摘 要

通过验证有限项（$n_0 \le n \le n_1$）特殊值来证明超几何恒等式这一想法最早由 Doron Zeilberger 于 1981 年提出，并由 Lily Yen 于 1993 年实现．自此，怎样估计出尽量小的 n_1，从而使得数值验证具有可行性，就成为人们感兴趣的研究课题．本文就主要研究了 n_1 的估计，以及其中涉及的一些数值计算问题．

一般来讲，n_1 可表示成下述三个数值的简单函数：欲证明等式中和式所满足的递归关系的阶数 L，该递归关系中首项系数的最大非负整数根 n_a，以及其中所有系数多项式的最高次数 n_f．本文的主要工作是在研究借助 Sister Celine 算法或 Zeilberger 算法得到的具体的符号线性方程组的数值性质的基础上，提出了估计 n_a 和 n_f 的一个新方法，实例表明，我们的估计结果与之前的工作相比有了极大的降低．

与现有方法类似，我们通过研究符号线性方程组多项式解的次数和高度的上界来估计 n_f 和 n_a．不同的是，我们基于具体的方程组进行估计．为此，我们首先研究了多项式的次数和高度的一些基本性质，得出了估计多项式矩阵行列式的次数和高度上界的公式，并对该公式的计算进行了讨论，特别是我们将次数上界的计算转换成了组合优化中经典的指派问题，从而实现了高度上界的快速计算．接下来我们给出了一个估计符号线性方程组多项式解的次数和高度上界的算法．同时，借助该算法的部分结果，我们还给出了一个利用数值方法求解符号线性方程组的算法．

将上述估计符号线性方程组多项式解的次数和高度上界的算法与 Sister Celine 算法或 Zeilberger 算法相结合，我们最终提出了一个估计 n_1 的新方法，该方法对 q-超几何恒等式同样有效．大量实例表明，与以前的结果相比，我们的方法极大的降低了对 n_1 的估计，尤其是对 q-超几何恒等式，我们的方法不仅能够得到充分小的 n_1，而且运行速度也很快，这就使得利用数值验证法证明超几何恒

等式具有了实际意义上的可行性.

关键词: 超几何恒等式, 数值验证, 次数, 高度, Sister Celine 算法, Zeilberger 算法

Abstract

The stupefying idea of proving hypergeometric identities by checking a finite number of its special cases, say for $n_0 \leq n \leq n_1$, was first realized by Doron Zeilberger in 1981, and then implemented by Lily Yen in 1993. Since then, seeking a small upper bound for n_1 so that the numerical verification is practical has attracted researchers' interests. This thesis mainly studies the problem of estimating n_1 and involved numerical computation issues.

In general, estimating n_1 consists of estimating three numbers: the order L of the recurrence that the summation in the identity satisfies, the largest non-negative zero n_a of the leading coefficient of the recurrence, and the highest degree n_f of any of the coefficient polynomials of the recurrence. n_1 can be formulated as a simple function of these three numbers. In this thesis, we propose a new approach to estimate n_a and n_f, which, as examples indicate, are considerably smaller in comparison with previous results. Our approach relies largely on the study of the numerical aspects of the concrete symbolic linear systems produced by the classic Sister Celine's method and Zeilberger's algorithm.

As previous work, we estimate n_a and n_f by evaluating upper bounds of the degree and height of polynomial solution of the symbolic linear system also, while our approach is distinguished by the exploration of the concrete systems. To this end, we first introduce some basic properties of the degree and height of polynomials, and derive upper bounds formulas for the degree and height of the determinant of a polynomial matrix. Computing issues of these two formulas are also discussed, where, especially, we attack the computation of the degree bound by interpreting it into a classical combinatorial optimization problem, the assignment problem. Then we present an algorithm for estimating the upper bounds of the degree and height of the polynomial solution of a symbolic linear system of equations. As a byproduct, we offer a method for numerically solving symbolic linear systems.

Combining the above degree and height bound estimating algorithm with Sister Celine's method or Zeilberger's algorithm, we finally propose a new approach to derive sharper bounds for n_a and n_f, and consequently smaller n_1. Our approach is also applicable to q-hypergeometric identities. We test our method with plenty of examples. In comparison with previous results, our estimations of n_1 are greatly reduced. In addition, for the q case, our algorithm not only produces practical bounds for n_1, but also runs quickly, which implies that the idea of proving hypergeometric identities by numerical verifications has really been feasible.

Keywords: hypergeometric identities, numerical verification, degree, height, Sister Celine's method, Zeilberger's algorithm

南开大学学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师指导下，进行研究工作所取得的成果。除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人创作的、已公开发表或者没有公开发表的作品的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。本学位论文原创性声明的法律责任由本人承担。

学位论文作者签名：[illegible]

2009年 5 月 31 日

Chapter 1

Introduction

1.1 Background

Computer proofs, or automated proofs, of hypergeometric identities is a fairly recent subject. It was initiated by Sister Celine Fasenmyer [26, 27, 28], and flowered since series of work contributed by Zeilberger [76, 77, 78] and other researchers [31, 68, 69, 51].

Sister Celine's method, developed in Sister Celine Fasenmyer's doctoral dissertation [26] of 1945 under the supervision of Rainville, and two later papers [27, 28], is the first computerizable method for finding pure recurrence relations that are satisfied by sums. An exposition of this method is in Chapter 14 of Rainville's book [56]. Although this method is usually slower than the more recent Zeilberger's algorithm, it is also important because it has yielded The Fundamental Theorem for the existence of the recurrence satisfied by hypergeometric sums, which states that every proper hypergeometric term satisfies a k-free recurrence relation. The Fundamental Theorem underlies the later developments. This theorem also generalizes to multivariate sum, and to q- and multi-q-sums [69]. It has also been proved recently that a hypergeometric term has a k-free recurrence if and only if it is proper [12, 13, 35, 36].

Gosper's algorithm [30, 31] (see also [32, 54, 40]) is one of the landmarks in the theory of computerization of closed form summation. It plays a role to summation as finding antiderivatives is to integration. It not only definitively answers the question that whether or not a given hypergeometric term can be *indefinitely* summed, but also is vital in the operation of Zeilberger's algorithm and WZ method. Gosper's algorithm has been generalized to q-case [42], multibasic case and mixed case [18], etc. Paule [49] also proposed a new approach to indefinite hypergeometric summation which leads to the same algorithm as Gosper's by the notion of Greatest Factorial Factorization (GFF).

Zeilberger's algorithm [77, 78], extended from Gosper's algorithm by Doron Zeilberger, also known as the method of *creative telescoping*, is again for finding recurrence relations that are satisfied by *definite* sums, the same job as Sister Celine's method does. But it does that job a great deal faster. Zeilberger's algorithm occupies a central position in the study of *definite* sums, it has made the computerized proofs of a whole generation of identities possible. Zeilberger's algorithm can also be easily adapted to the q-case [69, 42, 50, 19]. The termination problem of (q-)Zeilberger's algorithm has just been fixed quite recently by several researchers [5, 6, 9, 14, 20, 44].

What Sister Celine's method and Zeilberger's algorithm can produce is a recurrence that the sum satisfies, and it is very likely that the recurrence happens not to be of first order, in which case, we need some techniques to find the "simple" solution of the recurrence. This problem has been extensively studied for both ordinary and q recurrence relations in piles of literature, and bundles of algorithms have been proposed for finding the polynomial solution [52, 8], rational solution [2, 3, 4, 7, 21, 64], hypergeometric solution [52, 10, 53, 22], and the more general d'Alembertian solution [11].

Summarizing, the above classical algorithms of computer proofs of identities are in the following framework, which was recognized by Zeilberger [76]:

Proving an identity like

$$\sum_k F(n,k) = f(n)$$

can be performed in the following steps.

1. Find a recurrence relation that is satisfied by the summation on the left hand side.
2. Substitute $f(n)$ into the same recurrence to verify that it satisfies the same recurrence.
3. Check that both sides agree for enough corresponding initial values.

Step 1 is just the mission of Sister Celine's method and Zeilberger's algorithm. Gosper's algorithm can find $f(n)$ directly from $F(n,k)$ if $F(n,k)$ is

Gosper-summable, while algorithms for solving recurrences can find $f(n)$ from the recurrence that $f(n)$ satisfies.

Beyond the above traditional framework, another stupefying idea, proving identities by numerical verification, also arose.

It was Zeilberger [75, 76] who first realized that when he studied Sister Celine's method by means of Stanley's notions of P-recursiveness and D-finiteness [61]. His original idea was to contend that every identity involving sums of products of binomial coefficients can be verified by checking a finite number of its special cases, say for $n_0 \leq n \leq n_1$. Here we adopt the notation n_1 as used originally by Lily Yen in her doctoral dissertation [73] of 1993 under the supervision of Herbert Wilf. Of course, for a given integer n, the identity is immediately verified. Since in the traditional framework, getting recurrence relations satisfied by hypergeometric terms by Sister Celine's method or Zeilberger's algorithm always involves solving a (generally large) symbolic linear system of equations and can be prohibitively difficult on a computer [79]. The possibility of checking a finite number of cases numerically is very appealing indeed.

Following the general idea that the estimate for n_1 is safe if it is the order of the recurrence plus the size of the largest non-negative zero of the leading coefficient of the recurrence plus the highest degree in n of any of the coefficient polynomials, Lily Yen [72, 73] established the existence of n_1, and also gave the first *a priori* estimates of n_1 for hypergeometric identities. However her estimates are extremely too large to be practical-sized computations [54], and it naturally became an interesting question that how small we can make the estimate of n_1. In a later paper [74], Yen also showed that the estimates of n_1 for q-hypergeometric identities can be spectacularly reduced because of the generic variable "q", which eliminates the vanishing of the highest coefficient of the recurrence. In 2003, the estimates of n_1 for q-identities were further greatly reduced by Zhang and Li [82], and Zhang [80, 81], which employed the famous Wu's elimination method [70, 71].

As Yen [73] pointed out, "If more is known about the linear system of equations, the size of n_1 can be greatly reduced." We [33] tried to dig out more information from the concrete linear systems obtained from Sister Celine's method or Zeilberger's algorithm, and derived sharper bounds of n_1 for both ordinary and q-hypergeometric identities. These bounds have made the verifications more feasible in comparison with previous estimations. For instance, we get $n_1 = 4$ and $n_1 = 12091$ for identity $\sum_k \binom{n}{k} = 2^n$ and $\sum_k \binom{n}{k}^2 = \binom{2n}{n}$, while the bounds returned by Yen [72] are 10^{11} and 10^{115} respectively. For the q-Chu-Vandermonde identity, our approach returns $n_1 = 25$, while Yen [74] and Zhang [80] return 2358 and 191 respectively.

This thesis is organized as follows. In the next section, we provide some notations and basic definitions that are used throughout the entire thesis. In Chapter 2, we will introduce the degree and height of polynomials, the very information we dig out from the system, and present an algorithm for estimating the upper bounds for the degree and height of the polynomial solution of a symbolic linear system of equations. Computing issues involved in the algorithm are discussed as well. As a byproduct, we also offer a method for numerically solving symbolic linear system. In Chapter 3, we will show the flow of derivation of n_1 by applying our method to the linear systems obtained from both Sister Celine's method and Zeilberger's algorithm. Plenty of examples are presented in the end.

1.2 Notations and Basic Definitions

In this section, we recall some notations and basic definitions in the thesis.

As usual, we denote the set of integers, non-negative integers, rational numbers, real numbers and complex numbers by $\mathbb{Z}$, $\mathbb{N}$, $\mathbb{Q}$, $\mathbb{R}$ and $\mathbb{C}$ respectively. Let K be a field of characteristic zero, and let $[n]$ denote the integer set $\{1, \ldots, n\}$.

In this thesis, we adopt the notation

$$(x)_n = \begin{cases} x(x+1)\cdots(x+n-1), & n \geq 0; \\ \frac{1}{(x+n)_{-n}}, & n < 0, \end{cases}$$

to denote the generalized rising factorial. Note that ordinary rising factorial, also known as rising factorial power [32, p. 48] or Pochhammer symbol, is usually only defined for $n \geq 0$.

A function f from $D \subseteq \mathbb{Z}$ to K is said to be a *hypergeometric term* if the ratio of two consecutive terms is a rational function, i.e.,

$$\frac{f(n+1)}{f(n)} = r(n),$$

where $r(x)$ if a rational function from $\mathbb{Z}$ to K.

Similarly, a bivariate function F from $D \subseteq \mathbb{Z}^2$ to K is called a (*bivariate*) *hypergeometric term* if

$$\frac{F(n+1,k)}{F(n,k)} \quad \text{and} \quad \frac{F(n,k+1)}{F(n,k)}$$

are both bivariate rational functions from $\mathbb{Z}^2$ to K.

Definition 1.2.1 *A function $F(n,k)$ is a (bivariate) proper hypergeometric term if it can be written as*

$$F(n,k) = P(n,k)\frac{\prod_{i=1}^{uu}(c_i)_{a_in+b_ik}}{\prod_{i=1}^{vv}(w_i)_{u_in+v_ik}}x^k, \tag{1.2.1}$$

where x is an indeterminate over, say, the complex numbers, and

1. *$P(n,k)$ is a bivariate polynomial,*
2. *$a_i, b_i, u_i, v_i \in \mathbb{Z}$ are specific integers,*
3. *c_i, w_i are constants, and*
4. *$uu, vv \in \mathbb{N}$ are finite, non-negative, specific integers.*

For our purpose, we focus on a class of *special* proper hypergeometric terms which are in the same form as (1.2.1) with tightened conditions:

1. $P(n,k)$ is a bivariate polynomial from $\mathbb{Z}$ to $\mathbb{Q}$,
2. $c_i, w_i, x \in \mathbb{Q}$ are specific rational numbers, and
3. $F(n,k)$ does not depend on any other parameters.

In the rest of this thesis, it means this *special* proper hypergeometric term when we refer to *proper hypergeometric terms.*

For the q case, we have q-analogue of the above stuff.

Let q be transcendental over K. When q is a fixed non-zero complex number with $|q| < 1$, the q-shifted factorial is defined for any complex parameter a and integer k by

$$(a;q)_\infty = \prod_{k=0}^{\infty}(1-aq^k)$$

$$(a;q)_n = \frac{(a;q)_\infty}{(aq^n;q)_\infty} = \begin{cases} (1-a)(1-aq)\cdots(1-aq^{n-1}), & n>0; \\ 1, & n=0; \\ \frac{1}{(aq^n;q)_{-n}}, & n<0. \end{cases}$$

We also use the notation $[a]_n$ to denote the q-shifted factorial as in [46].

For brevity, we write

$$(a_1, \ldots, a_m; q)_n = (a_1; q)_n \cdots (a_m; q)_n,$$

where n is an integer or ∞.

The natural q-analog of the binomial coefficients is defined by

$$\begin{bmatrix} n \\ m \end{bmatrix} = \frac{(q;q)_n}{(q;q)_m (q;q)_{n-m}} = \frac{(1-q^n)(1-q^{n-1})\cdots(1-q^{n-m+1})}{(1-q)\cdots(1-q^m)}.$$

We refer the readers to the classic literature [15, 16, 29] for more q-theory stuff.

A function f from $D \subseteq \mathbb{Z}$ to the rational function field $K(q)$ is said to be a *q-hypergeometric term* if the ratio of two consecutive terms is a rational function, i.e.,

$$\frac{f(n+1)}{f(n)} = r(q^n),$$

where $r(x)$ is a rational function of x over $K(q)$.

A bivariate function F from $D \subseteq \mathbb{Z}^2$ to $K(q)$ is called a (*bivariate*) *q-hypergeometric term* if there are bivariate rational functions $r(x, y)$ and $s(x, y)$ over $K(q)$ such that

$$\frac{F(n+1,k)}{F(n,k)} = r(q^n, q^k) \quad \text{and} \quad \frac{F(n,k+1)}{F(n,k)} = s(q^n, q^k).$$

As in the ordinary case, we focus on a class of special (*bivariate*) *q-proper hypergeometric terms* which can be written in the form

$$F(n,k) = P(q^n, q^k) \frac{\prod_{i=1}^{uu} [c_i]_{a_i n + b_i k}}{\prod_{i=1}^{vv} [w_i]_{u_i n + v_i k}} q^{Jk(k-1)/2} z^k,$$

in which

1. $P(q^n, q^k)$ is a bivariate Laurent polynomial in q^n, q^k over $K(q)$,
2. $a_i, b_i, u_i, v_i, J \in \mathbb{Z}$ are specific integers,
3. $c_i, w_i, z \in K(q)$ are specific constants,

4. $uu, vv \in \mathbb{N}$ are finite, non-negative, specific integers and

5. $F(n,k)$ does not depend on any other parameters.

In the following chapters, when speaking of *q-proper hypergeometric term* without any additional qualification, then this *special q-proper hypergeometric term* is meant.

Chapter 2

The Degree and Height of Polynomials and Polynomial Determinant

The degree and height of polynomials play a key role in both Yen's [72, 73] and our method [33] to estimate n_1. This chapter is mainly concerned with the degree and height of polynomials and polynomial determinant, which underly the estimating algorithms in the next chapter. First we show some basic properties of the degree and height of polynomials, and derive formulas for calculating the upper bounds of the degree and height of a polynomial determinant. Next we talk about the computation issue of the degree and height bounds. Then we present the DHB algorithm for computing upper bounds on the degree and height of a non-trivial polynomial solution of a symbolic linear system of equations. Finally, we offer a numerical method to solve symbolic linear systems based on the result of the DHB algorithm.

2.1 The Degree and Height

First let us recall the well-known polynomial remainder theorem in algebra which states that the remainder, r, of a polynomial, $P(x)$, divided by a linear divisor, $x - a$, is equal to $P(a)$. Then for polynomials with integer coefficients, we can easily derive the following lemma.

Lemma 2.1.1 *Let $P(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_t x^t \in \mathbb{Z}[x]$ be a univariate polynomial in x and $n \geq t \geq 0, a_t \neq 0$. If x_0 is a non-zero integer root of $P(x)$, then $x_0 | a_t$. Consequently, for any integer $x > |a_t|$, $P(x) \neq 0$.*

Lemma 2.1.1 implies that the trailing coefficient can be taken as an upper bound of the largest non-zero integer root of a polynomial. However, the trailing coefficients may be eliminated during additions. For example, given $P_1 = 2x + 1$ and $P_2 = 2x - 1$, then $P_1 + P_2 = 4x$. Thus it is impossible to tell the trailing coefficient of $P_1 + P_2$ by tracking only the trailing coefficients of P_1 and P_2.

