A Fast Algorithm for MacMahon's Partition Analysis

Guoce Xin *
Department of Mathematics
Brandeis University, Waltham, USA
maxima@brandeis.edu

Submitted: Aug 26, 2004; Accepted: Aug 30, 2004; Published: Sep 9, 2004
Mathematics Subject Classifications: 11Y60, 05A17

Abstract

This paper deals with evaluating constant terms of a special class of rational functions, the Elliott-rational functions. The constant term of such a function can be read off immediately from its partial fraction decomposition. We combine the theory of iterated Laurent series and a new algorithm for partial fraction decompositions to obtain a fast algorithm for MacMahon's Omega calculus, which (partially) avoids the "run-time explosion" problem when eliminating several variables. We discuss the efficiency of our algorithm by investigating problems studied by Andrews and his coauthors; our running time is much less than that of their Omega package.

1 Introduction

Zeilberger [20] proved a conjecture of Chan et al. [11] by proving an identity equivalent to

$$\operatorname*{CT}_{x_1}\cdots\operatorname*{CT}_{x_n} \frac{1}{\prod_{i=1}^{n}(1-x_i)} \frac{1}{\prod_{i<j}(x_i-x_j)} = C_1\cdots C_{n-1}, \tag{1.1}$$

where C_k's are the Catalan numbers.

This identity should be interpreted as taking *iterated constant terms* [10]; i.e., in applying CT_{x_n} to the displayed rational function, we expand it as a Laurent series in x_n; the result is still a rational function and we can apply $\mathrm{CT}_{x_{n-1}}, \ldots, \mathrm{CT}_{x_1}$ to it iteratively.

The idea behind the above treatment is to give a proper series expansion of $1/(x_i - x_j)$ for every i and j, so that all of the expansions are compatible. Once we have determined the relations between the x's, there is no confusion about their series expansion. For instance, we can let $1 > x_1 > \cdots > x_n$. For the particular rational function in equation (1.1), which is symmetric in the x's, there is no confusion after a total ordering on the x's is given.

*Thanks to Ira Gessel

Here we present a slightly different, but more efficient, approach, by means of applying the theory of the field of *iterated Laurent series*. We first treat the rational function in question as an iterated Laurent series, by which we mean we expand it as a Laurent series in x_n, then a Laurent series in x_{n-1}, and so on. Then we take the constant term. This idea led to the study of the field of iterated Laurent series in [18, Ch. 2], which applies to MacMahon's Partition Analysis.

MacMahon's Partition Analysis is suited for solving problems of counting solutions to linear Diophantine equations and inequalities. Using MacMahon's approach, problems such as counting lattice points in a convex polytope, counting integral solutions to a system of linear Diophantine equations, and computing Ehrhart quasi-polynomials, become evaluations of the constant term of an *Elliott-rational function*: a rational function whose denominator has only factors of the form $A-B$, where A and B are both monomials. An example of such is the rational function in (1.1).

MacMahon's technique has been restudied by Andrews et. al. using computer algebra in a series papers [1–9]. New algorithms have been found and computer programs such as the Omega package have been developed.

The constant term (in one variable) of an Elliott-rational function can be read off immediately if its partial fraction decomposition is given. However, the coefficients of a rational function must lie in a field to guarantee the existence of its partial fraction decompositions, and the classical algorithm for partial fraction decomposition is rather slow because the coefficients contain many other variables. The above two problems are solved by applying the theory of iterated Laurent series and a new algorithm for partial fraction decompositions in [17].

In section 2, we give the basic theory of iterated Laurent series. The fundamental structure theorem tells us when a formal Laurent series is an iterated Laurent series. In section 3, we introduce MacMahon's partition analysis. In section 4, we develop an efficient algorithm for MacMahon's partition analysis by combining the theory of iterated Laurent series and a new algorithm for partial fraction decompositions. The theory of iterated Laurent series is crucial in avoiding the "run-time explosion" problem [7, p. 9] when eliminating several variables. In section 5, we use our Maple package to test the efficiency of our algorithm. We investigate problems related to k-gons, generalized Putnam problems, and magic squares [4; 6; 7; 9]. The known formulas are obtained within seconds, and several new formulas are produced in minutes. Finally in section 6, we point out several ways to accelerate the computer program. There are also ways to make the computation easier that are hard to implement on the computer. As an example, we give a simple proof of the formula for k-gon partitions in [4].

2 The Field of Iterated Laurent Series

By a formal Laurent series in $x_1, \dots, x_n$, we mean a series that can be written in the form

$$\sum_{i_1=-\infty}^{\infty} \cdots \sum_{i_n=-\infty}^{\infty} a_{i_1\dots i_n} x_1^{i_1} \cdots x_n^{i_n},$$

where $a_{i_1 \dots i_n}$ are elements in a field K. For formal Laurent series, the definition of the constant term operator is clear:

Definition 2.1 (Natural Definition). *The operator* CT_{x_j} *acts on a formal series in* $x_1, \dots, x_n$ *with coefficients* $a_{i_1,\dots,i_n}$ *in* K *by*

$$\mathop{\mathrm{CT}}_{x_j} \sum_{(i_1,\dots,i_n)\in\mathbb{Z}^n} a_{i_1,\dots,i_n} x_1^{i_1}\cdots x_n^{i_n} = \sum_{(i_1,\dots,i_n)\in\mathbb{Z}^n, i_j=0} a_{i_1,\dots,i_n} x_1^{i_1}\cdots x_n^{i_n}.$$

The simplest way to apply the natural definition would be to work with all formal series $\sum_{i_1,\dots,i_n} a_{i_1,\dots,i_n} x_1^{i_1}\cdots x_n^{i_n}$, where $(i_1,\dots,i_n)$ ranges over all elements of $\mathbb{Z}^n$. Unfortunately, they do not form a ring. Therefore we usually work in a ring, such as the ring of Laurent series $K((x_1,\dots,x_n))$: formal series of monomials where the exponents of the variables are bounded from below. But we need a larger ring or even a field that includes all rational functions, because many constant term evaluation problems involves rational functions.

Let K be a field. We define $K\langle\langle x_1\rangle\rangle$ to be the field of Laurent series $K((x_1))$, and define the *field of iterated Laurent series* $K\langle\langle x_1,\dots,x_n\rangle\rangle$ inductively to be $K\langle\langle x_1,\dots,x_{n-1}\rangle\rangle((x_n))$, which is the field of Laurent series in x_n with coefficients in $K\langle\langle x_1,\dots,x_{n-1}\rangle\rangle$. Thus an iterated Laurent series is first regarded as a Laurent series in x_n, then a Laurent series in x_{n-1}, and so on. An iterated Laurent series obviously has a unique formal Laurent series expansion. However, it is not obvious which formal series are in $K\langle\langle x_1,\dots,x_n\rangle\rangle$. The fundamental structure theorem solves this problem nicely.

We define a total ordering $\preceq$ on monomials by representing $x_1^{i_1}\cdots x_n^{i_n}$ by $(i_1,\dots,i_n)\in\mathbb{Z}^n$, where $\mathbb{Z}^n$ is ordered reverse lexicographically. So $x_i^s \prec x_j$ for all $i<j$ and $s\in\mathbb{Z}$. We define the *support* of a formal Laurent series by

$$\mathrm{supp} \sum_{(i_1,\dots,i_n)\in\mathbb{Z}^n} a_{i_1,\dots,i_n} x_1^{i_1}\cdots x_n^{i_n} := \{\,(i_1,\dots i_n) \mid a_{i_1,\dots,i_n}\neq 0\,\}.$$

Recall that a totally ordered set S is *well-ordered* if each nonempty subset of S contains a minimal element.

