, particularly the lecturers at the Institute of Algebra and RISC.

And, finally, I want to thank my family and friends for their enduring support.

Abstract

About a century ago P. A. MacMahon introduced Partition Analysis as a method for solving problems in connection with systems of linear Diophantine inequalities and equations. Due to its arithmetical complexity the method lay dorment until G. E. Andrews, P. Paule and A. Riese discovered that it could be implemented in today's computer algebra systems and presented their `Omega` package written in Mathematica. Inspired by their work, Guo-Niu Han was able to derive an algorithm for more general expressions than those covered with the `Omega` package, which resulted in his `GenOmega` package written in Maple. In this thesis we present an account of these accomplishments and introduce the new `GenOmega` package which was implemented in the course of the thesis. This new `GenOmega` package is written in Mathematica and is also based on Han's algorithm but is more general than the Maple version.

Zusammenfassung

Vor etwa einem Jahrhundert stellte P. A. MacMahon Partition Analysis als eine Methode vor, Probleme im Zusammenhang mit linearen diophantischen Gleichungs- und Ungleichungssystemen zu lösen. Aufgrund ihres rechnerischen Aufwandes lag die Methode brach, bis G. E. Andrews, P. Paule und A. Riese entdeckten, dass sie in heutigen Computeralgebra-Systemen implementiert werden konnte, und ihr Mathematica-Package `Omega` präsentierten. Inspiriert durch ihre Arbeit entwickelte Guo-Niu Han einen Algorithmus, der allgemeinere Ausdrücke bewältigen konnte als der von Andrews et. al., was zu seinem Maple-Package `GenOmega` führte. In dieser Arbeit geben wir eine Darstellung dieser Leistungen und stellen das neue `GenOmega`-Package vor, das im Zuge dieser Arbeit implementiert wurde. Dieses neue `GenOmega`-Package wurde in Mathematica geschrieben und basiert ebenfalls auf Han's Algorithmus, ist allerdings noch allgemeiner als die Maple-Version.

Contents

Chapter 1

Introduction

1.1 Overview

In [22] MacMahon introduced Partition Analysis as a method for solving combinatorial problems in connection with systems of linear Diophantine inequalities and equations. Andrews gives a very good review of MacMahon's partition analysis. He writes [1, page 3]:

> "Our method of proof is the method of Partition Analysis first suggested by Cayley [15] but primarily developed by P. A. MacMahon in [19], [20], [21], surveyed by him again in [23, Vol. 2, Section VIII, 91–170] and recapped in his Collected Papers [24, Ch. 10, pp. 1119–1314].
>
> A careful examination of MacMahon's work shows that MacMahon developed a magnificent machine which according to MacMahon's own admission [22, Vol. 2, p. 187] failed to prove the major theorems on plane partitions. Very much to his credit, MacMahon developed alternative tools necessary to prove his plane partition discoveries. However that admission of the failure of Partition Analysis to prove general results on plane partitions led to near total neglect of the method. The single major exception to this comment is the beautiful solution to the Anand-Dumir-Gupta Conjecture by Richard Stanley [25]. However, since 1973 there has been (to my knowledge) no further work on Partition Analysis."

He continues [1, pp. 4–6]:

"Since MacMahon's method has not been used for some time, we provide a general review here. Perhaps we best begin with his own words [22, vol. 2, pp. 91–92].

'Consider i numbers in descending order of magnitude

$$\alpha_1,\ \alpha_2,\ \alpha_3, \ldots, \alpha_i.$$

The order is defined by the Diophantine relations

$$\begin{aligned} \alpha_1 &\geq \alpha_2, \\ \alpha_2 &\geq \alpha_3, \\ &\vdots \\ \alpha_{i-1} &\geq \alpha_i, \end{aligned}$$

and subject to them we consider the sum

$$\sum x^{\alpha_1+\alpha_2+\alpha_3+\cdots+\alpha_i}.$$

Now observe that the algebraic fraction

$$\frac{1}{(1-\lambda_1 x)\,(1-\frac{\lambda_2}{\lambda_1} x)\,(1-\frac{\lambda_3}{\lambda_2} x)\,\cdots\,(1-\frac{\lambda_i}{\lambda_{i-1}} x)},$$

when expanded in ascending powers of x, has the general term

$$(\lambda_1)^{\alpha_1} \left(\frac{\lambda_2}{\lambda_1}\right)^{\alpha_2} \left(\frac{\lambda_3}{\lambda_2}\right)^{\alpha_3} \cdots \left(\frac{\lambda_i}{\lambda_{i-1}}\right)^{\alpha_i} x^{\alpha_1+\alpha_2+\alpha_3+\cdots+\alpha_i},$$

or

$$\lambda_1^{\alpha_1-\alpha_2}\, \lambda_2^{\alpha_2-\alpha_3}\, \cdots\, \lambda_{i-1}^{\alpha_{i-1}-\alpha_i}\, \lambda_i^{\alpha_i}\, x^{\alpha_1+\alpha_2+\alpha_3+\cdots+\alpha_i},$$

and that if the Diophantine relations are to be satisfied this must be free from negative powers of λ_1, λ_2, λ_3, ...

It is thus evident that we have only to expand the fraction

$$\frac{1}{(1-\lambda_1 x)\,(1-\frac{\lambda_2}{\lambda_1} x)\,(1-\frac{\lambda_3}{\lambda_2} x)\,\cdots\,(1-\frac{\lambda_i}{\lambda_{i-1}} x)},$$

and reject all the terms involving negative powers of λ_1, λ_2, λ_3, ..., and subsequently put

$$\lambda_1 = \lambda_2 = \lambda_3 = \cdots = \lambda_i = 1$$

to obtain the desired sum

$$\sum x^{\alpha_1+\alpha_2+\alpha_3+\cdots+\alpha_i}.$$

The performance of these operations upon the fraction we shall denote by prefixing the symbol $\Omega_{\geq}$, so that

$$\sum x^{\alpha_1+\alpha_2+\alpha_3+\cdots+\alpha_i} = \Omega_{\geq} \frac{1}{(1-\lambda_1 x)\,(1-\frac{\lambda_2}{\lambda_1} x)\,(1-\frac{\lambda_3}{\lambda_2} x)\cdots(1-\frac{\lambda_i}{\lambda_{i-1}} x)}.'$$

More generally

$$\Omega_{\geq} \sum_{s_1=-\infty}^{\infty} \cdots \sum_{s_r=-\infty}^{\infty} A_{s_1,s_2,\ldots,s_r} \lambda_1^{s_1} \lambda_2^{s_2} \ldots \lambda_r^{s_r} := \sum_{s_1=0}^{\infty} \cdots \sum_{s_r=0}^{\infty} A_{s_1,s_2,\ldots,s_r},$$

where $A_{s_1,s_2,\ldots,s_r}$ may be functions of several complex variables.

Let us now return to MacMahon. He notes that to make effective use of his method, one requires a number of simple identities for the Ω symbol. As he remarks [22, Vol. 2 p. 102].

'We may add conveniently the easily verifiable results

$$\Omega_{\geq} \frac{1}{(1-\lambda x)(1-\frac{y}{\lambda})(1-\frac{z}{\lambda})} = \frac{1}{(1-x)(1-xy)(1-xz)},$$

$$\Omega_{\geq} \frac{1}{(1-\lambda x)(1-\lambda y)(1-\frac{z}{\lambda})} = \frac{1-xyz}{(1-x)(1-y)(1-xz)(1-yz)},$$

$$\Omega_{\geq} \frac{1}{(1-\lambda x)(1-\frac{y}{\lambda^2})} = \frac{1}{(1-x)(1-x^2y)},$$

$$\Omega_{\geq} \frac{1}{(1-\lambda^2 x)(1-\frac{y}{\lambda})} = \frac{1+xy}{(1-x)(1-xy^2)},$$

$$\Omega_{\geq} \frac{1}{(1-\lambda x)(1-\lambda y)(1-\frac{z}{\lambda^2})} = \frac{1+xyz-x^2yz-xy^2z}{(1-x)(1-y)(1-x^2z)(1-y^2z)},$$

$$\Omega_{\geq} \frac{1}{(1-\lambda x)(1-\frac{y}{\lambda^s})} = \frac{1}{(1-x)(1-x^sy)},$$

$$\Omega_{\geq} \frac{1}{(1-\lambda^s x)(1-\frac{y}{\lambda})} = \frac{1+xy\frac{1-y^{s-1}}{1-y}}{(1-x)(1-xy^s)}.''$$

We have to remark that s in the above identities is a nonnegative integer.

To get a better feeling for what is happening we will prove the third identity:

$$\begin{aligned}\Omega_{\geq}\frac{1}{(1-\lambda x)(1-\frac{y}{\lambda^2})} &= \Omega_{\geq}\sum_{i,j\geq 0} x^i\lambda^{i-2j}y^j = \Omega_{\geq}\sum_{k,j\geq 0} x^{k+2j}\lambda^k y^j \\ &= \sum_{k,j\geq 0} x^{k+2j}y^j = \frac{1}{(1-x)(1-x^2y)}.\end{aligned}$$

There are two main advantages of Partition Analysis:

- One is not required to have a detailed understandig of the underlying theory of the problem. Partition Analysis yields a solution quite automatically.
- The problem can be considered in full generality so that possible refinements of the problem present themselves automatically.

After the above cited paper followed a series of other papers [2–12] where Andrews, P. Paule and A. Riese managed to implement the method in Mathematica for expressions of the form

$$\Omega_{\geq}\frac{\lambda^a}{(1-x_1\lambda^{j_1})\cdots(1-x_n\lambda^{j_n})(1-\frac{y_1}{\lambda^{k_1}})\cdots(1-\frac{y_m}{\lambda^{k_m}})},$$

where n and m are nonnegative integers, the j_i and k_i are positive integers and a is any integer, and where they managed to derive many results in the field of combinatorics.

Based on their work, Guo-Niu Han was able to derive an algorithm for more general expressions of the form

$$\Omega_{\geq}\frac{U(\lambda)}{A(\lambda)B(1/\lambda)},$$

where $U(\lambda)$ is a Laurent polynomial in λ and $A(\lambda)$ and $B(\lambda)$ are polynomials in λ, which led to the Maple package `GenOmega`.

In the course of this thesis a more general version of `GenOmega` was implemented in Mathematica.

In the next chapter a more detailed introduction to the method of Partition Analysis will be given. We will go through Axel Riese's `Omega` package, Han's `GenOmega` package, the software `LattE` and its successor package `LattE macchiato`.

In Chapter 3 we will look at Guo-Niu Han's general algorithm in detail.

In Chapter 4 we will introduce the new `GenOmega` package written in Mathematica and prove new identities which could not be derived with the `Omega` package.

1.2 Notation

The symbols $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$ and $\mathbb{C}$ stand for the sets of nonnegative integers, all integers, rational, real and complex numbers. We denote the set of natural numbers without 0 by $\mathbb{N}^*$.

Let $R \supseteq \mathbb{Q}$ be a ring. We denote the ring of polynomials over R with indeterminate x by $R[x]$ and we denote the ring of formal power series over R with indeterminate x by $R[[x]]$. In the multivariate case with indeterminates $x_1, \ldots, x_n$ we write $R[x_1, \ldots, x_n]$ and $R[[x_1, \ldots, x_n]]$, respectively. We will not distinguish between polynomials and polynomial functions.

Let K be a field and $x \notin K$. We denote the smallest field extension of K containing x by $K(\!(x)\!)$. In the multivariate case with elements $x_1, \ldots, x_n \notin K$ we write $K(\!(x_1, \ldots, x_n)\!)$.

Let R be a ring and $p \in R[t, t^{-1}]$ be a Laurent polynomial in t. The low degree $\mathrm{ldeg}(p)$ of p in t is defined to be the degree of $p(1/t)$ in t. If we want to emphasize that we mean the degree or low degree with respect to t, we write $\deg_t(p)$ or $\mathrm{ldeg}_t(p)$, respectively.

Let K be a field and $f = \sum_{i=0}^{\infty} a_i x^i \in K[[x]]$. Then we denote for all $k \in \mathbb{N}$ the k-th coefficient of f with respect to x by $\langle x^k \rangle(f)$, i.e.

$$\langle x^k \rangle(f) := a_k.$$

For a matrix

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$$

we denote its determinant by

$$\det(A) = \begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{vmatrix}.$$

Chapter 2

Partition Analysis

2.1 Introduction

In [22] MacMahon introduced Partition Analysis as a method for solving combinatorial problems in connection with systems of linear Diophantine inequalities and equations. To this end, he defined the Omega operator $\Omega_{\geq;\,\lambda_1,\ldots,\lambda_r}$ for $r \in \mathbb{N}^*$ informally as follows:

$$\Omega_{\geq;\,\lambda_1,\ldots,\lambda_r} \sum_{n_1=-\infty}^{\infty} \cdots \sum_{n_r=-\infty}^{\infty} A(n_1,n_2,\ldots,n_r)\lambda_1^{n_1}\lambda_2^{n_2}\ldots\lambda_r^{n_r} := \sum_{n_1=0}^{\infty} \cdots \sum_{n_r=0}^{\infty} A(n_1,n_2,\ldots,n_r), \tag{2.1}$$

where $A(n_1,n_2,\ldots,n_r)$ is generally some rational function of variables like x, y or z. In short, for every nonnegative exponent n_i the corresponding λ_i is set to 1 and for every negative exponent n_i the λ_i is set to 0 by the Omega operator. If $r = 1$, we also use the notation $\Omega_{\geq,\,\lambda}$ instead of $\Omega_{\geq;\,\lambda_1}$.

For a better understanding we consider some examples which are taken from [6] and [9].

Problem 2.1.1 *Find all nonnegative integer solutions a, b to the inequality $2a \geq 3b$.*

The corresponding generating function is the following

$$f(x,y) = \sum_{\substack{a,\,b\geq 0\\ 2a\geq 3b}} x^a y^b.$$

In order to get rid of the inequality constraints we make use of the Omega operator and rewrite the generating function to

$$f(x,y) = \Omega_{\geq,\lambda} \sum_{a,\,b\geq 0} \lambda^{2a-3b} x^a y^b.$$

By definition of the Omega operator exactly those summands $x^a y^b$ remain for which $2a - 3b \geq 0$. Now by geometric series expansion we obtain

$$f(x,y) = \Omega_{\geq,\lambda} \frac{1}{(1-x\lambda^2)(1-\frac{y}{\lambda^3})}.$$

MacMahon calls this the *crude generating function*. Now we have to eliminate λ. To this end, MacMahon gives a catalog [22, Vol. II, pp. 102-103] of fundamental evaluations of the Omega operator. One of these evaluations is the following,

$$\Omega_{\geq,\lambda} \frac{1}{(1-x\lambda)(1-\frac{y}{\lambda^s})} = \frac{1}{(1-x)(1-x^s y)}, \tag{2.2}$$

where s is a nonnegative integer. By partial fraction decomposition (assuming x and y to be positive real numbers), we get

$$f(x,y) = \frac{1}{2}\ \Omega_{\geq,\lambda} \frac{1}{(1-\sqrt{x}\lambda)(1-\frac{y}{\lambda^3})} + \frac{1}{2}\ \Omega_{\geq,\lambda} \frac{1}{(1+\sqrt{x}\lambda)(1-\frac{y}{\lambda^3})}.$$

Now we are able to apply rule (2.2):

$$\begin{aligned} f(x,y) &= \frac{1}{2}\ \frac{1}{(1-\sqrt{x})(1-\sqrt{x^3}y)} + \frac{1}{2}\ \frac{1}{(1+\sqrt{x})(1-\sqrt{x^3}y)} \\ &= \frac{1+x^2y}{(1-x)(1-x^3y^2)}. \end{aligned}$$

By geometric series expansion we obtain

$$f(x,y) = \sum_{\alpha,\,\beta\geq 0} x^{\alpha+3\beta} y^{2\beta} (1+x^2y)$$

from which we can deduce that

$$\{(a,b) \in \mathbb{N}^2 : 2a \geq 3b\} = \{(m+n+\lceil n/2 \rceil, n) : (m,n) \in \mathbb{N}^2\}.$$

Problem 2.1.2 *Let $t_3(n)$ be the number of non-congruent triangles (a_1, a_2, a_3) with perimiter n whose sides a_1, a_2 and a_3 have integer length. For example, $t_3(9) = 3$, corresponding to the triangles $(3,3,3)$, $(2,3,4)$ and $(1,4,4)$. Find $t_3(n)$ for general n.*

The corresponding generating function is

$$T_3(q) = \sum_{n\geq 3} t_3(n)q^n = \sum\nolimits^* q^{a_1+a_2+a_3},$$

where $\sum^*$ is the restricted sum over all positive integer triples (a_1, a_2, a_3) satisfying $a_1 \leq a_2 \leq a_3$ and $a_1 + a_2 > a_3$. We rewrite this to the crude generating function,

$$\begin{aligned} T_3(q) &= \Omega_{\geq;\,\lambda_1,\,\lambda_2,\,\lambda_3} \sum_{a_1\geq 1,\ a_2,\,a_3\geq 0} \lambda_1^{a_2-a_1}\lambda_2^{a_3-a_2}\lambda_3^{a_1+a_2-a_3-1}q^{a_1+a_2+a_3} \\ &= \Omega_{\geq;\,\lambda_1,\,\lambda_2,\,\lambda_3} \frac{q\lambda_1^{-1}}{(1-q\frac{\lambda_3}{\lambda_1})(1-q\frac{\lambda_1\lambda_3}{\lambda_2})(1-q\frac{\lambda_2}{\lambda_3})}. \end{aligned}$$

In order to eliminate the λ_i's we use the following rule, which can be proven easily by geometric series expansion and index substitution,

$$\Omega_{\geq,\,\lambda} \frac{\lambda^{-s}}{(1-\lambda A)(1-\frac{B}{\lambda})} = \frac{A^s}{(1-A)(1-AB)}, \tag{2.3}$$

where s is a nonnegative integer and the variables A and B are free of λ. Applying this rule to $T_3(q)$ we obtain by successive elimination of λ_2, λ_1 and λ_3

$$\begin{aligned} T_3(q) &= \Omega_{\geq;\,\lambda_1,\,\lambda_3} \frac{q\lambda_1^{-1}}{(1-q\frac{\lambda_3}{\lambda_1})(1-\frac{q}{\lambda_3})(1-q^2\lambda_1)} \\ &= \Omega_{\geq,\,\lambda_3} \frac{q^3}{(1-q^2)(1-q^3\lambda_3)(1-\frac{q}{\lambda_3})} \\ &= \frac{q^3}{(1-q^2)(1-q^3)(1-q^4)}. \end{aligned}$$

From this it is easily seen that $t_3(n)$ equals the number of partitions of $n-3$ into parts 2, 3 and 4, that is the cardinality of the set $\{(i,j,k) \in \mathbb{N}^3 : 2i+3j+4k = n-3\}$. For example, $t_3(9) = 3$ corresponds to the partitions $6 = 2+2+2 = 3+3 = 4+2$.

In order to be able to treat systems of linear Diophantine equalities in a similar fashion, MacMahon introduced a different Omega operator.

Definition 2.1.3 *The operator* $\Omega_{=;\,\lambda_1,\,\cdots,\,\lambda_r}$ *is given by*

$$\Omega_{=;\,\lambda_1,\,\cdots,\,\lambda_r} \sum_{n_1=-\infty}^{\infty} \cdots \sum_{n_r=-\infty}^{\infty} A(n_1, n_2, \ldots, n_r)\lambda_1^{n_1}\lambda_2^{n_2}\ldots\lambda_r^{n_r} := A(0, \ldots, 0).$$

This means, all non-trivial power-products in the λ_i's are eliminated by the $\Omega_{=;\lambda_1,\cdots,\lambda_r}$-operator. If $r=1$, we also use the notation $\Omega_{=,\lambda}$ instead of $\Omega_{=;\lambda_1}$.

We consider the following example which again is taken from [6].