A safe way is to track the largest absolute value of the coefficients, which clearly is an upper bound of the trailing coefficient. Recall that the largest absolute value of the coefficients of a polynomial P is called the *height* of P, usually denoted by the l^∞-polynomial norm $||P||_\infty$, or $|P|$ as a shorthand.

Now let us see some properties of the degree and height of univariate polynomials. Here we denote the degree of a polynomial $P(x)$ by $\deg(P)$ and define $\deg(0) = -\infty$.

Lemma 2.1.2 *Let K be a field of characteristic zero and $P_1, P_2, \ldots, P_n \in K[x]$ be polynomials in x with degrees $d_1, d_2, \ldots, d_n$, respectively. Then*

(1) $\deg(P_1 + P_2 + \cdots + P_n) \le \max(d_1, d_2, \ldots, d_n)$;

(2) $\deg(P_1 P_2 \cdots P_n) = d_1 + d_2 + \cdots + d_n$.

Suppose further that K is a subfield of $\mathbb{C}$, the field of complex numbers, and the heights of $P_1, \ldots, P_n$ are $h_1, \ldots, h_n$, respectively. Then

(3) $|P_1 + P_2 + \cdots + P_n| \le h_1 + h_2 + \cdots + h_n$;

(4) $|P_1 P_2 \cdots P_n| \le \prod_{i=1}^{n-1} \left(\min \left(\sum_{j=1}^{i} d_j,\ d_{i+1} \right) + 1 \right) \cdot \prod_{i=1}^{n} h_i.$

Proof. The first three assertions are obvious, we thereby only prove the fourth.

First, let us consider the case of $n = 2$. Suppose

$$P_1 = \sum_{i=0}^{d_1} a_i x^i \quad \text{and} \quad P_2 = \sum_{j=0}^{d_2} b_j x^j,$$

then

$$P_1 P_2 = \sum_{k=0}^{d_1+d_2} \left(\sum_{i+j=k} a_i b_j \right) x^k.$$

Let $c_k = \sum_{i+j=k} a_i b_j$. Note that in c_k, the number of terms in the summation is no larger than $\min(d_1, d_2) + 1$, we have, for each k, $|c_k| \le (\min(d_1, d_2) + 1)\, h_1 h_2$. Thus,

$$|P_1 P_2| = \max_{0 \le k \le d_1 + d_2} \{|c_k|\} \le (\min(d_1, d_2) + 1)\, h_1 h_2.$$

The conclusion follows immediately by induction on n. ∎

Denote by H the right hand side of the inequality in assertion (4) of Lemma 2.1.2. Note that H depends on the order of P_i's degrees, while $|P_1 P_2 \cdots P_n|$ is free of that order. Fix $d_1, \ldots, d_n$, for any permutation σ of $[n]$, let

$$D_\sigma(d_1, \ldots, d_n) = \prod_{i=1}^{n-1} \left(\min \left(\sum_{j=1}^{i} d_{\sigma(j)},\ d_{\sigma(i+1)} \right) + 1 \right), \tag{2.1.1}$$

$$H_\sigma(P_1, \ldots, P_n) = D_\sigma \cdot \prod_{i=1}^{n} h_i. \tag{2.1.2}$$

Clearly, the minimum of H_σ among all the permutations will be large enough to be an upper bound of $|P_1 P_2 \cdots P_n|$. The following lemma tells when it reaches the minimum.

Lemma 2.1.3 *Given non-negative integers $d_1, \ldots, d_n$, σ ranges over all permutations of $[n]$, D_σ is minimal when $d_{\sigma(1)} \ge d_{\sigma(2)} \ge \cdots \ge d_{\sigma(n)}$.*

Proof. Given an initial order of $d_1, \ldots, d_n$, suppose that they are not in descending order, then there must exist two consecutive terms d_i and d_{i+1} such that $d_i < d_{i+1}$. Consider the identical permutation $\mathbf{1}$ and the transposition $\tau = (i, i+1)$. By definition, we have $D_{\mathbf{1}} = \prod_{j=1}^{n-1} b_j$, where

$$\begin{aligned} b_1 &= \min(d_1,\ d_2) + 1, \\ &\vdots \\ b_{i-1} &= \min(d_1 + d_2 + \cdots + d_{i-1},\ d_i) + 1, \\ b_i &= \min(d_1 + d_2 + \cdots + d_i,\ d_{i+1}) + 1, \\ &\vdots \\ b_{n-1} &= \min(d_1 + d_2 + \cdots + d_{n-1},\ d_n) + 1. \end{aligned}$$

Similarly, we have $D_\tau = \prod_{j=1}^{n-1} b'_j$ with $b'_j = b_j$ except for the following two terms:

$$\begin{aligned} b'_{i-1} &= \min(d_1 + d_2 + \cdots + d_{i-1},\ d_{i+1}) + 1, \\ b'_i &= \min(d_1 + d_2 + \cdots + d_{i-1} + d_{i+1},\ d_i) + 1 = d_i + 1. \end{aligned}$$

We distinguish three cases:

Case 1. If $d_1 + d_2 + \cdots + d_{i-1} \leq d_i < d_{i+1}$, then

$$b_{i-1} = b'_{i-1} = d_1 + d_2 + \cdots + d_{i-1} + 1$$

and

$$b_i \geq \min(d_i,\ d_{i+1}) + 1 = d_i + 1 = b'_i.$$

Thus we have $D_1 \geq D_\tau$.

Case 2. If $d_i < d_1 + d_2 + \cdots + d_{i-1} \leq d_{i+1}$, then $b_{i-1} = b'_i = d_i + 1$ and

$$b_i \geq \min(d_1 + d_2 + \cdots + d_{i-1},\ d_{i+1}) + 1 = b'_{i-1}.$$

We still have $D_1 \geq D_\tau$.

Case 3. If $d_i < d_{i+1} < d_1 + d_2 + \cdots + d_{i-1}$, then

$$b_{i-1} = b'_i = d_i + 1 \quad \text{and} \quad b_i = b'_{i-1} = d_{i+1} + 1,$$

implies that $D_1 = D_\tau$.

Summarizing, D_σ will not increase by swapping two consecutive ascending terms, thus we can always do this swap to decrease D_σ until $d_{\sigma(1)}, \ldots, d_{\sigma(n)}$ is descending. Therefore, D_σ is minimal when $d_{\sigma(1)} \geq d_{\sigma(2)} \geq \cdots \geq d_{\sigma(n)}$. ∎

Moreover, provided that $d_{\sigma(1)} \geq d_{\sigma(2)} \geq \cdots \geq d_{\sigma(n)}$, D_σ can be reduced to the following form.

$$D_\sigma = \left(d_{\sigma(2)} + 1\right)\left(d_{\sigma(3)} + 1\right)\cdots\left(d_{\sigma(n)} + 1\right). \tag{2.1.3}$$

As a simple example, let $P_1(x) = x + 1$, $P_2(x) = x^2 + 2$, $P_3(x) = x^3 + 1$. Then $d_1 = 1$, $d_2 = 2$, $d_3 = 3$ and hence $D_{123} = D_{213} = 8$, $D_{132} = D_{312} = D_{321} = D_{231} = 6$. We see that D_{321} is minimal in the set $\{D_\sigma\}$.

Given polynomials $P_1, \ldots, P_n$ and $d_i = \deg(P_i)$, let $\mathrm{minD}(d_1, \ldots, d_n)$ and $\mathrm{minH}(P_1, \ldots, P_n)$ denote the minimum of D_σ and H_σ for $\sigma \in S_n$, respectively, where S_n denotes the set of all the permutations of $[n]$. Then combining Lemma 2.1.2 and Lemma 2.1.3, we can derive upper bounds on the degree and height of a univariate polynomial determinant.

Theorem 2.1.4 *Let K be a field of characteristic zero and $M = [p_{ij}(x)]_{n\times n}$ be a matrix whose entries are polynomials in $K[x]$, and denote the set of permutations of $[n]$ by S_n. Then the degree of* $\det(M)$*, determinant of M, is bounded by*

$$\deg(\det(M)) \leq \max_{\pi\in S_n}\left\{d_{1,\pi(1)} + d_{2,\pi(2)} + \ldots + d_{n,\pi(n)}\right\} \triangleq \mathcal{D}(M), \tag{2.1.4}$$

where $d_{i,j} = \deg(p_{ij}(x))$. Suppose further that $K \subseteq \mathbb{C}$, then the height of $\det(M)$ *is bounded by*

$$|\det(M)| \leq \sum_{\pi\in S_n} \mathrm{minH}\left(p_{1,\pi(1)}(x), \ldots, p_{n,\pi(n)}(x)\right) \triangleq \mathcal{H}(M). \tag{2.1.5}$$

Proof. Starting from the definition of the determinant, we have

$$\begin{aligned}
\deg(\det(M)) &= \deg\left(\sum_{\pi\in S_n}\mathrm{sgn}(\pi)\prod_{i=1}^{n} p_{i,\pi(i)}\right)\\
&\leq \max_{\pi\in S_n}\left\{\deg\left(\prod_{i=1}^{n} p_{i,\pi(i)}\right)\right\} && \text{(Lemma 2.1.2 (1))}\\
&= \max_{\pi\in S_n}\left\{d_{1,\pi(1)} + d_{2,\pi(2)} + \cdots + d_{n,\pi(n)}\right\}, && \text{(Lemma 2.1.2 (2))}
\end{aligned}$$

here $\mathrm{sgn}(\pi)$ denotes the signature of the permutation π: $+1$ if π is an even permutation and -1 if it is odd. $\mathrm{sgn}(\pi)$ is also known as the signature of the number of inversions of the product of the permutation.

In view of the definition of determinant, inequality (2.1.5) can be proved similarly by using Lemma 2.1.2 (3), 2.1.2 (4) and Lemma 2.1.3. ∎

For example, let

$$M = [p_{ij}(x)]_{3\times 3} = \begin{bmatrix} 1 & 0 & -2 \\ x+1 & -x & 2 \\ 0 & 1 & 1 \end{bmatrix},$$

we have

$$\begin{aligned}
\max\{d_{1,\pi(1)} + d_{2,\pi(2)} + d_{3,\pi(3)} \mid \pi \in S_3\} &= 1,\\
\sum_{\pi\in S_3}\mathrm{minH}\left(p_{1,\pi(1)}(x),\ p_{2,\pi(2)}(x),\ p_{3,\pi(3)}(x)\right) &= 5.
\end{aligned}$$

In fact, $\det(M) = -3x-4$, whose degree and height are 1 and 4, respectively. Clearly, inequalities (2.1.4) and (2.1.5) hold.

Since π ranges over all the permutations of $[n]$, it is straightforward that $\mathcal{D}(M)$ and $\mathcal{H}(M)$ are invariant under the transpose of matrix and permutation of rows and columns.

Corollary 2.1.5 *Let M be a square polynomial matrix, M^T be the transpose of M, M_1 be the matrix obtained by swapping any two rows of M and M_2 be the matrix obtained by swapping any two column of M_2, $\mathcal{D}(M)$ and $\mathcal{H}(M)$ are defined as in (2.1.4) and (2.1.5), respectively. We have*

1. $\mathcal{D}(M) = \mathcal{D}(M^T) = \mathcal{D}(M_1) = \mathcal{D}(M_2)$,
2. $\mathcal{H}(M) = \mathcal{H}(M^T) = \mathcal{H}(M_1) = \mathcal{H}(M_2)$.

Note that $\mathcal{D}(M)$ depends only on the degrees of the entries and $\mathcal{H}(M)$ depends on both the degrees and the heights of the entries. Therefore, we need only play with numbers to obtain these two upper bounds, which makes the computation faster than calculating $\det(M)$ explicitly. In implementation, we can directly take as input the degree matrix and height matrix of M to compute $\mathcal{D}(M)$ and $\mathcal{H}(M)$. However, the complexity of computing $\mathcal{D}(M)$ and $\mathcal{H}(M)$ as formulas (2.1.4) and (2.1.5) requires running over all $n!$ permutations, so it will soon become extremely slow when the order of M grows. In the next section, we will discuss the computing issue of $\mathcal{D}(M)$ and $\mathcal{H}(M)$.

2.2 Computing $\mathcal{D}(M)$ and $\mathcal{H}(M)$

The complexity of computing $\mathcal{D}(M)$ and $\mathcal{H}(M)$ straightforwardly by their definitions (2.1.4) and (2.1.5) are $O(n \cdot n!)$ and $O(n \log n \cdot n!)$, both super exponential! More precisely, the complexity of $\mathcal{H}(M)$ is $O((n \log n + 3n)n!)$, here we suppose a sorting algorithm with complexity $O(n \log n)$ is employed, and $O(3n)$ is the complexity of computing (2.1.3) plus the multiplication of h_i's as in (2.1.2). Due to the high complexity, even for slightly larger M, the executing time of computing in this way is not acceptable. As our experiments on `Maple` (version 9.5) with a `Windows` installed PC machine of 2 GHz CPU (AMD Athlon 64 Processor 3200+) and 1 GB memory indicate, it quickly, usually in less than one second, returns $\mathcal{D}(M)$ and $\mathcal{H}(M)$ for matrices of size up to 7-by-7. However, for a 9-by-9 matrix it takes 71 seconds, and for a 10-by-10 matrix, the time sharply goes up to 768 seconds. For the 19-by-19 matrix produced by the Dixon sum identity, the program did not terminate in a whole day!

In fact, the problem of evaluating polynomial determinant and computing its degree and zeros has been studied extensively (e.g., [45, 60, 34, 37, 65, 24]). [34] presented a numerical method for evaluating the degree of a polynomial determinant based on an early result on the Smith-MacMillan form of a rational matrix [63]. This method seems to be a little bit complex, while [24] proposed a much simpler recursive method for obtaining an upper bound of the degree.

On the other hand, given the degree of the determinant of a univariate polynomial matrix, it is possible to calculate the determinant by using the interpolation technique, i.e., once we have determined the degree of a univariate polynomial determinant, we can use $d+1$ values for the variable and calculate the determinant $d+1$ times to obtain $d+1$ points, which would interpolate the polynomial. Once the whole polynomial is evaluated, its height is of course clear.

In this thesis, we will attack the computation of $\mathcal{D}(M)$ by interpreting it into the classical assignment problem.

The assignment problem is a fundamental combinatorial optimization problem:

Given a number of agents and a number of tasks, the agents will be assigned to perform the tasks, incurring some cost that may vary depending on the agent-task assignment. The goal is to find an assignment whose cost is minimized subject to the following requirements,

1. *Any agent can be assigned to perform any task,*
2. *Each agent is assigned no more than one task,*
3. *Each task is assigned to exactly one agent.*

If the numbers of agents and tasks are equal and the total cost of the assignment for all tasks is equal to the sum of the costs for each agent (or the sum of the costs for each task, which is the same), then the problem is called the linear assignment problem. Usually, when speaking of the assignment problem without any additional qualification, then the linear assignment problem is meant. In terms of graph theory language, the assignment problem consists of finding a maximum weight matching in a weighted bipartite graph.

The assignment problem can be formulated into a linear 0-1 integer programming.

$$\begin{aligned}
&\text{minimize} \quad \sum_{i\in A}\sum_{j\in T} c(i,j)x_{ij} \\
&\text{subject to} \\
&\sum_{j\in T} x_{ij} = 1, \qquad \text{for } i \in A, \\
&\sum_{i\in A} x_{ij} = 1, \qquad \text{for } j \in T, \\
&x_{ij} = 0 \text{ or } 1 \quad \text{for } i \in A, j \in T,
\end{aligned}$$

where A and T are the sets of agents and tasks respectively, $c : A \times T \to \mathbb{R}$ is the cost function, x_{ij} represents the assignment of agent i to task j, taking value 1 if the assignment is done and 0 otherwise. The first and second constraints correspond to requirement 2 and 3, respectively.

For our case, the sets of agents and tasks are both $[n]$, a permutation π of $[n]$ corresponds to an assignment, $c(i,j)$ maps (i,j) to d_{ij}, the degree of (i,j)-th entry of M, and $\mathcal{D}(M)$ is the very objective function. Noting that our objective function is maximization, which is essentially equivalent to a minimization one by a trick that subtracting the objective function from a large enough value.

There are various ways to solve assignment problems. Certainly it can be formulated as a linear program by applying linear programming relaxation, i.e., modifying the last constraint into $x_{ij} \geq 0$, and then the famous simplex method (c.f. [23, 48]) or interior method [38] can be used to solve it. In addition, it can also be formulated as a network problem and solved by the network simplex method which may runs quickly. The best known algorithm for solving the linear assignment problem may be the Hungarian method [43] of which the computation complexity is $O(n^3)$. The author refers the readers to [48] for details about linear programming, the assignment problem and the Hungarian method.

As to the computation of $\mathcal{H}(M)$, we are not so lucky as $\mathcal{D}(M)$. The definition of D_π, $\min(D_\pi)$ and thus $\min(H_\pi)$ are all non-linear, and the absence of an explicit formula for $\min(D_\pi)$ makes the situation even worse.

We have considered giving up finding the $\min(D_\pi)$ and use the recursive method for calculating the degree upper bound of polynomial determinant in [24] to calculate the height bound. However, because of the lack of tracking the height upper bound in the division operation of polynomials like in the addition, subtraction and multiplication operations as in Lemma 2.1.2, the unwinding technique in

[24] is not applicable to the height calculation, and the absence of this technique will result extremely large height bounds. For example, for the following degree matrix and height matrix extracted from Example 3.5.2,

$$M_D = \begin{bmatrix} 2 & 2 & 0 & 0 & 0 \\ 2 & 2 & 0 & 1 & 1 \\ 2 & 2 & 0 & 1 & 2 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad M_H = \begin{bmatrix} 2 & 2 & 1 & 1 & 1 \\ 10 & 4 & 2 & 1 & 2 \\ 18 & 2 & 1 & 3 & 3 \\ 3 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix},$$

formula (2.1.4) and (2.1.5) return $\mathcal{D}(M) = 6, \mathcal{H}(M) = 12089$, while the recursive method in [24] returns 7 as the degree bound and 4×10^{20} (approxmate) as the height bound without applying the unwinding technique.