Theorem 2.2 (Fundamental Structure). *A formal series in* $x_1,\dots,x_n$ *belongs to* $K\langle\langle x_1,\dots,x_n\rangle\rangle$ *if and only if it has a well-ordered support.*

The proof of this theorem is omitted. For details, see [18, Proposition 2.1.2]. The result gives us an overview about when a formal Laurent series is an iterated Laurent series.

The fundamental structure theorem, together with the simple and useful fact that *any subset of a well-ordered set is well-ordered*, justify the application of the natural definition in $K\langle\langle x_1,\dots,x_n\rangle\rangle$ because of the following three properties:

*P*1. $\mathrm{CT}_{x_i} : K\langle\langle x_1,\dots,x_n\rangle\rangle \to K\langle\langle x_1,\dots,\hat{x}_i,\dots,x_n\rangle\rangle$. This property is necessary to make the natural definition applicable.

*P*2. $\mathrm{CT}_{x_k}\sum_i F_i = \sum_i \mathrm{CT}_{x_k} F_i$. This property is the key to converting many problems into simple algebraic computations.

*P*3. $\mathrm{CT}_{x_i} \mathrm{CT}_{x_j} F = \mathrm{CT}_{x_j} \mathrm{CT}_{x_i} F$. This property may significantly simplify the constant term evaluations.

We define the *order* $\operatorname{ord}(f)$ of an iterated Laurent series f to be the minimum of its support, which is well-ordered by the fundamental structure theorem. We have the following composition law.

Proposition 2.3 (Composition Law). *Suppose that f belongs to $K\langle\langle x_1, \dots, x_n \rangle\rangle$ and $\operatorname{ord}(f) > \operatorname{ord}(1)$. Then for any $b_i \in K$ for all i,*

$$\sum_{i=0}^{\infty} b_i f^i$$

is well defined and belong to $K\langle\langle x_1, \dots, x_n \rangle\rangle$, in the sense that all of its coefficients are finite sum of nonzero elements in K.

This result is a consequence of a general result for Malcev-Neumann series [18, Theorem 3.1.7]. As a consequence, the series expansion of $1/(1-f)$ for $\operatorname{ord}(f) > \operatorname{ord}(1)$ is just $1 + f + f^2 + \cdots$. More generally, for two iterated Laurent series A and B with $\operatorname{ord}(A) < \operatorname{ord}(B)$, the expansion of $1/(A-B)$ is

$$\frac{1}{A-B} = \frac{1}{A} \frac{1}{1-B/A} = \sum_{k \geq 0} B^k / A^{k+1}.$$

For instance, in $K\langle\langle x_1, x_2, x_3 \rangle\rangle$, we have

$$\frac{1}{x_1^2 x_2^4 - x_3} = \sum_{k \geq 0} x_3^k / (x_1^2 x_2^4)^{k+1}.$$

In the field $K\langle\langle x_1, \dots x_n \rangle\rangle$, we define a total ordering on the variables, which produces a total ordering on its group of monomials. This total ordering plays a central role in series expansion. By thinking of iterated Laurent series as numbers, $\operatorname{ord}(f) > \operatorname{ord}(1)$ means that f is much smaller than 1, or $f = o(1)$. Similarly $\operatorname{ord}(B) > \operatorname{ord}(A)$ means that B is much smaller than A, or $B = o(A)$.

The analogous situation for complex variables would be informally written as $1 \gg x_1 \gg \cdots \gg x_n$ when expanding rational functions into Laurent series, where $\gg$ means "much greater". See [16] and [14, p. 231].

The the following three computational rules are frequently used in constant term evaluations. Let $F, G \in K\langle\langle x_1, \dots, x_n \rangle\rangle$.

1. Linearity: $\mathrm{CT}_{x_i}(aF + bG) = a \,\mathrm{CT}_{x_i} F + b \,\mathrm{CT}_{x_i} G$, if a and b are independent of x_i.
2. If F can be written as $\sum_{k \geq 0} a_k x_i^k$, then $\underset{x_i}{\mathrm{CT}} F = F|_{x_i = 0}$.
3. $\underset{x_i}{\operatorname{Res}} \dfrac{\partial}{\partial x_i} F = 0$.

Remark 2.4. Depending on the working field, rational functions $Q(x_1, x_2, \ldots, x_m)$ may have as many as $m!$ different expansions. More precisely, if σ is a permutation of $[m]$, then $Q(\mathbf{x})$ will have a unique expansion in $K\langle\langle x_{\sigma_1}, x_{\sigma_2}, \ldots, x_{\sigma_m}\rangle\rangle$. The expansions of $Q(\mathbf{x})$ for different σ are usually different. So we need to specify the working field whenever a reciprocal comes into account.

Iterated Laurent series is to obtained by defining a total ordering on its variables (this idea is not new, e.g., [14; 16]). In fact, it is a special kind of Malcev-Neumann series, which has been studied in [18], and has applications to MacMahon's partition analysis.

3 MacMahon's Partition Analysis

MacMahon's Partition Analysis is used for counting the solutions to a system of linear Diophantine equations and inequalities, and the number of lattice points in a convex polytope. Such problems can be converted into evaluating the constant terms of certain *Elliott-rational functions*. This conversion has been known as MacMahon's partition analysis, and has been given a new life by Andrews et al. in a series of papers [1–9]

Definition 3.1. *An* Elliott-rational *function is a rational function that can be written in such a way that its denominator can be factored into the products of one monomial minus another, with the* 0 *monomial allowed.*

In the one-variable case, this concept reduces to the generating function of a quasi-polynomial.

MacMahon's idea was to introduce new variables $\lambda_1, \lambda_2, \ldots$ to replace linear constraints. For example, suppose we want to count the nonnegative integral solutions to the linear equation $2a_1 - 3a_2 + a_3 + 2 = 0$. We can compute the generating function of such solutions as the following:

$$\sum_{\substack{a_1,a_2,a_3\geq 0\\ 2a_1-3a_2+a_3+2=0}} x_1^{a_1}x_2^{a_2}x_3^{a_3} = \sum_{a_1,a_2,a_3\geq 0} \operatorname*{CT}_{\lambda} \lambda^{2a_1-3a_2+a_3+2}x_1^{a_1}x_2^{a_2}x_3^{a_3}.$$

Now apply the formula for the sum of a geometric series. It becomes

$$\operatorname*{CT}_{\lambda} \frac{\lambda^2}{(1-\lambda^2 x_1)(1-\lambda^{-3}x_2)(1-\lambda x_3)}.$$

The above expression is a power series in x_i but not in λ.

It is clear that if there are r linear equations, we can compute their solutions by introducing r variables Λ, short for $\lambda_1, \ldots, \lambda_r$. Thus counting solutions of a system of linear Diophantine equations can be converted into evaluating the constant term of an Elliott-rational function.

Theorem 3.2. *If F is Elliott-rational, then the constant terms of F are still Elliott-rational.*

This result follows from "The method of Elliott" (see [13, p. 111–114]) developed from the following identity. Note that we have not specified the working field yet.

Lemma 3.3 (Elliott Reduction Identity). *For positive integers j and k,*

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

Elliott's argument is that after finitely many applications of the above identity to an Elliott-rational function, we will get a sum of rational functions, in which every denominator has either all factors of the form $1-x\lambda^i$, or all factors of the form $1-y/\lambda^i$. Now taking the constant term of each summand is easy.