Problem 2.1.4 *Find all nonnegative integer solutions a, b, c to the Diophantine equation* $2a = 3b + c$.

One can easily see that this problem is equivalent to Problem 2.1.1, but nevertheless we show how to solve this with the Omega operator.

As in the former examples we first translate the problem into the corresponding crude generating function,

$$\begin{aligned} g(x,y,z) &:= \sum_{\substack{a,b,c\geq 0\\ 2a=3b+c}} x^a y^b z^c = \Omega_{=,\lambda} \sum_{a,b,c\geq 0} \lambda^{2a-3b-c} x^a y^b z^c \\ &= \Omega_{=,\lambda} \frac{1}{(1-x\lambda^2)(1-\frac{y}{\lambda^3})(1-\frac{z}{\lambda})}. \end{aligned}$$

In order to eliminate λ we apply a method which MacMahon called *The Method of Elliott*; see [22, Vol. 2, Section VIII, pp. 111-114].

Theorem 2.1.5 (Elliott Reduction) *For positive integers j and k,*

$$\frac{1}{(1-x\lambda^j)(1-y\lambda^{-k})} = \frac{1}{1-xy\lambda^{j-k}} \left(\frac{1}{1-x\lambda^j} + \frac{1}{1-y\lambda^{-k}} - 1 \right).$$

Applying this theorem to the first two factors $\frac{1}{1-x\lambda^2}$ and $\frac{1}{1-\frac{y}{\lambda^3}}$ of the crude generating function results in

$$g(x,y,z) = \Omega_{=,\lambda} \frac{1}{1-\frac{xy}{\lambda}} \left(\frac{1}{1-x\lambda^2} + \frac{1}{1-\frac{y}{\lambda^3}} - 1 \right) \frac{1}{1-\frac{z}{\lambda}}.$$

After expanding this expression we see that the two summands obtained from $\frac{1}{1-\frac{y}{\lambda^3}}$ and -1 involve only negative powers of λ in the factors of the denominator which are reduced to 1 by the $\Omega_{=,\lambda}$-operator. Hence we obtain

$$g(x,y,z) = \Omega_{=,\lambda} \frac{1}{(1-x\lambda^2)(1-\frac{xy}{\lambda})(1-\frac{z}{\lambda})}.$$

Now we make use of the following fact [22, Vol. 2, Section VIII, p. 105]:

$$\Omega_{=,\lambda} \frac{1}{(1-\lambda^2 x)(1-\frac{y}{\lambda})(1-\frac{z}{\lambda})} = \frac{1+xyz}{(1-xy^2)(1-xz^2)}.$$

With this we obtain

$$g(x,y,z) = \frac{1 + x^2yz}{(1 - x^3y^2)(1 - xz^2)},$$

which is the desired form of the generating function corresponding to our problem.

In the next section we will introduce some software packages that do these computations automatically.

2.2 Software

A hundred years ago MacMahon was forced to do all these computations by hand with the help of his catalog of fundamental evaluations. Nowadays we have computer algebra systems to assist us. There are several software packages that help us solve the kind of problems introduced in the last section. Some are based on the work of MacMahon, others use a different approach. In the following three subsections we are going to introduce some of them.

2.2.1 The Omega Package

Andrews et al. sought to implement MacMahon's method in the computer algebra system Mathematica which resulted in a series of papers [3]– [12] and the development of the package `Omega2` (see [6])—a much improved version of the package `Omega` described in [3].

Instead of the informal definition given by (2.1) we will use their definition of the Omega operator in the context of `Omega2` applications; see for example [6].

Definition 2.2.1 *The operator* $\Omega_{\geq;\,\lambda_1,\ldots,\lambda_r}$ *with* $r \in \mathbb{N}^*$ *is given by*

$$\Omega_{\geq;\,\lambda_1,\ldots,\lambda_r} \sum_{s_1=-\infty}^{\infty} \cdots \sum_{s_r=-\infty}^{\infty} A_{s_1,\,s_2,\,\ldots,\,s_r} \lambda_1^{s_1} \lambda_2^{s_2} \ldots \lambda_r^{s_r} := \sum_{s_1=0}^{\infty} \cdots \sum_{s_r=0}^{\infty} A_{s_1,\,s_2,\,\ldots,\,s_r},$$

where the domain of the $A_{s_1,\,s_2,\,\ldots,\,s_r}$ *is the field of rational functions over* $\mathbb{C}$ *in several complex variables and the* λ_i *are restricted to a neighborhood of the*

circle $|\lambda_i| = 1$. *In addition, the* $A_{s_1,s_2,\ldots,s_r}$ *are required to be such that any of the* $2^r - 1$ *sums*

$$\sum_{s_{i_1}=-\infty}^{\infty} \cdots \sum_{s_{i_j}=-\infty}^{\infty} A_{s_1,s_2,\ldots,s_r}$$

is absolute convergent within the domain of the definition of $A_{s_1,s_2,\ldots,s_r}$.

The authors emphasize that it is essential to treat everything analytically rather than formally, because the method relies on unique Laurent series representations of rational functions. Again, for $r = 1$ we write $\Omega_{\geq,\lambda}$ instead of $\Omega_{\geq;\lambda_1}$.

Remark 2.2.2 Later, in the context of `GenOmega` applications, we will use a slightly different definition of the Omega operator; see Chapters 3 and 4.

In [6] the authors studied the expression

$$\Omega_{\geq\lambda} \frac{\lambda^a}{(1-x_1\lambda^{j_1})\cdots(1-x_n\lambda^{j_n})(1-\frac{y_1}{\lambda^{k_1}})\cdots(1-\frac{y_m}{\lambda^{k_m}})},$$

where n and m are nonnegative integers, the j_i and k_i are positive integers and a is any integer, and constructed an algorithm based on a generalized recurrence which led to the `Omega2` package.

The Mathematica package `Omega2` is freely available at [28]. After loading the package by typing `<<Omega2.m` the following functions are available:

`Omega[f,{t1,...,tr}]` denotes the Omega operator $\Omega_{\geq}$ acting on `f` with respect to the variables `t1`,...,`tr`.

`Omega[f,t]` denotes the Omega operator $\Omega_{\geq}$ acting on `f` with respect to the variable `t`.

`OmegaEq[f,{t1,...,tr}]` denotes the Omega operator $\Omega_{=}$ acting on `f` with respect to the variables `t1`,...,`tr`.

`OmegaEq[f,t]` denotes the Omega operator $\Omega_{=}$ acting on `f` with respect to the variable `t`.

`OSum[expr,{ineq1,...,ineqr},z]` sets up the crude generating function of `expr` with summation variables satisfying the ineqalities `ineq1`,...,`ineqr` and Omega variables `z1`,...,`zr`. For each inequality the procedure introduces an Omega variable `zi`. The variable `z` is a Mathematica symbol

and `expr` is a power product with Mathematica symbols as basis and the summation variables as exponents. Unless specified explicitly by the inequalities, all summation variables are assumed to range over the nonnegative integers. The output is of the form `Omega[f,{`$\mathtt{z}_1$`,...,`$\mathtt{z}_r$`}]`. To evaluate the result, apply the procedure `OR`.

`OEqSum[expr,{`$\mathtt{eq}_1$`,...,`$\mathtt{eq}_k$`,`$\mathtt{ineq}_1$`,... `$\mathtt{ineq}_r$`},z]` sets up the crude generating function of `expr` with summation variables satisfying the equalities $\mathtt{eq}_1$,...,$\mathtt{eq}_k$ and ineqalities $\mathtt{ineq}_1$,...,$\mathtt{ineq}_r$, and with Omega variables $\mathtt{z}_1$,...,$\mathtt{z}_{(k+r)}$. For each equality and inequality the procedure introduces an Omega variable $\mathtt{z}_i$. The variable `z` is a Mathematica symbol and `expr` is a power product with Mathematica symbols as basis and the summation variables as exponents. Unless specified explicitly by the $\mathtt{ineq}_i$, all summation variables are assumed to range over the nonnegative integers. The output is of the form `OmegaEq[f,{`$\mathtt{z}_1$`,...,`$\mathtt{z}_s$`}]` with $s := k + r$. To evaluate the result, apply the procedure `OEqR`.

`OR[expr,l]` **or** `OR[Omega[expr,l]]` eliminates the Omega variable `l` from the Omega expression `expr` which is of the form

$$\frac{LP(l)}{(1 \pm pp_1(l)) \cdots (1 \pm pp_d(l))},$$

where $LP(l)$ is a Laurent polynomial in `l` and the $pp_i(l)$ are power products in `l` and some other variables with integer exponents. This procedure corresponds to the $\Omega_{\geq}$ operator.

`OEqR[expr,l]` **or** `OEqR[OmegaEq[expr,l]]` eliminates the Omega variable `l` from the Omega expression `expr` which is of the form

$$\frac{LP(l)}{(1 \pm pp_1(l)) \cdots (1 \pm pp_d(l))},$$

where $LP(l)$ is a Laurent polynomial in `l` and the $pp_i(l)$ are power products in `l` and some other variables with integer exponents. This procedure corresponds to the $\Omega_{=}$ operator.

`OR[expr,{`$\mathtt{l}_1$`,...,`$\mathtt{l}_n$`}]` **or** `OR[Omega[expr,{`$\mathtt{l}_1$`,...,`$\mathtt{l}_n$`}]]` eliminates several variables $\mathtt{l}_1$,...,$\mathtt{l}_n$ from `expr` in one stroke. Here, `expr` must be of the form

$$\frac{LP(l_1, \ldots, l_n)}{(1 \pm pp_1(l_1, \ldots, l_n)) \cdots (1 \pm pp_d(l_1, \ldots, l_n))}.$$

This procedure corresponds to the $\Omega_{\geq}$ operator.

`OEqR[expr,{l1,...,ln}]` **or** `OEqR[OmegaEq[expr,{l1,...,ln}]]` eliminates several variables `l1,...,ln` from `expr` in one stroke. Again, `expr` must be of the form

$$\frac{LP(l_1,\ldots,l_n)}{(1 \pm pp_1(l_1,\ldots,l_n))\cdots(1 \pm pp_d(l_1,\ldots,l_n))}.$$

This procedure corresponds to the $\Omega_{=}$ operator.

Let us now solve the examples from the last section with `Omega2`.

Problem 2.1.1

We start with the solution of Problem 2.1.1 which read: *Find all nonnegative integer solutions a, b to the inequality* $2a \geq 3b$.

```
In[1]:= <<Omega2.m

Out[1]= Omega Package by Axel Riese (in cooperation with
        George E. Andrews and Peter Paule) - ©RISC Linz
        - V 2.48 (01/10/08))

In[2]:= OSum[x^a y^b, {2a≥3b}, λ]

        Assuming a ≥ 0
        Assuming b ≥ 0

Out[2]= Ω_{≥,λ1} 1/((1 - y/λ1^3)(1 - xλ1^2))

In[3]:= OR[%]

        Eliminating λ1...

Out[3]= (1 + x^2 y)/((1 - x)(1 - x^3 y^2))
```

Partition on Triangles

We continue with the solution of Problem 2.1.2 which read: *Let* $t_3(n)$ *be the number of non-congruent triangles with perimiter* n *whose sides have integer length. Find* $t_3(n)$ *for general* n.

```
In[4]:= OSum[q^(a1+a2+a3), {a3 ≥ a2, a2 ≥ a1, a1+a2 > a3, a1 ≥ 1}, λ]

        Assuming a2 ≥ 0
        Assuming a3 ≥ 0
```

$$\texttt{Out[4]=}\quad \Omega_{\geq,\lambda_1,\lambda_2,\lambda_3}\frac{q}{\lambda_2(1-\frac{q\lambda_1}{\lambda_3})(1-\frac{q\lambda_3}{\lambda_2})(1-\frac{q\lambda_2\lambda_3}{\lambda_1})}$$

```
In[5]:= OR[%]

        Eliminating λ2...
        Eliminating λ1...
        Eliminating λ3...
```

$$\texttt{Out[5]=}\quad \frac{q^3}{(1-q^2)(1-q^3)(1-q^4)}$$

With the help of the Omega package we can extend Problem 2.1.2 and generate all triples (a_1, a_2, a_3) where a_1, a_2 and a_3 are the lengths of the three sides of the triangles satisfying our conditions. That is, we are looking for the full generating function

$$S_3(x_1, x_2, x_3) = \sum\nolimits^* x_1^{a_1} x_2^{a_2} x_3^{a_3},$$

where $\sum^*$ denotes the restricted summation over all positive integer triples (a_1, a_2, a_3) satisfying $a_1 \leq a_2 \leq a_3$ and $a_1 + a_2 > a_3$.

```
In[6]:= OR[OSum[x1^a1 x2^a2 x3^a3, {a3 ≥ a2, a2 ≥ a1, a1+a2 > a3,
        a1 ≥ 1}, λ]]

        Assuming a2 ≥ 0
        Assuming a3 ≥ 0
        Eliminating λ2...
        Eliminating λ1...
        Eliminating λ3...
```

$$\texttt{Out[6]=}\quad \frac{x_1x_2x_3}{(1-x_2x_3)(1-x_1x_2x_3)(1-x_1x_2x_3^2)}$$

By geometric series expansion, namely

$$S_3(x_1, x_2, x_3) = \sum_{n_1,n_2,n_3\geq 0} x_1^{n_2+n_3+1} x_2^{n_1+n_2+n_3+1} x_3^{n_1+n_2+2n_3+1},$$

we obtain a representation for all non-congruent triangles with sides of integer length,

$$\{(n_2+n_3+1, n_1+n_2+n_3+1, n_1+n_2+2n_3+1) \in \mathbb{N}^3 : n_1, n_2, n_3 \in \mathbb{N}\}.$$

Problem 2.1.4

We continue with the solution of Problem 2.1.4 which was: *Find all non-negative integer solutions a, b, c to the Diophantine equation* $2a = 3b + c$.

```
In[7]:= OEqSum[x^a y^b z^c,{2a == 3b+c},λ]

        Assuming a ≥ 0
        Assuming b ≥ 0
        Assuming c ≥ 0
```

$$\texttt{Out[7]=}\quad \Omega_{=,\lambda_1} \frac{1}{(1-\frac{y}{\lambda_1^3})(1-\frac{z}{\lambda_1})(1-x\lambda_1^2)}$$

```
In[8]:= OEqR[%]

        Eliminating λ1 ...
```

$$\texttt{Out[8]=}\quad \frac{1+x^2yz}{(1-x^3y^2)(1-xz^2)}$$

Magic Squares

As a more involved application of the Omega operator for equality consider the following problem which was taken from [29].

Problem 2.2.3 *Let us consider magic squares of order 3 given by*

a_1	a_2	a_3
a_4	a_5	a_6
a_7	a_8	a_9

where $a_1, \ldots, a_9 \in \mathbb{N}$ *and all the sums of the rows, columns and diagonals, respectively, are equal, i.e.*

$$\begin{aligned} a_{10} &:= a_1+a_2+a_3 = a_4+a_5+a_6 = a_7+a_8+a_9 \\ &= a_1+a_4+a_7 = a_2+a_5+a_8 = a_3+a_6+a_9 \\ &= a_1+a_5+a_9 = a_3+a_5+a_7. \end{aligned}$$

How many such magic squares are there for fixed $a_{10} = n$*?*

First we set up the corresponding generating function. We immediately consider the general form where the a_i's are encoded separately.

$$f(x_1, \ldots, x_{10}) := \sum\nolimits^{*} x_1^{a_1} \cdots x_{10}^{a_{10}},$$

where $\sum^*$ is the restricted sum over all positive integer triples $(a_1, \ldots, a_{10})$ satisfying the above eight conditions. Then we use `OEqSum` to set up the corresponding crude generating function.

```
In[1]:= OEqSum[Product[x_i^a_i,{i,10}],
        {a1+a2+a3==a10, a4+a5+a6==a10, a7+a8+a9==a10
        a1+a4+a7==a10, a2+a5+a8==a10, a3+a6+a9==a10
        a1+a5+a9==a10, a3+a5+a7==a10,}, λ]

        Assuming a1 ≥ 0
        Assuming a2 ≥ 0
        Assuming a3 ≥ 0
        Assuming a4 ≥ 0
        Assuming a5 ≥ 0
        Assuming a6 ≥ 0
        Assuming a7 ≥ 0
        Assuming a8 ≥ 0
        Assuming a9 ≥ 0
        Assuming a10 ≥ 0
```

$$\begin{aligned}\texttt{Out[1]=}\quad & \Omega_{=,\lambda_1,\ldots,\lambda_8}\, 1/((1 - x_4\lambda_2\lambda_4)\,(1 - x_2\lambda_1\lambda_5)\,(1 - x_8\lambda_3\lambda_5) \\ & (1 - x_6\lambda_2\lambda_6)\,(1 - x_1\lambda_1\lambda_4\lambda_7)\,(1 - x_9\lambda_3\lambda_6\lambda_7) \\ & \left(1 - \tfrac{x_{10}}{\lambda_1\lambda_2\lambda_3\lambda_4\lambda_5\lambda_6\lambda_7\lambda_8}\right)\,(1 - x_3\lambda_3\lambda_4\lambda_8)\,(1 - x_3\lambda_1\lambda_6\lambda_8) \\ & (1 - x_5\lambda_2\lambda_5\lambda_7\lambda_8))\end{aligned}$$

Now we can invoke `OEqR`.

```
In[2]:= OEqR[%]

        Eliminating λ1 ...
        Eliminating λ8 ...
        Eliminating λ7 ...
        Eliminating λ6 ...
        Eliminating λ5 ...
        Eliminating λ4 ...
        Eliminating λ3 ...
        Eliminating λ2 ...
```

$$\begin{aligned}\texttt{Out[2]=}\quad & (1 + x_1\,x_2\,x_3\,x_4\,x_5\,x_6\,x_7\,x_8\,x_9\,x_{10}^3 - \\ & x_1^2\,x_2^2\,x_3^2\,x_4^2\,x_5^2\,x_6^2\,x_7^2\,x_8^2\,x_9^2\,x_{10}^6 - x_1^3\,x_2^3\,x_3^3\,x_4^3\,x_5^3\,x_6^3\,x_7^3\,x_8^3\,x_9^3\,x_{10}^9)/ \\ & ((1 - x_1^2\,x_3\,x_5\,x_6^2\,x_7\,x_8^2\,x_{10}^3)\,(1 - x_1\,x_2^2\,x_5\,x_6^2\,x_7^2\,x_9\,x_{10}^3) \\ & (1 - x_1\,x_3^2\,x_4^2\,x_5\,x_8^2\,x_9\,x_{10}^3)\,(1 - x_2^2\,x_3\,x_4^2\,x_5\,x_7\,x_9^2\,x_{10}^3))\end{aligned}$$

From this we can extract the exact look of the magic squares. For example, the coefficient of x_{10}^n in the series expansion of the above expression in x_{10} tells us what the magic squares with row, column and diagonal sums n look like.

If we are only interested in the number of such squares, we set x_{10} to y and all the other x_i to x.

```
In[3]:= % /.x10 → y /.x_ → x // Cancel
```

$$\texttt{Out[3]=}\quad \frac{-1-2\,x^9\,y^3-x^{18}\,y^6}{(-1+x^9\,y^3)^3}$$

The coefficient of $x^{3n}\,y^n$ in the series expansion of this expression equals

$$2m^2 + 2m + 1,$$

if n is divisible by 3 and $n = 3m$, and

$$0$$

otherwise.

For further information about `Omega2` we refer to [6].

2.2.2 GenOmega

Based on the work of Andrews et al., Han [17] derived an algorithm with which more general expressions can be treated than with the Omega package. More precisely, Han solves the following problem; see [17, Problem 2].