An alternative way is to find an upper bound for $\mathcal{H}(M)$. Investigating the definition of $\mathcal{H}(M)$ again, minH$(\cdot)$ actually consists of two parts:

$$\text{minH}\left(p_{1,\pi(1)}(x), \ldots, p_{n,\pi(n)}(x)\right) = \text{minD}\left(d_{1,\pi(1)}, \ldots, d_{n,\pi(n)}\right) \cdot \prod_{i=1}^{n} h_{i,\pi(i)},$$

where $d_{i,\pi(i)} = \deg(p_{i,\pi(i)})$ and $h_{i,\pi(i)} = |p_{i,\pi(i)}|$. The former degree part appears to be more complicated than the latter height part. If we can find an upper bound MD of minD$\left(d_{1,\pi(1)}, \ldots, d_{n,\pi(n)}\right)$ for all $\pi \in S_n$, then we can enlarge $\mathcal{H}(M)$ as follows.

$$\begin{aligned} \mathcal{H}(M) &= \sum_{\pi \in S_n} \text{minH}\left(p_{1,\pi(1)}(x), \ldots, p_{n,\pi(n)}(x)\right) \\ &= \sum_{\pi \in S_n} \text{minD}\left(d_{1,\pi(1)}, \ldots, d_{n,\pi(n)}\right) \cdot \prod_{i=1}^{n} h_{i,\pi(i)} \\ &\le MD \cdot \sum_{\pi \in S_n} \prod_{i=1}^{n} h_{i,\pi(i)}. \end{aligned}$$

The latter part is in fact the permanent of the matrix M_H, where $M_H = (h_{ij})_{n \times n}$ is the height matrix of M, i.e., $h_{ij} = |p_{ij}|$.

Permanent is a well-known combinatorial object, it has two graph-theoretic interpretations: as the sum of weights of cycle covers of a directed graph, and as the sum of weights of perfect matchings in a bipartite graph. The permanent of an n-by-n matrix $A = (a_{ij})_{n \times n}$ is defined as

$$\text{perm}(A) = \sum_{\pi \in S_n} \prod_{i=1}^{n} a_{i,\pi(i)},$$

the sum here extends over all permutations of $[n]$.

This definition, as a sum of products of sets of matrix entries that lie in distinct rows and columns, is quite similarly to the corresponding formula of determinant, and differs only in that the determinant assigns a ± 1 *sign of the permutation* to each of these products, the permanent does not.

Sadly, given the above similar definition between the permanent and the more complicated-looking determinant, the complexity of the best-known algorithms for computing permanent grows exponentially with the size of the matrix [39, p. 499], while the determinant can be computed by Gaussian elimination in polynomial time $O(n^3)$. In fact, Valiant [62] has proved that computing permanent is #P-complete, putting the computation of the permanent in a class of problems believed to be even more difficult than NP. The fastest known (as of [57, p. 4]) general exact algorithm for computing permanent is based on the Ryser's formula [59], whose complexity is $O(n \cdot 2^n)$.

Now let us come back to MD. Let $rd_i = \max\{d_{ij} \mid 1 \le j \le n\}$ and $cd_j = \max\{d_{ij} \mid 1 \le i \le n\}$ denote the maximum value of the i-th row and j-th column respectively. Obviously, for any $\pi \in S_n$, we have

$$\begin{aligned}\mathrm{minD}\left(d_{1,\pi(1)}, \ldots, d_{n,\pi(n)}\right) &\le \mathrm{minD}(rd_1, \ldots, rd_n),\\ \mathrm{minD}\left(d_{1,\pi(1)}, \ldots, d_{n,\pi(n)}\right) &\le \mathrm{minD}(cd_1, \ldots, cd_n).\end{aligned}$$

Thus $MD = \min(\mathrm{minD}(rd_1, \ldots, rd_n), \mathrm{minD}(cd_1, \ldots, cd_n))$ is a safe upper bound of $\mathrm{minD}\left(d_{1,\pi(1)}, \ldots, d_{n,\pi(n)}\right)$.

The complexity of computing the above MD is $O(n^2 + n\log n + 2n)$, here $O(n^2)$, $O(n\log n)$ and $O(2n)$ are for finding rd_i's (cd_i's), sorting rd_i's (cd_i's) and computing (2.1.3) respectively. Clearly, this approximating computation is much faster than computing by the definition. For example, noting the time we mentioned in the beginning of this section, it takes only less than one second to compute MD for the 19-by-19 matrix yielded from the Dixon sum identity. However, the approximating value is usually much larger than the exact value. For example, for the following 7-by-7 degree matrix produced in Example 3.5.7,

$$\begin{bmatrix} 7 & 7 & 7 & 1 & 1 & 1 & 1\\ 8 & 8 & 8 & 2 & 3 & 3 & 3\\ 7 & 7 & 7 & 1 & 2 & 3 & 3\\ 6 & 6 & 6 & 0 & 1 & 2 & 3\\ 5 & 5 & 5 & 0 & 0 & 1 & 2\\ 4 & 4 & 4 & 0 & 0 & 0 & 1\\ 3 & 3 & 3 & 0 & 0 & 0 & 0 \end{bmatrix},$$

it returns $MD = 53760$ while the exact $\mathcal{D}(M) = 26$, which puts us in the dilemma of speed and precision. It might be an interesting question to find how small MD could be.

2.3 The Degree-Height Bound Algorithm

First we give an algorithm that refines Yen's idea [73] to find a non-trivial solution of linear system of equations.

Input: Linear system $M\mathbf{x} = 0$, here $M \in D^{l\times m}$ is an $l \times m$ $(l < m)$ matrix of rank ρ, and D is an integral domain. $\mathbf{x} = [x_1, x_2, \ldots, x_m]^T$.

Output: A non-trivial solution of the system.

1. Permute the rows and columns of M, so that the top left corner sub-matrix $\widetilde{M}$ formed by the first ρ rows and ρ columns is of rank ρ. Meanwhile, permute the unknowns x_j's according to the column permutation. Denote the new system by $M'\mathbf{x}' = 0$.

2. Let $\mathbf{y}$ be the column vector formed by the first ρ rows of the $(\rho+1)$-th column of M' and let $\tilde{\mathbf{x}} = [x'_1, \ldots, x'_\rho]^T$. We now consider the inhomogenous linear system $\widetilde{M}\tilde{\mathbf{x}} = -\mathbf{y}$.

3. Let $\widetilde{M_i}$ be the matrix obtained by replacing the i-th column of $\widetilde{M}$ with $-\mathbf{y}$. By Cramer's rule,

$$\left[\frac{\det \widetilde{M_1}}{\det \widetilde{M}}, \frac{\det \widetilde{M_2}}{\det \widetilde{M}}, \ldots, \frac{\det \widetilde{M_\rho}}{\det \widetilde{M}}\right]^T$$

   is a solution to $\widetilde{M}\tilde{\mathbf{x}} = -\mathbf{y}$. Hence,

$$[\det \widetilde{M_1}, \det \widetilde{M_2}, \ldots, \det \widetilde{M_\rho}, \det \widetilde{M}, 0, \ldots, 0]^T$$

   is a non-trivial solution to $M'\mathbf{x}' = 0$.

4. Corresponding to the column permutation in step 1, rearrange the elements of $\mathbf{x}'$. Then we finally obtain a non-trivial solution to the system $M\mathbf{x} = 0$.

In the above construction, we see that each entry of the solution is the determinant of a $\rho \times \rho$ sub-matrix of M up to a sign. Thus, when M is a polynomial matrix, the degrees and heights of these determinants are bounded by Theorem 2.1.4. Moreover, the following two lemmas enable us to evaluate the degree and height bounds of all the entries of the solution without evaluating ρ and calculating $\mathcal{D}(\cdot)$ and $\mathcal{H}(\cdot)$ for all the $\rho \times \rho$ sub-matrices of M.

Lemma 2.3.1 *Let K be a field of characteristic zero and $A = [p_{ij}(n)]$ be a square matrix whose entries are non-zero polynomials in $K[n]$. Then for any square sub-matrix A' of A, we have $\mathcal{D}(A') \leq \mathcal{D}(A)$. Suppose further that $p_{ij}(x) \in \mathbb{Z}[x]$ for all i, j. Then $\mathcal{H}(A') \leq \mathcal{H}(A)$.*

Lemma 2.3.2 *Let K be a field of characteristic zero and $A = [p_{ij}(n)], A' = [p'_{ij}(n)]$ be two square matrices of the same order whose entries are polynomials in $K[n]$. Suppose that $\deg(p'_{ij}) \leq \deg(p_{ij})$ for all i, j, then $\mathcal{D}(A') \leq \mathcal{D}(A)$. Suppose further that $K \subseteq \mathbb{C}$ and $|p'_{ij}| \leq |p_{ij}|$ for all i, j, then $\mathcal{H}(A') \leq \mathcal{H}(A)$.*

Lemma 2.3.1 converts the computation of $\rho \times \rho$ sub-matrices to that of $l \times l$ sub-matrices and enables us to avoid computing the rank ρ, which is equivalent to solving the system. Proper use of Lemma 2.3.2 further reduces the computation to one special $l \times l$ sub-matrix. To this end, we define two kinds of transformations on a matrix.

Definition 2.3.3 *Let K be a field of characteristic zero and $A = [p_{ij}(n)]$ be a matrix whose entries are polynomials in $K[n]$.*

A 0-1 augment is the transformation of replacing each zero entry of A with 1. Denote the resulting matrix by $\overline{A}$.

Suppose that $K \subseteq \mathbb{C}$. A DH augment is the transformation as follows: choose a column, say the k-th column, of the matrix A, replace those entries $p_{ij}(x)$ that satisfy $|p_{ij}| < |p_{ik}|$ or $\deg(p_{ij}) < \deg(p_{ik})$ with $h \cdot x^d$ where $h = \max(|p_{ij}|, |p_{ik}|)$ and $d = \max(\deg(p_{ij}), \deg(p_{ik}))$, and finally delete the k-th column from A. We call this special chosen column the augmenting column. The resulting matrix is denoted by $\widehat{A}$.

Now we are ready to present the degree-height bound algorithm (**DHB algorithm** in short).

Input: An $l \times m$ $(l < m)$ matrix $M = [p_{ij}(n)]$ whose entries are polynomials in $\mathbb{Z}[n]$.

Output: Two integers d and h.

1. Repeat DH augment on M and set $M = \widehat{M}$ until M becomes an $l \times l$ square matrix.
2. Do 0-1 augment on M and set $M = \overline{M}$.
3. Return $d = \mathcal{D}(M),\ h = \mathcal{H}(M)$.

Remark 2.3.4 Notice that the DH augment involves the selection of a special column – the augmenting column. In order to obtain smaller bounds, we might prefer to an as small as possible *column, e.g., the one with the minimum height sum or degree sum.*

Remark 2.3.5 The 0-1 augment and DH augment can also be defined on the degree matrix $M_D = (d_{ij})$ and height matrix $M_H = (h_{ij})$ of the polynomial matrix M with slight modifications: 0-1 augment involves replacing $-\infty$ entries into 0 in M_D and replacing zero entries in M_H into 1, and DH augment involves replacing d_{ij} with d_{ik} if $d_{ij} < d_{ik}$ and replacing h_{ij} with h_{ik} if $h_{ij} < h_{ik}$. Then in implementation, we can directly take the two numerical matrix M_D and M_H as input to the DHB algorithm to save time and space.

The following theorem shows that d and h are upper bounds on the degree and height of a non-trivial polynomial solution to the system $M\mathbf{x} = 0$.

Theorem 2.3.6 *Suppose that $M = [p_{ij}(n)]_{l \times m}$ ($l < m$) is a matrix whose entries are polynomials in $\mathbb{Z}[n]$. Let d and h be the integers returned by the DHB algorithm. Then there exists a non-trivial polynomial solution $\mathbf{x} = [x_1(n), x_2(n), \ldots, x_m(n)]^T$ of the system $M\mathbf{x} = 0$ such that* $\deg(x_i(n)) \leq d$ *and* $|x_i(n)| \leq h$ *for all* $i = 1, 2, \ldots, m$.

Proof. For convenience, we denote the resulting matrices after step 1 and step 2 in the DHB algorithm by M_1 and M_2, respectively. Let A be an arbitrary $l \times m$ ($l < m$) matrix whose entries are polynomials in n. From the definition of DH augment, for each square sub-matrix $B = [p_{ij}(n)]$ of A, there exists a square sub-matrix $C = [q_{ij}(n)]$ of $\widehat{A}$ such that $\deg(p_{ij}) \leq \deg(q_{ij})$ and $|p_{ij}| \leq |q_{ij}|$. By Lemma 2.3.2, we have $\mathcal{D}(B) \leq \mathcal{D}(C)$ and $\mathcal{H}(B) \leq \mathcal{H}(C)$. Repeating the discussion, we derive that for each square sub-matrix B of M, there exists a square sub-matrix C of M_1 such that $\mathcal{D}(B) \leq \mathcal{D}(C)$ and $\mathcal{H}(B) \leq \mathcal{H}(C)$. One more step shows that this also holds for M_2.

Now by Lemma 2.3.1, for each square sub-matrix C of M_2, we have $\mathcal{D}(C) \leq \mathcal{D}(M_2)$ and $\mathcal{H}(C) \leq \mathcal{H}(M_2)$. Therefore, for each square sub-matrix B of M, we finally have

$$\mathcal{D}(B) \leq \mathcal{D}(M_2) \quad \text{and} \quad \mathcal{H}(B) \leq \mathcal{H}(M_2).$$

Since there exists a non-trivial solution to the system $M\mathbf{x} = 0$ represented by the determinants of sub-matrices of M up to a sign, the theorem follows. ∎

Sometimes we may need only the bound on part of the unknowns, say x_j's, $j \in S \subseteq \{1, 2, \ldots, m\}$. Then in the DHB algorithm we may use the following *partial DH augment* instead:

Choose a column, say the k-th ($k \in S$) column, of the matrix A, replace those entries $p_{ij}(n)$ ($j \in S$) that satisfy $|p_{ij}| < |p_{ik}|$ or $\deg(p_{ij}) < \deg(p_{ik})$ with $h \cdot n^d$ where $h = \max(|p_{ij}|, |p_{ik}|)$ and $d = \max(\deg(p_{ij}), \deg(p_{ik}))$, and finally delete the k-th column from A. We call the resulting algorithm the *partial DHB algorithm*. An extreme case is that $|S| = 1$, which corresponds to the operation that directly delete the column that S indicates.

Notice that d and h obtained by the partial DHB algorithm are the upper bounds for the degrees and heights of the x_j's with $j \in S$ which constitute part of a non-trivial solution of the system $M\mathbf{x} = 0$. This can be proved by the same arguments as in Theorem 2.3.6.

2.4 Numerically Solving Symbolic Linear System of Equations

In this section, we introduce a numerical way to solve the symbolic linear system of equations.

First recall the classical two steps to find polynomial solutions of linear recurrences with polynomial coefficients.

1. Compute an upper bound d for the possible degrees of polynomial solutions of the recurrence.

2. Given d, find polynomial solution(s) with degree(s) no larger than d.

Step 1 can be performed by Abramov's [1] and Petkovšek's [52] approaches. The most common way to perform step 2 is the method of undetermined coefficients, which has been also used to find non-zero polynomial solutions of first-order recurrence in Gosper's algorithm [31] and arbitrary order recurrence in algorithm Poly [52]. Some more sophisticated methods for this purpose are also devised, e.g., [8] and the interpolation techniques [17, 67].

Inspired by this, we split the problem of solving symbolic linear system of equations into the following two subproblems:

1. Estimate an overall upper bound d, or a sequence of upper bounds d_i's for the possible degrees of polynomial solutions of the system.

2. Given d or d_i's, find all polynomial solution(s) with degree(s) no larger than d.

In fact, the first problem has been fixed by the (partial) DHB algorithm introduced in Section 2.3, and the method of undetermined coefficients can be used to solve the second problem as follows, first plug generic polynomials of degree d (or d_i's) for the unknown polynomials $x_i(n)$'s into the symbolic system, then equate the coefficients of like powers of n, and solve the resulting linear algebraic equations for the unknown coefficients of $x_i(n)$'s. Note that these equations are linear since the original system is linear in $x_i(n)$'s. Finally, substitute the coefficients back into the generic polynomials to retrieve the polynomial solutions.

In the following, we walk through a running example to illustrate the general idea.