Theorem 3.2 reduces the evaluation of $\mathrm{CT}_\Lambda F$ to the univariate case $\mathrm{CT}_\lambda F$ by iteration. Unfortunately, the Elliott reduction algorithm is not efficient in practice. Other algorithms have been developed, and computer programs have been set up, such as the "Omega" package [6]. But we can do much better by the partial fraction method and working in a field of iterated Laurent series.

Before going further, let us review some of the work in [6]. The key ingredient in their argument is MacMahon's Omega operator $\Omega_{\geq}$, which is defined by:

$$\underset{\geq}{\Omega} \sum_{s_1=-\infty}^{\infty} \cdots \sum_{s_r=-\infty}^{\infty} A_{s_1,\dots,s_r} \lambda_1^{s_1}\cdots\lambda_r^{s_r} := \sum_{s_1=0}^{\infty}\cdots\sum_{s_r=0}^{\infty} A_{s_1,\dots,s_r},$$

where the domain of the $A_{s_1,\dots,s_r}$ is the field of rational functions over $\mathbb{C}$ in several complex variables and λ_i are restricted to a neighborhood of the circle $|\lambda_i|=1$. In addition, the $A_{s_1,\dots,s_r}$ are required to be such that any of the 2^r-1 sums

$$\sum_{s_{i_1}=0}^{\infty}\cdots\sum_{s_{i_j}=0}^{\infty} A_{s_{i_1},\dots,s_{i_j}}$$

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

Another operator $\Omega_{=}$ is given by

$$\underset{=}{\Omega} \sum_{s_1=-\infty}^{\infty} \cdots \sum_{s_r=-\infty}^{\infty} A_{s_1,\dots,s_r} \lambda_1^{s_1}\cdots\lambda_r^{s_r} := A_{0,\dots,0}.$$

Andrews et al. emphasized in [6] that it is essential to treat everything analytically rather than formally because the method relies on unique Laurent series representations of rational functions.

It is not hard to see that their definition always works if we are working in a ring such as the ring of formal power series in $\mathbf{x}$ with coefficients Laurent polynomials in Λ, where $\mathbf{x}$ is short for $x_1,\dots,x_n$ and Λ is short for $\lambda_1,\dots,\lambda_r$. In fact, this approach was used by Han in [12].

By Theorem 3.2, it suffices to consider the case of $r=1$, since the general case can be done by iteration. In the previous work by Andrews et al. and by Han, the problem was reduced to evaluating the constant term (with respect to λ) of a rational function of the form

$$\frac{\lambda^k}{\prod_{1\le i\le m}(1-\lambda^{j_i}x_i)\prod_{1\le i\le n}(1-y_i/\lambda^{k_i})}. \tag{3.1}$$

This treatment has assumed the obvious geometric expansion:

$$\frac{1}{1-\lambda^{j_i}x_i}=\sum_{s=0}^{\infty}\lambda^{sj_i}x_i^s \quad \text{and} \quad \frac{1}{1-y_i/\lambda^{k_i}}=\sum_{s=0}^{\infty}\lambda^{-sk_i}y_i^s.$$

In other words, for each factor f in the denominator, f has positive powers in λ indicates that the series expansion of $1/f$ contains only nonnegative powers in λ; and f has negative powers in λ indicates that $1/f$ contains only nonpositive powers in λ. In our approach, these indications are dropped off after defining a total ordering.

We find it better to do this kind of work in a certain field of iterated Laurent series, because in such a field, we can use the theory of partial fraction decompositions in $K(\lambda)$ for any field K and any variable λ. We illustrate this idea by solving a problem in [6, p. 252] with the partial fraction method.

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

Solution. First of all, using geometric series summations we translate the problem into a form which MacMahon calls the *crude generating function*, namely

$$f(x,y):=\sum_{a,b\ge 0, 2a-3b\ge 0} x^a y^b = \mathop{\Omega}_{\ge} \sum_{a,b\ge 0}\lambda^{2a-3b}x^a y^b = \mathop{\Omega}_{\ge} \frac{1}{(1-\lambda^2 x)(1-\lambda^{-3}y)},$$

where everything is regarded as a power series in x and y but not in λ.

Now by converting into partial fractions in λ, we have

$$\frac{1}{(1-\lambda^2 x)(1-\lambda^{-3}y)}=\frac{y(1+\lambda x^2 y+\lambda^2 x)}{(1-x^3y^2)(\lambda^3-y)}+\frac{1+\lambda x^2 y}{(1-x^3y^2)(1-\lambda^2 x)}.$$

Where the right-hand side of the above equation is expanded as a power series in x and y, the second term contains only nonnegative powers in λ, and the first term,

$$\frac{y(1+\lambda x^2 y+\lambda^2 x)}{(1-x^3y^2)(a^3-y)}=\frac{y}{1-x^3y^2}\frac{\lambda^{-3}+\lambda^{-2}x^2y+\lambda^{-1}x}{1-\lambda^{-3}y}$$

contains only negative powers in λ. Thus by setting $\lambda = 1$ in the second term, we obtain

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

By a geometric series expansion, it is easy to deduce that

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

$\square$

In solving the above problem, we see that partial fraction decomposition helps in evaluating constant terms, and that only part of the partial fraction is needed.

4 Algorithm by Partial Fraction Decomposition

Working in the field of iterated Laurent series has two advantages. First, the expansion of a rational function into Laurent series is determined by the total ordering "$\preceq$ " on its monomials, so we can temporarily forget its expansion as long as we work in this field. Second, the fact that F is a rational function in λ with coefficients in a certain *field* permits us to apply the theory of partial fraction decompositions.

Note that the idea of using partial fraction decompositions in this context was first adopted by Stanley in [14, p. 229–231], but without the use of computers, this idea was thought to be impractical.

MacMahon's partition analysis always works in a ring like $K[\Lambda, \Lambda^{-1}][[\mathbf{x}]]$, where Λ^{-1} is short for $\lambda_1^{-1}, \dots, \lambda_r^{-1}$. This ring can be embedded into a field of iterated Laurent series, such as $K\langle\langle \Lambda, \mathbf{x} \rangle\rangle$.

While working in the field of iterated Laurent series, it is convenient to use the operator PT_λ, which is formally defined by

$$\mathop{\mathrm{PT}}_{\lambda} \sum_{n=-\infty}^{\infty} a_n \lambda^n = \sum_{n=0}^{\infty} a_n \lambda^n,$$

whose validity is justified by the fundamental structure theorem.

MacMahon's operators can be realized as the following.

$$\mathop{\Omega}_{\geq} F(\Lambda, \mathbf{x}) = \mathop{\mathrm{PT}}_{\Lambda} F(\Lambda, \mathbf{x}) \Big|_{\Lambda=(1,\dots,1)}, \tag{4.1}$$

$$\mathop{\Omega}_{=} F(\Lambda, \mathbf{x}) = \mathop{\mathrm{CT}}_{\Lambda} F(\Lambda, \mathbf{x}) = \mathop{\mathrm{PT}}_{\Lambda} F(\Lambda, \mathbf{x}) \Big|_{\Lambda=(0,\dots,0)}. \tag{4.2}$$

So it suffices to find $\mathrm{PT}_\Lambda F$. In fact, it is well-known that $\Omega_{\geq}$ can be realized by $\Omega_{=}$ by introducing new variables, just as the PT operators can be realized by the CT operators (see [18, Ch. 1]). So either an algorithm for $\mathrm{PT}_\Lambda F$ or an algorithm for $\mathrm{CT}_\Lambda F$ will be sufficient for our purpose. Generally speaking, PT is more suitable for the algorithm, and CT is more suitable for theoretical analysis.