Problem 2.2.4

Given

$$A(t) = 1 + a_1 t + a_2 t^2 + \cdots + a_n t^n = \prod_{i=1}^{n} (1 - x_i t),$$

$$B(t) = 1 + b_1 t + b_2 t^2 + \cdots + b_m t^m = \prod_{j=1}^{m} (1 - y_j t)$$

as polynomials in $K := \mathbb{Q}(\!(a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_m)\!)[t]$ *and* $U(t)$ *as a Laurent polynomial in* $K[t, t^{-1}]$ *with* $x_i \neq 1$ *and* $x_i y_j \neq 1$ *for all* i, j.

Compute *the Omega expression:*

$$\Omega_{\geq,\lambda} \frac{U(\lambda)}{A(\lambda)B(1/\lambda)}$$

by using only the expansions $1 + a_1 t + a_2 t^2 + \cdots + a_n t^n$ *and* $1 + b_1 t + b_2 t^2 + \cdots + b_m t^m$ *without having to determine and use the roots* x_i *and* y_i.

Note: The x_i, y_i are viewed as elements from $\tilde{K}$, the algebraic closure of K.

Based on his algorithm the Maple package `GenOmega` was developed. It can be downloaded from [30].

After loading the package into Maple with `read(‘genomega.mpl‘):` the following functions are available:

`GENO(A,B,U,t)` eliminates `t` in $\frac{U(t)}{A(t)B(1/t)}$, where `A` and `B` are two polynomials in `t` with $A(1) \neq 0$, $A(0) \neq 0$ and $B(0) \neq 0$, and where for all roots x_i of `A` and all roots y_j of `B` the constraint $x_i y_j \neq 1$ holds , and where `U` is a Laurent polynomial in `t` (see the notation of Problem 2.2.4).

`GENO(f,t,q)` eliminates `t` in $f(t;q)$, where $f(t;q)$ is a formal power series in `q`.

`GENO(f,[t1,...,tr],q)` eliminates `t1,...,tr` in $f(t1,\ldots,tr;q)$, where $f(t1,\ldots,tr;q)$ is a formal power series in `q`.

Since no function like `OSum` is available here, you have to find the crude generating function by yourself. Also, only procedures for the Omega operator for inequality are implemented.

Let us now solve the problems from Section 2.1 with `GenOmega`.

Problem 2.1.1

We again start with the solution of Problem 2.1.1: *Find all nonnegative integer solutions a, b to the inequality $2a \geq 3b$.*

```
> read(‘genomega.mpl‘):

  "General Omega Package"
  "Guoniu Han - ©IRMA Strasbourg - V0.6(07/31/2003)
  "Help()"

> f:=1/((1-y/l^3)*(1-x*l^2));
```

$$f := \frac{1}{(1-\frac{y}{l^3})(1-xl^2)}$$

```
> GENO(f,l,x);
```

$$\frac{1+yx^2}{(x-1)(y^2x^3-1)}$$

We could also have chosen `GENO(f,l,y)` instead of `GENO(f,l,x)`. The output is the same.

Partition on Triangles

We continue with the solution of Problem 2.1.2: *Let $t_3(n)$ be the number of non-congruent triangles with perimiter n whose sides have integer length. Find $t_3(n)$ for general n.*

```
> f:=q/(l2*(1-q*l1/l3)*(1-q*l3/l2)*(1-q*l2*l3/l1));
```

$$f := \frac{q}{l_2(1-\frac{ql_1}{l_3})(1-\frac{ql_3}{l_2})(1-\frac{ql_2l_3}{l_1})}$$

```
> GENO(f,[l1,l2,l3],q);
```

$$-\frac{q^3}{(q+1)^2(q-1)^3(q^2+q+1)(q^2+1)}$$

The last output is the same as $\frac{q^3}{(1-q^2)(1-q^3)(1-q^4)}$ which is the output of the `Omega2` package.

Let us also consider the expansion of this problem made in the last subsection with the help of `GenOmega`. That is, we are looking for the full generating function

$$S_3(x_1,x_2,x_3) = \sum{}^* x_1^{a_1}x_2^{a_2}x_3^{a_3},$$

where $\sum^*$ denotes the restricted summation over all positive integer triples (a_1,a_2,a_3) satisfying $a_1 \le a_2 \le a_3$ and $a_1+a_2 > a_3$.

```
> f:=x1/(l2*(1-x3*l1/l3)*(1-x1*l3/l2)*(1-x2*l2*l3/l1));
```

$$f := \frac{x_1}{l_2(1-\frac{x_3l_1}{l_3})(1-\frac{x_1l_3}{l_2})(1-\frac{x_2l_2l_3}{l_1})}$$

```
> GENO(f,[l1,l2,l3],x1);
```

$$-\frac{x_1x_2x_3}{(x_2x_3-1)(-1+x_1x_2x_3)(-1+x_3^2x_1x_2)}$$

Problem 2.1.4

We continue with the solution of Problem 2.1.4: *Find all nonnegative integer solutions a, b, c to the Diophantine equation* $2a = 3b + c$.

Since there is no function in `GenOmega` that corresponds to the $\Omega_{=,\lambda}$-operator, we have to use some relation which allows us to represent the $\Omega_{=,\lambda}$-operator in terms of the $\Omega_{\ge,\lambda}$-operator. One of these relations is the following (see [22, Vol. II, Sect. VIII, p. 104]),

$$\Omega_{=,\lambda}F(\lambda) = \Omega_{\ge,\lambda}F(\lambda) + \Omega_{\ge,\lambda}F(1/\lambda) - F(1). \tag{2.4}$$

```
> f:=1/((1-y/l^3)*(1-z/l)*(1-x*l^2));
```

$$f := \frac{1}{(1-\frac{y}{l^3})(1-\frac{z}{l})(1-xl^2)}$$

```
> g:=1/((1-y*l^3)*(1-z*l)*(1-x/l^2));
```

$$g := \frac{1}{(1-yl^3)(1-zl)(1-\frac{x}{l^2})}$$

```
> h:=1/((1-y)*(1-z)*(1-x));
```

$$h := \frac{1}{(1-y)(1-z)(1-x)}$$

```
> simplify(GENO(f,l,x)+GENO(g,l,x)-h);
```

$$\frac{1+x^2yz}{(y^2x^3-1)(z^2x-1)}$$

Magic Squares

Last, we have a look at Problem 2.2.3 on magic squares of order 3. Using the relation given in (2.4), neither the general generating function which we investigated with `Omega2` nor the generating function for the special problem stated in Problem 18 could be handled in reasonable time.

Since Han's algorithm is more general than the algorithm of Andrews et al., there are expressions that cannot be handled by `Omega2` but can be handled by `GenOmega`. In [17] Han gives such an example, namely the following,

$$\Omega_{\geq,\lambda}\frac{1}{(1+x\lambda+y\lambda^5)(1+a/\lambda)}.$$

The Omega package returns the error message

`Omega::Input:` $1/((1+a/\lambda)(1+x\lambda+y\lambda^5))$ `is not a valid input`

while `GenOmega` returns the correct result

$$\frac{a^4y-a^3y+a^2y-ay-1}{(a^5y+ax-1)(x+y+1)}.$$

Concerning efficiency, in [17, page 5] Han points out that he does not claim that `GenOmega` is faster than `Omega2`, because he did not effectuate a complete testing for a large number of examples in various environments. But in a preliminary test `GenOmega` was often faster than `Omega2` by a factor of 3 to 6.

Han's algorithm will be viewed in detail in Chapter 3.

2.2.3 LattE and LattE Macchiato

`LattE` and `LattE macchiato` are not based on the theory of Partition Analysis. However, since some of the problems that can be treated with Partition Analysis can also be treated with `LattE` and `LattE macchiato`, we include a short description of them here. The information and examples used in this subsection have been taken from [16, Chapters "Introduction", "Input Files" and "Running `LattE`"] and [31].

2.2.3.1 LattE

`LattE` is free software, available at [32] and runs on Linux systems. It was developed by a team directed by J. A. DeLoera at UC Davis. The name "`LattE`" stands for "**Latt**ice point **E**numeration". It contains the first implementation of Barvinok's algorithm [13]. The software's main function is to count the lattice points contained in convex polyhedra defined by linear equations and inequalities with integer coefficients. The polyhedra can be of any (reasonably small) dimension, and `LattE` uses an algorithm that runs in polynomial time, if the dimension is fixed.

Given a convex polyhedron

$$P = \{u \in \mathbb{R}^d : Au \leq b\},$$

where the matrix A and the vector b are integral and d is a nonnegative integer, a finite representation of the (in general infinite) power series

$$f(P; x) = \sum_{\alpha \in P \cap \mathbb{Z}^d} x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_d^{\alpha_d}$$

is computed. Here each lattice point is given by one monomial. The fundamental theorem of Barvinok [13] states that $f(P; x)$ can be written as a sum of short rational functions — in polynomial time if the dimension of the polyhedron is fixed.

In the following the operations that `LattE` v1.1 can perform on bounded convex polyhedra (more commonly referred to as *polytopes*) are listed. Let's assume that a description of a polytope P is given in the file "fileName" and that a cost vector is pecified in the file "fileName.cost" (needed for the optimization part).

Tasks performed by `LattE` v1.1:

1. Count the number of lattice points in P.
   `./count fileName`

2. Count the number of lattice points in nP, the dilation of P by the integer factor n.
   `./count dil n fileName`

3. Calculate a rational function that encodes the *Ehrhart series* associated with the polytope. By definition, the n-th coefficient in the Ehrhart series equals the number of lattice points in nP. For more details on Ehrhart counting functions see, for example, Chapter 4 of [26].
   `./ehrhart fileName`

4. Calculate a rational function that encodes the *Ehrhart series* associated with the polytope. By definition, the first $n+1$ terms of the Ehrhart series associated with the polytope.
   `./ehrhart n fileName`

5. Maximize or minimize a given linear function of the lattice points in P.
   `./maximize fileName ./minimize fileName`

In addition to these basic functions, there are more specific calls to `LattE`, for example to use the homogenized Barvinok algorithm instead of the original one in order to count the lattice points.

Inequality Description

For computations involving a polytope P described by a system of inequalities $Ax \leq b$, where $A \in \mathbb{Z}^{m \times d}$ and $b \in \mathbb{Z}^m$, the `LattE` readable input file would be of the following form:

$$\begin{array}{ll} m & d+1 \\ b & -A. \end{array}$$

Example 2.2.5 *Let* $P = \{(x, y) : x \leq 1, y \leq 1, x + y \leq 1, x \geq 0, y \geq 0\}$. *Thus*

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \\ -1 & 0 \\ 0 & -1 \end{pmatrix}, \; b = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

and the `LattE` *input file would be as such:*

```
5    3
1   -1    0
1    0   -1
1   -1   -1
0    1    0
0    0    1
```

Equations

In `LattE`, polytopes are represented by linear constraints, i.e. equalities or inequalities. By default a constraint is an inequality of type $Ax \leq b$ unless the line numbers of constraints that are linear equalities are specified in a single additional line.

Example 2.2.6 *Let* P *be as in the previous example, but now require* $x+y = 1$ *instead of* $x + y \leq 1$, *thus,*

$$P = \{(x, y) : x \leq 1, y \leq 1, x + y = 1, x \geq 0, y \geq 0\}.$$

Then the `LattE` *input file that describes* P *would read as:*

```
5    3
1   -1    0
1    0   -1
1   -1   -1
0    1    0
0    0    1
linearity 1 3
```

The last line states that among the 5 *inequalities one is to be considered an equality, namely the third one.*

Nonnegativity Constraints

Since it is often very cumbersome to state all nonnegativity constraints for the variables one by one, there is a more convenient way to do so.

Example 2.2.7 *Let P be as in the previous example, then the* `LattE` *input file that describes P could also read as:*

```
3    3
1   -1    0
1    0   -1
1   -1   -1
linearity 1 3
nonnegative 2 1 2
```

The last line states that there are two nonnegativity constraints and that the first and second variables are required to be nonnegative.

Command Syntax

The basic syntax to invoke the various functions of `LattE` is:

```
./count fileName
./ehrhart fileName
./maximize fileName
./minimize fileName
```

Note that the last two functions require a cost vector specified in the file "fileName.cost". For more information on Ehrhart series and optimization in `LattE` see [16].

Additionally, a variety of options can be used. All options should be space-delimited in the command.

Counting

1. Count the number of lattice points in polytope P, where P is given in "fileName".
   `./count fileName`

2. Count the number of lattice points in nP, the dilation of P by the integer factor n.
   `./count dil n fileName`

3. Count the number of lattice points in the interior of the polytope P, where P is given in "fileName".
   `./count int fileName`

4. Use the homogenized Barvinok algorithm to count the number of lattice points in the polytope P, where P is given in "fileName". Use it if the number of vertices of P is big coompared to the number of constraints.
   `./count homog fileName`

As we have access to the improved version `LattE macchiato` we do not explain how to download and install `LattE`. For more information on `LattE` see [16].

2.2.3.2 LattE Macchiato

`LattE macchiato` is an improved version of `LattE`, derived from the latest release 1.2. It contains amongst others the following improvements:

- Using GNU Autoconf and Automake, it easily builds and runs on a variety of UNIX systems, rather than just on GNU/Linux on the x86 architecture. This includes Mac OS X systems.
- Various micro-optimizations and code clean-ups make the code more robust and much faster than `LattE`.
- For polyhedral computations, it can optionally use the library `4ti2` instead of CDD+ and `cddlib`. For many instances, this gives another big reduction in running time.
- New algorithms are implemented:
  - Primal irrational decomposition
  - All-primal irrational decomposition
  - Exponential substitution
  - Non-unimodular enumeration

  A combination of these new algorithms is much faster (for some examples, by several orders of magnitude) than the traditional algorithms implemented in `LattE`.

Download

The current version of `LattE` "`for tea`, `too`", a source code distribution of `LattE macchiato` and `4ti2` and all required libraries can be downloaded from [33].

Installation

Be sure to install the most recent version. It is self-contained and thus easy to build and install:

```
tar -xfz latte-for-tea-too-1.2-mk-0.9.3.tar.gz
cd latte-for-tea-too-1.2-mk-0.9.3
./configure
make
```

When the compilation has finished (it takes a while), one can start using it:

```
dest/bin/count dest/share/latte/examples/magic4x4
```

2.2.3.3 Examples

Let us now solve special instances of the examples from Section 2.1 with `LattE macchiato`.

Partition on Triangles

We start with the solution of Problem 2.1.2 which was: *Let $t_3(n)$ be the number of non-congruent triangles with perimiter n whose sides have integer length. Find $t_3(n)$ for general n.* We consider the case $n = 9$.

Recall that we have the following constraints:

$$\begin{aligned} a_2 - a_3 &\leq 0, \\ a_1 - a_2 &\leq 0, \\ -a_1 - a_2 + a_3 &\leq -1, \\ -a_1 &\leq -1, \\ a_1 + a_2 + a_3 &= 9. \end{aligned}$$

The input file reads as:

```
5    4
0    0   -1    1
0   -1    1    0
-1   1    1   -1
-1   1    0    0
9   -1   -1   -1
linearity 1 5
nonnegative 3 1 2 3
```

`LattE macchiato` returns:

```
***** Total number of lattice points:  3 ****
```

Total time: 0.1 sec

Magic Squares

We solve Problem 2.2.3, which was: *How many magic squares are there for fixed $a_{10} = n$?* We consider the case $n = a_{10} = 15$:

Recall that we have the following constraints:

$$\begin{aligned} 15 &:= a_1 + a_2 + a_3 = a_4 + a_5 + a_6 = a_7 + a_8 + a_9 \\ &= a_1 + a_4 + a_7 = a_2 + a_5 + a_8 = a_3 + a_6 + a_9 \\ &= a_1 + a_5 + a_9 = a_3 + a_5 + a_7. \end{aligned}$$

The input file reads as:

```
8    10
15   -1   -1   -1    0    0    0    0    0    0
15    0    0    0   -1   -1   -1    0    0    0
15    0    0    0    0    0    0   -1   -1   -1
15   -1    0    0   -1    0    0   -1    0    0
15    0   -1    0    0   -1    0    0   -1    0
15    0    0   -1    0    0   -1    0    0   -1
15   -1    0    0    0   -1    0    0    0   -1
15    0    0   -1    0   -1    0   -1    0    0
linearity 8 1 2 3 4 5 6 7 8
nonnegative 9 1 2 3 4 5 6 7 8 9
```

`LattE macchiato` returns:

```
***** Total number of lattice points:  61 ****
```

Total time: 0.17 sec

We solve the same Problem for $n = a_{10} = 150$.

The input file reads as:

```
8      10
150   -1   -1   -1    0    0    0    0    0    0
150    0    0    0   -1   -1   -1    0    0    0
150    0    0    0    0    0    0   -1   -1   -1
150   -1    0    0   -1    0    0   -1    0    0
150    0   -1    0    0   -1    0    0   -1    0
150    0    0   -1    0    0   -1    0    0   -1
150   -1    0    0    0   -1    0    0    0   -1
150    0    0   -1    0   -1    0   -1    0    0
linearity 8 1 2 3 4 5 6 7 8
nonnegative 9 1 2 3 4 5 6 7 8 9
```

`LattE macchiato` returns:

```
***** Total number of lattice points:  5101 ****
```

Total time: 0.16 sec

Chapter 3

Han's General Algorithm

3.1 Introduction

For the definition of Ω, Han [17] fixes two alphabets $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_r\}$ and $\Gamma = \{p, q, x, y, z, x_1, x_2, \ldots\}$. The expression

$$F := \sum_{s_1=-\infty}^{\infty} \cdots \sum_{s_r=-\infty}^{\infty} A_{s_1,\ldots,s_r} \lambda_1^{s_1} \cdots \lambda_r^{s_r}$$

is a multiple Laurent series in Λ and the coefficients $A_{s_1,\ldots,s_r}$ are ordinary formal series in Γ. The MacMahon operator $\Omega_{\geq} := \Omega_{\geq}[\Lambda, \Gamma]$ is defined by

$$\Omega_{\geq} F := \sum_{s_1=0}^{\infty} \cdots \sum_{s_r=0}^{\infty} A_{s_1,\ldots,s_r}$$

.

Contrary to the classical definition [1–12], which treats the functions analytically, he adds a second alphabet Γ, which enables him to work formally.

Based on the work of Andrews et al., Han derives an algorithm with which more general expressions can be treated than with the Omega package. More precisely, Han solves the following problem; see [17, Problem 2].

Problem 3.1.1

Given

$$A(t) = 1 + a_1 t + a_2 t^2 + \cdots + a_n t^n = \prod_{i=1}^{n} (1 - x_i t),$$

$$B(t) = 1 + b_1 t + b_2 t^2 + \cdots + b_m t^m = \prod_{j=1}^{m} (1 - y_j t)$$

as polynomials in $K := \mathbb{Q}(\!(a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_m)\!)[t]$ *and* $U(t)$ *as a Laurent polynomial in* $K[t, t^{-1}]$ *with* $x_i \neq 1$ *and* $x_i y_j \neq 1$ *for all* i, j.

Compute *the Omega expression:*

$$\Omega_{\geq,\lambda} \frac{U(\lambda)}{A(\lambda)B(1/\lambda)}$$

by using only the expansions $1 + a_1 t + a_2 t^2 + \cdots + a_n t^n$ *and* $1 + b_1 t + b_2 t^2 + \cdots + b_m t^m$ *without having to determine and use the roots* x_i *and* y_i.

Note: The x_i, y_i are viewed as elements from $\tilde{K}$, the algebraic closure of K.

The two conditions $x_i \neq 1$ and $x_i y_j \neq 1$ for all i, j are necessary as will be seen in step A1 in the next section and in Theorem 3.3.6, respectively. Throughout this chapter the notations of Problem 3.1.1 have been kept.

3.2 The Algorithm

Definition 3.2.1 *Let* $f(t)$ *be a polynomial in* t *with constant term equal to* 1. *For nonnegative* i *the complete homogeneous symmetric function* $h_i(f)$ *is implicitly defined by*

$$\frac{1}{f(t)} = \sum_{i \geq 0} h_i(f) t^i,$$

for negative i *it is defined as* 0. *For* $f(t) = \prod_{j=1}^{r} (1 - z_j t)$ *it is also common to write* $h_i(z_1, z_2, \ldots, z_r)$ *instead of* $h_i(f)$.