Consider $S(n) = \sum_k \binom{n}{k}^2$, which will be discussed again in example 3.5.2. By Zeilberger's algorithm [46], we will produce the following linear symbolic system

$$\begin{bmatrix} n^2+2n+1 & n^2+2n+1 & n^2+2n+1 & -1 & -1 & -1 \\ 4n^2+10n+6 & 2n^2+4n+2 & 0 & -2 & n & 2n+2 \\ 6n^2+18n+13 & n^2+2n+1 & 0 & 0 & 2n+3 & -n^2+3 \\ 2n+3 & 0 & 0 & 0 & 0 & -n-1 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} e_0(n) \\ e_1(n) \\ e_2(n) \\ x_0(n) \\ x_1(n) \\ x_2(n) \end{bmatrix} = 0. \tag{2.4.1}$$

The DHB algorithm will indicate that the degrees of e_i's and x_i's are no larger than 7. Thus write e_i's and x_i's in the following generic form:

$$e_i(n) = \sum_{k=0}^{7} a_{ik} n^k, \text{ for } i = 0, 1, 2, \tag{2.4.2}$$

$$x_i(n) = \sum_{k=0}^{7} a_{3+i,k} n^k, \text{ for } i = 0, 1, 2, \tag{2.4.3}$$

where a_{ik}'s are the undetermined coefficients. Substitute them into the system (2.4.1), then the equations become polynomials in n with unknown coefficients a_{ik}. Now equating to zero the coefficients of each power of n in these polynomials, we get a linear algebraic system with 47 equations and 48 unknowns (see figure 4.1), the solution of this system is shown in figure 4.2, where $a_{23}, a_{24}, a_{25}, a_{40}, a_{41}$ are the five free variables. Then substitute the solution back into (2.4.2) and (2.4.3), we finally obtain

$$\begin{cases} e_0(n) = & 0 \\ e_1(n) = & -4a_{25}n^5 + (2a_{25} - 4a_{24})n^4 + (-4a_{23} - 4a_{25} + 2a_{24})n^3 \\ & +(2a_{23} - 2a_{43})n^2 + (-4a_{23} - 4a_{21} - 12a_{25} + 6a_{24} + a_{43})n \\ & -24a_{23} - 6a_{21} - 72a_{25} + 36a_{24} + 6a_{43} \\ e_2(n) = & a_{25}n^5 + a_{24}n^4 + a_{23}n^3 + (2a_{25} - a_{24} + \frac{1}{2}a_{43})n^2 + a_{21}n \\ & +8a_{23} + 2a_{21} + 24a_{25} - 12a_{24} - 2a_{43} \\ x_0(n) = & -24a_{23} - 6a_{21} - 72a_{25} + 36a_{24} + 6a_{43} \\ & +(-48a_{23} - 15a_{21} - 144a_{25} + 72a_{24} + 12a_{43})n \\ & +(-24a_{23} - 12a_{21} - 78a_{25} + 39a_{24} + \frac{9}{2}a_{43})n^2 \\ & +(-3a_{21} - 3a_{43} - 3a_{23} - 12a_{25} + 6a_{24})n^3 \\ & +(-6a_{23} - \frac{3}{2}a_{43} - 6a_{25})n^4 + (-3a_{23} - 3a_{25} - 6a_{24})n^5 \\ & +(-6a_{25} - 3a_{24})n^6 - 3a_{25}n^7 \\ x_1(n) = & 8a_{23} + 2a_{21} + 24a_{25} - 12a_{24} - 2a_{43} \\ & +(4a_{21} + 12a_{23} + 36a_{25} - 18a_{24} - 3a_{43})n \\ & +(2a_{23} + 2a_{21} + 8a_{25} - 4a_{24})n^2 \\ & +a_{43}n^3 + (2a_{23} + 2a_{25})n^4 + 2a_{24}n^5 + 2a_{25}n^6 \end{cases}$$

Moreover, taking $a_{25} = a_{24} = a_{23} = a_{43} = 0$ and $a_{21} = 1$, e_i's can be reduced to

$$\begin{cases} e_0(n) = 0 \\ e_1(n) = -4n - 6 \\ e_2(n) = n + 2 \end{cases} \tag{2.4.4}$$

On the other hand, part of the solution of the linear symbolic system (2.4.1) is

$$\begin{cases} e_0(n) = 0 \\ e_1(n) = -\frac{4n+6}{n+2} e_2(n) \\ e_2(n) = e_2(n) \end{cases}$$

where $e_2(n)$ is a free variable. Taking $e_2(n) = n + 2$, we obtain the same polynomial solution as (2.4.4).

In the above discussion, we applied DHB algorithm to produce an overall degree upper bound for all the e_i's and x_i's. Alternatively, we can also take advantage of the partial DHB algorithm to compute the degree bounds of e_i's and x_i's separately. For example, taking $S = \{1\}$ in the partial DHB algorithm will return the degree bound of $e_0(n)$, taking $S = \{2\}$ will return $e_1(n)$'s degree bound, etc. This strategy requires calling the partial DHB algorithm multi-times, but will produce smaller degree bounds.

Now we formulate the general steps to numerically solve linear symbolic system of equations.

Input: A linear system $M\mathbf{x} = 0$, where matrix $M = [p_{ij}(n)]$ is of order $l \times m$ $(l < m)$, and $p_{ij}(n)$'s are polynomials in $\mathbb{Z}[n]$.

Output: Polynomial solution(s) of the system.

1. Given symbolic system $M\mathbf{x} = 0$, apply (partial) DHB algorithm to get the degree bound(s) of x_i's.

2. Write x_i's in the generic form like (2.4.2) and (2.4.3) with undetermined coefficients a_{ik}'s, and plug into $M\mathbf{x} = 0$, then the equations become polynomials in n.

3. Equate to zero the coefficients of each power of n in these polynomials to get a linear algebraic system of equations for the unknown coefficients a_{ik}'s.

4. Solve the linear algebraic system and substitute the solution back into the expression of x_i's.

$$
\left\{\begin{array}{r}
-a_{20}+a_{40}+a_{50}+a_{30}-a_{00}-a_{10}=0\\
a_{41}-2a_{10}-2a_{20}-a_{21}+a_{31}-a_{01}-2a_{00}-a_{11}+a_{51}=0\\
-2a_{27}-a_{16}-a_{26}-2a_{07}-a_{06}-2a_{17}=0\\
-a_{17}-a_{27}-a_{07}=0\\
-2a_{01}+a_{32}-2a_{11}-a_{02}+a_{42}-a_{00}+a_{52}-a_{12}-a_{22}-a_{20}-a_{10}-2a_{21}=0\\
-a_{14}+a_{44}-a_{24}-a_{22}-a_{04}-2a_{23}+a_{54}+a_{34}-2a_{03}-2a_{13}-a_{02}-a_{12}=0\\
a_{33}+a_{53}-a_{23}-a_{13}-a_{11}-2a_{02}-a_{21}-a_{01}-2a_{12}-a_{03}-2a_{22}+a_{43}=0\\
-a_{23}-a_{15}-a_{03}-2a_{04}-a_{25}+a_{45}-a_{13}-a_{05}+a_{55}+a_{35}-2a_{24}-2a_{14}=0\\
-a_{06}-a_{26}-a_{24}-2a_{15}-a_{14}-a_{04}+a_{36}+a_{46}-a_{16}+a_{56}-2a_{05}-2a_{25}=0\\
a_{57}-a_{15}-a_{27}-a_{07}-a_{17}-a_{05}-2a_{16}+a_{37}-a_{25}-2a_{26}+a_{47}-2a_{06}=0\\
6a_{00}+2a_{10}-2a_{30}+2a_{50}=0\\
10a_{00}+2a_{51}+6a_{01}+4a_{10}+2a_{11}+2a_{50}+a_{40}-2a_{31}=0\\
4a_{06}+4a_{17}+2a_{57}+10a_{07}+2a_{16}+a_{47}=0\\
2a_{17}+4a_{07}=0\\
10a_{01}+4a_{00}+2a_{52}+2a_{12}+6a_{02}+4a_{11}+2a_{51}+a_{41}+2a_{10}-2a_{32}=0\\
2a_{54}+2a_{12}+2a_{53}+2a_{14}+10a_{03}+a_{43}+4a_{13}-2a_{34}+4a_{02}+6a_{04}=0\\
4a_{12}+6a_{03}+10a_{02}+2a_{13}+2a_{11}+2a_{53}+4a_{01}+a_{42}-2a_{33}+2a_{52}=0\\
2a_{15}+2a_{55}-2a_{35}+2a_{13}+2a_{54}+a_{44}+4a_{14}+6a_{05}+4a_{03}+10a_{04}=0\\
2a_{14}+2a_{56}+a_{45}+6a_{06}+2a_{16}+4a_{15}+2a_{55}+4a_{04}-2a_{36}+10a_{05}=0\\
2a_{57}+4a_{05}+a_{46}-2a_{37}+2a_{15}+4a_{16}+10a_{06}+6a_{07}+2a_{56}+2a_{17}=0\\
-13a_{00}-3a_{50}-a_{10}-3a_{40}=0\\
-13a_{01}-18a_{00}-3a_{41}-2a_{40}-2a_{10}-a_{11}-3a_{51}=0\\
-2a_{17}-18a_{07}-6a_{06}-a_{16}+a_{56}-2a_{47}=0\\
-6a_{07}-a_{17}+a_{57}=0\\
-2a_{11}-3a_{52}-2a_{41}-13a_{02}-a_{10}-18a_{01}+a_{50}-3a_{42}-6a_{00}-a_{12}=0\\
-3a_{54}-13a_{04}-3a_{44}+a_{52}-a_{12}-a_{14}-2a_{13}-18a_{03}-2a_{43}-6a_{02}=0\\
-6a_{01}-2a_{12}-3a_{53}-13a_{03}-a_{11}-18a_{02}+a_{51}-2a_{42}-3a_{43}-a_{13}=0\\
-18a_{04}-2a_{14}-6a_{03}-a_{13}-13a_{05}+a_{53}-3a_{55}-2a_{44}-3a_{45}-a_{15}=0\\
-2a_{15}-3a_{56}-3a_{46}-18a_{05}-a_{14}-2a_{45}-a_{16}-6a_{04}-13a_{06}+a_{54}=0\\
-a_{15}-2a_{46}-a_{17}-13a_{07}-2a_{16}-6a_{05}-18a_{06}+a_{55}-3a_{47}-3a_{57}=0\\
-a_{50}+3a_{00}=0\\
2a_{00}+3a_{01}-a_{51}-a_{50}=0\\
-a_{57}+2a_{07}=0\\
3a_{02}-a_{51}+2a_{01}-a_{52}=0\\
3a_{04}+2a_{03}-a_{54}-a_{53}=0\\
2a_{02}-a_{52}-a_{53}+3a_{03}=0\\
3a_{05}-a_{54}-a_{55}+2a_{04}=0\\
3a_{06}+2a_{05}-a_{55}-a_{56}=0\\
3a_{07}-a_{57}-a_{56}+2a_{06}=0\\
a_{00}=0\\
a_{01}=0\\
a_{02}=0\\
a_{04}=0\\
a_{03}=0\\
a_{05}=0\\
a_{06}=0\\
a_{07}=0
\end{array}\right.
$$

Figure 4.1: The linear algebraic system for a_{ik}

$$\left\{\begin{array}{l}
a_{00} = 0\\
a_{01} = 0\\
a_{02} = 0\\
a_{03} = 0\\
a_{04} = 0\\
a_{05} = 0\\
a_{06} = 0\\
a_{07} = 0\\
a_{16} = 0\\
a_{17} = 0\\
a_{26} = 0\\
a_{27} = 0\\
a_{47} = 0\\
a_{50} = 0\\
a_{51} = 0\\
a_{52} = 0\\
a_{53} = 0\\
a_{54} = 0\\
a_{55} = 0\\
a_{56} = 0\\
a_{57} = 0\\
a_{10} = -3a_{40}\\
a_{11} = -3a_{41} + 4a_{40}\\
a_{12} = -6a_{23} - 24a_{25} + 12a_{24} - 2a_{41} + 4a_{40}\\
a_{13} = 2a_{24} - 4a_{23} - 4a_{25}\\
a_{14} = 2a_{25} - 4a_{24}\\
a_{15} = -4a_{25}\\
a_{20} = a_{40}\\
a_{21} = a_{41} - \frac{3}{2}a_{40}\\
a_{22} = \frac{1}{2}a_{41} - a_{40} + 2a_{23} + 8a_{25} - 4a_{24}\\
a_{23} = a_{23}\\
a_{24} = a_{24}\\
a_{25} = a_{25}\\
a_{30} = -3a_{40}\\
a_{31} = -\frac{3}{2}a_{40} - 3a_{41}\\
a_{32} = -6a_{23} - 24a_{25} + 12a_{24} - \frac{15}{2}a_{41} + 9a_{40}\\
a_{33} = -48a_{25} + 24a_{24} - 6a_{41} + \frac{21}{2}a_{40} - 15a_{23}\\
a_{34} = -12a_{23} - 24a_{25} + 9a_{24} - \frac{3}{2}a_{41} + 3a_{40}\\
a_{35} = -6a_{24} - 3a_{23} - 3a_{25}\\
a_{36} = -6a_{25} - 3a_{24}\\
a_{37} = -3a_{25}\\
a_{40} = a_{40}\\
a_{41} = a_{41}\\
a_{42} = 2a_{41} - 3a_{40} + 2a_{23} + 8a_{25} - 4a_{24}\\
a_{43} = 12a_{25} - 6a_{24} + a_{41} - 2a_{40} + 4a_{23}\\
a_{44} = 2a_{23} + 2a_{25}\\
a_{45} = 2a_{24}\\
a_{46} = 2a_{25}
\end{array}\right.$$

Figure 4.2: Solution to the system in Fig. 4.1.

Chapter 3

Proving Hypergeometric Identities by Numerical Verifications

In this chapter, we present our approach to proving hypergeometric identities by numerical verifications. We first briefly describe the general idea of numerical verifications accompanied with a running example. Next we show how to apply our approach based on Sister Celine's method to estimate n_1 for hypergeometric identities. Then we combine our method with Zeilberger's algorithm to estimate n_1. The q-analogue is also discussed. Plenty of examples of both ordinary and q cases are presented at last.

3.1 Overview of the Approach

Before going into details of our estimation algorithms, let us first summarize the general idea of estimating n_1 and walk through an example.

Suppose that we aim to prove an identity of the form

$$\sum_k F(n,k) = f(n), \quad n \geq n_0, \tag{3.1.1}$$

where n_0 is a non-negative integer, $F(n,k)$ is a proper hypergeometric term over $\mathbb{Q}$, and $f(n)$ is a hypergeometric term.

Usually, under appropriate hypothesis, there exists polynomial coefficients recurrence for $S(n) = \sum_k F(n,k)$ such that

$$\sum_{i=0}^{L} a_i(n)S(n+i) = 0, \quad n \geq n_0, \tag{3.1.2}$$

where L is a non-negative integer, a_i's are (not all zero) polynomials depending only on n. If $a_L(n) \neq 0$ when $n > n_a$, we have

$$S(n+L) = -\frac{1}{a_L(n)} \sum_{i=0}^{L-1} a_i(n) S(n+i), \quad \text{for all } n \geq n'_a = \max(n_a + 1, n_0).$$

Then $S(n)$ is completely determined by its initial values: $S(n_0)$, $S(n_0+1)$,..., $S(n'_a + L - 1)$. Once proving that $f(n)$ satisfies the same recurrence and $S(n)$ agrees with $f(n)$ on these initial values, we immediately have $S(n) = f(n)$ for all $n \geq n_0$.

Given the recurrence (3.1.2) for $S(n)$, to verify that $f(n)$ and $S(n)$ satisfy the same recurrence is just to show $\sum_{i=0}^{L} a_i(n) f(n+i) = 0$, or equally

$$R(n) = \sum_{i=0}^{L} a_i(n) \frac{f(n+i)}{f(n)} = 0.$$

Notice that since $f(n)$ is hypergeometric, $R(n)$ is a rational function of n. Suppose that the degree of the numerator polynomial of $R(n)$ is less than or equal to n_f. Then $R(n)$ is identical to 0 if and only if $R(n) = 0$ holds for $n = n_0, n_0 + 1, \ldots, n_0 + n_f$, which in turn can be verified by checking $S(n) = f(n)$ for $n = n_0, \ldots, n_0 + n_f + L$.

Now let $n_1 = \max(n'_a + L - 1, n_0 + n_f + L)$, then we can safely claim that identity (3.1.1) holds for all $n \geq n_0$ if and only if it holds for $n = n_0, \ldots, n_1$.

Among the above three numbers L, n_a and n_f, n_a is usually the most difficult to estimate and in most cases the largest. Yen used the following method to estimate n_a. First use Sister Celine's method to obtain a linear system of equations in some unknowns related to the $a_i(n)$'s, then apply Cramer's rule to the system formally and thus get estimates for the degree d and the height m of the coefficients of $a_i(n)$, finally take md as n_a by Proposition 3.6 in [73].

Beyond Sister Celine's method, we also explore the techniques due to Mohammed and Zeilberger [46] to obtain the linear system of equations. Advantages of the latter techniques are mainly that the resulting system is directly in the unknowns $a_0(n), \ldots, a_L(n)$ and has smaller size. In addition, we dig out more information from the concrete linear system of equations so that the estimates for the height become more accurate. Our approach sharply reduces the estimates of n_1 for hypergeometric identities, and is also applicable to the q-hypergeometric cases.

Now we take an example to illustrate the general idea of our method. Let us consider the identity

$$\sum_k k\binom{n}{k} = n2^{n-1}, \quad n \geq 0.$$

By the **Theorem** in [46], there exist polynomials $a_0(n)$, $a_1(n), x_0(n), x_1(n)$ such that

$$a_0(n)F(n,k) + a_1(n)F(n+1,k) - G(n,k+1) + G(n,k) = 0, \quad n \geq 0,$$

where $G(n,k) = (x_0(n) + x_1(n)k)\binom{n}{k-1}$. Simplifying the above equation and equating to zero the coefficients of each power of k, we get the following linear system of equations after eliminating the common factor $-(n+1)$ in the third equation.

$$\begin{bmatrix} 1 & 0 & 0 & -2 \\ n+1 & n+1 & 2 & -n \\ 0 & 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_0(n) \\ a_1(n) \\ x_0(n) \\ x_1(n) \end{bmatrix} = 0.$$

By Cramer's rule, we know that

$$[a_0(n), a_1(n), x_0(n), x_1(n)]^T$$
$$= \left[\begin{vmatrix} 2 & 0 & 0 \\ n & n+1 & 2 \\ -1 & 0 & 1 \end{vmatrix}, \begin{vmatrix} 1 & 2 & 0 \\ n+1 & n & 2 \\ 0 & -1 & 1 \end{vmatrix}, \begin{vmatrix} 1 & 0 & 2 \\ n+1 & n+1 & n \\ 0 & 0 & -1 \end{vmatrix}, \begin{vmatrix} 1 & 0 & 0 \\ n+1 & n+1 & 2 \\ 0 & 0 & 1 \end{vmatrix} \right]^T$$

is a non-trivial solution to the above system. Instead of computing these determinants, we estimate their degrees and heights using only the degree and height of each entry. Theorem 2.1.4 indicates that $\deg(a_i(n)) \leq 1, i = 0, 1$ and their heights are less than or equal to 5 (In fact $a_0(n) = 2n + 2$ and $a_1(n) = -n$). Lemma 2.1.1 guarantees that taking $n_a = 5$ is large enough. Noting further that $n_f = 2$ and $L = 1$, we finally get $n_1 = 6$.

3.2 Estimating n_1 for Hypergeometric Identities Based on Sister Celine's Method

In this section, we try our approach of estimating n_1 on the system of equations produced by Sister Celine's method.

First let us recall Sister Celine's method. Given bivariate hypergeometric function $F(n,k)$, Sister Celine's method will find a recurrence for $F(n,k)$ of the form

$$\sum_{i=0}^{I}\sum_{j=0}^{J}\alpha_{i,j}(n)F(n-j,k-i)=0. \tag{3.2.1}$$

The general steps of Sister Celine's method are as follows: try fixed values of I and J from, say, $I = J = 1$, take coefficients $\alpha_{i,j}(n)$ as undetermined in the recurrence formula (3.2.1). Then divide each term in the sum by $F(n, k)$, simplify each ratio into rational function of n and k, place the entire reduced expression over a single common denominator, and collect the numerator as a polynomial in k. Finally, equating to zero the coefficients of each power of k in the numerator polynomial, we get a system of linear equations for the unknown coefficients $\alpha_{i,j}$. If the solution exists, we are done. If not, increase I and/or J and try the whole thing again.