Now we need an algorithm to evaluate $\mathrm{PT}_\lambda F(\lambda)$ with

$$F(\lambda) = \frac{P(\lambda)}{\prod_{1 \leq i \leq n} (\lambda^{j_i} - z_i)}$$

where $P(\lambda)$ is a polynomial in λ, j_i are nonnegative integers, and z_i are independent of λ. Note that we allow z_i to be zero, so that the case of $P(\lambda)$ being a Laurent polynomial is covered. Our approach is different from the previous algorithms, which deal with rational functions expressed as in (3.1) (the difference will be further explained in the next section). It based on the following known fact, which says that once the partial fraction decomposition of F is given, $\mathrm{PT}_\lambda F$ can be read off immediately.

Theorem 4.1. *Suppose that the factors in the denominator of F are pairwise relatively prime, and that the partial fraction decomposition of F is*

$$F = f(\lambda) + \sum_{1\le i\le n} \frac{p_i(\lambda)}{\lambda^{j_i} - z_i},$$

where $f(\lambda)$ is a polynomial in λ, and $p_i(\lambda)$ is a polynomial of degree less than j_i for each i. Then

$$\operatorname*{PT}_{\lambda} F = f(\lambda) + \sum_i \frac{p_i(\lambda)}{\lambda^{j_i} - z_i}, \tag{4.3}$$

where the sum ranges over all i such that $z_i \prec \lambda^{j_i}$.

Proof. The condition that z_i is independent of λ implies that either $\lambda^{j_i} \prec z_i$ or $z_i \prec \lambda^{j_i}$. In the former case, we observe that the expansion of $p_i(\lambda)/(\lambda^{j_i} - z_i)$ into Laurent series contains only negative powers in λ, hence has no contribution when applying PT_λ. In the latter case, the expansion contains only nonnegative powers in λ. Thus the the theorem follows. □

To apply Theorem 4.1, we need to know the partial fraction decompositions of the given rational function. In fact, we need only part of the partial fraction decompositions. Thus we need an efficient algorithm for the partial fraction decompositions. More ideally, an algorithm that only give us the necessary parts. The classical algorithm does not seem to work nicely. We use the new algorithm in [17] developed from the following Theorem 4.2.

To state the theorem, we need some concepts. Let K be a field. For $N, D \in K[t]$ with $D \neq 0$, N/D can be uniquely written as the summation of a polynomial p and a proper fraction (or rational function) r/D. We denote by $\mathsf{Poly}(N/D)$ the polynomial part, which is p, and by $\mathsf{Frac}(N/D)$ the fractional part, which is r/D.

Suppose that $N, D \in K[t]$ and D is factored into pairwise relatively prime factors $D = D_1 \cdots D_k$. Then the ppfraction (short for polynomial and proper fraction) expansion of N/D with respect to $D_1, \ldots, D_k$ is the decomposition of N/D as

$$N/D = p + r_1/D_1 + \cdots r_k/D_k$$

such that p, r_i are polynomials and $\deg(r_i) < \deg(D_i)$ for every i. We denote the above r_i/D_i by $\mathsf{Frac}(N/D, D_i)$, the fractional part of N/D with respect to D_i.

Theorem 4.2 (Theorem 2.3 [17]). *For any $N, D \in K[t]$ with $D \neq 0$, if $D_1, \ldots D_k \in K[t]$ are pairwise relatively prime, and $D = D_1 \cdots D_k$, then*

$$\frac{N}{D} = \mathsf{Poly}\left(\frac{N}{D}\right) + \mathsf{Frac}\left(\frac{N}{D}, D_1\right) + \cdots + \mathsf{Frac}\left(\frac{N}{D}, D_k\right)$$

is the ppfraction expansion of N/D with respect to $(D_1, \ldots, D_k)$. Moreover, if $1/(D_1 D_i) = s_i/D_1 + p_i/D_i$, then

$$\mathsf{Frac}(N/D, D_1) = \mathsf{Frac}(N s_2 s_3 \cdots s_k / D_1).$$

By Theorem 4.2, we need two formulas to develop our algorithm. One is for the fractional part of $p(\lambda)/(\lambda^j - a)$, and the other for the partial fraction decomposition of $(\lambda^j - a)^{-1}(\lambda^k - b)^{-1}$. These are given as Propositions 4.3 and 4.6 respectively.

Let n mod k be the remainder of n when divided by k. We have

Proposition 4.3. *The fractional part of $p(\lambda)/(\lambda^j - a)$ can be obtained by replacing λ^d with $\lambda^{(d \bmod j)} a^{\lfloor d/j \rfloor}$ in $p(\lambda)$ for all d, and dividing the result by $\lambda^j - a$.*

Proof. By linearity, it suffice to show that the remainder of λ^d when divided by $\lambda^j - a$ equals $\lambda^{(d \bmod j)} a^{\lfloor d/j \rfloor}$, which is trivial. □

It is easy to see that this operation takes time linear in the number of nonzero terms of $p(\lambda)$, where we assume fast arithmetic operations.

Remark 4.4. Observe that the numerator of the fractional part of $p(\lambda)/(\lambda^j - a)$ is always a Laurent polynomial in all variables.

Lemma 4.5. *For positive integers j and k, if $a^k \neq b^j$, then the following is a partial fraction expansion.*

$$\frac{1}{(\lambda^j - a)(\lambda^k - b)} = \frac{1}{b^j - a^k} \mathsf{Frac}\left(\frac{\sum_{i=0}^{k-1} \lambda^{ij} a^{k-1-i}}{\lambda^k - b} \right) - \frac{1}{b^j - a^k} \mathsf{Frac}\left(\frac{\sum_{i=0}^{j-1} \lambda^{ik} b^{j-1-i}}{\lambda^j - a} \right) \tag{4.4}$$

Proof. First we show that if $a^k \neq b^j$, then $\lambda^j - a$ and $\lambda^k - b$ are relatively prime. If not, say ξ is their common root in a field extension, then $\xi^j = a$ and $\xi^k = b$. Thus we have $a^k = (\xi^j)^k = \xi^{jk} = (\xi^k)^j = b^j$, a contradiction.

We have

$$\begin{aligned} \frac{b^j - a^k}{(\lambda^j - a)(\lambda^k - b)} &= \frac{\lambda^{jk} - a^k}{(\lambda^j - a)(\lambda^k - b)} - \frac{\lambda^{jk} - b^j}{(\lambda^j - a)(\lambda^k - b)} \\ &= \frac{\sum_{i=0}^{k-1} \lambda^{ij} a^{k-1-i}}{\lambda^k - b} - \frac{\sum_{i=0}^{j-1} \lambda^{ik} b^{j-1-i}}{\lambda^j - a}. \end{aligned}$$

Now the polynomial part of $\frac{b^j - a^k}{(\lambda^j - a)(\lambda^k - b)}$ is clearly 0. Thus the sum of the polynomial parts of the two terms on the right side of the above equation also equals 0. So taking the fractional part of both sides and then dividing both sides by $b^j - a^k$ gives the desired result. □

Now if $\gcd(j, k)$ is not 1, then we can replace $\lambda^{\gcd(j,k)}$ with μ and apply the above lemma. This gives us the following result.