The following two lemmas are taken from [3, Lemmas 2.1–2.2].

Lemma 3.2.2 *For any integer* a,

$$\Omega_{\geq,\lambda} \frac{\lambda^a}{(1 - x_1\lambda)(1 - x_2\lambda)\cdots(1 - x_n\lambda)} = \Omega_{\geq,\lambda} \sum_{j=0}^{\infty} h_j(x_1, x_2, \ldots, x_n) \lambda^{a+j}$$

$$= \begin{cases} \frac{1}{(1-x_1)(1-x_2)\cdots(1-x_n)}, & \text{if } a \geq 0, \\ \frac{1}{(1-x_1)(1-x_2)\cdots(1-x_n)} - \sum_{j=0}^{-a-1} h_j(x_1, \ldots, x_n), & \text{if } a < 0. \end{cases}$$

Lemma 3.2.3 *For any integer a,*

$$\begin{aligned}\Omega_{\geq,\lambda}\frac{\lambda^a}{(1-\frac{y_1}{\lambda})(1-\frac{y_2}{\lambda})\cdots(1-\frac{y_m}{\lambda})} &= \Omega_{\geq,\lambda}\sum_{j=0}^{\infty}h_j(y_1,y_2,\ldots,y_m)\lambda^{a-j}\\ &= \begin{cases}0, & \text{if } a<0,\\ \sum_{j=0}^{a}h_j(y_1,\ldots,y_m), & \text{if } a\geq 0.\end{cases}\end{aligned}$$

The algorithm for solving Problem 3.1.1 is described in the following five steps.

A1 First, we can assume that $A(t) \neq 1$ and $B(t) \neq 1$. It is easy to evaluate the Ω expression when $A(t) = 1$ or $B(t) = 1$ (see Lemmas 3.2.2 and 3.2.3; here it is required that $x_i \neq 1$ for all i).

A2 Then, we can assume that $A(t)$ is of the form $W(t)^k$ where $W(t)$ is an irreducible polynomial over K. If it is not, we use the partial fraction decomposition

$$\frac{1}{A(t)} = \frac{U_1(t)}{A_1(t)} + \cdots + \frac{U_l(t)}{A_l(t)}$$

and the linearity of the Ω operator.

A3 We can also assume that the numerator $U(t)$ satisfies $\deg(U) < n$ and $\mathrm{ldeg}(U) < m$. If not, we decompose the Laurent polynomial $U(t) = U^+(t) + U^-(t)$ where $U^+(t)$ (resp. $U^-(t)$) is the "positive part" (resp. "negative part") of $U(t)$. Notice that $U^+(t)$ and $U^-(1/t)$ are two polynomials. We can find four polynomials Q_1, Q_2, R_1, R_2 satisfying

$$\begin{aligned}U^+(t) &= Q_1(t)A(t) + R_1(t), \quad \deg(R_1) < n\\ U^-(1/t) &= Q_2(t)B(t) + R_2(t), \quad \deg(R_2) < m.\end{aligned}$$

Let $R(t) = R_1(t) + R_2(1/t)$ be a new Laurent polynomial with $\deg(R) < n$ and $\mathrm{ldeg}(R) < m$. We have

$$U(t) = Q_1(t)A(t) + Q_2(1/t)B(1/t) + R(t)$$

and

$$\Omega_{\geq,t}\frac{U(t)}{A(t)B(1/t)} = \Omega_{\geq,t}\frac{Q_1(t)}{B(1/t)} + \Omega_{\geq,t}\frac{Q_2(1/t)}{A(t)} + \Omega_{\geq,t}\frac{R(t)}{A(t)B(1/t)}.$$

Because the first two terms are easy to evaluate (see step A1), it suffices to evaluate the third term, which satisfies $\deg(R) < n$ and $\mathrm{ldeg}(R) < m$.

A4 If $A(t)$ is an irreducible polynomial, i.e., $k = 1$ in step A2, we do the following (see Theorem 3.3.6): Let the polynomial $C(t)$ be the Bézout coefficient in $C(t)\cdot t^m B(1/t)+K(t)\cdot A(t) = 1$, and the polynomial $D(t)$ be the remainder of the division of $t^m U(t)C(t)$ by $A(t)$. Then

$$\Omega_{\geq,t}\frac{U(t)}{A(t)B(1/t)} = \frac{D(1)}{A(1)}.$$

A5 Now we consider the case $k \geq 2$ in step A2. Let $W(t) = (1 - X_1 t)(1 - X_2 t)\cdots(1 - X_s t)$. We have (see Theorem 3.3.10)

$$\Omega_{\geq,\lambda}\frac{U(\lambda)}{A(\lambda)B(1/\lambda)} = S|_{z_1\to X_1,\ldots,z_s\to X_s},$$

with

$$S = \sum_{i=1}^{s}\frac{1}{(k-1)!}\left[\left(\frac{\partial}{\partial t}\right)^{k-1}\frac{t^{n-1}U(1/t)}{(1-t)B(t)}\left(\frac{t-z_i}{\prod_{i'}(t-z_{i'})}\right)^k\right]\Bigg|_{t\to z_i},$$

i.e.,

$$\Omega_{\geq,\lambda}\frac{U(\lambda)}{A(\lambda)B(1/\lambda)} = \sum_{i=1}^{s}\frac{1}{(k-1)!}\left[\left(\frac{\partial}{\partial t}\right)^{k-1}\frac{t^{n-1}U(1/t)}{(1-t)B(t)t^n}\left(\frac{t-X_i}{W(1/t)}\right)^k\right]\Bigg|_{t\to X_i}.$$

Moreover, if $\mathrm{ldeg}(U) \leq m-1$, then the i-th summand of S is a rational fraction, whose numerator $N(z_i)$ and denominator $D(z_i)$ are two polynomials in z_i such that $\deg_{z_i}(N(z_i)) < \deg_{z_i}(D(z_i))$.

We calculate this sum the following way (see Proposition 3.3.13). Let $Q(t) = q_0 + q_1 t + q_2 t^2 + \cdots + q_l t^l$, $S(t) = s_{\alpha_1}t^{\alpha_1} + s_{\alpha_2}t^{\alpha_2} + \cdots + s_{\alpha_d}t^{\alpha_d}$ with $0 \leq \alpha_1 < \alpha_2 < \cdots < \alpha_d \leq l-1$, and R be the resultant of the two polynomials $t^n A(1/t)$ and $Q(t) + (u_1 - q_{\alpha_1})t^{\alpha_1} + (u_2 - q_{\alpha_2})t^{\alpha_2} + \cdots + (u_d - q_{\alpha_d})t^{\alpha_d}$. Then

$$\sum_{i=1}^{n}\frac{S(x_i)}{Q(x_i)} = \left.\frac{\left(s_{\alpha_1}\frac{\partial}{\partial u_1} + \cdots + s_{\alpha_d}\frac{\partial}{\partial u_d}\right)R}{R}\right|_{u_1\to q_{\alpha_1},\ldots,u_d\to q_{\alpha_d}}.$$

3.3 Development of the Algorithm

The following Theorem is the first step in the solution of Problem 3.1.1.

Theorem 3.3.1 *If* $\deg(U) \leq n-1$, *then:*

$$\Omega_{\geq,\lambda}\frac{U(\lambda)}{A(\lambda)B(1/\lambda)} = \sum_{i=1}^{n}\frac{x_i^{n-1}U(1/x_i)}{(1-x_i)B(x_i)\prod_{j\neq i}(x_i-x_j)}.$$

The formula holds even if the x_j*'s are not all distinct.*

For the proof we need the following definitions and lemmas.

Definition 3.3.2 *Let* $Z = \{z_1, \ldots, z_r\}$ *be an alphabet. Then*

$$\Delta(Z) := \begin{vmatrix} z_1^{r-1} & z_2^{r-1} & \cdots & z_r^{r-1} \\ \vdots & \vdots & \ddots & \vdots \\ z_1^2 & z_2^2 & \cdots & z_r^2 \\ z_1^1 & z_2^1 & \cdots & z_r^1 \\ 1 & 1 & \cdots & 1 \end{vmatrix}.$$

Definition 3.3.3 *Let* $d = d_1 + d_2 + \cdots + d_r$ *with* $d_1 \geq d_2 \geq \cdots \geq d_r$, *where each* d_j *is a non-negative integer, be a partition and* $Z = \{z_1, \ldots, z_r\}$ *be an alphabet. The corresponding Schur polynomial* $S_{(d_1,d_2,\ldots,d_r)} \in \mathbb{Z}[z_1, z_2, \ldots, z_r]$ *is given by*

$$S_{(d_1,d_2,\ldots,d_r)} = \frac{\det R}{\Delta(Z)}$$

where

$$R = \begin{pmatrix} z_1^{d_1+r-1} & z_2^{d_1+r-1} & \cdots & z_r^{d_1+r-1} \\ z_1^{d_2+r-2} & z_2^{d_2+r-2} & \cdots & z_r^{d_2+r-2} \\ \vdots & \vdots & \ddots & \vdots \\ z_1^{d_r} & z_2^{d_r} & \cdots & z_r^{d_r} \end{pmatrix}.$$

If the numbers $d_i + r - i$ *are not all distinct then* $S_{(d_1,d_2,\ldots,d_r)}$ *is* 0.

The content of the following lemma is taken from [18, equation (3.4) on page 41].

Lemma 3.3.4 *Each Schur polynomial $S_{(d_1,d_2,\ldots,d_r)} \in \mathbb{Z}[z_1, z_2, \ldots, z_r]$ can be expressed as a polynomial in the complete symmetric functions $h_i(z_1, z_2, \ldots, z_r)$, namely by*

$$S_{(d_1,d_2,\ldots,d_r)} = \det(h_{d_i-i+j}(z_1, z_2, \ldots, z_r))_{1\leq i,j\leq r}.$$

Lemma 3.3.5 *Let $Z = \{z_1, z_2, \ldots, z_r\}$ be an alphabet. For all integers $\alpha \geq -(r-1)$ we have*

$$h_\alpha(z_1, z_2, \ldots, z_r) = \begin{vmatrix} z_1^{\alpha+r-1} & z_2^{\alpha+r-1} & \cdots & z_r^{\alpha+r-1} \\ z_1^{r-2} & z_2^{r-2} & \cdots & z_r^{r-2} \\ \vdots & \vdots & \ddots & \vdots \\ z_1^1 & z_2^1 & \cdots & z_r^1 \\ z_1^0 & z_2^0 & \cdots & z_r^0 \end{vmatrix} / \Delta(Z).$$

Proof. If $\alpha < 0$, we have

$$\begin{vmatrix} z_1^{\alpha+r-1} & z_2^{\alpha+r-1} & \cdots & z_r^{\alpha+r-1} \\ z_1^{r-2} & z_2^{r-2} & \cdots & z_r^{r-2} \\ \vdots & \vdots & \ddots & \vdots \\ z_1^1 & z_2^1 & \cdots & z_r^1 \\ z_1^0 & z_2^0 & \cdots & z_r^0 \end{vmatrix} = 0,$$

since one of the rows occurs twice because of the range of α.

If $\alpha \geq 0$, we use the partition $d = d_1 + d_2 + \cdots + d_r$ where $d_1 = \alpha$ and $d_j = 0$ for all j where $2 \leq j \leq r$. The resulting Schur polynomial $S_{(d_1,\ldots,d_r)} \in \mathbb{Z}[z_1, z_2, \ldots, z_r]$ is given by

$$S_{(d_1,\ldots,d_r)} = \begin{vmatrix} z_1^{\alpha+r-1} & z_2^{\alpha+r-1} & \cdots & z_r^{\alpha+r-1} \\ z_1^{r-2} & z_2^{r-2} & \cdots & z_r^{r-2} \\ \vdots & \vdots & \ddots & \vdots \\ z_1^1 & z_2^1 & \cdots & z_r^1 \\ z_1^0 & z_2^0 & \cdots & z_r^0 \end{vmatrix} / \Delta(Z).$$

Using Lemma 3.3.4, we know (we write h_i instead of $h_i(z_1, \ldots, z_r)$):

$$S_{(d_1,\ldots,d_r)} = \begin{vmatrix} h_\alpha & h_{\alpha+1} & h_{\alpha+2} & \cdots & h_{\alpha+r-1} \\ 0 & h_0 & h_1 & \cdots & h_{r-2} \\ 0 & 0 & h_0 & \cdots & h_{r-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & h_0 \end{vmatrix} = h_\alpha(z_1, \ldots, z_r).$$

■

Now we can prove Theorem 3.3.1.

Proof of Theorem 3.3.1. Let $a \in \mathbb{Z}, a \leq n-1$. With the notation of Problem 3.1.1 we have

$$\begin{aligned} \Omega_{\geq,\lambda} \frac{\lambda^a}{A(\lambda)B(1/\lambda)} &= \Omega_{\geq,\lambda} \sum_{k,j\geq 0} h_k(A)h_j(B)\lambda^{a+k-j} \\ &= \sum_{k,j,a+k-j\geq 0} h_k(A)h_j(B) \\ &= \sum_{k\geq 0} \left[h_k(A) \sum_{0\leq j\leq a+k} h_j(B) \right]. \end{aligned}$$

With $C(t) := (1-t)B(t)$ we get

$$\frac{1}{C(t)} = \frac{1}{B(t)(1-t)} = \sum_{i\geq 0} h_i(B)t^i \sum_{j\geq 0} t^j = \sum_{l\geq 0} \left[\sum_{0\leq i\leq l} h_i(B) \right] t^l,$$

i.e., $h_l(C) = \sum_{i=0}^{l} h_i(B)$. We then have

$$\Omega_{\geq,\lambda} \frac{\lambda^a}{A(\lambda)B(1/\lambda)} = \sum_{k\geq 0} h_k(A)h_{a+k}(C).$$

Since for all k with $-(n-1) \leq k < 0$ we have $h_k(A) = 0$, we get

$$\Omega_{\geq,\lambda} \frac{\lambda^a}{A(\lambda)B(1/\lambda)} = \sum_{k\geq -(n-1)} h_k(A)h_{a+k}(C).$$

Let $Z = \{z_1, z_2, \ldots, z_n\}$ be an alphabet. With Lemma 3.3.5 we get that for every $k \geq -(n-1)$

$$h_k(z_1, z_2, \ldots, z_n) = \begin{vmatrix} z_1^{k+n-1} & z_2^{k+n-1} & \cdots & z_n^{k+n-1} \\ z_1^{n-2} & z_2^{n-2} & \cdots & z_n^{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ z_1^1 & z_2^1 & \cdots & z_n^1 \\ z_1^0 & z_2^0 & \cdots & z_n^0 \end{vmatrix} / \Delta(Z).$$

By multiplying both sides by $h_{a+k}(C)$ and summing over k from $-(n-1)$ up to ∞, we get:

$$\Delta(Z)\cdot\sum_{k\geq-(n-1)}h_k(z_1,\ldots,z_n)h_{a+k}(C)=\sum_{k\geq-(n-1)}\begin{vmatrix} z_1^{k+n-1}h_{a+k}(C) & \cdots \\ z_1^{n-2} & \cdots \\ \cdots & \cdots \\ z_1^0 & \cdots \end{vmatrix}$$

$$=\begin{vmatrix} z_1^{n-1-a}\sum_{k\geq-(n-1)}z_1^{a+k}h_{a+k}(C) & \cdots \\ z_1^{n-2} & \cdots \\ \cdots & \cdots \\ z_1^0 & \cdots \end{vmatrix}=\begin{vmatrix} \frac{z_1^{n-1-a}}{C(z_1)} & \cdots \\ z_1^{n-2} & \cdots \\ \cdots & \cdots \\ z_1^0 & \cdots \end{vmatrix}.$$

In the last step we required the equality $h_{a+k}(C)=0$ to hold for all k with $-(n-1)\leq k<-a$.

Now by expanding the determinant along the first row we get:

$$\begin{aligned}\sum_{k\geq-(n-1)}h_k(z_1,\ldots,z_n)h_{a+k}(C) &= \frac{1}{\Delta(Z)}\sum_{i=1}^{n}\frac{z_i^{n-1-a}}{C(z_i)}(-1)^{i+1}\Delta(Z\setminus\{z_i\})\\ &= \sum_{i=1}^{n}\frac{z_i^{n-1}/z_i^a}{(1-z_i)B(z_i)\prod_{j\neq i}(z_i-z_j)}.\end{aligned}$$

By substitution of z_i by x_i for all $i\in\{1,\ldots,n\}$ we get

$$\Omega_{\geq,\lambda}\frac{\lambda^a}{A(\lambda)B(1/\lambda)}=\sum_{i=1}^{n}\frac{x_i^{n-1}/x_i^a}{(1-x_i)B(x_i)\prod_{j\neq i}(x_i-x_j)}.$$

With this, Theorem 3.3.1 is proved in the case $U(t)=t^a$ and $a\leq n-1$. The linear combination of all those particular cases provides a complete proof of Theorem 3.3.1. ■

None of the summands in the equation in Theorem 3.3.1 contains any of the y_j, but the x_i still appear. In order to eliminiate those as well, Han makes use of two methods.

- The first one uses the Lagrange interpolation formula (see Theorem 3.3.6). He points out that the algorithm is elegant and simple, but becomes inefficient, for instance when the degree of the polynomial $A(t)$ is large, since it involves the Euclidean division of two polynomials, which is known to have a complexity of high order if the coefficients of the two polynomials are polynomials in several variables.

- The second method consists of two steps:
  - first, re-express the i-th summand in Theorem 3.3.1 in a form that depends only on x_i (Theorem 3.3.10);
  - then, symmetrize the new summands as rational functions in one variable (Proposition 3.3.13).

  This method involves the resultant of two polynomials. As the resultant is a multiplicative operator, this method is much faster than the first one if $A(t)$ is not an irreducible polynomial.