The following well-known fundamental theorem guarantees that Sister Celine's method will succeed if I, J are large enough.

The Fundamental Theorem (See [54, p. 65]). *Let $F(n, k)$ be a proper hypergeometric term. Then $F(n, k)$ satisfies a k-free recurrence relation. That is to say, there exist positive integers I, J and polynomials $\alpha_{i,j}(n)$ for $i = 0, \ldots, I$; $j = 0, \ldots, J$, not all zero, such that*

$$\sum_{i=0}^{I}\sum_{j=0}^{J}\alpha_{i,j}(n)F(n-j,k-i)=0 \tag{3.2.2}$$

holds at every point (n, k) at which $F(n, k) \neq 0$ and all of the values of F that occur in (3.2.2) *are well defined.*

The Fundamental Theorem in the case of one summation variable was first proved by Verbaeten [66]. The cases of two and more summation variables and q and multi-q were proved by Wilf and Zeilberger [69] in a constructive Sister Celine's method-like way.

Summarizing, by Sister Celine's method, we can get a homogeneous linear system $M\mathbf{x} = 0$ for the unknowns $\mathbf{x} = [\alpha_{0,0}(n), \alpha_{0,1}(n), \ldots, \alpha_{I,J}(n)]^T$, whose degree and height upper bounds can then be computed by the DHB algorithm introduced in Section 2.3. However, the objects we actually care are the upper bounds of the leading coefficient of the recurrence satisfied by the summation of $F(n, k)$ over k. Before showing the relation between the two groups of coefficients, we introduce the concept of admissible hypergeometric term that is first proposed in [69] for yielding a homogeneous recurrence relation for the summation from the k-free recurrence for the summand in the case of *standard boundary conditions*, which refers to the fact that the summand vanishes outside the range of summation.

For a fixed integer n, let $B(n) = [a(n), b(n)]$ denote a maximal interval of integer values of k for which $F(n,k)$ is well defined and non-zero. Suppose that there are intervals $\alpha(n) \le k < a(n)$ and $b(n) < k \le \beta(n)$ in which $F(n,k)$ is well defined and is equal to 0. Then the interval $B(n)$ is called *a natural support* of $F(n,k)$.

Definition 3.2.1 *An admissible hypergeometric term $F(n,k)$ is one in which, for all sufficiently large n there is a natural support $B(n)$ such that $B(n)$ is compact and*

$$B(n) \subseteq B(n+1) \subseteq B(n+2) \subseteq \cdots, \quad n > n_0,$$

and the intervals of zero values that surround $B(n)$ satisfy

$$\beta(n-j) \ge b(n) + I \quad \text{and} \quad \alpha(n-j) \le a(n) - I$$

for $0 \le j \le J$ and $n > n_0$, where I and J are the orders of a k-free recurrence that $F(n,k)$ satisfies.

Now we are ready to reveal the relation between the coefficients of the recurrences of the summand and the summation.

Theorem 3.2.2 *Let $F(n,k)$ be an admissible proper hypergeometric term satisfying the k-free recurrence relation (3.2.2). Then $S(n) = \sum_{k\in B(n)} F(n,k)$ satisfies*

$$a_0(n)S(n) + a_1(n)S(n-1) + \cdots + a_J(n)S(n-J) = 0, \tag{3.2.3}$$

where

(1) $a_0(n), \ldots, a_J(n) \in \mathbb{Z}[n]$,

(2) $a_0(n), \ldots, a_J(n)$ *are not all zero,*

(3) $a_j(n)$ *can be written in form of non-negative linear combination of α_{ij}'s. Moreover, let D_α and H_α denote the maximal degree and maximal height of α_{ij}'s, D and H denote the maximal degree and maximal height of a_j's, then*

$$D \le D_\alpha, \tag{3.2.4}$$

$$H \le H_\alpha \cdot \max_{0\le m\le I} \left\{ \binom{m+\theta_m+1}{m+1} + \binom{I-\theta_m}{m+1} \right\}, \tag{3.2.5}$$

where $\theta_m = \lfloor \frac{I-m}{2} \rfloor$.

Proof. First we introduce the concept of linear recurrence operators (see [54] for details). Let N and K be the linear recurrence operator such that

$$NF(n,k) = F(n-1,k), \quad KF(n,k) = F(n,k-1), \quad n,k \in \mathbb{Z}.$$

Iterating, we have that for $r \in \mathbb{Z}$,

$$N^rF(n,k) = F(n-r,k), \quad K^rF(n,k) = F(n,k-r), \quad n,k \in \mathbb{Z}.$$

Then the recurrence (3.2.2) can be rewritten using the notations of operator as

$$LF(n,k) = 0,$$

where

$$L(n,N,K) = \sum_{i=0}^{I}\sum_{j=0}^{J} \alpha_{i,j}(n)N^jK^i.$$

Now we proceed with our proof. Suppose

$$L(n,N,K) = (K-1)^m S(n,N,K), \tag{3.2.6}$$

where

$$S(n,N,K) = \sum_{i=0}^{I-m}\sum_{j=0}^{J} \beta_{i,j}(n)N^jK^i,$$

and $S(n,N,1) \neq 0$. Denote $a^{(i)} = a(a-1)\cdots(a-i+1)$, then

$$\begin{aligned}&(k)^m L(n,N,K)\\ &= (K-1)\left(\sum_{i=0}^{m-1} m^{(i)}(k+1)^{(m-i)}(K-1)^{m-1-i}S(n,N,K)\right) + m!S(n,N,K).\end{aligned}$$

Because $F(n,k)$ is admissible, we have

$$0 = \sum_{k=a(n)}^{b(n)+I} k^{(m)}LF(n,k) = \sum_{k=a(n)}^{b(n)+I} m!SF(n,k) = m!\sum_{j=0}^{J} f(n-j)\left(\sum_{i=0}^{I-m}\beta_{i,j}(n)\right).$$

Then $S(n,N,1) \neq 0$ guarantees that $a_j(n) = \sum_{i=0}^{I-m}\beta_{i,j}(n)$ are not all zero. Therefore $S(n)$ satisfies the following non-trivial recurrence relation with integer coefficients.

$$a_0(n)S(n) + a_1(n)S(n-1) + \cdots + a_J(n)S(n-J) = 0.$$

In the following we prove assertion (3). First by comparing equation (3.2.6) with the equation

$$\sum_{i,j}\alpha_{i,j}(n)N^jK^i=\sum_{i,j}\alpha_{i,j}(n)N^j(K-1+1)^i=\sum_{i,j}\sum_{r=0}^{i}\binom{i}{r}\alpha_{i,j}(n)N^j(K-1)^r,$$

we get

$$S(n,N,K)=\sum_{i,j}\sum_{r=m}^{i}\binom{i}{r}\alpha_{i,j}(n)N^j(K-1)^{r-m}.$$

Thus

$$\beta_{i,j}(n)=\sum_{l=m+i}^{I}\alpha_{l,j}(n)\sum_{r=m+i}^{l}(-1)^{r-m-i}\binom{l}{r}\binom{r-m}{i}=\sum_{l=m+i}^{I}\binom{l-i-1}{m-1}\alpha_{l,j}(n).$$

Therefore,

$$a_j(n)=\sum_{i=0}^{I-m}\beta_{i,j}(n)=\sum_{i=0}^{I-m}\sum_{l=m+i}^{I}\binom{l-i-1}{m-1}\alpha_{l,j}(n)=\sum_{l=m}^{I}\binom{l}{m}\alpha_{l,j}(n)$$

is a non-negative linear combination of $\alpha_{i,j}(n)$'s, consequently, $D\le D_\alpha$ is obvious.

On the other hand, for the second part of assertion (3), first note that

$$\begin{aligned}&\sum_{i,j}\alpha_{i,j}(n)N^jK^i=(K-1)^m\sum_{i,j}\beta_{i,j}(n)N^jK^i\\ &\qquad\Longleftrightarrow\sum_{i,j}\alpha_{I-i,j}(n)N^j(K^{-1})^i=(1-K^{-1})^m\sum_{I-m-i,j}\beta_{i,j}(n)N^j(K^{-1})^i,\end{aligned}$$

we have

$$\beta_{I-m-i,j}(n)=(-1)^m\sum_{l=m+i}^{I}\binom{l-i-1}{m-1}\alpha_{I-l,j}(n),$$

or

$$\beta_{i,j}(n)=(-1)^m\sum_{l=0}^{i}\binom{m-1+i-l}{m-1}\alpha_{l,j}(n).$$

Therefore

$$\begin{aligned}a_j(n)&=\sum_{i=0}^{\theta_m}(-1)^m\sum_{l=0}^{i}\binom{m-1+i-l}{m-1}\alpha_{l,j}(n)+\sum_{i=\theta_m+1}^{I-m}\sum_{l=m+i}^{I}\binom{l-i-1}{m-1}\alpha_{l,j}(n)\\&=(-1)^m\sum_{l=0}^{\theta_m}\binom{m+\theta_m-l}{m}\alpha_{l,j}(n)+\sum_{l=m+\theta_m+1}^{I}\binom{l-\theta_m-1}{m}\alpha_{l,j}(n).\end{aligned}$$

By the definition of H and H_α, we finally get

$$\begin{aligned} H &\le H_\alpha\left(\sum_{l=0}^{\theta_m}\binom{m+\theta_m-l}{m}+\sum_{l=m+\theta_m+1}^{I}\binom{l-\theta_m-1}{m}\right) \\ &= H_\alpha\left(\binom{m+\theta_m+1}{m+1}+\binom{I-\theta_m}{m+1}\right). \end{aligned}$$

Summarizing, combining Sister Celine's method, the DHB algorithm and Theorem 3.2.2, we are able to get the upper bound of the heights of $a_j(n)$'s. Note that although $a_0(n),\ldots,a_J(n)$ are not all zero, we have no guarantee that the specific leading coefficient $a_0(n)$ is non-zero, thus it is necessary to find the height upper bound of all the $a_j(n)$'s. Moreover, because $a_j(n)$'s are polynomials with integer coefficients, we can safely take $n_a = r.h.s.$ of (3.2.5).

Suppose we want to prove the identity $\sum_k F(n,k) = f(n)$ for $n \ge n_0$. The next job is to estimate n_f. Actually, n_f is given by an upper bound of the degree of the numerator polynomial of

$$R(n) = \sum_{j=0}^{J} a_j(n)\frac{f(n-j)}{f(n)}.$$

Let $A(n)$ be the common denominator of $\frac{f(n-j)}{f(n)}$ for $j = 0,\ldots,J$, then the degree of the numerator polynomial of $R(n)$ is bounded by the largest degree of $a_i(n)$ (bounded by D_α in Theorem 3.2.2) plus the largest degree of $A(n)\cdot\frac{f(n-j)}{f(n)}$ for $j = 0,\ldots,J$, denoted by d_f. Hence $n_f = D_\alpha + d_f$ is a safe estimation for verifying that $f(n)$ and $S(n)$ satisfy the same recurrence.

Note that we adopt the descending recurrence in accordance with the conventions of Sister Celine's method, the formulation of n_1 is slightly different from the discussions in Section 3.1 where the ascending recurrence is used. Given n_a and recurrence (3.2.3), we have

$$S(n) = -\frac{1}{a_0(n)}\sum_{j=1}^{J} a_j(n)S(n-j), \quad \text{for all } n \ge n'_a = \max(n_a+1, n_0+J).$$

Thus $S(n)$ is completely determined by its initial values: $S(n_0)$,$S(n_0+1)$,. . .,$S(n'_a-1)$.

On the other hand, to verify that $f(n)$ and $S(n)$ satisfy the same recurrence is equivalent to prove the rational function

$$R(n) = \sum_{j=0}^{J} a_j(n) \frac{f(n-j)}{f(n)}$$

is identical to zero. Given the degree bound n_f of the numerator polynomial of $R(n)$, $R(n) \equiv 0$ can be verified by checking $R(n) = 0$ for $n = n_0, n_0+1, \ldots, n_0+n_f$, which in turn is equivalent to check $S(n) = f(n)$ for $n = n_0, \ldots, n_0 + n_f$.

In conclusion, taking $n_1 = \max(n'_a - 1, n_0 + n_f)$, we can safely claim that $\sum_k F(n,k) = f(n)$ holds for all $n \geq n_0$ if and only if it holds for $n = n_0, \ldots, n_1$. Now we give the complete algorithm to estimate n_1 for admissible proper hypergeometric identities based on Sister Celine's method.

Sister Celine-DHB Algorithm

Input: An admissible proper hypergeometric term $F(n,k)$ over $\mathbb{Q}$, a hypergeometric term $f(n)$, and n_0.

Output: An integer n_1 such that $\sum_k F(n,k) = f(n)$ holds for $n \geq n_0$ if and only if it holds for $n = n_0, \ldots, n_1$.

1. Let $M\mathbf{x} = 0$ be the homogeneous linear system produced by applying Sister Celine's method on $F(n,k)$, where $M = [p_{ij}(n)]_{l \times m}$ $(l < m)$, $\mathbf{x} = [\alpha_{0,0}(n), \alpha_{0,1}(n), \ldots, \alpha_{I,J}(n)]^T$. Suppose that Sister Celine' method succeeds at order (I, J).

2. Apply DHB algorithm to M to get (d, h), set $D_\alpha = d,\ H_\alpha = h$.

3. By Theorem 3.2.2(3), let

$$d_a = D_\alpha, \tag{3.2.7}$$

$$n_a = H_\alpha \cdot \max_{0 \leq m \leq I} \left\{ \binom{m + \theta_m + 1}{m+1} + \binom{I - \theta_m}{m+1} \right\}, \tag{3.2.8}$$

where $\theta_m = \left\lfloor \frac{I-m}{2} \right\rfloor$.

4. Compute the common denominator $A(n)$ of $\frac{f(n-j)}{f(n)}, j = 0, \ldots, J$ and then find the largest degree d_f of $A(n) \cdot \frac{f(n-j)}{f(n)}$. Set $n_f = d_a + d_f$.

5. Return $n_1 = \max(n'_a - 1, n_0 + n_f)$, where $n'_a = \max(n_a + 1, n_0 + J)$.

3.3 Estimating n_1 for Hypergeometric Identities Based on Zeilberger's Algorithm

We follow the notation in [46] to write a proper hypergeometric term over $\mathbb{Q}$ as

$$F(n,k) = POL(n,k) \cdot H(n,k), \tag{3.3.1}$$

with $POL(n,k)$ being a polynomial in n and k, and

$$H(n,k) = \frac{\prod_{j=1}^{A}(a_j'')_{a_j'n+a_jk} \prod_{j=1}^{B}(b_j'')_{b_j'n-b_jk}}{\prod_{j=1}^{C}(c_j'')_{c_j'n+c_jk} \prod_{j=1}^{D}(d_j'')_{d_j'n-d_jk}} z^k,$$

where $a_j, a_j', b_j, b_j', c_j, c_j', d_j, d_j'$ are non-negative integers, and $z, a_j'', b_j'', c_j'', d_j'' \in \mathbb{Q}$.

Let

$$L = \max\left(\sum_{j=1}^{A} a_j + \sum_{j=1}^{D} d_j, \ \ \sum_{j=1}^{B} b_j + \sum_{j=1}^{C} c_j\right). \tag{3.3.2}$$

[46] showed that there exist polynomials $e_0(n), \ldots, e_L(n)$ in n and a rational function $R(n,k)$ such that $G(n,k) = R(n,k)F(n,k)$ satisfies

$$\sum_{i=0}^{L} e_i(n)F(n+i,k) = G(n,k+1) - G(n,k). \tag{3.3.3}$$

More precisely, let

$$\overline{H}(n,k) = \frac{\prod_{j=1}^{A}(a_j'')_{a_j'n+a_jk} \prod_{j=1}^{B}(b_j'')_{b_j'n-b_jk}}{\prod_{j=1}^{C}(c_j'')_{c_j'(n+L)+c_jk} \prod_{j=1}^{D}(d_j'')_{d_j'(n+L)-d_jk}} z^k, \tag{3.3.4}$$

$$u(k) = z\prod_{j=1}^{A}(a_j'n + a_jk + a_j'')_{a_j} \prod_{j=1}^{D}(d_j'(n+L) - d_jk + d_j'' - d_j)_{d_j}, \tag{3.3.5}$$

$$v(k) = \prod_{j=1}^{B}(b_j'n - b_jk + b_j'' - b_j)_{b_j} \prod_{j=1}^{C}(c_j'(n+L) + c_jk + c_j'')_{c_j}. \tag{3.3.6}$$

Let further

$$h(k) = \sum_{i=0}^{L} e_i(n)POL(n+i,k) \cdot \frac{H(n+i,k)}{\overline{H}(n,k)} \tag{3.3.7}$$

and $X(k) = \sum_{i=0}^{m} x_i(n)k^i$, where $m = \deg h - \max(\deg u, \deg v)$, $x_i(n)$ and $e_i(n)$ are unknown polynomials in n to be determined. Note that $h(k)$ is a polynomial in k. Then there is a non-trivial solution $e_0(n), \dots, e_L(n), x_0(n), \dots, x_m(n)$ of the equation

$$u(k)X(k+1) - v(k-1)X(k) - h(k) = 0. \tag{3.3.8}$$

Moreover, Equation (3.3.3) holds for $G(n,k) = v(k-1)X(k)\overline{H}(n,k)$.

The results of Mohammed and Zeilberger do not ensure that $e_0(n), \dots, e_L(n)$ are not all zeros, however, we can prejudge it and this bad situation rarely happens. Suppose that $e_0(n), \dots, e_L(n)$ are all zeros, then $h(k) = 0$ but $X(k) \neq 0$. By (3.3.8), we have

$$\frac{v(k-1)}{u(k)} = \frac{X(k+1)}{X(k)}. \tag{3.3.9}$$

It is well known that the rational function $\frac{v(k-1)}{u(k)}$ has a unique Gopser-Petkovšek representation [52] (GP representation, in short):

$$\frac{v(k-1)}{u(k)} = \frac{a(k)}{b(k)} \frac{c(k+1)}{c(k)}.$$

In most cases, $\frac{a(k)}{b(k)} \neq 1$, which implies that Equation (3.3.9) does not hold. Under this situation, we can claim that $e_0(n), \dots, e_L(n)$ are non-trivial.