Let

$$\mathcal{F}(\lambda^j - a, \lambda^k - b) = \frac{\sum_{i=0}^{j'-1} \lambda^{ik} b^{j'-1-i}}{a^{k'} - b^{j'}},$$

where $j' = j/\gcd(j, k)$ and $k' = k/\gcd(j, k)$.

Proposition 4.6. *For positive integers j and k, if $a^k \neq b^j$, then we have*

$$\mathsf{Frac}\left(\frac{1}{(\lambda^j - a)(\lambda^k - b)}, \lambda^j - a\right) = \mathsf{Frac}\left(\frac{\mathcal{F}(\lambda^j - a, \lambda^k - b)}{\lambda^j - a}\right), \tag{4.5}$$

Remark 4.7. Note that a similar result appeared in [6, Theorem 1], but their proof was lengthy.

Now by Theorem 4.2, we have the following:

Theorem 4.8. *With the notation of Theorem 4.1, the polynomial $p_s(\lambda)$ equals the remainder of*

$$P(\lambda) \prod_{i=1, i\neq s}^{n} \mathcal{F}(\lambda^{j_s} - a_s, \lambda^{j_i} - a_i),$$

when divided by $\lambda^{j_i} - z_i$ as a polynomial in λ.

In Theorem 4.1, we assumed that $\lambda^{j_i} - z_i$ and $\lambda^{j_k} - z_k$ are relatively prime. Now let us consider the case that $\lambda^{j_i} - z_i$ and $\lambda^{j_k} - z_k$ have a nontrivial common factor. This happens if and only if $z_i^{j_k} = z_k^{j_i}$, which can be easily checked. Andrews et al. [6] suggested that we temporarily regard z_i and z_j as two different variables. After the computation, we replace them. We find an alternate approach, which has been implemented in our computer program and will be discussed in the next section.

Thus the above argument, Theorem 4.1, and Theorem 4.8 together will give us an fast algorithm for evaluating $\mathrm{CT}_\lambda F$.

Remark 4.9. From Remark 4.4, Theorem 4.1, and Theorem 4.8, we see that $\mathrm{PT}_\lambda F$ is Elliott-rational when F is. This is another way to prove Theorem 3.2.

Example 4.10. Count all triples (a, b, c) in $\mathbb{N}^3$ such that they satisfy the triangle inequalities.

Similar problems have been done, such as counting non-congruent triangles with integral side lengths [15, Exercise 4.16]. We are going to illustrate our new approach by this example.

Solution. We solve the following three Diophantine inequalities: $a+b-c \geq 0$, $b+c-a \geq 0$, and $c+a-b \geq 0$. The generating function of these solutions is equal to $\Omega_{\geq} F(\Lambda, \mathbf{x})$, where

$$F(\Lambda) = \frac{1}{((1 - \lambda_1\lambda_3 x_1/\lambda_2)(1 - \lambda_1\lambda_2 x_2/\lambda_3)(1 - \lambda_2\lambda_3 x_3/\lambda_1))}.$$

Although $F(\Lambda, \mathbf{x})$ is in $K[\Lambda, \Lambda^{-1}][[\mathbf{x}]]$, we shall work in a field of iterated Laurent series. We will chose $K\langle\langle \Lambda, \mathbf{x} \rangle\rangle$, and $K\langle\langle \Lambda, x_3, x_2, x_1 \rangle\rangle$ and compare the results.

We will apply $\Omega_{\geq}$ to λ_3, λ_2, λ_1 subsequently. The first step, applying $\Omega_{\geq}$ to λ_3 makes no difference for the two working fields. Applying Theorems 4.1 and 4.8 to the factors of

$F(\Lambda)$ containing λ_3, we get

$$\Omega_{\geq,\lambda_3} F(\Lambda, \mathbf{x}) =$$
$$\frac{\lambda_2 \lambda_1^2 x_1}{\left({\lambda_1}^2 x_1 - \lambda_2^2 x_3\right)\left(1 - {\lambda_1}^2 x_2 x_1\right)(\lambda_1 x_1 - \lambda_2)} - \frac{\lambda_1 {\lambda_2}^2 x_3}{\left({\lambda_1}^2 x_1 - {\lambda_2}^2 x_3\right)\left(1 - {\lambda_2}^2 x_2 x_3\right)(\lambda_1 - \lambda_2 x_3)}$$

Denote by F_1 and F_2 the above two summands. At this stage, we note that the expansion of $\left({\lambda_1}^2 x_1 - \lambda_2^2 x_3\right)^{-1}$ does not exist in $K[\Lambda, \Lambda^{-1}][[\mathbf{x}]]$, and generally there is no advantage in getting rid of the factor $\lambda_1^2 x_1 - \lambda_2^2 x_3$ in the denominator by combining the above two summands into one rational function. This will be further explained in the next section.

Now when applying $\Omega_{\geq}$ on λ_2 to F_1 and F_2, the results are different for the two working fields. Let us look at F_1, especially the expansion of $\left({\lambda_1}^2 x_1 - \lambda_2^2 x_3\right)^{-1}$. The expansion in $K\langle\langle\Lambda, \mathbf{x}\rangle\rangle$ contains only nonnegative powers in λ_2, while the expansion in $K\langle\langle\Lambda, x_3, x_2, x_1\rangle\rangle$ contains only negative powers in λ_2. The situation for F_2 is similar. The conclusion is that working in $K\langle\langle\Lambda, x_3, x_2, x_1\rangle\rangle$ is better: applying $\Omega_{\geq}$ on λ_2 to F_1 will gives us 0, and we have

$$\Omega_{\geq,\lambda_3,\lambda_2} F(\Lambda, \mathbf{x})$$
$$= \frac{\lambda_1 (\lambda_1 + x_3) x_2}{\left(-x_3 + {\lambda_1}^2 x_2\right)\left(-1 + {\lambda_1}^2 x_2 x_1\right)(x_2 x_3 - 1)} + \frac{\lambda_1 x_3}{\left(-x_3 + {\lambda_1}^2 x_2\right)(-1 + x_1 x_3)(-x_3 + \lambda_1)}$$

Applying $\Omega_{\geq}$ on λ_1 to the two summands of the above equation and simplifying gives us

$$\Omega_{\geq,\lambda_3,\lambda_2,\lambda_1} F(\Lambda, \mathbf{x}) = \frac{1 + x_3 x_2 x_1}{(1 - x_1 x_3)(1 - x_2 x_1)(1 - x_2 x_3)}.$$

□

Remark 4.11. It is left to the reader to check that for the above example, $\mathbb{C}\langle\langle\Lambda, x_2, x_3, x_1\rangle\rangle$ is the best working field.