Theorem 3.3.6 *If the Laurent polynomial U has degree $\deg(U) \leq n-1$ and low degree $\operatorname{ldeg}(U) \leq m-1$, then there exists a unique polynomial $D(t)$ of degree smaller than n such that*

$$D(t) \equiv \frac{U(t)}{B(1/t)} \mod A(t).$$

Moreover,

$$\Omega_{\geq,\lambda} \frac{U(\lambda)}{A(\lambda)B(1/\lambda)} = \frac{D(1)}{A(1)}.$$

Proof. Since $\gcd(t^m B(1/t), A(t)) = 1$, as can be easily seen (here we need $x_i y_j \neq 1$ for all i, j), Bézout's Theorem implies that there exists a polynomial $C(t)$ such that $C(t) \cdot t^m B(1/t) + F(t) \cdot A(t) = 1$, i.e., $C(t) \equiv \frac{1}{t^m B(1/t)} \mod A(t)$. Then the polynomial $D(t)$ is the remainder of the division of $t^m U(t) C(t)$ by $A(t)$. For the second part it suffices to verify that

$$\sum_{i=1}^{n} \frac{x_i^{n-1} U(1/x_i)}{B(x_i)} \frac{\prod_{j \neq i}(1 - x_j t)}{\prod_{j \neq i}(x_i - x_j)} = D(t).$$

For this we use the Lagrange interpolation formula which states that for $g(t) = (t - s_1)(t - s_2) \cdots (t - s_r)$ with distinct s_j's and for any polynomial $f(t)$ we have

$$\operatorname{rem}(f, g) = \sum_{k=1}^{r} f(s_k) \frac{\prod_{j \neq k}(t - s_j)}{\prod_{j \neq k}(s_k - s_j)}$$

and from the fact that for all polynomials f, g with $g \neq 0$, and constants $c \neq 0$

$$\operatorname{rem}(f, g) = \operatorname{rem}(f, c \cdot g).$$

We have

$$A(t) = \prod_{i=1}^{n}(1 - x_i t) = \prod_{i=1}^{n}(-x_i)\prod_{i=1}^{n}(t - 1/x_i).$$

Therefore,

$$\begin{aligned} D(t) &= \operatorname{rem}(t^m\, U(t)\, C(t),\, A(t)) = \operatorname{rem}(t^m\, U(t)\, C(t),\, \prod_{i=1}^{n}(t - 1/x_i)) \\ &= \sum_{i=1}^{n} x_i^{-m}\, U(1/x_i)\, C(1/x_i)\, \frac{\prod_{j\neq i}(t - \frac{1}{x_j})}{\prod_{j\neq i}(\frac{1}{x_i} - \frac{1}{x_j})} \\ &= \sum_{i=1}^{n} x_i^{-m}\, U(1/x_i)\, C(1/x_i)\, \frac{\prod_{j\neq i}(1 - x_j t)\, \prod_{j\neq i}(-\frac{1}{x_j})}{\prod_{j\neq i}(x_i - x_j)\, \prod_{j\neq i}(-\frac{1}{x_i x_j})} \\ &= \sum_{i=1}^{n} x_i^{-m}\, U(1/x_i)\, C(1/x_i)\, \frac{\prod_{j\neq i}(1 - x_j t)}{\prod_{j\neq i}(x_i - x_j)\, (\frac{1}{x_i})^{n-1}}. \end{aligned}$$

Since

$$C(1/x_i)\, x_i^{-m} B(x_i) + F(1/x_i)\, A(1/x_i) = 1$$

and

$$A(1/x_i) = 0$$

it follows that

$$C(1/x_i) = x_i^m \frac{1}{B(x_i)}.$$

Therefore,

$$D(t) = \sum_{i=1}^{n} \frac{x_i^{n-1} U(\frac{1}{x_i})}{B(x_i)}\, \frac{\prod_{j\neq i}(1 - x_j t)}{\prod_{j\neq i}(x_i - x_j)}.$$

■

In the following, the summands of Theorem 3.3.1 are going to be evaluated according to the second method mentioned above. First Han eliminates all the x_j $(j \neq i)$ and keeps only x_i in the i-th summand. For an arbitrary polynomial $A(t)$, he considers the partial fraction decomposition

$$\frac{1}{A(t)} = \frac{U_1(t)}{A_1(t)} + \cdots + \frac{U_l(t)}{A_l(t)}.$$

Since all the $A_i(t)$'s are powers of square-free polynomials, it suffices to study the case where $A(t)$ itself is a power of a square-free polynomial. First, he considers the case where the polynomial $A(t)$ is square-free, i.e., all the x_j $(1 \leq j \leq n)$ are distinct.

Lemma 3.3.7 *Let $f(t)$ and $a(t)$ be polynomials in t over a field F and let $x \in F$ be a root of $f(t)$ with multiplicity 1. Then we have for all $k \in \mathbb{N}^*$*

$$\left(\frac{\partial}{\partial t}\right)^k (a(t)f(t)^k)\bigg|_{t\to x} = k!a(x)f'(x)^k.$$

Proof. We prove this lemma by induction on k.

For $k = 1$,

$$\left(\frac{\partial}{\partial t}\right)^k (a(t)f(t)^k)\bigg|_{t\to x} = (a'(t)f(t) + a(t)f'(t))|_{t\to x} = 1!a(x)f'(x).$$

We assume the lemma holds for $k \geq 1$ and show that it holds for $k+1$.

$$\begin{aligned}\left(\frac{\partial}{\partial t}\right)^{k+1} (a(t)f(t)^{k+1})\bigg|_{t\to x} &= \left(\frac{\partial}{\partial t}\right)^k (a'(t)f(t)^{k+1})\bigg|_{t\to x} \\ &\quad + \left(\frac{\partial}{\partial t}\right)^k ((k+1)a(t)f(t)^k f'(t))\bigg|_{t\to x} \\ &= k!a'(x)f(x)f'(x)^k + k!(k+1)a(x)f'(x)^{k+1} \\ &= (k+1)!\, a(x)f'(x)^{k+1}.\end{aligned}$$

■

Lemma 3.3.8 *Let $a \in \mathbb{Z}$ and let $Z = \{z_1, \ldots, z_n\}$ be an alphabet. If $\deg(U) \leq n-1$ and if $x_i \neq x_j$ for all $i \neq j$, then*

$$\Omega_{\geq,\lambda}\frac{\lambda^a}{A(\lambda)B(1/\lambda)} = \sum_{i=1}^{n} \frac{z_i^{n-1}U(1/z_i)}{(z_i-1)B(z_i)\bar{A}(1/z_i)z_i^{n-1}}\bigg|_{z_1\to x_1,\ldots,z_n\to x_n},$$

where $\bar{A}(t) = t\frac{\partial}{\partial t}A(t)$. Moreover, if $\operatorname{ldeg}(U) \leq m-1$, then the numerator $N(z_i)$ and the denominator $D(z_i)$ of the i-th summand of the uninstantiated sum are two polynomials in z_i such that $\deg_{z_i}(N(z_i)) < \deg_{z_i}(D(z_i))$.

Proof. By Theorem 3.3.1 and Lemma 3.3.7,

$$\begin{aligned}\prod_{j\neq i}(x_i - x_j) &= \frac{\prod_{j=1}^n (t-x_j)}{t-x_i}\bigg|_{t\to x_i} = \frac{t^n A(1/t)}{t-x_i}\bigg|_{t\to x_i} \\ &= \frac{\frac{\partial}{\partial t}t^n A(1/t)}{\frac{\partial}{\partial t}(t-x_i)}\bigg|_{t\to x_i} = -x_i^{n-1}\bar{A}(1/x_i).\end{aligned}$$

If $\mathrm{ldeg}(U) \leq m-1$, then $\deg_{z_i}(N(z_i)) = n-1+\mathrm{ldeg}(U) \leq n+m-2$ and $\deg_{z_i}(D(z_i)) = m+n$. ■

Next consider the case of the power of a square-free polynomial.

Definition 3.3.9 *Let $k \in \mathbb{Z}$, R a commutative ring with 1, such that $R \supseteq \mathbb{Q}$, and $x \in R$. Then*

$$\binom{x}{k} := \begin{cases} \frac{x(x-1)\cdots(x-k+1)}{k!}, & \text{if } k \geq 0, \\ 0, & \text{otherwise.} \end{cases}$$

Theorem 3.3.10 *Let $Z = \{z_1, \ldots, z_n\}$ be an alphabet. If $\deg(U) \leq n-1$ and if $A(t) = W(t)^k$ $(k \geq 1)$ with $W(t) = (1-X_1t)(1-X_2t)\cdots(1-X_st)$ such that $X_i \neq X_j$ for $1 \leq i < j \leq s$, then:*

$$\Omega_{\geq,\lambda}\frac{U(\lambda)}{A(\lambda)B(1/\lambda)} = S|_{z_1\to X_1,\ldots,z_s\to X_s},$$

with

$$S = \sum_{i=1}^{s} \frac{1}{(k-1)!}\left[\left(\frac{\partial}{\partial t}\right)^{k-1} \frac{t^{n-1}U(1/t)}{(1-t)B(t)}\left(\frac{t-z_i}{\prod_{i'}(t-z_{i'})}\right)^k\right]\Bigg|_{t\to z_i},$$

i.e.,

$$\Omega_{\geq,\lambda}\frac{U(\lambda)}{A(\lambda)B(1/\lambda)} = \\ = \sum_{i=1}^{s} \frac{1}{(k-1)!}\left[\left(\frac{\partial}{\partial t}\right)^{k-1} \frac{t^{n-1}U(1/t)}{(1-t)B(t)t^n}\left(\frac{t-X_i}{W(1/t)}\right)^k\right]\Bigg|_{t\to X_i}.$$

Moreover, if $\mathrm{ldeg}(U) \leq m-1$, then the i-th summand of S is a rational fraction, whose numerator $N(z_i)$ and denominator $D(z_i)$ are two polynomials in z_i such that $\deg_{z_i}(N(z_i)) < \deg_{z_i}(D(z_i))$.

The proof of this theorem is based on the following lemma.

Lemma 3.3.11 *Let $f(t) \in \mathbb{F}[[t]]$, where $\mathbb{F} \supseteq \mathbb{Q}$ is a field, and let $Z = \{z_1, z_2, \ldots, z_k\}$ be an alphabet. Then*

$$\sum_{i=1}^{k} \frac{f(z_i)}{\prod_{j\neq i}(z_i - z_j)}\Bigg|_{z_1,z_2,\ldots,z_k\to t} = \frac{1}{(k-1)!}\left(\frac{\partial}{\partial t}\right)^{k-1} f(t).$$

Proof. By linearity it suffices to take $f(t) = t^\alpha$ with $\alpha \in \mathbb{N}$, so that

$$\sum_{i=1}^{k} \frac{z_i^\alpha}{\prod_{j\neq i}(z_i - z_j)} = \frac{1}{\Delta(Z)} \begin{vmatrix} z_1^\alpha & z_2^\alpha & \cdots & z_k^\alpha \\ z_1^{k-2} & z_2^{k-2} & \cdots & z_k^{k-2} \\ \vdots & \vdots & \ddots & \vdots \\ z_1^1 & z_2^1 & \cdots & z_k^1 \\ 1 & 1 & \cdots & 1 \end{vmatrix} = h_{\alpha-k+1}(Z),$$

where we invoked Lemma 3.3.5 in the last step. Therefore,

$$\begin{aligned} \sum_{i=1}^{k} \frac{z_i^\alpha}{\prod_{j\neq i}(z_i - z_j)}\bigg|_{z_1,z_2,\ldots,z_k\to t} &= h_{\alpha-k+1}(t,t,\ldots,t) \\ &= \langle u^{\alpha-k+1}\rangle \left(\frac{1}{(1-ut)^k}\right). \end{aligned}$$

Since for all $l \in \mathbb{N}$

$$\langle u^l\rangle \left(\frac{1}{(1-ut)^k}\right) = \binom{l+k-1}{k-1} t^l,$$

we have

$$\begin{aligned} \sum_{i=1}^{k} \frac{z_i^\alpha}{\prod_{j\neq i}(z_i - z_j)}\bigg|_{z_1,z_2,\ldots,z_k\to t} &= \binom{\alpha}{k-1} t^{\alpha-k+1} \\ &= \frac{\alpha(\alpha+1)\cdots(\alpha-k+2)}{(k-1)!} t^{\alpha-k+1} \\ &= \frac{1}{(k-1)!}\left(\frac{\partial}{\partial t}\right)^{k-1} t^\alpha. \end{aligned}$$

■

Proof of Theorem 3.3.10. Let $z_1 = z_{11} = z_{12} = \cdots = z_{1k}$, $z_2 = z_{21} = z_{22} = \cdots = z_{2k}$, $\ldots$, $z_s = z_{s1} = z_{s2} = \cdots = z_{sk}$. Also let $F(t) := \frac{t^{n-1}U(1/t)}{(1-t)B(t)}$. It follows from Theorem 3.3.1 that

$$\Omega_{\geq,\lambda} \frac{U(\lambda)}{A(\lambda)B(1/\lambda)} = \sum_{i=1}^{s}\sum_{j=1}^{k} \frac{F(z_{ij})}{\prod_{(i',j')\neq(i,j)}(z_{ij} - z_{i'j'})}\bigg|_{z_{11},\ldots,z_{1k}\to X_1,\ldots,z_{s1},\ldots,z_{sk}\to X_s}.$$

We have

$$\begin{aligned}
&\sum_{i=1}^{s}\sum_{j=1}^{k}\frac{F(z_{ij})}{\prod_{(i',j')\neq(i,j)}(z_{ij}-z_{i'j'})}\\
&= \sum_{i=1}^{s}\sum_{j=1}^{k}\frac{F(t)}{\prod_{i'\neq i,\,j'=1,\ldots,k}(t-z_{i'j'})\prod_{j'\neq j}(t-z_{ij'})}\Bigg|_{t\to z_{ij}}\\
&= \sum_{i=1}^{s}\sum_{j=1}^{k}\frac{F(t)}{\prod_{i'\neq i}(t-z_{i'})^k\prod_{j'\neq j}(z_{ij}-z_{ij'})}\Bigg|_{t\to z_i}.
\end{aligned}$$

Note that $\frac{F(t)}{\prod_{i'\neq i}(t-z_{i'})^k}$ is a formal power series in $\hat{K}[[t]]$ with $\hat{K} := K(\!(z_1,\ldots,z_s)\!)$ since $t^{n-1}U(1/t)$ is a polynomial in $\hat{K}[t]$ and hence can be viewed as a formal power series in $\hat{K}[[t]]$, the fractions $\frac{1}{1-t}$, $\frac{1}{B(t)}$ and $\frac{1}{\prod_{i'\neq i}(t-z_{i'})^k}$ can also be viewed as formal power series in $\hat{K}[[t]]$ and the product of two formal power series in $\hat{K}[[t]]$ is again a formal power series in $\hat{K}[[t]]$.

From the previous lemma we have

$$\begin{aligned}
&\sum_{j=1}^{k}\frac{F(t)}{\prod_{i'\neq i}(t-z_{i'})^k\prod_{j'\neq j}(z_{ij}-z_{ij'})}\Bigg|_{z_{ij}\to t}\Bigg|_{t\to z_i}\\
&= \frac{1}{(k-1)!}\left(\frac{\partial}{\partial t}\right)^{k-1}\frac{F(t)}{\prod_{i'\neq i}(t-z_{i'})^k}\Bigg|_{t\to z_i}\\
&= \frac{1}{(k-1)!}\left(\frac{\partial}{\partial t}\right)^{k-1}\frac{F(t)(t-z_i)^k}{\prod_{i'}(t-z_{i'})^k}\Bigg|_{t\to z_i}.
\end{aligned}$$

We get

$$\begin{aligned}
&\Omega_{\geq,\lambda}\frac{U(\lambda)}{A(\lambda)B(1/\lambda)} =\\
&= \sum_{i=1}^{s}\frac{1}{(k-1)!}\left(\frac{\partial}{\partial t}\right)^{k-1}\frac{F(t)(t-z_i)^k}{\prod_{i'}(t-z_{i'})^k}\Bigg|_{t\to z_i}\Bigg|_{z_1\to X_1,\ldots,z_s\to X_s}\\
&= \sum_{i=1}^{s}\frac{1}{(k-1)!}\left[\left(\frac{\partial}{\partial t}\right)^{k-1}\frac{F(t)}{t^n}\left(\frac{t-X_i}{W(1/t)}\right)^k\right]\Bigg|_{t\to X_i}.
\end{aligned}$$

Now if $\mathrm{ldeg}(U)\leq m-1$, then $N_I := t^{n-1}U(1/t)(t-z_i)^k$ is a polynomial in t the degree of which is smaller than or equal to $n+m-2+k$ and

$D_I := (1-t)B(t)t^n \prod_{i'}(1-z_{i'}/t)^k$ is a polynomial in t of degree $n+m+1$. We denote

$$\left(\frac{\partial}{\partial t}\right)^{k-1} \frac{N_I}{D_I} =: \frac{N_F}{D_F}.$$

When the operator $\frac{\partial}{\partial t}$ acts on a rational fraction, it increases the difference between the degree of the denominator and the degree of the numerator. Thus

$$\deg(D_F) - \deg(N_F) \geq (n+m+1) - (n+m-2+k) + (k-1) = 2.$$

Let $\tilde{W}(t) := \prod_{i'}(1-z_{i'}t)$. Since $\tilde{W}(1/z_i) = 0$ and hence z_i is a zero of both N_F and D_F, the substitution of z_i for t cannot be made immediately. With each differentiation step the exponent of $\tilde{W}(1/t)$ increases by 1. Therefore, $D_F = f(t)\tilde{W}(1/t)^{2k-1}$ for some polynomial $f(t)$ with $f(t) \neq 0$. It follows that z_i is a zero of D_F with multiplicity $2k-1$. Since we have

$$\begin{aligned}
\frac{N_I}{D_I} &= \frac{t^{n-1}U(1/t)(t-z_i)^k}{(1-t)B(t)t^n\tilde{W}(1/t)^k} \\
&= \frac{t^{n-1}U(1/t)(t-z_i)^k}{(1-t)B(t)t^n\prod_{j=1}^{s}(1-z_j/t)^k} \\
&= \frac{t^{n-1}U(1/t)(t-z_i)^k}{(1-t)B(t)\prod_{j=1}^{s}(t-z_j)^k} \\
&= \frac{t^{n-1}U(1/t)}{(1-t)B(t)\prod_{j\neq i}(t-z_j)^k}.
\end{aligned}$$

where no more z_i occurs, we know that z_i can be reduced in N_F/D_F. Hence, we know that the multiplicity of z_i in N_F and D_F is the same, namely $2k-1$, as stated above. Consequently, with Lemma 3.3.7 we have

$$\left(\frac{\partial}{\partial t}\right)^{k-1} \frac{N_I}{D_I}\bigg|_{t\to z_i} = \frac{\left(\frac{\partial}{\partial t}\right)^{2k-1} N_F\Big|_{t\to z_i}}{\left(\frac{\partial}{\partial t}\right)^{2k-1} D_F\Big|_{t\to z_i}}.$$

In $\left(\frac{\partial}{\partial t}\right)^{2k-1} N_F$ the z_i occurs only in expressions of the form $t-z_i$, hence

$$\deg_t\left(\left(\frac{\partial}{\partial t}\right)^{2k-1} N_F\right) \geq \deg_{z_i}\left(\left(\frac{\partial}{\partial t}\right)^{2k-1} N_F\bigg|_{t\to z_i}\right)$$

and

$$\deg_t\left(\left(\frac{\partial}{\partial t}\right)^{2k-1} D_F\right) = \deg_{z_i}\left(\left(\frac{\partial}{\partial t}\right)^{2k-1} D_F\bigg|_{t\to z_i}\right).$$

Therefore,

$$\deg_{z_i}\left(\left(\frac{\partial}{\partial t}\right)^{2k-1} N_F\Big|_{t\to z_i}\right) - \deg_{z_i}\left(\left(\frac{\partial}{\partial t}\right)^{2k-1} D_F\Big|_{t\to z_i}\right) \geq 2.$$

■

Now, the sum in Theorem 3.3.1 is of the form $\sum_{i=1}^{n} f(x_i)$, where each summand is a rational function that depends on a single x_i. The next step is to calculate that sum by using the expansion $A(t) = 1 + a_1 t + \cdots + a_n t^n$ without having to determine and use the roots x_i. This will be done in Proposition 3.3.13.

Lemma 3.3.12 *Let $Q(t) = q_0 + q_1 t + q_2 t^2 + \cdots + q_d t^d + \cdots + q_l t^l \in K[t]$ with $0 \leq d \leq l-1$ such that $Q(x_i) \neq 0$ for each $1 \leq i \leq n$. Then*

$$\sum_{i=1}^{n} \frac{x_i^d}{Q(x_i)} = \frac{\frac{\partial}{\partial u} R}{R}\Bigg|_{u\to q_d},$$

where R is the resultant of the two polynomials $t^n A(1/t)$ and $Q(t)+(u-q_d)t^d$.

Proof. Let $\hat{Q}(t) := Q(t) + (u - q_d)t^d$. Because

$$R = \hat{Q}(x_1)\hat{Q}(x_2)\cdots\hat{Q}(x_n),$$

we have

$$\frac{\partial}{\partial u}\log(R) = \frac{\frac{\partial}{\partial u}R}{R} = \sum_{i=1}^{n} \frac{\frac{\partial}{\partial u}\hat{Q}(x_i)}{\hat{Q}(x_i)} = \sum_{i=1}^{n} \frac{x_i^d}{\hat{Q}(x_i)}.$$

Thus

$$\frac{\frac{\partial}{\partial u}R}{R}\Bigg|_{u\to q_d} = \sum_{i=1}^{n} \frac{x_i^d}{\hat{Q}(x_i)}\Bigg|_{u\to q_d} = \sum_{i=1}^{n} \frac{x_i^d}{Q(x_i)}.$$

■

If the numerator is not a monomial, Lemma 3.3.12 can be applied several times by linearity. But to avoid the calculation of the resultant several times, Han gives the following Proposition, that can be proved in the same way as the Lemma.

