J. Korean Math. Soc. **36** (1999), No. 2, pp. 267–298

A DIOPHANTINE CONSTRUCTION OF AN EXACT ALGEBRAIC FORMULA FOR GRADED PARTITION FUNCTIONS

SUN T. SOH

ABSTRACT. A geometric construction of an exact algebraic formula for graded partition functions, of which a special one is the classical unrestricted partition function $p(n)$, from a diophantine point of view is presented. Moreover, the involved process allows us to compute the values of a graded partition function in an inductive manner with a geometrically built-in self-error-checking ability at each step for correctness of the computed values of the partition function under consideration.

1. Introduction

One first reviews notation and terminology.

NOTATION. Let $\mathbf{A} = \{1, 2, 3, 4, \ ..., q\ \}$ and n a positive integer $\leq q$.

$p(n)$ = the number of partitions of n into natural integers without any restrictions.

$p_{\mathbf{A}}(n)$ = that of n into parts belonging to $\mathbf{A}$.

$p_{\mathbf{A},m}(n)$ = that of n into parts not exceeding m and belonging to $\mathbf{A}$.

$p_{\mathbf{A}}^{(m)}(n)$ = that of n into at most m summands, all belonging to $\mathbf{A}$.

$p_{\mathbf{A}}^{(d)}(n)$ = that of n into distinct summands, all belonging to $\mathbf{A}$.

$p_{\mathbf{O}}(n)$ = that of n into odd positive integers belonging to $\mathbf{A}$.

$p_{\mathbf{E}}(n)$ = that of n into even positive integers belonging to $\mathbf{A}$.

Received March 12, 1998.

1991 Mathematics Subject Classification: 11P81, 14M10, 14N10.

Key words and phrases: complete intersections, computation of partition functions.

This research was supported by Korean Ministry of Education through Research Fund BSRI-97-1438.

One may recall that for a given positive integer n the number $p(n)$ is equal to the number of non-isomorphic abelian groups of order p^n for prime p [7].In representation theory, one may also recall that $p(n)$ is equal to that of conjugacy classes of the symmetric group S_n, and hence is equal to that of irreducible representation of S_n, and hence is equal to that of different Young diagrams or Ferrers diagrams, and hence is equal to that of orthonormal basic characters χ_V for $\mathbf{C}_{class}(S_n)$.[4]. This indicates that the study of partition functions is no longer an isolated subject.

On the other hand, as the partition functions in the above Notation indicate, it is very easy to define a new partition function by introducing somewhat mathematically meaningful restrictions on other partition functions of which a very special one is the well-known classical unrestricted partition function $p(n)$. This aspect is certainly a property of partition functions in general.

When one considers formulas for a given partition function, one should first note that there are basically two different types of them, either exact, i.e., **complete**, with which one computes the value of the function for a single natural number, or **recursive**, with which one computes the values in a recursive manner. For different reasons, we usually need both types.

The purpose of this paper is to establish a geometric construction of a new *single* exact, i.e., complete, algebraic formula for "graded" partition functions, which is defined in Section 2, of which specializations are, for instances, each of the above partition functions. Here, the word "algebraic" exactly means that we do not need any approximated additions, subtractions and/or multiplications of truncated fractions of real numbers (which is indeed the case of the Rademacher's formula, Eq. (3) below, for the classical unrestricted partition function $p(n)$) in computing the values of the partition function under consideration. Moreover, one also obtains, as an easy corollary of the main result, a **quasi-recursive** formula associated to the complete one with a built-in error-checking and/or correcting ability. As long as actual computation with computer is concerned, this recursive formula and its generalizations in [14] are new ones providing us with some fast algorithms, according to Prof. George E. Andrews at Penn. St. Univ.

As one shall see clearly later, these results are obtained as a corollary of the author's previous results on a homogeneous complete intersection in Algebraic Geometry, by the process of passing through the areas of the so-called Formal Geometry to which, for instance, the theory of q-series belongs.