Up to now, we have fixed L. It remains to estimate n_a and n_f.

Notice that Equation (3.3.8) is in fact a system of linear equations in the unknowns $e_0(n), \dots, e_L(n)$ and $x_0(n), \dots, x_m(n)$, which can be written as $M\mathbf{x} = 0$. By multiplying the common denominators, we may assume that each entry of M is a polynomial in $\mathbb{Z}[n]$. Thus there exists a non-trivial solution whose entries are all polynomials in $\mathbb{Z}[n]$. If $e_L(n) \neq 0$, then we only need to apply the partial DHB algorithm on $S = \{L\}$ to get the degree and height bounds of $e_L(n)$. However, there is no such guarantee, and also we do not know which is the maximal i that $e_i(n) \neq 0$. Therefore we have to estimate the degree upper bound d_a and height upper bound h_a of all the e_i's, which can be obtained by applying the partial DHB algorithm (taking $S = \{1, \dots, L+1\}$, note that the i-th $(1 \leq i \leq L+1)$ column corresponds to e_{i-1}) to matrix M.

Now suppose that $F(n,k)$ has finitely supports, i.e., for each n, there are only finitely many $k \in \mathbb{Z}$ such that $F(n,k) \neq 0$. Summing over k on both sides of Equation (3.3.3) leads to a recurrence satisfied by $S(n) = \sum_k F(n,k)$:

$$\sum_{i=0}^{L} e_i(n) S(n+i) = 0. \tag{3.3.10}$$

Let $e_{L'}(n)$ be the last non-zero polynomial reading from e_0 to e_L. Since $e_{L'}(n)$ is a polynomial with integer coefficients, we have $e_{L'}(n) \neq 0$ for $n > h_a$ by Lemma 2.1.1. Therefore, we may take $n_a = h_a$.

Similarly, n_f is given by an upper bound of the degree of the numerator polynomial of

$$R(n) = \sum_{i=0}^{L} e_i(n) \frac{f(n+i)}{f(n)}.$$

Let $D(n)$ be the common denominator of $\frac{f(n+i)}{f(n)}$ for $i = 0, \dots, L$. Then the degree of the numerator polynomial of $R(n)$ is bounded by the largest degree of $e_i(n)$ (bounded by d_a computed above) plus the largest degree of $\frac{f(n+i)}{f(n)} \cdot D(n)$ for $i = 0, \dots, L$.

In conclusion, we get the following algorithm for estimating n_1 for proper hypergeometric identities based on Zeilberger's algorithm.

Zeilberger-DHB Algorithm

Input: A proper hypergeometric term $F(n,k)$ over $\mathbb{Q}$ with finitely supports, a hypergeometric term $f(n)$, and n_0.

Output: An integer n_1 such that $\sum_k F(n,k) = f(n)$ holds for $n \geq n_0$ if and only if it holds for $n = n_0, \dots, n_1$.

1. Write $F(n,k)$ in the form of (3.3.1).

2. Compute L by (3.3.2).

3. Compute $\overline{H}(n,k), u(k), v(k)$ and $h(k)$ by (3.3.4)–(3.3.7).

4. Compute the GP representation of

$$\frac{v(k-1)}{u(k)} = \frac{a(k)}{b(k)} \frac{c(k+1)}{c(k)}.$$

   If $\frac{a(k)}{b(k)} = 1$, the algorithm fails. Otherwise continue the following procedures.

5. Equate to zero the coefficients of each power of k in (3.3.8) to get a system of linear equations $M\mathbf{x} = 0$ in the unknowns $e_i(n), 0 \leq i \leq L$ and $x_j(n), 0 \leq j \leq m$.

6. Apply the partial (taking $S = \{1, \dots, L+1\}$) DHB algorithm to M to get (d,h), set $d_a = d$ and $h_a = h$.

7. Compute the common denominator $D(n)$ of $\frac{f(n+i)}{f(n)}, i = 0, \ldots, L$ and then find the largest degree d_f of $\frac{f(n+i)}{f(n)} \cdot D(n)$. Set $n_f = d_a + d_f$.
8. Return $n_1 = \max(n'_a + L - 1, n_0 + n_f + L)$, where $n'_a = \max(h_a + 1, n_0)$.

3.4 Estimating n_1 for q-Hypergeometric Identities

Estimating n_1 for q-hypergeometric identities can be done just as the q-analogue of the ordinary case. In this thesis, we only discuss estimating n_1 based on Zeilberger's algorithm for the q case. For summation of q-hypergeometric terms, we have the following q-analogue of linear recurrence (3.3.10).

$$\sum_{i=0}^{L} a_i(q^n, q)S(n+i) = 0, \ n \geq n_0,$$

where the $a_i(q^n, q)$'s are polynomials in q^n and q over K. The following lemma shows that n_a can be set to be one plus the degree of $a_L(q^n, q)$ in q.

Lemma 3.4.1 ([74]) *Let $P(q^n, q)$ be a non-zero polynomial in q^n and q over K. Suppose that the degree of $P(q^n, q)$ in q is m. Then $P(q^n, q) \neq 0$ for all $n \geq m+1$.*

Follow the techniques in [46], write a q-proper hypergeometric term $F(n, k)$ in the following form

$$F(n,k) = POL(n,k)\frac{\prod_{j=1}^{A}[a''_j]_{a'_j n + a_j k}\prod_{j=1}^{B}[b''_j]_{b'_j n - b_j k}}{\prod_{j=1}^{C}[c''_j]_{c'_j n + c_j k}\prod_{j=1}^{D}[d''_j]_{d'_j n - d_j k}} q^{Jk(k-1)/2} z^k, \tag{3.4.1}$$

where $POL(n, k)$ is a Laurent polynomial in q^n and q^k over $K(q)$, $a_j, a'_j, b_j, b'_j, c_j, c'_j, d_j, d'_j$ are non-negative integers, $z, a''_j, b''_j, c''_j, d''_j \in K(q)$ and $J \in \mathbb{Z}$.

The q-Theorem in [46] states that $F(n, k)$ has a telescoped recurrence

$$\sum_{i=0}^{L} e_i(q^n, q)F(n+i, k) = G(n, k+1) - G(n, k) \tag{3.4.2}$$

of order

$$L = \max\left(J + \sum_{j=1}^{A} a_j^2, \ \sum_{j=1}^{C} c_j^2\right) + \max\left(-J + \sum_{j=1}^{D} d_j^2, \ \sum_{j=1}^{B} b_j^2\right). \tag{3.4.3}$$

Moreover, the coefficients $e_0(q^n, q), \ldots, e_L(q^n, q)$ together with some extra unknowns $x_{-m_1}(q^n, q), \ldots, x_{m_2}(q^n, q)$ satisfy a system of linear equations $M\mathbf{x} = 0$, where the entries of M are polynomials in q^n and q. The deduction is analogous to the ordinary case in Section 3.3, see [46] for details.

Different from the ordinary case, in the q-case, we need the degree bound in q of the e_i's for the estimation of n_a by Lemma 3.4.1, and their degree bound in q^n for the estimation of n_f, while the height does not make any sense in this case. So we need to introduce the counterpart of DH augment and DHB algorithm for the q-case.

Definition 3.4.2 *Let $A = [p_{ij}(x)]$ be a matrix whose entries are polynomials in $K[x]$. Degree augment is the transformation as follows:*

Choose a column, say the k-th column, of matrix A, replace those entries p_{ij} that satisfy $\deg(p_{ij}) < \deg(p_{ik})$ with x^d where $d = \deg(p_{ik})$, and finally delete the k-th column from A. Denote the resulting matrix by $\widehat{A}$.

q-DB Algorithm

Input: An $l \times m$ $(l < m)$ matrix $M = [p_{ij}(q^n, q)]$ whose entries are polynomials in q^n and q.

Output: Two integers d_q and d_{q^n}.

1. Let $M' = M$, regard the entries of M' as polynomials in q, then repeat degree augment on M' and set $M' = \widehat{M'}$ until M' becomes an $l \times l$ square matrix. Denote the resulting matrix by M_q.

2. Let $M' = M$, regard the entries of M' as polynomials in q^n, then repeat degree augment on M' and set $M' = \widehat{M'}$ until M' becomes an $l \times l$ square matrix. Denote the resulting matrix by M_{q^n}.

3. Do 0-1 augment on M_q and M_{q^n} respectively, and set $M_q = \overline{M_q}, M_{q^n} = \overline{M_{q^n}}$.

4. Return $d_q = \mathcal{D}(M_q),\ d_{q^n} = \mathcal{D}(M_{q^n})$.

The counterpart of Theorem 2.3.6 for the q-case, which guarantees that the output d_q and d_{q^n} are degree bounds in q and q^n, can be deduced in a thorough similar way, so does the *partial q-DB algorithm*, the counterpart of partial DHB algorithm. Similar to the ordinary case, q-DB Algorithm can also directly accept as input the q-degree matrix and q^n-degree matrix, whose entries are the degrees of q and q^n of each corresponding entry in M.

Now we are ready to give the algorithm for estimating n_1 for the q-case.

q-Zeilberger-DB Algorithm

Input: A q-proper hypergeometric term $F(n,k)$ with finitely supports, a q-hypergeometric term $f(n)$, and n_0.

Output: An integer n_1 such that $\sum_k F(n,k) = f(n)$ holds for $n \geq n_0$ if and only if it holds for $n = n_0, \ldots, n_1$.

1. Write $F(n,k)$ in the form of (3.4.1).
2. Compute L by (3.4.3).
3. Compute $\overline{H}(n,k), u(q^k), v(q^k)$ and $h(q^k)$ as in [46].
4. Compute the q-analogue of the GP representation [42] of
$$\frac{v(q^{k-1})}{u(q^k)} = \frac{a(q^k)}{b(q^k)}\frac{c(q^{k+1})}{c(q^k)}.$$
If $\frac{a(q^k)}{b(q^k)} = 1$, the algorithm fails. Otherwise continue the following procedures.
5. By equation (3.4.2), get a system of linear equations $M\mathbf{x} = 0$ in the unknowns $e_i(q^n,q), 0 \leq i \leq L$ and extra $x_j(q^n,q), -m_1 \leq j \leq m_2$.
6. Apply the partial q-DB algorithm on the e_i's to M to get the bounds d_q and d_{q^n}, and set $n_a = d_q, d_a = d_{q^n}$.
7. Compute the common denominator $D(q^n)$ of $\frac{f(n+i)}{f(n)}, i = 0, \ldots, L$ and then find the largest degree d_f of $D(q^n) \cdot \frac{f(n+i)}{f(n)}$. Set $n_f = d_a + d_f$.
8. Return $n_1 = \max(n'_a + L - 1, n_0 + n_f + L)$, where $n'_a = \max(n_a + 1, n_0)$.

3.5 Examples

We have implemented all the algorithms in this thesis in Maple program, see Appendix for the Maple source codes. Our implementation requires the `hsum` and `qsum` [41], `ZEILBERGER` and `qZEILBERGER` [47] as prerequisite packages.

3.5.1 Examples of the Ordinary Case

Example 3.5.1 Estimate n_1 *for*

$$\sum_k \binom{n}{k} = 2^n, \quad n \geq 0.$$

We try both Sister Celine's method and Zeilberger's algorithm on this example.

By Sister Celine's method, there exists a non-trivial k-free recurrence relation in the form of (3.2.2) for the summand $F(n,k) = \binom{n}{k}$ when $I = J = 1$, and we can get the following linear symbolic system for the unknown coefficients $\alpha_{i,j}$'s.

$$\begin{cases} (n^2+n)\alpha_{0,0} + (n^2+n)\alpha_{0,1} = 0 \\ n\alpha_{0,0} + (2n+1)\alpha_{0,1} + (-n)\alpha_{1,0} + (-n-1)\alpha_{1,1} = 0 \\ -\alpha_{0,1} + \alpha_{1,1} = 0 \end{cases}$$

Simplifying each equation and writing the system in the form of $M\mathbf{x} = 0$, we have

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ n & 2n+1 & -n & -(n+1) \\ 0 & -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha_{0,0} \\ \alpha_{0,1} \\ \alpha_{1,0} \\ \alpha_{1,1} \end{bmatrix} = 0.$$

By criterion of the least height sum, we choose the third column as the augmenting column, then the DH augment returns

$$\widehat{M} = \begin{bmatrix} 1 & 1 & 0 \\ n & 2n+1 & -(n+1) \\ 0 & -1 & 1 \end{bmatrix}.$$

Sequentially, 0-1 augment transforms $\widehat{M}$ into

$$M = \overline{\widehat{M}} = \begin{bmatrix} 1 & 1 & 1 \\ n & 2n+1 & -(n+1) \\ 1 & -1 & 1 \end{bmatrix}.$$

Then it follows that

$$\mathcal{D}(M) = 1 \quad \text{and} \quad \mathcal{H}(M) = 8,$$

thus $D_\alpha = 1$ and $H_\alpha = 8$. By (3.2.7) and (3.2.8), we set $d_a = 1, n_a = 8 \times 2 = 16$.

To get d_f, noting that $\frac{2^{n+j}}{2^n} = 2^j$, we have $d_f = 0$, and thus $n_f = 1$. Finally, $n_1 = \max(16,1) = 16$.

In the following, we present the process using Zeilberger's algorithm.

First write the summand $F(n,k) = \binom{n}{k}$ in the form of (3.3.1),

$$F(n,k) = \frac{(1)_n}{(1)_{n-k}(1)_k}.$$

A straightforward computation produces that $L = 1$, $u(k) = n - k + 1$ and $v(k) = k+1$. Since the GP representation of $\frac{v(k-1)}{u(k)}$ is $\frac{k}{(n-k+1)} \neq 1$, we are ensured that $e_0(n), e_1(n)$ are not both zeros. Now Equation (3.3.8) becomes

$$(e_0(n) - 2x_0(n))k + (n+1)(x_0(n) - e_0(n) - e_1(n)) = 0.$$

Equating the coefficients of each power of k to 0 yields

$$\begin{bmatrix} 1 & 0 & -2 \\ -1 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} e_0(n) \\ e_1(n) \\ x_0(n) \end{bmatrix} = 0.$$

By the partial DHB algorithm in which we take $S = \{1,2\}$ and choose the second column (with the least height sum) as the augmenting column in DH augment, we get $d_a = 0$ and $h_a = 3$, together with the fact that $d_f = 0$, we finally have $n_1 = \max(4,1) = 4$.

Note that the bound given by Yen [72] for this example was 10^{11}.

Example 3.5.2 Estimate n_1 for

$$\sum_k \binom{n}{k}^2 = \binom{2n}{n}.$$

For this example, Sister Celine's method and Zeilberger's algorithm will produce results with great differences.

By Sister Celine's method, there exists a non-trivial k-free recurrence for the summand $F(n,k) = \binom{n}{k}^2$ when $I = 2, J = 3$, and the system for the unknown coefficients $\alpha_{i,j}$'s is of size 11 (equations) by 12 (unknowns). Sister Celine-DHB algorithm will return $d_a = 40$, $n_a = 34804330863895756800$ and $d_f = 2$, and thus $n_1 = \max(n_a, d_a + d_f) = n_a \approx 3.5 \times 10^{19}$.

On the other hand, by Zeilberger's algorithm, we have $L = 2$, $m = 2$ and arrive at the following 5-by-6 linear system for e_i's and x_i's.

$$\begin{bmatrix} n^2+2n+1 & n^2+2n+1 & n^2+2n+1 & -1 & -1 & -1 \\ 4n^2+10n+6 & 2n^2+4n+2 & 0 & -2 & n & 2n+2 \\ 6n^2+18n+13 & n^2+2n+1 & 0 & 0 & 2n+3 & -n^2+3 \\ 2n+3 & 0 & 0 & 0 & 0 & -n-1 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} e_0(n) \\ e_1(n) \\ e_2(n) \\ x_0(n) \\ x_1(n) \\ x_2(n) \end{bmatrix} = \mathbf{0}. \tag{3.5.1}$$

Extracting the degrees and heights, we obtain the degree matrix M_D and height matrix M_H,

$$M_D = \begin{bmatrix} 2 & 2 & 2 & 0 & 0 & 0 \\ 2 & 2 & -\infty & 0 & 1 & 1 \\ 2 & 2 & -\infty & -\infty & 1 & 2 \\ 1 & -\infty & -\infty & -\infty & -\infty & 1 \\ 0 & -\infty & -\infty & -\infty & -\infty & -\infty \end{bmatrix}, \quad M_H = \begin{bmatrix} 2 & 2 & 2 & 1 & 1 & 1 \\ 10 & 4 & 0 & 2 & 1 & 2 \\ 18 & 2 & 0 & 0 & 3 & 3 \\ 3 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Take M_D and M_H as input to the partial DHB algorithm in which we take $S = \{1, 2, 3\}$ and choose the third column (with least height sum among the first three columns) as the augmenting column, the DH augment and 0-1 augment transform M_D and M_H into

$$M_D = \begin{bmatrix} 2 & 2 & 0 & 0 & 0 \\ 2 & 2 & 0 & 1 & 1 \\ 2 & 2 & 0 & 1 & 2 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad M_H = \begin{bmatrix} 2 & 2 & 1 & 1 & 1 \\ 10 & 4 & 2 & 1 & 2 \\ 18 & 2 & 1 & 3 & 3 \\ 3 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Further computation returns $d_a = 6$ and $h_a = 12089$, thus $n'_a = h_a + 1 = 12090$. Moreover, we have $d_f = 2$. Therefore $n_1 = \max(n'_a + L - 1, n_0 + d_a + d_f + L) = 12091$.

Note that the bound given by Yen [72] for this example was 10^{115}.

In the following examples, we show only the results based on Zeilberger's algorithm.

Example 3.5.3 The WZ dual identity of $\sum_k \binom{n}{k} = 2^n$ *[54, p. 133].*

$$\sum_k (-1)^{n+k} \binom{n}{k} 2^k = 1, \quad n \geq 0.$$

The Zeilberger-DHB algorithm produces a 2-by-3 system and $L = 1$, $d_a = 0$, $h_a = 3$, $d_f = 0$, and finally returns $n_1 = 4$, exactly the same to its original case in Example 3.5.1.