Proposition 3.3.13 *Let* $Q(t) = q_0 + q_1 t + q_2 t^2 + \cdots + q_l t^l \in K[t]$ *with* $Q(x_i) \neq 0$ *for each* $1 \leq i \leq n$. *If* $S(t) = s_{\alpha_1} t^{\alpha_1} + s_{\alpha_2} t^{\alpha_2} + \cdots + s_{\alpha_d} t^{\alpha_d} \in K[t]$ *with* $0 \leq \alpha_1 < \alpha_2 < \cdots < \alpha_d \leq l - 1$, *then*

$$\sum_{i=1}^{n} \frac{S(x_i)}{Q(x_i)} = \left. \frac{\left(s_{\alpha_1} \frac{\partial}{\partial u_1} + \cdots + s_{\alpha_d} \frac{\partial}{\partial u_d} \right) R}{R} \right|_{u_1 \to q_{\alpha_1}, \ldots, u_d \to q_{\alpha_d}},$$

where R *is the resultant of the two polynomials* $t^n A(1/t)$ *and* $Q(t) + (u_1 - q_{\alpha_1}) t^{\alpha_1} + (u_2 - q_{\alpha_2}) t^{\alpha_2} + \cdots + (u_d - q_{\alpha_d}) t^{\alpha_d}$.

3.4 A Note on Applicability

If one wants to compute Omega expressions where several Omega variables are involved, this can be accomplished by successively eliminating the Omega variables. However, in every such elimination step the current Omega expression has to be valid (see Problem 3.1.1) with respect to the Omega variable to be eliminated. Unfortunately, this cannot be guaranteed even if the original Omega expression is valid with respect to every Omega variable. For more details, see Sections 4.3.5.2 and 4.4.3.

Chapter 4

The new GenOmega Package

4.1 Introduction

As mentioned in 2.2.2, Han implemented the algorithm described in the last chapter in the computer algebra system Maple. This implementation contained a few bugs, some of which will be described below and which were corrected in our Mathematica version which was implemented in Mathematica 6.0. Also, more general input polynomials are allowed in our version than in Han's. In addition to that, not only (as it had been in the Maple package) the Omega operator for inequality has been implemented in the new package, but also the Omega operator for equality.

Since we found the procedures for setting up the crude generating functions provided in Axel Riese's `Omega2` package very useful, we also included them in this package. However, we did not use any of the elimination procedures from `Omega2`.

After instructions for the installation of the package and the listing of the provided procedures, we give a description of each of these procedures with some examples.

In the last section of this chapter we prove some identities which could not be derived with the `Omega2` package.

4.2 Installation

The implementation consists of the Mathematica source code file `GenOmega.m` which can be downloaded at [34].

Make sure Mathematica finds the directory where the file is located.

The package can be loaded by typing the Mathematica command `<<GenOmega.m`. Subsequently, all the global functions of the package are available.

4.3 The Global Functions

The package provides six global functions which can be divided into two parts.

1. *Setting up the crude generating function:*

   `Omega`
   `OmegaEq`
   `OSum`
   `OEqSum`

2. *Applying the Omega operators for inequality and equality:*

   `Geno`
   `GenoEq`

Remark 4.3.1 The global functions of the first part, namely `Omega`, `OmegaEq`, `OSum` and `OEqSum`, are taken from Axel Riese's `Omega2` package.

4.3.1 Omega

The function `Omega[f,{t1,...,tr}]` denotes the Omega operator $\Omega_{\geq}$ acting on `f` with respect to the variables `t1`,...,`tr`.

Example.

```
In[1]:=  Omega[1/((1-q*t1)(1-2*t2)), {t1,t2}]
```

```
Out[1]=  Ω≥,t1,t2 1/((1-q t1)(1-2 t2))
```

```
In[2]:=  f := Σ_{i=0}^{∞} Σ_{j=0}^{∞} (q*t1)^i (2*t2)^j;
         Omega[f, {t1,t2}]
```

```
Out[3]=  Ω≥,t1,t2 1/((1-q t1)(1-2 t2))
```

If the second argument of `Omega` contains only one element `t`, the command `Omega[f,t]` also works.

Example.

```
In[4]:= Omega[1/(1-q*t), t]
```

$$\texttt{Out[4]=}\quad \Omega_{\geq,t}\,\frac{1}{(1-q\,t)}$$

4.3.2 OmegaEq

The function `OmegaEq[f,{t1,...,tr}]` denotes the Omega operator $\Omega_{=}$ acting on `f` with respect to the variables `t1`,...,`tr`.

Example.

```
In[1]:= OmegaEq[1/((1-q*t1)(1-2*t2)), {t1,t2}]
```

$$\texttt{Out[1]=}\quad \Omega_{=,t_1,t_2}\,\frac{1}{(1-q\,t_1)(1-2\,t_2)}$$

```
In[2]:= f := Sum_{i=0}^{Infinity} Sum_{j=0}^{Infinity} (q*t1)^i (2*t2)^j;
        OmegaEq[f, {t1,t2}]
```

$$\texttt{Out[3]=}\quad \Omega_{=,t_1,t_2}\,\frac{1}{(1-q\,t_1)(1-2\,t_2)}$$

If the second argument of `OmegaEq` contains only one element `t`, the command `OmegaEq[f,t]` also works.

Example.

```
In[4]:= OmegaEq[1/(1-q*t), t]
```

$$\texttt{Out[4]=}\quad \Omega_{=,t}\,\frac{1}{(1-q\,t)}$$

4.3.3 OSum

The procedure `OSum[expr,{ineq1,...,ineqr},z]` sets up the crude generating function of `expr` with summation variables satisfying the ineqalities `ineq1`,...,`ineqr` and Omega variables `z1`,...,`zr`. For each inequality the procedure introduces an Omega variable `zi`. The variable `z` is a Mathematica

symbol and `expr` is a power product with Mathematica symbols as basis and the summation variables as exponents. Unless specified explicitly by the inequalities, all summation variables are assumed to range over the nonnegative integers. The output is of the form `Omega[f,{z1,...,zr}]`. To evaluate the result, apply the procedure `Geno`.

Example.

```
In[1]:=  OSum[x^a y^b u^c, {a≤b,b≤c}, z]

         Assuming a ≥ 0
         Assuming b ≥ 0
         Assuming c ≥ 0
```

$$\texttt{Out[1]=}\quad \Omega_{\geq,\, z_1,\, z_2}\, \frac{1}{(1-\frac{x}{z_1})(1-\frac{y\, z_1}{z_2})(1-u\, z_2)}$$

4.3.4 OEqSum

The procedure `OEqSum[expr,{eq1,...,eqk,ineq1,... ineqr},z]` sets up the crude generating function of `expr` with summation variables satisfying the equalities `eq1`,...,`eqk` and ineqalities `ineq1`,...,`ineqr`, and with Omega variables `z1`,...,`z(k+r)`. For each equality and inequality the procedure introduces an Omega variable `zi`. The variable `z` is a Mathematica symbol and `expr` is a power product with Mathematica symbols as basis and the summation variables as exponents. Unless specified explicitly by the `ineqi`, all summation variables are assumed to range over the nonnegative integers. The output is of the form `OmegaEq[f,{z1,...,z(k+r)}]`. To evaluate the result, apply the procedure `GenoEq`.

Example.

```
In[1]:=  OEqSum[x^a y^b u^c, {a+b==0,c≥1}, z]

         Assuming a ≥ 0
         Assuming b ≥ 0
```

$$\texttt{Out[1]=}\quad \Omega_{=,\, z_1}\, \frac{u}{(1-u)(1-x\, z_1)(1-y\, z_1)}$$

```
In[2]:= OEqSum[x^a y^b u^c, {a+b+c==0,c≥1}, z]

        Assuming a ≥ 0
        Assuming b ≥ 0
```

$$\texttt{Out[2]=}\quad \Omega_{=,\,z_1}\,\frac{u\,z_1}{(1-u\,z_1)(1-x\,z_1)(1-y\,z_1)}$$

4.3.5 Geno

Depending on the number and type of the input parameters, there are four procedures with the name `Geno`.

4.3.5.1 Geno[A,B,U,t]

The procedure `Geno[A,B,U,t]` eliminates the Omega variable `t` in $\Omega_{\geq,\,t}\frac{U(t)}{A(t)B(1/t)}$, where `A` and `B` are two polynomials in `t` with $A(1) \neq 0$ and where for all roots x_i of `A` and all roots y_j of `B` the constraint $x_i y_j \neq 1$ holds, and where `U` is a Laurent polynomial in `t` (see the notation of Problem 3.1.1). It is not necessary that `A` and `B` have constant terms $\neq 0$.

The procedure corresponds to the function `GENO(A,B,U,t)` in Han's Maple package, except that several bugs have been eliminated and general polynomials `A` and `B` in `t` are allowed as input. In Han's Maple package the procedure only works for certain polynomials as will be seen in the examples that follow.

Example.

As a first example let us compute the Omega expression $\Omega_{\geq,\,\lambda}\frac{\lambda^2}{(1-2\lambda)(1-5/\lambda)}$.

```
In[1]:= Geno[1-2λ, 1-5λ, λ^2, λ]
```

$$\texttt{Out[1]=}\quad -\frac{29}{9}$$

The function `GENO` in Han's Maple package does not work in the cases where the input polynomial `B` equals 1. For example, if you try to eliminate λ in the Omega expression $\Omega_{\geq,\,\lambda}\frac{\lambda}{1-5\lambda+3\lambda^2}$, it will not be able to derive a solution.

```
> GENO(1-5*l+3*l^2, 1, l, l);

  Error,(in Omega_Frac) numeric exception:  division by
  zero
```

Another error occurs in the computation of $\Omega_{\geq,\lambda}\frac{\lambda}{(1-2\lambda)(1-5\lambda)}$.

```
> GENO((1-2*l)*(1-5*l), 1, l, l);

  Error,(in OmegaLH) invalid subscript selector
```

The new Mathematica package on the other hand is able to handle these expressions.

```
In[2]:=  Geno[1-5λ+3λ^2, 1, λ, λ]

Out[2]=  −1

In[3]:=  Geno[(1-2λ)(1-5λ), 1, λ, λ]

Out[3]=  1/4
```

Similarly, the Maple package cannot handle examples where at least one of the input polynomials `A` and `B` is a polynomial in `t` with low degree greater than zero.

Example.

```
> GENO(l-2*l^2, 1-5*l, 1, l);

  Error,(in Omega_Frac) numeric exception:  division by
  zero
```

Since

$$\Omega_{\geq,\lambda}\frac{1}{(\lambda-2\lambda^2)(1-5/\lambda)} = \Omega_{\geq,\lambda}\frac{\lambda^{-1}}{(1-2\lambda)(1-5/\lambda)},$$

this problem can be avoided.

```
> GENO(1-2*l, 1-5*l, l^(-1), l);

  2/9
```

Nevertheless, it would be nice if the procedure could handle it automatically. The Mathematica package is able to do so.

```
In[4]:=  Geno[λ-2λ^2, 1-5λ, 1, λ]

Out[4]=  2/9
```

4.3.5.2 `Geno[f,{t1,...,tr}]`

The procedure `Geno[f,{t1,...,tr}]` successively eliminates `t1,...,tr` in $\Omega_{\geq;t_1,\ldots,t_r} f(t_1,\ldots,t_r)$.

The procedure is not the same as the function `GENO(f,[t1,...,tr],q)` in Han's Maple package.

Remark 4.3.2 If only one Omega variable is present one can treat the series involved formally as is mentioned in the description of Han's algorithm above. But if there are several Omega variables present one has to take care. If we treat the series formally, we get for example the following:

$$\sum_{n=0}^{\infty} q^n\lambda^n = \frac{1}{1-q\lambda} = \frac{q\lambda}{1-q\lambda} + 1 = -\frac{1}{1-\frac{1}{q\lambda}} + 1 = -\sum_{n=1}^{\infty} q^{-n}\lambda^{-n}, \qquad (4.1)$$

but

$$\Omega_{\geq,\lambda} \sum_{n=0}^{\infty} q^n\lambda^n = \sum_{n=0}^{\infty} q^n = \frac{1}{1-q}$$

and

$$\Omega_{\geq,\lambda} - \sum_{n=1}^{\infty} q^{-n}\lambda^{-n} \;=\; 0.$$

Hence, it can happen that in the elimination process of the i-th Omega variable the series is changed in the way mentioned in (4.1) and the elimination of the $(i+1)$-th variable λ cannot be done correctly. Therefore, in this case it is essential to let Mathematica know that it has to keep track of whether λ or $1/\lambda$ is in the factors of the numerator and denominator, if λ is an Omega variable that should be eliminated in a later step.

Example.

We compute

$$\Omega_{\geq;\lambda_1,\lambda_2} \frac{1}{(1-\lambda_1/\lambda_2)\,(1+1/\lambda_1)}.$$

If we just eliminate λ_1 and λ_2 successively, we obtain

```
In[1]:=  Geno[1-λ1/λ2, 1+λ1, 1, λ1]

Out[1]=  λ2^2/((-1+λ2) (1+λ2))

In[2]:=  Geno[(-1+λ2)(1+λ2), 1, λ2^2, λ2]

         Power::infy :  Infinite expression 1/0 encountered.

         ∞::indet :  Indeterminate expression 0
         ComplexInfinity encountered.

         Power::infy :  Infinite expression 1/0 encountered.

Out[2]=  Indeterminate.
```

In the elimination step of λ_1 we have to tell Mathematica that we have $1/\lambda_2$ in the Omega expression, since λ_2 is eliminated in a later step. We accomplish this by substituting $1/\lambda_2$ by μ_2. Then we eliminate λ_1.

```
In[3]:=  Geno[1-λ1 μ2, 1+λ1, 1, λ1]

Out[3]=  -1/((-1+μ2) (1+μ2))
```

Next, λ_2 has to be eliminated, hence μ_2 has to be substituted by $1/\lambda_2$.

```
In[4]:=  Geno[1, (-1+λ2)(1+λ2), -1, λ2]

Out[4]=  1
```

This is the way `Geno[f,{t1,...,tr}]` proceedes.

```
In[5]:=  Geno[1/((1-λ1/λ2) (1+1/λ1)), {λ1, λ2}]

Out[5]=  1
```

Example.

We compute

$$\Omega_{\geq;\,\lambda_1,\,\lambda_2,\,\lambda_3}\frac{1}{(1-\lambda_1/(\lambda_2\,\lambda_3))\,(1+\lambda_2/\lambda_1)\,(1-2\lambda_3)}.$$

Again, if we just eliminate λ_1 and λ_2 successively, we obtain

```
In[1]:= Geno[1-λ1/(λ2 λ3), 1+λ2 λ1, 1/(1-2λ3), λ1]
```

$$\text{Out[1]} = -\frac{\lambda_2\,\lambda_3^2}{(1+\lambda_3)\,(-1+2\lambda_3)\,(-1+\lambda_2\,\lambda_3)}$$

```
In[2]:= Geno[-1+λ2 λ3, 1, -λ2 λ3^2/((1 + λ3)(-1 + 2λ3)), λ2]
```

$$\text{Out[2]} = -\frac{\lambda_3^2}{(-1+\lambda_3)\,(1+\lambda_3)\,(-1+2\lambda_3)}$$

```
In[3]:= Geno[(-1+λ3)(1+λ3)(-1+2λ3), 1, -λ3^2, λ3]

        Power::infy :  Infinite expression 1/0 encountered.

        ∞::indet :  Indeterminate expression 0
        ComplexInfinity encountered.

        Power::infy :  Infinite expression 1/0 encountered.

Out[3]= Indeterminate.
```

In the elimination step of λ_1 we have to tell Mathematica that we have $1/\lambda_2$ and $1/\lambda_3$ in the omega expression, since λ_2 and λ_3 are eliminated in later steps. As before, we accomplish this by substituting $1/\lambda_2$ by μ_2 and $1/\lambda_3$ by μ_3. Then we eliminate λ_1.

```
In[4]:= Geno[1-λ1 μ2 μ3, 1+λ2 λ1, 1/(1-2λ3), λ1]
```

$$\text{Out[4]} = \frac{1}{(-1+2\lambda_3)\,(-1+\mu_2\,\mu_3)\,(1+\lambda_2\,\mu_2\,\mu_3)}$$

Next, λ_2 has to be eliminated, hence μ_2 has to be substituted by $1/\lambda_2$. We leave μ_3 as it is.

```
In[5]:= Geno[1, -1+λ2 μ3, 1/((-1+2λ3)(1+μ3)), λ2]
```

$$\text{Out[5]} = -\frac{1}{(-1+2\lambda_3)\,(1+\mu_3)}$$

In the last step λ_3 has to be eliminated, hence μ_3 has to be substituted by $1/\lambda_3$.

```
In[6]:= Geno[-1+2λ3, 1+λ3, -1, λ3]
```

$$\text{Out[6]} = -\frac{1}{3}$$

The procedure `Geno[f,{t1,...,tr}]` yields

`In[7]:=` `Geno[`$\frac{1}{(1-\lambda_1/(\lambda_2\,\lambda_3))\,(1+\lambda_2/\lambda_1)\,(1-2\lambda_3)}$`, {`$\lambda_1, \lambda_2, \lambda_3$`}]`

`Out[7]=` $-\frac{1}{3}$

The procedure `GENO(f,[t1,...,tr],q)` in Han's Maple package makes use of the fact that Maple factors in such a way that it "remembers" if there was a t or a $1/t$ in the denominator of the original expression for every variable t. For example,

```
> factor(1/((1-2*t)*(1-5*t+5*t^2)));
```

$$-\frac{1}{(-1+2t)(1-5t+5t^2)}$$

```
> factor(1/((1-2*t)*(1-5/t+5/t^2)));
```

$$-\frac{t^2}{(-1+2t)(t^2-5t+5)}\,.$$

In Han's Maple package actually the function `normal` is used.

```
> normal(1/((1-2*t)*(1-5*t+5*t^2)));
```

$$-\frac{1}{(-1+2t)(1-5t+5t^2)}$$

```
> normal(1/((1-2*t)*(1-5/t+5/t^2)));
```

$$-\frac{t^2}{(-1+2t)(t^2-5t+5)}\,.$$

As one can see, if there is $1/t$ in a factor of the original expression, the exponents of t occur in decreasing order in the corresponding factor and if there is t in a factor of the original expression, the exponents of t occur in increasing order in the corresponding factor. To Mathematica, this makes no difference. Hence, we chose the way explained above to realize a procedure where several Omega variables could be eliminated automatically.