5 The Maple Package

Lemma 5.1. *Let j_i be positive integers and let z_i be monomials. If $\lambda^{j_1} - z_1$ is not relatively prime to $\lambda^{j_2} - z_2$, nor to $\lambda^{j_3} - z_3$, then $\lambda^{j_2} - z_2$ and $\lambda^{j_3} - z_3$ are not relatively prime.*

Proof. By the proof of Lemma 4.5, we have $z_1^{j_2} = z_2^{j_1}$ and $z_1^{j_3} = z_3^{j_1}$. It is easy to see that $z_2^{j_3 j_1} = z_3^{j_2 j_1}$. Now the fact that z_i is a monomial (and hence has coefficient 1) implies that $z_2^{j_3} = z_3^{j_2}$. □

Thus to obtain the complete algorithm, we need to handle the situation when $\lambda^{j_1} - z_1, \ldots, \lambda^{j_k} - z_k$ are not relatively prime to each other. For this situation, we have not succeeded in applying the suggestion of the last section: we tried to let $z_i = z_i v_i$ and do the computation, and finally replace v_i with 1. But the problem is that the last step can only be done after simplification, for which the rational function will be too big for Maple to deal with. The following example explains why this is not a fast approach: evaluating

$$\Omega_{\geq} \frac{1}{(1-\lambda x)^{10}(1-y/\lambda)^{8}}.$$

Our current program uses a modified ppfraction expansion as follows. Suppose that N, D, p_i belong to $K[\lambda]$, and that $P = p_1 \cdots p_k$ is relatively prime to D. Then we can obtain a formula for $\mathsf{Frac}(N/PD, P)$ satisfying our needs:

Write $N/(p_1 D) = r_1/p_1 + N_1/D$ with $\deg(r_1) < \deg(p_1)$. Then $r_1/p_1 = \mathsf{Frac}(N/p_1 D, p_1)$ can be easily obtained.

Now write $N_1/(p_2 D) = r_2/p_2 + N_2/D$. Then $N/(p_1 p_2 D) = r_1/(p_1 p_2) + r_2/p_2 + N_2/D$.

In general, we have

$$\frac{N}{p_1 \cdots p_k D} = \frac{r_1}{p_1 \cdots p_k} + \cdots + \frac{r_k}{p_k} + \frac{N_k}{D},$$

with $\deg(r_i) < \deg(p_i)$. Now it is easy to see that

$$\mathsf{Frac}\left(\frac{N}{PD}, P\right) = \frac{r_1}{p_1 \cdots p_k} + \cdots + \frac{r_k}{p_k}.$$

The recurrence formula for r_i and N_i is given by

$$\frac{r_i}{p_i} = \mathsf{Frac}\left(\frac{N_{i-1}}{p_i D}, p_i\right) \qquad \text{and } N_i = \frac{N_{i-1} - r_i D}{p_i},$$

where $N_0 = N$. Note that we shall let Maple compute N_i with respect to λ.

Now we can give the algorithm for computing $\mathrm{PT}_\lambda\, F(\lambda)$ as follows.

1. Collect the factors in the denominator of F into several groups, such that the factors in different groups are relatively prime and factors in a same group are not.
2. For each group having a contribution, find its corresponding fractional part of F.
3. Take the sum of the results obtained from step 2, and add the polynomial part of F.

Remark 5.2. We will simplify only if needed.

The factors in the denominator of F that are independent of λ should be factored out to speed up the calculation. This has been implemented in our computer program.

The algorithm for $\Omega_\Lambda F(\Lambda, \mathbf{x})$ is described as follows.

1. Fix a total ordering on $\mathbf{x}$ and a total ordering on Λ. Suppose we are working in $\mathbb{C}\langle\!\langle \Lambda, \mathbf{x} \rangle\!\rangle$.

2. Eliminate λ_1 by computing $\mathrm{PT}_{\lambda_1} F$ and then replacing λ_1 with 1.

3. For each rational functions obtained from step 2, eliminate λ_2.

4. Eliminate all the λ's, and finally simplify.

This approach partially solves the "run-time explosion" problem existing in Omega Calculus. Let us analyze a simple situation by considering $\Omega_{\geq} F(\lambda)$, where

$$F(\lambda) = \frac{p(\lambda)}{\prod_{i=1}^{m}(1 - x_i/\lambda)\prod_{j=1}^{n}(1 - y_j\lambda)}.$$

The result after eliminating λ and combining terms will have a denominator of mn factors: $(1 - x_i y_j)$ with $1 \leq i \leq m$ and $1 \leq j \leq n$. Such factors potentially contain the other variables that are going to be eliminated. This explains the existence of the run-time explosion problem.

In our approach, the result after eliminating λ will be a sum of n rational functions (with a possible polynomial part), each with a denominator of $m + n$ factors. Now it is crucial that for each rational function, we can apply the theory of iterated Laurent series to eliminate the other variables.

A Maple package implementing the above algorithm is available online at [19]. Here is an example of how to use this program after downloading this package. The current program uses $\mathrm{E\_Oge}(F, \mathbf{x}, \Lambda)$ to compute $\Omega_{\geq} F(\Lambda, \mathbf{x})$ in the field $\mathbb{C}\langle\langle \Lambda, \mathbf{x} \rangle\rangle$, where $\mathbf{x}$ is realized by $[x_1, \cdots, x_n]$ in maple and Λ is realized similarly.

Example 5.3. Compute the generating function of k-gon partitions, which are partitions that can be the side lengths of a k-gon.

This problem was first studied in [4], where the generating functions of k-gon partitions are obtained only for $k \leq 6$ by using the authors' Omega package. We will discuss in the next section about their formula for general k.

In the following $F(k)$ is the crude generating function of k-gon partitions

$$F(k) = \frac{x_1 a_1^{-1}}{(1 - x_1 \frac{a_k}{a_1})(1 - x_2 \frac{a_1 a_k}{a_2}) \cdots (1 - x_{k-1} \frac{a_{k-2} a_k}{a_{k-1}})(1 - x_k \frac{a_{k-1}}{a_k})},$$

where we use a_i to replace λ_i. The function test(k) computes $\Omega_{\geq} F(k)$ and gives its normal expression.

```
> read "Ell.mpl";
> F:=proc(k)
> product(1-q*a[k]*a[i-1]/a[i],i=2..k-1);
> q/a[1]/((1-q*a[k]/a[1])*%*(1-q*a[k-1]/a[k]));
> end:
> va:=proc(k) seq(a[i],i=1..k) end:
```

```
> F(3);
```

$$qa_1^{-1}\left(1-\frac{qa_3}{a_1}\right)^{-1}\left(1-\frac{qa_3a_1}{a_2}\right)^{-1}\left(1-\frac{qa_2}{a_3}\right)^{-1}$$

```
> E_Oge(%,[q],[va(3)]);
```

$$-q^{-3}\left(q^{-2}-q^2\right)^{-1}\left(q^{-2}-q\right)^{-1}\left(1-q^{-2}\right)^{-1}$$

```
> test:=proc(n) F(n);va(n);E_Oge(%%,[q],[%]);normal(%);end:
> test(3);
```

$$-\frac{q^3}{(q^4-1)(q^3-1)(q^2-1)}$$

```
> test(4);
```

$$\frac{q^4(q^3-q^2+1)}{(q^3-1)(-1+q)(q^2-1)^2(q^3+q^2+q+1)(q^2-q+1)}$$

```
> test(5);
```

$$-\frac{(q^{10}+q^9+q^8+q^7+q^6+q^5+q^4+q^3+q^2+q+1)\,q^5}{(q^3-1)(-1+q)(q^4-q^3+q-1)(q^6+q^5-q-1)(-1+q^8)(q+1)(q^2+1)}$$

```
> test(6);
```

$$\frac{\left(q^{12}+q^{11}+q^{10}+q^9+q^8+2\,q^7+q^6+q^5+q^3+q^2+q+1\right)q^6}{(q^3-1)(q^5-1)(q^8+q^6-q^2-1)(q^4-1)(q^2-1)(q^6-q^4+q^2-1)(q+1)(q^4-q^3+q^2-q+1)}$$

```
> time(test(7));
```

$$34.765$$

All of the above are done in a personal computer. The time of test(7) is measured by seconds.