Example 3.5.4 Estimate n_1 for

$$\sum_k \binom{3n}{k} = 8^n.$$

This identity is obviously a variation of Example 3.5.1, by simply substituting n with $3n$. However, the result returned by the Zeilberger-DHB algorithm is different.

For this example, the algorithm yields an system of 4-by-5, and $L = 1$, $d_a = 4$, $h_a = 2173$, $d_f = 0$, thus $n_1 = 2173$, much larger than that of example 3.5.1.

Example 3.5.5 WZ dual of $\sum_k \binom{n}{k}^2 = \binom{2n}{n}$ [54, p. 132].

$$\sum_k (3k-2n)\binom{n}{k}^2\binom{2k}{k} = 0.$$

The algorithm produces a 10-by-12 system and $L = 4$, $d_a = 31$, $h_a = 2065670500895266682540596470$, $d_f = 0$, and finally returns $n_1 = h_a + 4 \approx 2 \times 10^{27}$, much larger than its original case in Example 3.5.2.

Example 3.5.6 [54, p. 113]

$$\sum_k \binom{n}{2k}\binom{2k}{k}4^{-k} = 2^{-(n-1)}\binom{2n-1}{n-1}, \quad n \geq 1$$

For this identity, the algorithm produces a 3-by-4 system, $L = 2$, $d_a = 1$, $h_a = 34$, and $n_1 = 36$.

Example 3.5.7 [54, p. 138]

$$\sum_k \frac{2^{k+1}(k+1)(2n-k-2)!}{(n-k-1)!} = \frac{(2n)!}{n!}.$$

For this identity, the algorithm produces a 4-by-5 system, $L = 1$, $d_a = 4$, $h_a = 569$, and $n_1 = 570$.

Example 3.5.8 [54, p. 111]

$$\sum_k (-1)^k \binom{n+1}{k} \binom{2n-2k+1}{n} = 1.$$

For this identity, we have a system of size 7-by-8, $L = 3$, $d_a = 26$, $h_a = 529198068667721384894668 8$, and $n_1 = h_a + 3 \approx 5.3 \times 10^{24}$.

Example 3.5.9 Reed-Dawson identities [58, p. 4]

$$\sum_{k=0}^{2n} (-1)^k \binom{2n}{k} 2^{-k} \binom{2k}{k} = 2^{-2n} \binom{2n}{n}, \tag{3.5.2}$$

and

$$\sum_{k=0}^{2n+1} (-1)^k \binom{2n+1}{k} 2^{-k} \binom{2k}{k} = 0. \tag{3.5.3}$$

The Reed-Dawson identities are also usually written in the following more compact form [54, p. 63].

$$\sum_k \binom{n}{k} \binom{2k}{k} (-2)^{n-k} = \begin{cases} 0, & \text{if } n \text{ is odd;} \\ \binom{n}{n/2}, & \text{if } n \text{ is even.} \end{cases} \tag{3.5.4}$$

For both summands in (3.5.2) and (3.5.3), the algorithm produces a system of size 7-by-9 and $L = 3$, $d_a = 14$. For (3.5.2), $h_a = 606735023907614 \approx 6 \times 10^{14}$, and for (3.5.3), $h_a = 5963877204989859 \approx 6 \times 10^{15}$. Therefore, we have $n_1 \approx 6 \times 10^{14}$ for (3.5.2) and $n_1 \approx 6 \times 10^{15}$ for (3.5.3) and (3.5.4).

Example 3.5.10 The identity that Knuth proposed in the foreword of [54].

$$\sum_k \binom{2n-2k}{n-k} \binom{2k}{k} = 4^n$$

The algorithm yields a system of size 9-by-11 and $L = 4$, $d_a = 26$, $h_a = 19915928653228019055415828480$, $d_f = 0$, and finally returns $n_1 = h_a + 4 \approx 2 \times 10^{28}$.

Identity	System Size	Computing $\mathcal{H}(M)$	Overall
$\sum_k \binom{n}{k} = 2^n$	2×3	< 0.001	< 0.001
$\sum_k \binom{n}{k}^2 = \binom{2n}{n}$	5×6	0.031	0.047
$\sum_k (-1)^{n+k}\binom{n}{k}2^k = 1$	2×3	< 0.001	< 0.001
$\sum_k \binom{3n}{k} = 8^n$	4×5	0.032	0.032
$\sum_k (3k-2n)\binom{n}{k}^2\binom{2k}{k} = 0$	10×12	768.079	768.219
$\sum_k \binom{n}{2k}\binom{2k}{k}4^{-k} = 2^{-(n-1)}\binom{2n-1}{n-1}$	3×4	< 0.001	< 0.001
$\sum_k \frac{2^{k+1}(k+1)(2n-k-2)!}{(n-k-1)!} = \frac{(2n)!}{n!}$	4×5	< 0.001	< 0.001
$\sum_k (-1)^k\binom{n+1}{k}\binom{2n-2k+1}{n} = 1$	7×8	0.828	0.984
$\sum_k (-1)^k\binom{2n}{k}2^{-k}\binom{2k}{k} = 2^{-2n}\binom{2n}{n}$	7×9	0.797	0.875
$\sum_k \binom{2n-2k}{n-k}\binom{2k}{k} = 4^n$	9×11	71.032	71.407
$\sum_k (-1)^k\binom{2n}{k}^3 = (-1)^n\frac{(3n)!}{n!^3}$	19×20	–	–

Table 3.1: Maple CPU Seconds to Estimate n_1 for hypergeometric identities.

Table 3.1 lists the times that are needed to estimate n_1 for the above identities. The experiments are carried out on the `Maple` platform with the same equipment that has been mentioned in the beginning of Section 2.2, i.e., a `Windows` installed PC machine with 2 GHz CPU (AMD Athlon 64 Processor 3200+) and 1 GB memory.

Column *System Size* indicates the size of the symbolic system of equations obtained by Zeilberger's algorithm [46] on each identity. Column *Computing $\mathcal{H}(M)$* indicates the times that are needed to compute $\mathcal{H}(M)$ by formula (2.1.5). Column *Overall* indicates the overall times that are needed for estimating n_1. "–" means that the computation did not terminate after 24 hours. Clearly, the computation of $\mathcal{H}(M)$ is the bottleneck of our approach, it occupies most of the computation time.

3.5.2 Examples of the q Case

Example 3.5.11 Estimate n_1 for a finite version of Jacobi's triple product identity [15]

$$\sum_k \begin{bmatrix} 2n \\ n+k \end{bmatrix} q^{\binom{k}{2}} z^k = (-z^{-1}q;q)_n(-z;q)_n, \quad n \geq 0.$$

The summand can be expressed in the form of

$$F(n,k) = \frac{[1]_{2n}}{[1]_{n+k}[1]_{n-k}} q^{k(k-1)/2} z^k.$$

By the q-Zeilberger-DB algorithm, we get $L = 1$ and the following homogeneous linear system.

$$\begin{bmatrix} qq^n & 0 & zq^n+1 & 0 \\ -q\left(1+q^2q^{2n}\right) & -q(q^3q^{4n} - q^2q^{2n} - qq^{2n}+1) & -z-q^2q^n & q\left(zqq^n+1\right) \\ qq^n & 0 & 0 & -z-qq^n \end{bmatrix} \cdot \begin{bmatrix} e_0(q^n,q) \\ e_1(q^n,q) \\ x_{-1}(q^n,q) \\ x_0(q^n,q) \end{bmatrix} = 0.$$

Extracting the q-degree matrix and q^n-degree matrix, we have

$$M_q = \begin{bmatrix} 1 & -\infty & 0 & -\infty \\ 3 & 4 & 2 & 2 \\ 1 & -\infty & -\infty & 1 \end{bmatrix}, \qquad M_{q^n} = \begin{bmatrix} 1 & -\infty & 1 & -\infty \\ 2 & 4 & 1 & 1 \\ 1 & -\infty & -\infty & 1 \end{bmatrix}.$$

Take M_q and M_{q^n} as the input to the partial q-DB algorithm in which we take $S = \{1,2\}$ and choose the second column for the degree augment, then the degree augment and 0-1 augment will transform M_q and M_{q^n} into

$$M_q = \begin{bmatrix} 1 & 0 & 0 \\ 4 & 2 & 2 \\ 1 & 0 & 1 \end{bmatrix}, \qquad M_{q^n} = \begin{bmatrix} 1 & 1 & 0 \\ 4 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}.$$

A straightforward calculation gives $d_q = 5$ and $d_{q^n} = 6$. Thus $n'_a = d_q + 1 = 6$. Furthermore, $\frac{f(n+1)}{f(n)} = (1 + z^{-1}q^{n+1})(1 + zq^n)$ implies $d_f = 2$. Finally, we derive that $n_1 = d_{q^n} + d_f + L = 9$.

Note that in this example, $d_q = 5 < 6 = d_{q^n} = 6$.

The bound given by Zhang [80] for this identity was 70.

Example 3.5.12 Compute n_1 for the q-Chu-Vandermonde identity (cf. [29]).

$$\sum_k q^{k^2} \begin{bmatrix} n \\ k \end{bmatrix}^2 = \begin{bmatrix} 2n \\ n \end{bmatrix}, \quad n \geq 0.$$

For this identity, the algorithm yields a system of size 5-by-6, and $L = 2$, $d_q = 23$, $d_{q^n} = 13$, $d_f = 6$, and finally returns $n_1 = 25$.

The bounds given by Yen [74] and Zhang [80] for this identity were 2358 and 191, respectively.

Example 3.5.13 Estimate n_1 for a finite form of Euler's pentagonal number theorem due to L. J. Rogers (cf. [15]).

$$\sum_k \frac{(-1)^k (q;q)_n q^{k(3k-1)/2}}{(q;q)_{n+k}(q;q)_{n-k}} = 1, \quad n \geq 0.$$

The algorithm produces a 7-by-8 system, and $L = 3$, $d_q = 38$, $d_{q^n} = 15$, and finally returns $n_1 = 41$.

The bound given by Zhang [80] for this example was 209.

Table 3.2 shows the times needed to estimate n_1 for the above identities.

Identity	System Size	Time
$\sum_k \begin{bmatrix} 2n \\ n+k \end{bmatrix} q^{\binom{k}{2}} z^k = (-z^{-1}q;q)_n(-z;q)_n$	3×4	0.047
$\sum_k q^{k^2} \begin{bmatrix} n \\ k \end{bmatrix}^2 = \begin{bmatrix} 2n \\ n \end{bmatrix}$	5×6	0.078
$\sum_k \frac{(-1)^k (q;q)_n q^{k(3k-1)/2}}{(q;q)_{n+k}(q;q)_{n-k}} = 1$	7×8	0.219

Table 3.2: Maple CPU Seconds to Estimate n_1 for q-hypergeometric identities.

Remark 3.5.14 Note that our approach runs quickly for the q case, because the bottleneck of the ordinary case, the computation of $\mathcal{H}(M)$ no longer exists. This can be viewed as a consequence of the non-vanishing property of the highest coefficient stated in Lemma 3.4.1.

Bibliography

[1] S. A. Abramov, Problems in computer algebra that are connected with a search for polynomial solutions of linear differential and difference equations, *Moscow Univ. Comput. Math. Cybernet.* **3** (1989) 63–68.

[2] S. A. Abramov, Denominators of rational solutions of linear difference equations, *Programmirovanie* **1** (1994) 49–52.

[3] S. A. Abramov, Rational solutions of linear difference and q-difference equations with polynomial coefficients, in: *Proc. ISSAC'95*, ACM Press, 1995, pp. 285–289.

[4] S. A. Abramov, Rational solutions of linear difference and q-difference equations with polynomial coefficients, *Program. Comput. Software* **21**(6) (1995) 273–278, Transl. from Programmirovanie **6** (1995) 3–11.

[5] S. A. Abramov, Applicability of Zeilberger's algorithm to hypergeometric terms, in: *Proc. ISSAC'02*, ACM Press, 2002, pp. 1–7.

[6] S. A. Abramov, When does Zeilberger's algorithm succeed?, *Adv. in Appl. Math.* **30** (2003) 424–441.

[7] S. A. Abramov and M. A. Barkatou, Rational solutions of first order linear difference systems, in: *Proc. ISSAC'98*, ACM Press, 1998, pp. 124–131.

[8] S. A. Abramov, M. Bronstein and M. Petkovšek, On polynomial solutions of linear operator equations, *Proc. ISSAC'95*, ACM Press, 1995, pp. 290–296.

[9] S. A. Abramov and H. Q. Le, A criterion for the applicability of Zeilberger's algorithm to rational functions, *Discrete Math.* **259** (2002) 1–17.

[10] S. A. Abramov, P. Paule and M. Petkovšek, q-Hypergeometric solutions of q-difference equations, *Discrete Math.* **180** (1998) 3–22.

[11] S. A. Abramov and M. Petkovšek, D'Alembertian solutions of linear differential and difference equations, in: *Proc. ISSAC'94*, ACM Press, 1994, pp. 169–174.

[12] S. A. Abramov and M. Petkovšek, Proof of a conjecture of Wilf and Zeilberger, Preprint Series of the Institute of Mathematics, Physics and Mechanics **39** (748) Ljubljana, 2001.

[13] S. A. Abramov and M. Petkovšek, On the structure of multivariate hypergeometric terms, *Adv. in Appl. Math.* **29** (2002) 386–411.

[14] S. A. Abramov and M. Petkovšek, Rational normal forms and minimal decompositions of hypergeometric terms, *J. Symbolic Comput.* **33** (2002) 521–543.

[15] G. E. Andrews, The Theory of Partitions, Encyclopedia of Mathematics and its Application, Vol. 2. Addison-Wesley, Reading, 1976. (Reissued: Cambridge University Press, London and New York, 1985.)

[16] G. E. Andrews, q-series: Their development and applications in analysis, number theory, combinatorics, physics, and computer algebra, CBMS series, vol. 66, Amer. Math. Soc., Providence, 1986.

[17] M. A. Barkatou, Rational solutions of matrix difference equations: the problem of equivalence and factorization, in: *Proc. ISSAC'99*, ACM Press, 1999, pp. 277–282.

[18] A. Bauer and M. Petkovšek, Multibasic and mixed hypergeometric Gosper-type algorithms, *J. Symbolic Comput.* **28** (1999) 711–736.

[19] H. Böing and W. Koepf, Algorithms for q-hypergeometric summation in computer algebra, *J. Symbolic Comput.* **28** (1999) 777–799.

[20] W. Y. C. Chen, Q.-H. Hou and Y. P. Mu, Applicability of the q-analogue of Zeilberger's algorithm, *J. Symbolic Comput.* **39** (2005) 155–170.

[21] W. Y. C. Chen, P. Paule and H. L. Saad, Converging to Gosper's algorithm *Adv. Appl. Math.* **41** (2008) 351-364.

[22] W. Y. C. Chen and H. L. Saad, On the Gosper-Petkovšek representation of rational functions, *J. Symbolic Comput.* **40** (2005) 955–963.

[23] G. B. Dantzig, Linear Programming and Extensions, Princeton University Press, Princeton, 1963.

[24] B. M. Dingle, Symbolic determinants: calculating the degree, Technical Report, Texas A&M University, 2005.

[25] A. C. Dixon, On the sum of the cubes of the coefficients in a certain expansion by the binomial theorem, *Messenger of Mathematics* **20** (1891) 79–80.

[26] M. C. Fasenmyer, Some generalized hypergeometric polynomials, Ph.D. dissertation, University of Michigan, 1945.

[27] M. C. Fasenmyer, Some generalized hypergeometric polynomials, *Bull. Amer. Math. Soc.* **53** (1947) 806–812.

[28] M. C. Fasenmyer, A note on pure recurrence relations, *Amer. Math. Monthly* **56** (1949) 14–17.

[29] G. Gasper and M. Rahman, Basic Hypergeometric Series, 2nd ed., Cambridge University Press, Cambridge, MA, 2004.

[30] R. W. Jr. Gosper, Indefinite hypergeometric sums in MaCSYMA, in: *Proc. 1977 MACSYMA Users' Conference*, Berkeley, 1977, pp. 237–251.

[31] R. W. Jr. Gosper, Decision procedure for indefinite hypergeometric summation, *Proc. Natl. Acad. Sci. USA* **75** (1978) 40–42.

[32] R. L. Graham, D. E. Knuth and O. Patashnik, Concrete Mathematics: A Foundation for Computer Science, 2nd ed., Addison-Wesley, Reading, MA, 1994.

[33] Q.-H. Guo, Q.-H. Hou and L. H. Sun, Proving hypergeometric identities by numerical verifications, *J. Symbolic Comput.* **43** (2008) 895–907.

[34] D. Henrion and M. Šebek, Improved polynomial matrix determinant computation, *IEEE Trans. Circuits Systems I Fund. Theory Appl.* **46** (1999) 1307–1308.

[35] Q.-H. Hou, Algebraic techniques in combinatorics, Ph.D. dissertation, Nankai University, 2001.

[36] Q.-H. Hou, k-free recurrences of double hypergeometric terms, *Adv. in Appl. Math.* **32** (2004) 468–484.

[37] M. HromEik and M. Šebek, New algorithms for polynomial matrix determinant based on FFT, *Procs. ECC'99*, Karlsruhe, 1999.

[38] N. Karmarkar, A new polynomial-time algorithm for linear programming, *Combinatorica* **4** (1984) 373–395.

[39] D. E. Knuth, The Art of Computer Programming, Vol. 2: Fundamental Algorithms, 3rd ed., Addison-Wesley, Reading, MA, 1998.

[40] W. Koepf, Hypergeometric Summation, Vieweg, Braunschweig/Wiesbaden, 1998.

[41] W. Koepf, `hsum` and `qsum`, Maple package on (q-) hypergeometric summation, from Homepage of Wolfram Koepf, available at: `http://www.mathematik.uni-kassel.de/~koepf/Publikationen/#down`.

[42] T. H. Koornwinder, On Zeilberger's algorithm and its q-analogue, *J. Comput. Appl. Math.* **48** (1993) 91–111.

[43] H. W. Kuhn, The Hungarian method for the assignment problem, *Naval Res. Logist. Q.* **2** (1955) 83–97.

[44] H. Q. Le, On the q-analogue of Zeilberger's algorithm to rational functions, *Program. Comput. Software* **27** (2001) 35–42.

[45] D. Manocha and J. F. Canny, Multipolynomial Resultants and Linear Algebra, in: *Proc. ISSAC'92*, ACM Press, 1992, 158–167.