Remark 4.3.3 As mentioned in Section 3.4, the procedure only works if the Omega expressions are valid (see Problem 3.1.1) in every step. Otherwise, an error message is displayed.

```
In[1]:= Geno[1/(λ^2+1+λ^-2), λ]

        Error in the elimination of λ.
        1/(1+1/λ^2+λ^2) is not a valid Omega expression.

Out[1]= $Failed.
```

Unfortunately, this cannot be guaranteed even if the original expression is valid with respect to every Omega variable. Consider for example the Omega expression

$$\Omega_{\geq;\,\lambda_1,\,\lambda_2,\,\lambda_3}\frac{1}{(1-\lambda_1^{-1}\lambda_2\lambda_3)(1-x\lambda_1-x\lambda_1\lambda_2^{-1}-x\lambda_1\lambda_3^{-1}-x\lambda_1\lambda_2^{-1}\lambda_3^{-1})},$$

which is the crude generating function of

$$\sum_{k=0}^{n}\left(\sum_{j=0}^{k}\binom{n}{k}\right)^2.$$

This expression is valid with respect to every λ_i. Nevertheless, in the elimination step of λ_3 an error occurs, since the Omega expression in the corresponding elimination step is not valid with respect to λ_3.

```
In[2]:= Geno[1/((1-λ1^-1 λ2 λ3)(1-x λ1-x λ1 λ2^-1-x λ1 λ3^-1-x λ1 λ2^-1 λ3^-1)),
        {λ1,λ2,λ3}]

Out[2]= Error in the elimination of λ3.
        -(-1+x+x λ3)/((-1+2x+x/λ3+x λ3)(-1+2x+2x λ3)) is not a
        valid Omega expression.
```

Other conditions are violated in the computation of the Omega expression

$$\Omega_{\geq;\,\lambda_1,\,\lambda_2}\frac{1}{(1-2\lambda_1\lambda_2)(1-\frac{1}{2\lambda_2})(1-\frac{2}{\lambda_1})}.$$

The conditions on the x_i and y_j are valid for both Omega variables λ_1 (with $x_1 = 2\lambda_2$ and $y_1 = 2$) and λ_2 (with $x_1 = 2\lambda_1$ and $y_1 = \frac{1}{2}$), respectively. If we eliminate λ_1 first, we get

```
In[3]:= Geno[1/((1-2λ1 λ2)(1-1/2 μ2)(1-2/λ1)), λ1]

Out[3]= -2/((-1+2λ2)(-1+4λ2)(-2+μ2)).
```

This expression equals $\frac{1}{(1-2\lambda_2)(1-4\lambda_2)(1-\frac{1}{2\lambda_2})}$, hence $x_1 = 2$, $x_2 = 4$ and $y_1 = \frac{1}{2}$. If we now try to eliminate λ_2 from this expression, we get

```
In[4]:=  Geno[-2/((-1+2λ2)(-1+4λ2)(-1+1/λ2)), λ2]

         One of the roots x_i of A(λ2) and one of the roots
         y_j of B(λ2) in -2/((-1+1/λ2)(-1+2λ2)(-1+4λ2)) do not
         satisfy x_i y_j ≠ 1.

Out[4]=  $Failed.
```

If we instead eliminate λ_2 first, we obtain

```
In[5]:=  Geno[1/((1-2λ1λ2)(1-1/(2λ2))(1-2μ1)), λ2]

Out[5]=  -1/((-1+λ1)(-1+2λ1)(-1+2μ1)).
```

This expression equals $\frac{1}{(1-\lambda_1)(1-2\lambda_1)(1-2/\lambda_1)}$, hence $x_1 = 1$, $x_2 = 2$ and $y_1 = 2$. If we try to eliminate λ_2 from this expression, we get

```
In[6]:=  Geno[-1/((-1+λ1)(-1+2λ1)(-1+2/λ1)), λ1]

         The denominator of -1/((-1+2/λ1)(-1+λ1)(-1+2λ1))
         has 1 - λ1 as a factor.

Out[6]=  $Failed.
```

See also Section 4.4.3.

4.3.5.3 `Geno[f,t]`

The procedure `Geno[f,t]` eliminates `t` in $\Omega_{\geq,t} f(t)$.

The procedure is not the same as the function `GENO(f,t,q)` in Han's Maple package.

`Geno[f,t]` corresponds to `Geno[f,{t}]`.

Example.

We compute $\Omega_{\geq,\lambda} \frac{1}{(1-2\lambda)\,(1-5\lambda)\,(1-a)}$.

```
In[1]:=  Geno[1/((1-2λ)(1-5λ)(1-a)), λ]
```

$$\texttt{Out[1]=}\quad -\frac{1}{4(-1+a)}$$

We can verify the solution with `Geno[A,B,U,t]` where `A` $= (1-2\lambda)(1-5\lambda)$, `B` $= 1$, `U` $= 1/(1-a)$ and `t` $= \lambda$.

```
In[2]:=  Geno[(1-2λ)(1-5λ), 1, 1/(1-a), λ]
```

$$\texttt{Out[2]=}\quad -\frac{1}{4(-1+a)}$$

4.3.5.4 `Geno[Omega[f,{t1,...,tr}]]`

In the case of the crude generating function being computed by `OSum` the procedure `Geno` can be applied directly to the output.

Example.

```
In[1]:=  OSum[x^a y^b u^c, {a≤b,b≤c}, z]

         Assuming a ≥ 0
         Assuming b ≥ 0
         Assuming c ≥ 0
```

$$\texttt{Out[1]=}\quad \Omega_{\geq,\, z_1,\, z_2}\, \frac{1}{(1-\frac{x}{z_1})(1-\frac{y\, z_1}{z_2})(1-u\, z_2)}$$

```
In[2]:=  Geno[%]
```

$$\texttt{Out[2]=}\quad -\frac{x}{(-1+u)(-1+x)(-1+xy)}$$

In the case of there being only one Omega variable `t`, `Geno` also works with an input of the form `Omega[f,t]`.

Example.

```
In[1]:=  OSum[x^a y^b, {a≤b}, z]

         Assuming a ≥ 0
         Assuming b ≥ 0
```

```
Out[1]=  Ω_{≥,z_1} 1/((1 - x/z_1)(1 - y z_1))

In[2]:=  Geno[%]

Out[2]=  1/((-1+y)(-1+xy))

In[3]:=  Geno[Omega[1/((1-x/z)(1-yz)),z]]

Out[3]=  1/((-1+y)(-1+xy))
```

4.3.6 GenoEq

In his Maple package Han only implemented procedures for the Omega operator for inequality. In the new Mathematica version for every `Geno` procedure there also exists a corresponding `GenoEq` procedure for the equality operator. This has been accomplished by making use of the identity

$$\Omega_{=,\lambda} F(\lambda) = \Omega_{\geq,\lambda} F(\lambda) + \Omega_{\geq,\lambda} F(1/\lambda) - F(1).$$

4.3.6.1 `GenoEq[A,B,U,t]`

The procedure `GenoEq[A,B,U,t]` eliminates the Omega variable `t` in $\Omega_{=,t} \frac{U(t)}{A(t)B(1/t)}$, where `A` and `B` are two polynomials in `t` with $A(1) \neq 0$ and where for all roots x_i of `A` and all roots y_j of `B` the constraint $x_i y_j \neq 1$ holds, and where `U` is a Laurent polynomial in `t` (see the notation of Problem 3.1.1). As with `Geno[A,B,U,t]`, it is not necessary that the constant terms of `A` and `B` are $\neq 0$.

Example.

As a first example let us compute the Omega expression $\Omega_{=,\lambda} \frac{\lambda^2}{(1-2\lambda)(1-5/\lambda)}$.

```
In[1]:=  GenoEq[1-2λ, 1-5λ, λ^2, λ]

Out[1]=  -25/9
```

As before, the procedure also works if `B` equals 1, and for polynomials `A` and `B` with constant term 0.

```
In[1]:=  GenoEq[1-5λ+3λ^2, 1, λ, λ]
```

```
Out[1]=  0
```

```
In[2]:=  GenoEq[(1-2λ)(1-5λ), 1, λ, λ]
```

```
Out[2]=  0
```

```
In[3]:=  GenoEq[1, (1-2λ)(1-5λ), λ, λ]
```

```
Out[3]=  7
```

```
In[4]:=  GenoEq[λ-2λ^2, 1-5λ, 1, λ]
```

```
Out[4]=  -2/9
```

4.3.6.2 GenoEq[f,{t$_1$,...,t$_r$}]

The procedure GenoEq[f,{t$_1$,...,t$_r$}] successively eliminates t$_1$,...,t$_r$ in $\Omega_{=;t_1,\ldots,t_r} f(t_1,\ldots,t_r)$.

The successive application of the Omega operator for equality is realised in the same way as in the case for inequality (see Section 4.3.5.2).

Example.

We compute $\Omega_{=;\lambda_1,\lambda_2} \frac{1}{(1-2\lambda_1+3\lambda_2)(1-a)}$.

```
In[1]:=  GenoEq[1/((1-2λ1+3λ2)(1-a/λ2)), {λ1,λ2}]
```

```
Out[1]=  1/(1+3a)
```

We can verify the solution by successively applying Geno[A,B,U,t] where again we have to be careful with Omega variables that should be eliminated in a later step.

Since λ_2 will be eliminated later on, we set $\mu_2 := 1/\lambda_2$.

```
In[2]:=  Geno[1-2λ1+3λ2, 1, 1/(1-aμ2), λ1]
```

```
Out[2]=  -1/((1+3λ2)(-1+aμ2))
```

We substitute $1/\lambda_2$ for μ_2.

In[3]:= Geno[1+3λ_2, -1+aλ_2, -1, λ_2]

Out[3]= $\frac{1}{1+3a}$

4.3.6.3 GenoEq[f,t]

The procedure GenoEq[f,t] eliminates t in $\Omega_{=,t} f(t)$.

GenoEq[f,t] corresponds to GenoEq[f,{t}].

Example.

We compute $\Omega_{=,\lambda} \frac{1}{(1-2\lambda)(1-5\lambda)(1-a)}$.

In[1]:= GenoEq[1/((1-2λ)(1-5λ)(1-a)), λ]

Out[1]= $-\frac{1}{(-1+a)}$

We can verify the solution with the function GenoEq[A,B,U,t] where A $= (1-2\lambda)(1-5\lambda)$, B $= 1$, U $= 1/(1-a)$ and t $= \lambda$.

In[2]:= Geno[(1-2λ)(1-5λ), 1, 1/(1-a), λ]

Out[2]= $-\frac{1}{(-1+a)}$

4.3.6.4 GenoEq[OmegaEq[f,{t_1,...,t_r}]]

In the case of the crude generating function being computed by OEqSum, the procedure GenoEq can be applied directly to the output.

Example.

In[1]:= OEqSum[x^a y^b u^c, {a+b==0,c$\geq$1}, z]

Assuming a $\geq$ 0
Assuming b $\geq$ 0

Out[1]= $\Omega_{=,z_1} \frac{u}{(1-u)(1-x\,z_1)(1-y\,z_1)}$

In[2]:= GenoEq[%]

Out[2]= $-\frac{u}{(-1+u)}$

4.4 Applications

4.4.1 Standard Type

The following Problem is taken from a seminar talk [27] given by Prof. Volker Strehl in November 2008 at RISC.

Problem 4.4.1 *Let $k \in \mathbb{N}$. Find a closed form for the sum*

$$\sum_{0 \le i < j < l \le k} (-1)^{i+j+l}(2i+1)(2j+1)(2l+1).$$

We accomplish this with the help of the new GenOmega package. Note: `Omega2` is also able to solve this problem.

Consider the generating function

$$F(x) = \sum_{k=0}^{\infty} \sum_{0 \le i < j < l \le k} (-1)^{i+j+l}(2i+1)(kj+1)(2l+1)x^k.$$

We have the following three constraints on i, j and l:

$$\begin{aligned} j &\ge i+1 \\ l &\ge j+1 \\ k &\ge l. \end{aligned}$$

Now we can set up the crude generating function for this problem.

$$\begin{aligned} F(x) &= \\ &= \Omega_{\ge;\lambda_1\,\lambda_2\,\lambda_3} \sum_{k,\,i,\,j,\,l \ge 0} (-1)^{i+j+l}(2i+1)(kj+1)(2l+1)x^k \lambda_1^{j-i-1}\lambda_2^{l-j-1}\lambda_3^{k-l} \\ &= \Omega_{\ge;\lambda_1\,\lambda_2\,\lambda_3} \frac{1-\lambda_1^{-1}}{(1+\lambda_1^{-1})^2} \frac{1-\lambda_1\lambda_2^{-1}}{(1+\lambda_1\lambda_2^{-1})^2} \frac{1-\lambda_2\lambda_3^{-1}}{(1+\lambda_2\lambda_3^{-1})^2} \frac{1}{\lambda_1\lambda_2} \frac{1}{1-\lambda_3 x}. \end{aligned}$$

We solve this by applying the procedure `Geno[f,{t1,...,tr}]` to $F(x)$ and $\{\lambda_1, \lambda_2, \lambda_3\}$. We obtain

$$F(x) = \frac{(-15+x)x^2}{(1+x)^5}.$$

To derive the desired closed form for the sum in Problem 4.4.1, we extract the coefficient of x^k in the series expansion of $F(x)$:

$$\langle x^k\rangle F(x) = \langle x^{k-3}\rangle \frac{1}{(1+x)^5} - 15\langle x^{k-2}\rangle \frac{1}{(1+x)^5}.$$

Since for all $r, s \in \mathbb{N}$

$$\langle u^r\rangle \left(\frac{1}{(1-ut)^s}\right) = \binom{r+s-1}{s-1} t^r,$$

we get

$$\begin{aligned}
\langle x^k\rangle F(x) &= \binom{k+1}{4}(-1)^{k-3} - 15\binom{k+2}{4}(-1)^{k-2} \\
&= \binom{k+1}{4}(-1)^{k-3} + 15\binom{k+2}{4}(-1)^{k-3} \\
&= \left(\binom{k+1}{4} + 15\binom{k+2}{4}\right)(-1)^{k-3}.
\end{aligned}$$

Hence, we obtain the following theorem.

Theorem 4.4.2 *Let $k \in \mathbb{N}$. Then*

$$\begin{aligned}
&\sum_{0\le i<j<l\le k} (-1)^{i+j+l}(2i+1)(kj+1)(2l+1) \\
&\quad = \left(\binom{k+1}{4} + 15\binom{k+2}{4}\right)(-1)^{k-3}.
\end{aligned}$$

This example can also be treated with symbolic summation methods. The following examples are a bit more "off the beaten track".

4.4.2 New Identities

In this section we will prove identities which could not be derived with the `Omega2` package. The identities proven here are variations of identities proven in Han's paper [17].

Proposition 4.4.3 *For natural numbers $n \geq 1$ we have*

$$\sum_{\substack{k,\,i,\,j\geq 0\\ 6k+2j-2i\geq 3n\\ n\geq k}} \binom{k}{i}\binom{n-k}{j} = 2^{n-3}(5+(-1)^n+4n).$$

For $n \in \mathbb{N}$ the following holds:

$$\sum_{\substack{k,\,i,\,j\geq 0\\ 6k+2j-2i\geq 3n\\ n\geq k}} (-1)^{n-k}\binom{k}{i}\binom{n-k}{j} = 2^{n-1}+(-1)^{n+1}(-2)^{\lfloor (n-1)/2\rfloor},$$

and, as a special case with $n \to 2n$,

$$\sum_{\substack{k,\,i,\,j\geq 0\\ 6k+2j-2i\geq 6n\\ 2n\geq k}} (-1)^{k}\binom{k}{i}\binom{2n-k}{j} = 2^{2n-1}-(-2)^{n-1}.$$

Proof. We consider the following generating function:

$$\begin{aligned}
F(x,y) &= \sum_{\substack{m,\,k,\,i,\,j\geq 0\\ 3k+2j\geq 3m+2i}} \binom{k}{i}\binom{m}{j}x^k y^m \\
&= \Omega_{\geq,\lambda} \sum_{m,\,k,\,i,\,j\geq 0} \binom{k}{i}\binom{m}{j}x^k y^m \lambda^{3k-2i-3m+2j} \\
&= \Omega_{\geq,\lambda} \sum_{m,\,k,\,i,\,j\geq 0} \binom{k}{i}\binom{m}{j}x^{i+k-i} y^{j+m-j} \lambda^{i+3k-3i}\lambda^{-j-3m+3j} \\
&= \Omega_{\geq,\lambda} \sum_{m,\,k,\,i,\,j\geq 0} \binom{k}{i}(x\lambda)^i (x\lambda^3)^{k-i}\binom{m}{j}(y/\lambda)^j (y/\lambda^3)^{m-j}
\end{aligned}$$

$$\begin{aligned}
&= \Omega_{\geq,\lambda} \sum_{m,\,k\geq 0} (x\lambda+x\lambda^3)^k (y/\lambda+y/\lambda^3)^m \\
&= \Omega_{\geq,\lambda} \frac{1}{(1-x\lambda-x\lambda^3)(1-y/\lambda-y/\lambda^3)}.
\end{aligned}$$

Using the Mathematica package `GenOmega` we obtain,

$$F(x,y) = \frac{1-xy+x^2y+x^3y+xy^2-x^2y^2}{(-1+2x)(1-4xy+x^3y-2x^2y^2+xy^3)}. \tag{4.2}$$

Setting $y = x$ in (4.2) yields

$$F(x,x) = \frac{1}{4} + \frac{1}{2(-1+2x)^2} - \frac{1}{8(-1+2x)} + \frac{1}{8(1+2x)}.$$

We now compute the n-th coefficient ($n \geq 1$) in the series expansion of $F(x,x)$ in x:

$$\begin{aligned}
\langle x^n \rangle F(x,x) &= \langle x^n \rangle \left(\frac{1}{2} \sum_{i,\, j \geq 0} (2x)^{i+j} + \frac{1}{8} \sum_{i \geq 0} (2x)^i + \frac{1}{8} \sum_{i \geq 0} (-2x)^i \right) \\
&= \frac{1}{2}(n+1)2^n + \frac{1}{8}2^n + \frac{1}{8}(-2)^n \\
&= 2^{n-3}(5 + (-1)^n + 4n).
\end{aligned}$$

This completes the proof for the first identity in Proposition 4.4.3, if one sets $m = n - k$.

For the proof of the second and the third identities in Proposition 4.4.3 we put $y = -x$ in (4.2) and obtain

$$F(x,-x) = -\frac{1}{2(-1+2x)} + \frac{1+2x}{2(1+2x^2)}.$$

Again, we compute the n-th coefficient in the series expansion of $F(x,-x)$ in x:

$$\begin{aligned}
\langle x^n \rangle F(x,-x) &= \langle x^n \rangle \left(\frac{1}{2} \sum_{i \geq 0} (2x)^i + \frac{1+2x}{2} \sum_{i \geq 0} (-2x^2)^i \right) \\
&= \begin{cases} \frac{1}{2}2^n + \frac{1}{2}(-2)^{n/2}, & \text{if } n \text{ is even,} \\ \frac{1}{2}2^n + (-2)^{(n-1)/2}, & \text{if } n \text{ is odd} \end{cases} \\
&= \begin{cases} 2^{n-1} - (-2)^{n/2-1}, & \text{if } n \text{ is even,} \\ 2^{n-1} + (-2)^{(n-1)/2}, & \text{if } n \text{ is odd} \end{cases} \\
&= 2^{n-1} + (-1)^{n+1}(-2)^{\lfloor (n-1)/2 \rfloor}.
\end{aligned}$$

This completes the proof for the second identity by setting $m = n - k$ and for the third identity by setting $m = 2n - k$. ■

Similarly, the following identities can be derived.