Example 5.4. A Putnam problem (B3 on the 2000 Putnam examination) was generalized in [7]. The generalized problems are converted to evaluating $CP(T,k,c) = \Omega_{\geq} P(T,k,c)$ where

$$P(T,k,c)=\frac{1}{(1-x_1(a_1\cdots a_T)^k a_1^{-(k(T-1)+c)})\cdots(1-x_T(a_1\cdots a_T)^k a_T^{-(k(T-1)+c)})}$$

for $k>c$ and for $k<c$ we have

$$P(T,k,c)=\frac{1}{(1-x_1(a_1\cdots a_T)^{-k} a_1^{(k(T-1)+c)})\cdots(1-x_T(a_1\cdots a_T)^{-k} a_T^{(k(T-1)+c)})}.$$

Note that the case of $k=c$ is trivial.

```
> read "Ell.mpl";
> P:=proc(T,k,c) if k>c then
> 1/product(1-x[i]*product(a[j]^k,j=1..T)/a[i]^(k*(T-1)+c),i=1..T);
> else
> 1/product(1-x[i]/product(a[j]^k,j=1..T)*a[i]^(k*(T-1)+c),i=1..T)fi;
> end:
```

```
> va:=proc(T) seq(a[i],i=1..T) end:
> vx:=proc(T) seq(x[i],i=1..T) end:
> CP:=proc(T,k,c) E_Oge(P(T,k,c),[vx(T)],[va(T)]);normal(%); end;
```

The case $T = 3$, $k = 2$ and $c = 1$ is given explicitly.

```
> P(3,2,1);
```

$$\left(1 - \frac{x_1 {a_2}^2 {a_3}^2}{{a_1}^3}\right)^{-1} \left(1 - \frac{x_2 {a_1}^2 {a_3}^2}{{a_2}^3}\right)^{-1} \left(1 - \frac{x_3 {a_1}^2 {a_2}^2}{{a_3}^3}\right)^{-1}$$

```
> CP(3,2,1);
```

$$-\frac{{x_1}^4 {x_2}^4 {x_3}^4 + {x_1}^3 {x_2}^3 {x_3}^3 + {x_2}^2 {x_3}^2 {x_1}^2 + x_2 x_1 x_3 + 1}{(-1 + x_3 {x_2}^2 {x_1}^2)(-1 + x_2 {x_3}^2 {x_1}^2)({x_2}^2 {x_3}^2 x_1 - 1)}$$

```
> time(CP(3,3,1));
```

$$0.171$$

```
> time(CP(3,2,3));
```

$$0.391$$

```
> time(CP(3,1,3));
```

$$0.235$$

```
> time(CP(3,1,4));
```

$$0.686$$

```
> time(CP(3,1,5));
```

$$2.172$$

```
> CP(4,1,3):factor(%);
```

Maple will give us the following result:

$$\frac{({x_4}^2 {x_3}^2 {x_2}^2 {x_1}^2 + 1 + x_2 x_4 x_3 x_1)}{(x_4 x_3 {x_1}^3 x_2 - 1)(x_1 x_4 {x_3}^3 x_2 - 1)(x_1 {x_4}^3 x_3 x_2 - 1)({x_2}^3 x_4 x_3 x_1 - 1)}$$
$$\big({x_2}^3 {x_4}^3 {x_3}^2 {x_1}^3 + {x_2}^3 {x_4}^3 {x_3}^3 {x_1}^3 + {x_4}^3 {x_3}^3 {x_2}^2 {x_1}^3 + {x_2}^3 {x_4}^2 {x_3}^3 {x_1}^3 + {x_2}^3 {x_4}^3 {x_3}^3 {x_1}^2$$
$$+ {x_1}^2 x_4 {x_3}^2 x_2 + {x_1}^2 {x_4}^2 x_3 x_2 + {x_2}^2 x_4 x_3 {x_1}^2 + x_4 x_3 x_2 {x_1}^2 + x_1 {x_4}^2 {x_3}^2 x_2$$
$$+ {x_2}^2 x_4 {x_3}^2 x_1 + x_4 {x_3}^2 x_2 x_1 + {x_2}^2 {x_4}^2 x_3 x_1 + {x_4}^2 x_3 x_2 x_1 + {x_2}^2 x_4 x_3 x_1 + 1\big)$$

It will take Maple more than 5 minutes to evaluate $CP(4, 2, 3)$.

We give a detailed comparison of the new algorithm and the Omega package in Table 1, where the unit of the run-time is seconds. Note that programs are not running on the same computer, and that the data for the Omega package comes from [7].

Table 1: Comparison of our new method and the Omega package

run-time for CP	$(3,3,1)$	$(3,2,3)$	$(3,1,3)$	$(3,1,4)$	$(3,1,5)$	$(4,1,3)$
New method	0.171	0.391	0.235	0.686	2.172	14.140
Omega package	7.14	58.95	100.55	643.86	-	-

Example 5.5. Magic squares of order n are n by n matrices with integral entries such that all the row sums and column sums are equal.

The crude generating function for Magic square is:

$$\frac{1}{1-t(\lambda_1\cdots\lambda_n\mu_1\cdots\mu_n)}\prod_{i=1}^{n}\prod_{j=1}^{n}\frac{1}{1-x_{i,j}/(\lambda_i\mu_j)}.$$

When evaluating the constant term in Λ and μ's, we use E_Oeq instead of E_Oge. Our maple program will reproduce the result for $n=3$ quickly. For $n=4$, we get a sum of 96 simple rational functions, which is less than the 254 of the Omega package. Moreover, if we set $x_{i,j}=x$ at the beginning, and finally replace x with 1, our program will reproduce the formula for the case $n=5$ in about two minutes, which is the generating function of the row sums for magic squares of order 5 [15, p. 234].

6 Ways to Accelerate the Program

Our program should have been accelerated by several ways, which are not implemented due to the author's lack of programming skills. These ways are listed as follows and explained by examples. we will manage to reduce the number of rational functions of the output, since the simplification of a sum of many rational functions is a bottle neck for Maple (also Mathematica).

1. The order of the variables to eliminate can make a difference for the computational time.

2. The total ordering on the x's can make a difference for the computational time, as will the total ordering on the λ's.

3. The following alternative formula of (4.3) can simplify the computation:

$$\operatorname*{PT}_{\lambda} F(\lambda) = F(\lambda) - \sum_i \frac{p_i(\lambda)}{\lambda^{j_i}-z_i}, \tag{6.1}$$

   where the sum ranges over all i such that $z_i \succ \lambda^{j_i}$.

(1) is a well-known fact. To take advantage from it, we use the fact that the number of rational functions produced by eliminating λ_i is equal to the number $cf(\lambda_i)$ of factors in the denominator of F that have contributions with respect to λ_i. If $cf(\lambda_{i_0})$ is the smallest among all the $cf(\lambda_i)$, then we shall eliminate the λ_{i_0} first. Note that This way does not guarantee the best result.

The first part of (2) can be explained by Example 4.10, which gives a simple example of how to take advantage of (2). The exact description will take time and is omitted. The second part is similar [18, Example 2.5.13].

Using (3) might produce fewer rational functions. This happens when the denominator of F has more factors with a contribution than those factors without a contribution. See the following example.

Example 6.1. Count all k-gons with nonnegative integral side lengths, which are not required to be in an increasing order.