[46] M. Mohammed and D. Zeilberger, Sharp upper bounds for the orders of the recurrences outputted by the Zeilberger and q-Zeilberger algorithms, *J. Symbolic Comput.* **39** (2005) 201–207.

[47] M. Mohammed and D. Zeilberger, `ZEILBERGER` and `qZEILBERGER`, from Homepage of Doron Zeilberger, available at: `http://www.math.rutgers.edu/~zeilberg/programs.html`.

[48] C. H. Papadimitriou and K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall, New Jersey, 1982.

[49] P. Paule, Greatest factorial factorization and symbolic summation, *J. Symbolic Comput.* **20** (1995) 235–268.

[50] P. Paule and A. Riese, A Mathematica q-analogue of Zeilberger's algorithm based on an algebraically motivated approach to q-hypergeometric telescoping, in: Special Functions, q-Series and Related Topics, *Fields Inst. Commun.* 14 (1997) 179–210.

[51] M. Petkovšek, Finding closed-form solutions of difference equations by symbolic methods, Ph.D. dissertation, Carnegie-Mellon University, CMU-CS-91-103, 1991.

[52] M. Petkovšek, Hypergeometric solutions of linear recurrences with polynomial coefficients, *J. Symbolic Comput.* **14** (1992) 243–264.

[53] M. Petkovšek, A generalization of Gosper's algorithm, *Discrete Math.* **134** (1994) 125–131.

[54] M. Petkovšek, H. S. Wilf and D. Zeilberger, "A=B", Wellesley, Massachusetts, 1996.

[55] R. Pirastu and V. Strehl, Rational summation and Gosper-Petkovšek representation, *J. Symbolic Comput.* **20** (1995) 617–635.

[56] E. D. Rainville, Special Functions, MacMillan, New York, 1960.

[57] G. A. Rempala and J. Wesolowski, Symmetric Functionals on Random Matrices and Random Matchings Problems, Springer Verlag, 2008.

[58] J. Riordan, An Introduction to Combinatorial Analysis, Wiley, New York, 1958.

[59] H. J. Ryser, Combinatorial Mathematics, The Carus mathematical monographs series, The Mathematical Association of America, 1963.

[60] M. Šebek, F. Kraus, S. Pejchová, H. Kwakernaak and D. Henrion, Numerical methods for zeros and determinant of polynomial matrix, *Procs. IEEE Mediterranean Symposium on New Directions in Control and Automation*, Crete, Greece, 1996, pp. 488–491.

[61] R. P. Stanley, Differentiably finite power series, *European J. Combin.* **1** (1980) 175–188.

[62] L. G. Valiant, The complexity of computing the permanent, *Theoret. Comput. Sci.* **8** (1979) 189–201.

[63] P. M. van Dooren, P. Dewilde and J. Vandewalle, On the determination of the Smith-MacMillan form of a rational matrix from its Laurent expansion, *IEEE Trans. Circuits Systems I Fund. Theory Appl.* **26** (1979) 180–189.

[64] M. van Hoeij, Rational solutions of linear difference equations, *Proc. ISSAC'98*, ACM Press, 1998, pp. 120–123.

[65] S. Vandebril, M. Van Barel, O. Ruatta and B. Mourrain, A new algorithm for solving multivariate polynomial problems by means of interpolation, Report TW 343, Department of Computer Science, K. U. Leuven, 2002.

[66] P. Verbaeten, The automatic construction of pure recurrence relations, *Proc. EUROSAM'74, ACM-SIGSAM Bulletin* **8** (1974) 96–98.

[67] C. Weixlbaumer, Solutions of difference equations with polynomial coefficients, Diploma thesis, RISC, Johannes Kepler University, 2001.

[68] H. S. Wilf and D. Zeilberger, Rational functions certify combinatorial identities, *J. Amer. Math. Soc.* **3** (1990) 147–158.

[69] H. S. Wilf and D. Zeilberger, An algorithmic proof theory for hypergeometric (ordinary and "q") multisum/integral identities, *Invent. Math.* **108** (1992) 575–633.

[70] W. Wu, A survey of developments of mathematics mechanization in China, in: W. Wu, M.-D. Cheng (Eds.), Chinese Mathematics into the 21st Century, Peking University Press, Beijing, 1991, pp. 15–40.

[71] W. Wu, Polynomial equations – Solving and its applications, in: Algorithms and Computation, Proceedings of ISAAC'94, Beijing, China, 1994, *Lecture Notes in Comput. Sci.* **834** (1994) 1–9.

[72] L. Yen, Contributions to the proof theory of hypergeometric identities, Ph.D. dissertation, University of Pennsylvania, 1993.

[73] L. Yen, A two-line algorithm for proving terminating hypergeometric identities, *J. Math. Anal. Appl.* **198** (1996) 856–878.

[74] L. Yen, A two-line algorithm for proving q-hypergeometric identities, *J. Math. Anal. Appl.* **213** (1996) 1–14.

[75] D. Zeilberger, All binomial identities are verifiable, *Proc. Natl. Acad. Sci. USA.* **78** (1981) 4000.

[76] D. Zeilberger, Sister Celine's technique and its generalizations, *J. Math. Anal. Appl.* **85** (1982) 114–145.

[77] D. Zeilberger, A fast algorithm for proving terminating hypergeometric identities, *Discrete Math.* **80** (1990) 207–211.

[78] D. Zeilberger, The method of creative telescoping, *J. Symbolic Comput.* **11** (1991) 195–204.

[79] D. Zeilberger, Theorems for a price: Tomorrow's semi-rigorous mathematical culture, *Notices Amer. Math. Soc.* **40** (1993) 978–981.

[80] B.-Y. Zhang, A new elementary algorithm for proving q-hypergeometric identities, *J. Symbolic Comput.* **35** (2003) 293–303.

[81] B.-Y. Zhang, Contributions to computer proofs of q-hypergeometric identities, Ph.D. dissertation, Nankai University, 2003.

[82] B.-Y. Zhang and J. Li, An improvement of the two-line algorithm for proving q-hypergeometric identities, *Discrete Math.* **268** (2003) 273–286.

致 谢

在此我要向导师陈永川教授致以诚挚的感谢！正是陈老师将我领进组合恒等式机器证明这一令人神往的领域。陈老师渊博的学识、严谨的学风、对学科发展的敏锐洞察力、忘我的工作精神和一丝不苟的工作态度为我树立了学习的榜样，也是我不断进步的源泉。在六年的硕士和博士学习期间，我所取得的每一点成绩都离不开陈老师的悉心指导和点拨。感谢陈老师为我们创造了良好的学习和工作环境，使我们可以安心尽享追求知识的快乐。还要感谢陈老师给了我很多参与各种工作的机会，使我的综合素质和能力得到了极大的提高。陈老师正直无私的品格和豁达的胸襟也为我树立了做人的楷模。我终生难忘陈老师的培养教育之恩。

感谢李学良教授和马万宝老师对我工作上的帮助和指导。感谢侯庆虎教授，侯老师是我学业上的良师益友，他深厚的学术造诣使我受益匪浅。感谢杨立波副教授、付梅副教授和姜洪华老师对我学习、工作和生活上的关心和帮助。感谢与我同室办公四年的徐昭和于名飞，与他们共同打拼的经历和在打拼中建立的友谊是我人生一笔宝贵的财富。衷心感谢组合数学中心的每位老师！

感谢我在机器证明小组的亲密战友，也是我曾经的室友：吕仑、周岳、涂建华，感谢好友李岩，感谢他们伴我一起走过四年时光，我会永远铭记和他们在一起的欢笑与汗水，铭记与他们情同兄弟的深厚友谊。感谢孙慧师妹，她为本文的写作做出了巨大的贡献，正是由于她无私热情的帮助，本文才得以顺利完成。

我还要深深地感谢我最亲爱的父母，感谢他们含辛茹苦将我养育成人，他们为我的成长，为我能顺利完成学业付出了巨大的心血。感谢我的姐姐给予我的支持和帮助。希望我取得的成绩能为我的家人带来欢欣与骄傲。

最后我要感谢我的女友赵嫚，她的爱使我的求学之路和整个人生不再寂寞，特以此文献给我们的五年爱情长跑。

Appendix

The Maple source codes for the Zeilberger-DHB algorithm

```
print(`Written by Qianghui Guo and Lisa Hui Sun`);
print(`Version of March 2009`);
print(`Prerequisite packages: ZEILBERGER, hsum9.mpl.`);

with(combinat, permute):
with(SolveTools):
read `ZEILBERGER`:
read `hsum9.mpl`:

#################################################################
# The main procedure for estimating degree and height bound
# for a given hypergeometric term.
#################################################################
esn1 := proc(F)

local res, matD, matH, L, st1, st2;

res := getDHMatrix(F);
matD := res[1];
matH := res[2];
L := res[3];

res := DHAugment(matD, matH, [seq(i,i=1..L)]);
matD := res[1];
matH := res[2];
matH := zeroneAugment(matH);

print(`degree and height: `, LeibnizDHperm(matD, matH));
end proc:
```

```
############################################
# Compute the degree and height matrix.
############################################

getDHMatrix := proc(F)
local i, ii, j, a, b, e, f, g , h, l, s, a1, b1, x,
      A, B, C, D, L, M, X, F1, F2, H, H1, H2,
      Num, Den, Den1, POL, covars, coeqns, rec,
      tempcoeqn, tempcoa, tempco1, tempco2, coegcd, elems,
      PMmapping, DMatrix, HMatrix, PMatrix, TMatrix, ty;

Num := 1;
Den := 1;
A := 0;
B := 0;
C := 0;
D := 0;
F1 := convert(F, factorial);
F2 := hafokh(F1, k, n); # hafokh is from the ZEILBERGER package.
l := nops(F2[1]);
s := nops(F2[2]);
POL := F2[3];
a := array(1..l);
a1 := array(1..l);
b := array(1..s);
b1 := array(1..s);

for i from 1 to l do
  a[i] := F2[1,i];
  a1[i] := coeff(a[i], k, 1);
  A := A + max(a1[i],0);
  B := B + max(-a1[i],0);
  Num := Num * a[i]!;
end do;

for j from 1 to s do
  b[j] := F2[2,j];
  b1[j] := coeff(b[j], k, 1);
  C := C + max(b1[j],0);
  D := D + max(-b1[j],0);
  Den := Den * b[j]!;
end do;

L := max(A+D, B+C);
e := array(0..L);
Den1 := subs(n=n+L, Den);
H := Num / Den;
H1 := Num / Den1;
H2 := simpcomb(subs(k=k+1,H1) / H1);
f := numer(H2);
g := denom(H2);
h := simpcomb(sum(e[ii]*subs(n=n+ii,POL)*subs(n=n+ii,H)/H1,ii=0..L));
M := degree(h,k) - max(degree(f,k), degree(g,k));

x := array(0..M);
X := sum(x[ii]*k^ii, ii=0..M);
rec := simplify(f*subs(k=k+1, X) - subs(k=k-1, g)*X - h);
coeqns := [coeffs(rec, k)];

DMatrix := Matrix(nops(coeqns), L+M+2, 0);
HMatrix := Matrix(nops(coeqns), L+M+2);

covars := NULL:
for i from 0 to L do
  covars := covars, e[i];
end do:
for i from 0 to M do
  covars := covars, x[i];
end do;

for i from 0 to degree(rec, k) do
  tempcol := coeffs(simplify(coeff(rec, k, i)), {covars});
  if (nops([tempcol]) = 1) then
    coegcd := tempcol;
  else
    coegcd := tempcol[1];
    for s from 2 to nops([tempcol]) do
```

```
      coegcd := gcd(coegcd, tempcol[s]);
    end do;
  end if;
  tempcoeqn := collect(simplify(coeff(rec,k,i)/coegcd),{covars});

  for j from 1 to L+M+2 do
    tempcoa := coeff(tempcoeqn, covars[j]);
    if (tempcoa <> 0) then
      DMatrix[i+1, j] := degree(tempcoa, n);
      TMatrix[i+1, j] := abs(tcoeff(tempcoa, n));
    end if;
    tempco2 := [coeffs(tempcoa, n)];
    elems := NULL;
    for l from 1 to nops(tempco2) do
      elems := elems, l;
    end do;
    HMatrix[i+1,j] := max(op(applyop(abs,{elems},[op(tempco2)])));
  end do;
end do;

##############################
# 0 stands for (-infinity, 0)
# c(<>0) stands for (0, c)
PMmapping := proc(i, j)
if (HMatrix[i, j] = 0) then
  return 0;
elif (DMatrix[i, j] = 0) then
  return HMatrix[i, j];
else
  return [DMatrix[i, j], HMatrix[i, j]];
end if;
end proc:
##############################

return [DMatrix, HMatrix, L+1];
end proc:
```

```
#########################################
# Find the minimum column of a matrix
# among specified columns in 'colSet'.
#########################################
findMinCol := proc(mat, colSet)

local hsum, col, mcol, minSum, hs, k;

hsum := NULL;
for col in colSet do
  hsum := hsum, convert(convert(mat[1..-1,col], list), `+`);
end do;
hsum := [hsum];

minSum := hsum[1];
mcol := 1;
for col from 2 to nops(hsum) do
  hs := hsum[col];
  if (hs < minSum) then
    minSum := hs;
    mcol := col;
  end if;
end do;

return colSet[mcol];
end proc:



###################################
# Apply DH augment to a matrix.
###################################

DHAugment := proc(matD, matH, colSet)

local augMatD, augMatH, mcol, rowDim, colDim,
      col, row, cols, cslen;

cols := colSet;
rowDim := nops(convert(matD[1..-1,1],list));
```

```
colDim := nops(convert(matD[1,1..-1],list));
augMatD := matD;
augMatH := matH;
while rowDim < colDim do
  mcol := findMinCol(augMatH, cols);
  for col in cols do
    for row from 1 to rowDim do
      augMatD[row,col] := max(augMatD[row,col], augMatD[row,mcol]);
      augMatH[row,col] := max(augMatH[row,col], augMatH[row,mcol]);
    end do;
  end do;
  colDim := colDim - 1;
  augMatD := augMatD[1..-1, [1..mcol-1,mcol+1..-1]];
  augMatH := augMatH[1..-1, [1..mcol-1,mcol+1..-1]];
  cols := cols[1..-2];
end do;
return [augMatD, augMatH];
end proc:



############################################
### Apply 0-1 augment to a matrix.
############################################
zeroneAugment := proc(mat)

local rowDim, colDim, row, col;

rowDim := nops(convert(mat[1..-1,1],list));
colDim := nops(convert(mat[1,1..-1],list));
for row from 1 to rowDim do
  for col from 1 to colDim do
    if (mat[row,col] = 0) then
      mat[row,col] := 1;
    end if;
  end do;
end do;

return mat;
end proc:
```

```
##########################################################
# Compute the D(M) and H(M) by their definition formulas,
# i.e., running over all the permutations.
##########################################################
LeibnizDHperm := proc(DMatrix, HMatrix)

local i, j, v, sj, II, JJ, x, w, V, X, Y,
      pi, pj, pk, rho, D, DS, H, d, h, n;

D := 0;
H := 0;
Y := 0;
n := nops(convert(DMatrix[1..-1,1],list));
v := array(1..n);
for i from 1 to n do
  v[i] := i;
end do;
DS := NULL;
d := 0;
h := 1;
for pj from 1 to n do
  DS := DS, DMatrix[pj,v[pj]];
end do;
DS := convert(-1*sort(-1*[DS]), list);
d := DS[1];
h := HMatrix[1, v[1]];
for pk from 2 to n do
  h := h * (min(d, DS[pk])+1) * HMatrix[pk, v[pk]];
  d := d + DS[pk];
end do;

D := max(D, d);
H := H + h;
for j from 2 to infinity while Y = 0 do
  II := 0;
  JJ := 0;
  for i from 2 to n do
    if v[i-1] < v[i] then
      II := max(i,II);
  end if;
end do;
for i from 1 to n do
  if v[II-1] < v[i] then
    JJ := max(JJ,i);
  end if;
end do;
x := v[II-1];
v[II-1] := v[JJ];
v[JJ] := x;
w := [];
for i from 1 to n do
  w := [op(w), v[i]];
end do;
for i from II to n do
  v[i] := w[n+II-i];
end do;

DS := NULL;
d := 0;
h := 1;
for pj from 1 to n do
  DS := DS, DMatrix[pj,v[pj]];
end do;
DS := convert(-1*sort(-1*[DS]), list);
d := DS[1];
h := HMatrix[1, v[1]];
for pk from 2 to n do
  h := h * (min(d, DS[pk])+1) * HMatrix[pk, v[pk]];
  d := d + DS[pk];
end do;
D := max(D, d);
H := H + h;

Y := 1;
for i from 2 to n do
  if v[i-1] > v[i] then
    X := 1;
    Y := Y * X;
  else
```

```
      X := 0;
      Y := Y * X;
    end if;
  end do;
end do;

return [D, H];
end proc:
```

个人简历 在学期间发表的学术论文与研究成果

个人简历

1981 年 7 月 23 日出生于河南省新乡市. 1999 年 9 月考入西安电子科技大学理学院数学科学系信息与计算科学专业学习, 于 2003 年 7 月本科毕业并获得理学学士学位. 同年 9 月保送至南开大学组合数学中心应用数学专业学习, 2006 年 7 月硕士毕业并获理学硕士学位. 同年 9 月经免试推荐继续在南开大学组合数学中心攻读应用数学博士学位至今. 期间受国家留学基金委 "国家建设高水平大学公派研究生项目" 资助, 于 2007 年 11 月至 2008 年 8 月赴英国爱丁堡大学信息学院数据库研究组访问学习.

在学期间的学术论文与研究成果

1. Qiang-Hui Guo, Qing-Hu Hou and Lisa H. Sun, Proving hypergeometric identities by numerical verifications, *J. Symbolic Comp.* **43** (2008) 895–907.
2. Qiang-Hui Guo and Heiko Müller, Detecting and representing changes in multi-version XML documents with key structure for archiving, preprint.

参加科研项目

1. 2005 年 1 月 — 2007 年 12 月, 国家自然科学基金委面上项目 —— q- 级数与机器证明. 项目批准号: 10401017.
2. 2005 年 4 月 — 2007 年 12 月, 天津市科委科技攻关计划重点科技攻关专项项目 —— 数学机械化. 合同编号: 05YFGZGX23800.