Proposition 4.4.4 *For $n \in \mathbb{N}$ the following holds:*

$$\begin{aligned}&\sum_{\substack{i,j,k,l,m\geq 0\\ 2n\geq 4l+i+k-j\\ n\geq m+l}} \binom{m}{i}\binom{l}{j}\binom{n-(m+l)}{k}\\ &\quad = \frac{1}{216}\left(8(-1)^n(7+3n)+2^n(5+3n)(32+27n)\right).\end{aligned}$$

For natural numbers $n \geq 1$ we have

$$\begin{aligned}&\sum_{\substack{i,j,k,l,m\geq 0\\ 2n\geq 4l+i+k-j\\ n\geq m+l}} (-1)^m\binom{m}{i}\binom{l}{j}\binom{n-(m+l)}{k}\\ &\quad = \frac{1}{12}\left(3(-2)^n+2(-1)^n+2^n(4+3n)\right)-\frac{1}{2}(-3)^{\lfloor (n-1)/2\rfloor},\end{aligned}$$

and, as a special case with $n \to 2n$,

$$\begin{aligned}&\sum_{\substack{i,j,k,l,m\geq 0\\ 4n\geq 4l+i+k-j\\ 2n\geq m+l}} (-1)^m\binom{m}{i}\binom{l}{j}\binom{2n-(m+l)}{k}\\ &\quad = \frac{1}{12}\left(2(-3)^n+3\cdot 4^n+4^{n+1}+3\cdot 2^{2n+1}n+2\right).\end{aligned}$$

Proof. We consider the following generating function:

$$\begin{aligned}F(x,y,z) &= \sum_{\substack{m,r,k,l,i,j\geq 0\\ 2r+2m\geq 2l+i+k-j}} \binom{m}{i}\binom{l}{j}\binom{r}{k}x^m y^l z^r\\ &= \Omega_{\geq,\lambda}\sum_{m,r,k,l,i,j\geq 0}\binom{m}{i}\binom{l}{j}\binom{r}{k}x^m y^l z^r \lambda^{2m-i-2l+j+2r-k}\\ &= \Omega_{\geq,\lambda}\sum_{m,r,k,l,i,j\geq 0}\binom{m}{i}\binom{l}{j}\binom{r}{k}x^{i+m-i}y^{j+l-j}z^{k+r-k}\\ &\qquad \cdot \lambda^{i+2m-2i}\lambda^{-j-2l+2j}\lambda^{k+2r-2k}\end{aligned}$$

$$\begin{aligned}
&= \Omega_{\geq,\lambda} \sum_{m,r,k,l,i,j\geq 0} \binom{m}{i} (x\lambda)^i (x\lambda^2)^{m-i} \binom{l}{j} (y/\lambda)^j (y/\lambda^2)^{l-j} \\
&\qquad \cdot \binom{r}{k} (z\lambda)^k (z\lambda^2)^{r-k} \\
&= \Omega_{\geq,\lambda} \sum_{m,l,r\geq 0} (x\lambda + x\lambda^2)^m (y/\lambda + y/\lambda^2)^l (z\lambda + z\lambda^2)^r \\
&= \Omega_{\geq,\lambda} \frac{1}{(1-x\lambda-x\lambda^2)(1-y/\lambda-y/\lambda^2)(1-z\lambda-z\lambda^2)}.
\end{aligned}$$

Using the Mathematica package `GenOmega` we obtain,

$$\begin{aligned}
F(x,y,z) &= \frac{1+x^2y-4xyz-4x^2yz-2xy^2z-3x^2y^2z-x^2y^3z+yz^2}{N} \\
&\quad + \frac{-4xyz^2-3xy^2z^2+5x^2y^2z^2-xy^3z^2+2x^2y^3z^2}{N} \qquad (4.3)
\end{aligned}$$

with $N = (-1+2x)(-1+3xy+x^2y+xy^2)(-1+2z)(-1+3yz+y^2z+yz^2)$.

Setting $y = x$ and $z = x$ in (4.3) yields

$$F(x,x,x) = \frac{1}{9(1+x)^2} + \frac{4}{27(1+x)} + \frac{3}{4(1-2x)^3} - \frac{1}{18(1-2x)^2} + \frac{5}{108(1-2x)}.$$

We now compute the n-th coefficient in the series expansion of $F(x,x,x)$ in x:

$$\begin{aligned}
&\langle x^n \rangle F(x,x,x) = \\
&= \langle x^n \rangle \left(\frac{1}{9} \sum_{i,j\geq 0} (-x)^{i+j} + \frac{4}{27} \sum_{i\geq 0} (-x)^i + \frac{3}{4} \sum_{i,j,k\geq 0} (2x)^{i+j+k} \right. \\
&\qquad \left. - \frac{1}{18} \sum_{i,j\geq 0} (2x)^{i+j} + \frac{5}{108} \sum_{i\geq 0} (2x)^i \right) \\
&= \frac{1}{9}(n+1)(-1)^n + \frac{4}{27}(-1)^n + \frac{3}{4} \sum_{i=0}^{n} (i+1) \cdot 2^n \\
&\qquad - \frac{1}{18}(n+1)2^n + \frac{5}{108} 2^n
\end{aligned}$$

$$\begin{aligned} &= \frac{1}{9}(n+1)(-1)^n + \frac{4}{27}(-1)^n + \frac{3}{4}\frac{(n+2)(n+1)}{2}2^n \\ &\qquad - \frac{1}{18}(n+1)2^n + \frac{5}{108}2^n \\ &= \frac{1}{216}\left(8(-1)^n(7+3n) + 2^n(5+3n)(32+27n)\right). \end{aligned}$$

This completes the proof for the first identity in Proposition 4.4.4, if one sets $r = n - (m + l)$.

For the proof of the second and the third identities in Proposition 4.4.4 we set $y = x$ and $z = -x$ in (4.3) and obtain

$$\begin{aligned} F(x,x,-x) &= \frac{1}{12} + \frac{1}{6(1+x)} + \frac{1}{4(1-2x)^2} + \frac{1}{12(1-2x)} \\ &\quad + \frac{1}{4(1+2x)} + \frac{1-3x}{6(1+3x^2)}. \end{aligned}$$

Again, we compute the n-th coefficient ($n \geq 1$) in the series expansion of $F(x, x, -x)$ in x:

$$\begin{aligned} \langle x^n\rangle F(x,x,-x) &= \langle x^n\rangle \left(\frac{1}{6}\sum_{i\geq 0}(-x)^i + \frac{1}{4}\sum_{i,\,j\geq 0}(2x)^{i+j} + \frac{1}{12}\sum_{i\geq 0}(2x)^i \right. \\ &\quad \left. + \frac{1}{4}\sum_{i\geq 0}(-2x)^i + \frac{1-3x}{6}\sum_{i\geq 0}(-3x^2)^i \right) \\ &= \begin{cases} \frac{1}{6}(-1)^n + \frac{1}{4}(n+1)2^n + \frac{1}{12}2^n \\ \quad + \frac{1}{4}(-2)^n + \frac{1}{6}(-3)^{n/2}, & \text{if } n \text{ is even,} \\ \frac{1}{6}(-1)^n + \frac{1}{4}(n+1)2^n + \frac{1}{12}2^n \\ \quad + \frac{1}{4}(-2)^n - \frac{1}{2}(-3)^{(n-1)/2}, & \text{if } n \text{ is odd} \end{cases} \end{aligned}$$

$$
\begin{aligned}
&= \begin{cases}
\frac{1}{6}(-1)^n + (n+1)2^{n-2} + \frac{1}{3}2^{n-2} \\
\quad + (-2)^{n-2} - \frac{1}{2}(-3)^{n/2-1}, & \text{if } n \text{ is even,} \\
\frac{1}{6}(-1)^n + (n+1)2^{n-2} + \frac{1}{3}2^{n-2} \\
\quad + (-2)^{n-2} - \frac{1}{2}(-3)^{(n-1)/2}, & \text{if } n \text{ is odd}
\end{cases} \\
&= \frac{1}{6}(-1)^n + (n+1)2^{n-2} + \frac{1}{3}2^{n-2} + (-2)^{n-2} \\
&\qquad + \frac{1}{2}(-3)^{\lfloor (n-1)/2 \rfloor} \\
&= \frac{1}{12}\left(3(-2)^n + 2(-1)^n + 2^n(4+3n)\right) - \frac{1}{2}(-3)^{\lfloor (n-1)/2 \rfloor}.
\end{aligned}
$$

This completes the proof for the second identity by setting $r = n - (m + l)$ and for the third identity by setting $r = 2n - (m + l)$. ■

4.4.3 Calkin's Identities

In [14] Calkin proved the identity stated in the following theorem.

Theorem 4.4.5 *For $n \in \mathbb{N}$*

$$\sum_{k=0}^{n} \left(\sum_{j=0}^{k} \binom{n}{j} \right)^3 = n2^{3n-1} + 2^{3n} - \frac{3n}{4} 2^n \binom{2n}{n}.$$

In [4] Andrews et. al. presented a proof of this identity which made use of Partition Analysis. However, they had to do several proof steps by hand since the `Omega2` package was not able to derive a solution from the crude generating function corresponding to the problem automatically, because the crude generating function did not satisfy the input conditions. In an attempt to find a solution of the crude generating function automatically with the help of the new `GenOmega` package, we investigated not only Calkin's identity of order 3 but also those of orders 1 and 2.

4.4.3.1 Calkin's Identity of order 1

Our goal is to find a closed form for the sum

$$\sum_{k=0}^{n}\sum_{j=0}^{k}\binom{n}{j}.$$

We set up the corresponding crude generating function

$$\begin{aligned} F(x) &= \sum_{n=0}^{\infty}\sum_{k=0}^{n}\sum_{j=0}^{k}\binom{n}{j}x^n \\ &= \Omega_{\geq;\,\lambda_1\,\lambda_2}\sum_{n,k,j\geq 0}\binom{n}{j}x^n\lambda_1^{n-k}\lambda_2^{k-j} \\ &= \Omega_{\geq;\,\lambda_1\,\lambda_2}\,\frac{1}{(1-\lambda_1^{-1}\lambda_2)(1-x\lambda_1-x\lambda_1\lambda_2^{-1})}. \end{aligned}$$

Now we invoke `GenOmega`.

```
In[1]:=  Geno[1/((1-λ1^-1 λ2)(1-x λ1-x λ1 λ2^-1)),{λ1,λ2}]

Out[1]=  -(-1+x)/(-1+2x)^2
```

By computing the n-th coefficient in the series expansion of this solution with respect to x we obtain the desired closed form,

$$\langle x^n\rangle\left(-\frac{-1+x}{(-1+2x)^2}\right) = 2^n + n2^{n-1}.$$

Hence,

Theorem 4.4.6 *For $n \in \mathbb{N}$*

$$\sum_{k=0}^{n}\sum_{j=0}^{k}\binom{n}{j} = 2^n + n2^{n-1}.$$

4.4.3.2 Calkin's Identity of order 2

We want to find a closed form for the sum

$$\sum_{k=0}^{n}\left(\sum_{j=0}^{k}\binom{n}{j}\right)^2.$$

As before, we set up the corresponding crude generating function

$$\begin{aligned}F(x) = &\\ &= \sum_{n=0}^{\infty}\sum_{k=0}^{n}\sum_{j_1=0}^{k}\binom{n}{j_1}\sum_{j_2=0}^{k}\binom{n}{j_2}x^n\\ &= \Omega_{\geq;\,\lambda_1\,\lambda_2\,\lambda_3}\sum_{n,k,j_1,j_2\geq 0}\binom{n}{j}x^n\lambda_1^{n-k}\lambda_2^{k-j_1}\lambda_3^{k-j_2}\\ &= \Omega_{\geq;\,\lambda_1\,\lambda_2\,\lambda_3}\frac{1}{(1-\lambda_1^{-1}\lambda_2\lambda_3)(1-x\lambda_1-x\lambda_1/\lambda_2-x\lambda_1/\lambda_3-x\lambda_1/\lambda_2/\lambda_3)}.\end{aligned}$$

We invoke `GenOmega`.

```
In[2]:= Geno[1/((1-λ1^-1 λ2 λ3)(1-x λ1-x λ1/λ2-x λ1/λ3-x λ1/λ2/λ3)),
        {λ1,λ2,λ3}]

        Error in the elimination of λ3.
        -(-1+x+x λ3)/((-1+2x+x/λ3+x λ3)(-1+2x+2x λ3)) is not a
        valid Omega expression.

Out[2]= Failed
```

As mentioned in Sections 3.4 and 4.3.5.2, `GenOmega` is not able to derive a solution if the Omega expression is not valid in every elimination step. Here, λ_3 cannot be eliminated. If one changes the order in which the Omega variables should be eliminated, the problem occurs nevertheless, but then λ_1 or λ_2 might be the variables that cannot be eliminated.

4.4.3.3 Calkin's Identity of order 3

We want to find a closed form for the sum

$$\sum_{k=0}^{n}\left(\sum_{j=0}^{k}\binom{n}{j}\right)^3.$$

Again, we set up the corresponding crude generating function

$$\begin{aligned} F(x) &= \sum_{n=0}^{\infty}\sum_{k=0}^{n}\sum_{j_1=0}^{k}\binom{n}{j_1}\sum_{j_2=0}^{k}\binom{n}{j_2}\sum_{j_3=0}^{k}\binom{n}{j_3}x^n \\ &= \Omega_{\geq;\,\lambda_1\,\lambda_2\,\lambda_3\,\lambda_4}\sum_{n,k,j_1,j_2,j_3\geq 0}\binom{n}{j}x^n\lambda_1^{n-k}\lambda_2^{k-j_1}\lambda_3^{k-j_2}\lambda_4^{k-j_3} \\ &= \Omega_{\geq;\,\lambda_1\,\lambda_2\,\lambda_3\,\lambda_4}\frac{1}{(1-\lambda_1^{-1}\lambda_2\lambda_3\lambda_4)\,G(x,\lambda_1,\lambda_2,\lambda_3,\lambda_4)}, \end{aligned}$$

with

$$\begin{aligned} G(x,\lambda_1,\lambda_2,\lambda_3,\lambda_4) &= 1 - x\lambda_1 - x\lambda_1/\lambda_2 - x\lambda_1/\lambda_3 - x\lambda_1/\lambda_2/\lambda_3 \\ &\quad - x\lambda_1/\lambda_4 - x\lambda_1/\lambda_2/\lambda_4 - x\lambda_1/\lambda_3/\lambda_4 \\ &\quad - x\lambda_1/\lambda_2/\lambda_3/\lambda_4. \end{aligned}$$

Invoking `GenOmega` results in the output

```
Error in the elimination of λ3.
-(-1+x+xλ3+xλ4+xλ3λ4)/((-1+2x+x/λ3+xλ3+x/λ4+x/(λ3λ4)+xλ4+xλ3λ4)(-1+2x+2xλ3+2xλ4+2xλ3λ4))
is not a valid Omega expression.

Out[3]= Failed.
```

The problem here is exactly the same as in the case of order 2.

Therefore, we have to conclude that `GenOmega` was not able to find solutions automatically for Calkin's Identities of orders 2 and 3.

Bibliography

[1] G. E. Andrews, *MacMahon's Partition Analysis I: The Lecture Hall Partition Theorem*, in "Mathematical essays in honor of Gian-Carlo Rota", Progr. Math., Vol. 161, pp. 1–22, Birkhäuser, Boston, 1998.

[2] G. E. Andrews, *MacMahon's Partition Analysis II: Fundamental Theorems*, Ann. Comb. **4**, pp. 327–338, 2000.

[3] G. E. Andrews, P. Paule and A. Riese, *MacMahon's Partition Analysis: The Omega Package*, Europ. J. Combinatorics **22**, pp. 887–904, 2001.

[4] G. E. Andrews, P. Paule and A. Riese, *MacMahon's Partition Analysis IV: Hypergeometric Multisums*, Séminaire Lotharingien de Combinatoire **B42i**, pp. 1–24, electronic journal, 1999.

[5] G. E. Andrews, P. Paule and A. Riese, *MacMahon's Partition Analysis V: Bijections, Recursions, and Magic Squares*, In: Algebraic Combinatorics and Applications, A. Betten and others (ed.), pp. 1–39, 2001, Springer, ISBN 3-5404-1110-0.

[6] G. E. Andrews, P. Paule and A. Riese, *MacMahon's Partition Analysis VI: A New Reduction Algorithm*, Ann. Comb. **5**, pp. 251–270, 2001.

[7] G. E. Andrews, P. Paule and A. Riese, *MacMahon's Partition Analysis VII: Constrained Compositions*, AMS Contemporary Mathematics **291**, pp. 11–27, 2001.

[8] G. E. Andrews, P. Paule and A. Riese, *MacMahon's Partition Analysis VIII: Plane Partition Diamonds*, Adv. in Appl. Math. **27**, pp. 231–242, 2001.

[9] G. E. Andrews, P. Paule and A. Riese, *MacMahon's Partition Analysis IX: k-Gon Partitions*, Bull. Austral. Math. Soc. **64**, pp. 321–329, 2001.

[10] G. E. Andrews, P. Paule and A. Riese, *MacMahon's Partition Analysis X: Plane Partitions with Diagonals*, Southeast Asian Journal Mathematics and Mathematical Sciences **3**, pp. 3–14, 2004.

[11] G. E. Andrews, P. Paule and A. Riese, *MacMahon's Partition Analysis XI: The Search for Modular Forms*, SFB 013, Technical report, 2006, SFB-report 2006-27.

[12] G. E. Andrews, P. Paule and A. Riese, *MacMahon's Partition Analysis XII: Plane Partitions*, SFB 013, Technical report, September 2006, SFB-report 2006-28.

[13] A. I. Barvinok and J. Pommersheim, *An Algorithmic Theory of Lattice Points in Polyhedra*, in: *New Perspectives in Algebraic Combinatorics* (Berkeley, CA, 1996-1997), 91-147, Math. Sci. Res. Inst. Publ. 38, Cambridge Univ. Press, Cambridge, 1999.

[14] N. j. Calkin, *A Curious Binomial Identity*, Discrete Math. **131**, pp. 335–337, 1994.

[15] A. Cayley, *On an Algebraical Operation*, Quart. J. of Pure and Appl. Math. **13**, pp. 369–375, 1875. [also Coll. Math Papers of A. Cayley, Vol. 9, pp. 537–542].

[16] J. A. DeLoera, D. Haws, R. Hemmecke, P. Huggins, J. Tauzer, R. Yoshida, *A User's Guide for* `LattE` *v1.1*, 2003, software package `LattE` is available at `http://www.math.ucdavis.edu/~latte/`

[17] G. N. Han, *A General Algorithm for the MacMahon Omega Operator*, Ann. of Comb. **7**, pp. 467–480, 2003.

[18] I. G. MacDonald, *Symmetric Functions and Hall Polynomials*, second edition, Clarendon Press, Oxford, 1995.

[19] P. A. MacMahon, *Memoir on the Theory of the Partition of Numbers — Part I*, Phil. Trans. **187**, pp. 619–673, 1897, (cf., Reference 8, pp. 1026–1080).

[20] P. A. MacMahon, *Memoir on the Theory of the Partition of Numbers — Part II*, Phil. Trans. **192**, pp. 351–401, 1899, (cf., Reference 8, pp. 1138–1188).

[21] P. A. MacMahon, *Memoir on the Theory of the Partition of Numbers — Part III*, Phil. Trans. **205**, pp. 35–58, 1906, (cf., Reference 8, pp. 1255–1277).

[22] P. A. MacMahon, *Combinatory Analysis*, 2 vols., Cambridge University Press, Cambridge, 1915-1916 (Reprinted: Chelsea, New York, 1960).

[23] P. A. MacMahon, *Collected Papers*, Vol. 1, *Combinatorics*, G. E. Andrews, ed., MIT Press, Cambridge, 1978.

[24] P. A. MacMahon, *Collected Papers*, Vol. 2, *Number Theory, Invariants and Applications*, G. E. Andrews, ed., MIT Press, Cambridge, 1986.

[25] R. P. Stanley, *Linear Homogeneous Diophantine Equations and Magic Labelings of Graphs*, Duke Math. J. **40**, pp. 607–632, 1973.

[26] R. P. Stanley, *Enumerative Combinatorics*, Volume I, Cambridge, 1997.

[27] Talk given by Prof. Volker Strehl on 26.11.2008 at RISC, Johannes Kepler University, Linz
`http://www.risc.uni-linz.ac.at/research/combinat/seminar/WS08.html`

[28] `http://www.risc.uni-linz.ac.at/research/combinat/risc/software/Omega`

[29] `http://www.risc.uni-linz.ac.at/research/combinat/software/source/mathematica/Omega/OmegaDemo.nb`

[30] `http://www-irma.u-strasbg.fr/~guoniu/software`

[31] `http://www.math.ucdavis.edu/~mkoeppe/latte/`

[32] `http://www.math.ucdavis.edu/~latte/`

[33] `http://www.math.ucdavis.edu/~mkoeppe/latte/download/latte-for-tea-too-1.2-mk-0.9.3.tar.gz`

[34] `http://www.risc.uni-linz.ac.at/research/combinat/risc/software/GenOmega`