Solution. Suppose the side lengths of a k-gon is given by $a_1, \ldots, a_k$. Then we have k inequalities, $a_1 + \cdots + a_k \geq 2a_i$ for all i.

Using formula (6.1) we can compute the generating function of k-gons without a computer. The eliminating order is $\lambda_k, \ldots, \lambda_1$.

$$\sum_{a_1+\cdots+a_k \geq 2a_i \text{ for all } i} x_1^{a_1} \cdots x_k^{a_k} = \Omega_{\geq} \frac{1}{(1 - x_1\lambda_1 \cdots \lambda_k/\lambda_1^2) \cdots (1 - x_k\lambda_1 \cdots \lambda_k/\lambda_k^2)}$$
$$= \Omega_{\geq} \frac{1}{(1 - x_1\lambda_1 \cdots \lambda_{k-1}/\lambda_1^2) \cdots (1 - x_{k-1}\lambda_1 \cdots \lambda_{k-1}/\lambda_{k-1}^2)(1 - x_k\lambda_1 \cdots \lambda_{k-1})}$$
$$- \frac{x_k\lambda_1 \cdots \lambda_{k-1}}{(1 - x_1x_k\lambda_1^2 \cdots \lambda_{k-1}^2/\lambda_1^2) \cdots (1 - x_{k-1}x_k\lambda_1^2 \cdots \lambda_{k-1}^2/\lambda_{k-1}^2)(1 - x_k\lambda_1 \cdots \lambda_{k-1})}.$$

Now notice that $\Omega_{\geq}$ acting on the second term is simply obtained by replacing λ_i with 1. Repeat the above computation. We get the final generating function:

$$\frac{1}{(1 - x_1) \cdots (1 - x_k)} - \sum_{i=1}^{k} \frac{x_i}{(1 - x_1x_i) \cdots (1 - x_kx_i)}.$$

□

However, it is probably better not to use (6.1) if the total degree of those factors without a contribution is much greater than those factors with a contribution. Also note that this formula is not easy to apply for the CT operator, we shall use $\mathrm{CT}_\lambda F(\lambda) = \mathrm{CT}_\lambda F(1/\lambda)$ instead.

There are also ways that may speed up the computation, but are not easy to implement by the computer. The following example is simplified by using different parameters for a given problem.

Example 6.2. Count k-gon partitions (revisited).

An exact formula for the generating function of k-gon partitions was given in [4, Theorem 1]. Here we give a simple proof by using different parameters and formula (6.1).

Solution. The problem is to find all $(a_1, \ldots, a_k) \in \mathbb{P}^k$ such that $1 \leq a_1 \leq a_2 \leq \cdots \leq a_k$, and $a_1 + \cdots + a_{k-1} > a_k$.

Let $b_1 = a_1 - 1$, $b_2 = a_2 - a_1, \ldots, b_k = a_k - a_{k-1}$. Then $a_i = 1 + b_1 + \cdots + b_i$ for all i, and it suffices to find all b_i such that $b_i \geq 0$, and $k - 3 + (k-2)b_1 + (k-3)b_2 + \cdots b_{k-2} \geq b_k$.

Thus the generating function for these b_i are given by

$$\underset{\geq}{\Omega} \frac{\lambda^{k-3}}{(1-x_1\lambda^{k-2})\cdots(1-x_{k-2}\lambda)(1-x_{k-1})(1-x_k/\lambda)}$$
$$= \frac{1}{(1-x_1)\cdots(1-x_k)} - \frac{x_k^{k-2}}{(1-x_1x_k^{k-2})(1-x_2x_k^{k-3})\cdots(1-x_{k-1})(1-x_k)}.$$

Now it is easy to convert this formula to [4, Theorem 1]. □

References

[1] G. E. Andrews, *MacMahon's partition analysis. I. The lecture hall partition theorem*, Mathematical essays in honor of Gian-Carlo Rota (Cambridge, MA, 1996), Progr. Math., vol. 161, Birkhäuser Boston, Boston, MA, 1998, pp. 1–22.

[2] ———, *MacMahon's partition analysis. II. Fundamental theorems*, Ann. Comb. **4** (2000), 327–338, Conference on Combinatorics and Physics (Los Alamos, NM, 1998).

[3] G. E. Andrews and P. Paule, *MacMahon's partition analysis. IV. Hypergeometric multisums*, Sém. Lothar. Combin. **42** (1999), Art. B42i, 24 pp. (electronic), The Andrews Festschrift (Maratea, 1998).

[4] G. E. Andrews, P. Paule, and A. Riese, *MacMahon's partition analysis. IX. k-gon partitions*, Bull. Austral. Math. Soc. **64** (2001), 321–329.

[5] ———, *MacMahon's partition analysis: the Omega package*, European J. Combin. **22** (2001).

[6] ———, *MacMahon's partition analysis. VI. A new reduction algorithm*, Ann. Comb. **5** (2001), 251–270, Dedicated to the memory of Gian-Carlo Rota (Tianjin, 1999).

[7] ———, *MacMahon's partition analysis. VII. Constrained compositions*, q-series with applications to combinatorics, number theory, and physics (Urbana, IL, 2000), Contemp. Math., vol. 291, Amer. Math. Soc., Providence, RI, 2001, pp. 11–27.

[8] ———, *MacMahon's partition analysis. VIII. Plane partition diamonds*, Adv. in Appl. Math. **27** (2001), 231–242, Special issue in honor of Dominique Foata's 65th birthday (Philadelphia, PA, 2000).

[9] G. E. Andrews, P. Paule, A. Riese, and V. Strehl, *MacMahon's partition analysis. V. Bijections, recursions, and magic squares*, Algebraic combinatorics and applications (Gößweinstein, 1999), Springer, Berlin, 2001, pp. 1–39.

[10] W. Baldoni-Silva and M. Vergne, *Residues formulae for volumes and Ehrhart polynomials of convex polytopes*, preprint, arXiv:math.CO/0103097, 2001.

[11] C. S. Chan, D. P. Robbins, and D. S. Yuen, *On the volume of a certain polytope*, Experiment. Math. **9** (2000), no. 1, 91–99.

[12] G. N. Han, *A general algorithm for the MacMahon Omega operator*, Ann. of Comb. **7** (2003), 467–480.

[13] P. A. MacMahon, *Combinatory Analysis*, vol. 2, Cambridge University Press, Cambridge, 1915–1916, Reprinted: Chelsea, New York, 1960.

[14] R. P. Stanley, *Combinatorial reciprocity theorems*, Adv. in Math. **14** (1974), 194–253.

[15] ———, *Enumerative Combinatorics*, 2 ed., vol. 1, Cambridge University Press, 1997.

[16] K. G. Wilson, *Proof of a conjecture by Dyson*, J. Math. Phys. **3** (1962), 1040–1043.

[17] G. Xin, *A fast algorithm for partial fraction decompositions*, J. Symbolic Comput. (2004), submitted.

[18] G. Xin, *The Ring of Malcev-Neumann Series and The Residue Theorem*, Ph.D. thesis, Brandeis University, 2004, arXiv:math.CO/0405133.

[19] Guoce Xin, *A Maple package for the constant terms of Elliott-rational functions*, 2004, http://www.brandeis.edu/~maxima/maple/Ell.mpl,.

[20] D. Zeilberger, *Proof of a conjecture of Chan, Robbins, and Yuen*, Elec. Trans. Numer. Anal. **9** (1999), 147–148.