EXAMPLE 1. Here are the value of $p(n)$ for various n which are taken from my computation of $p(n)$ for n up to 5000 in a tolerable amount of time of 80 Minutes in a small sized my home PC by simply expanding the following polynomial in t (i.e., the **direct enumeration of the generating function** of $p(n)$ modulo t^{5001}):

$$1 \cdot (1 + t + t^2 + ... + t^{5000}) \cdot (1 + t^2 + t^4 + ... + t^{5000})$$
$$\cdot (1 + t^3 + t^6 + ... + t^{4998}) \cdot \cdots \cdots \cdot (1 + t^{2500} + t^{5000})$$
$$\cdot (1 + t^{2501})(1 + t^{2502}) \cdot \cdots \cdot (1 + t^{4999}) \cdot (1 + t^{5000}), \text{modulo } t^{5001}$$

of which the n-th coefficient is equal to $p(n)$ for n up to 5000, and all this was due to a rapid development in software and hardware industry in recent decades. One may recall that such a modulo computation requires $O(n^2)$ of operations so that its time efficiency is not that bad. But, several computations for some large numbers certainly tell us that its space efficiency is not quite good. On the other hand, one is naturally lead to wonder whether these computed values are indeed all correct and one begins to wonder how to verify that they are all correct.

$p(10) = 42.$
$p(100) = 190,569,292.$ (9 digits)
$p(200) = 3,972,999,029,388.$ (13 digits)
$p(300) = 9,253,082,936,723,602.$ (16 digits)
$p(400) = 6,727,090,051,741,041,926.$ (19 digits)
$p(500) = 2,300,165,032,574,323,995,027.$ (22 digits)
$p(600) = 458,004,788,008,144,308,553,622.$ (24 digits)
$p(700) = 60,378,285,202,834,474,611,028,659.$ (26 digits)
$p(800) = 5,733,052,172,321,422,504,456,911,979.$ (28 digits)
$p(900) = 415,873,681,190,459,054,784,114,365,430.$ (30 digits)
$p(1000) = 24,061,467,864,032,622,473,692,149,727,991.$ (32 digits)
$p(2000) = 4,720,819,175,619,413,888,601,432,406,799,959,512,$
$200,344,166.$ (46 digits)
$p(3000) = 496,025,142,797,537,184,410,324,879,054,927,095,334,$
$462,742,231,683,423,624.$ (57 digits)
$p(4000) = 1,024,150,064,776,551,375,119,256,307,915,896,842,$
$122,498,030,313,150,910,234,889,093,895.$ (67 digits)

$$p(5000) = 169{,}820{,}168{,}825{,}442{,}121{,}851{,}975{,}101{,}689{,}306{,}431{,}361{,}\\ 757{,}683{,}049{,}829{,}233{,}322{,}203{,}824{,}652{,}329{,}144{,}349.\\ \text{(75 digits)}$$

As this example indicates, the number $p(n)$ grows so fantastically fast as n increases that it was probably this reason why most mathematicians in the past tried to figure out its asymptotic growth, utilizing, for instance, some analytic methods. Moreover, even if one is able to compute numbers such as those in the above example, the result itself does not tell us what's really going on during computation of such big numbers, and consequently one does not have any insight into the inter-related arithmetic behavior of the values of $p(n)$ for various n at all.

It is well known that Euler and his successors introduced the method of generating functions, which is still commonly utilized in the study of theory of partitions in connection with some algebraic methods such as decomposition into partial fractions, or in connection with analytic methods such as Cauchy's theorem on residues, contour integrations, or Tauberian theorems, etc.

Since the power series expansion of the generating function of $p(n)$ is invertible in the ring of formal power series $\mathbf{Z}[[t]]$ which is an object of the so-called Formal Geometry, as its constant term is $1 \neq 0$, one obtains its inverse described in Theorem 2 below and from the fact that their product must be equal to 1, one finally obtains a famous recursive formula for $p(n)$:

THEOREM 1. (Euler) *Given a natural number n, one has*

$$(1) \quad p(n) = p(n-1) + p(n-2) - p(n-5) - p(n-7) + \cdots \\ +(-1)^{j+1} p(n-n_j) + \cdots,$$

where the sum is taken until the so-called "pentagonal numbers" $n_j = j(3j \pm 1)/2$ are $\leq n$ and by definition $p(0) = 1$.

One can develop an **algorithm** based on this recursive formula and it is well known that it only needs $O(n^{3/2})$ operations so that one may say that it is very efficient.

THEOREM 2. (Euler's pentagonal number theorem)

$$(2) \qquad \prod_{j=1}^{\infty}(1-t^j) = 1 + \sum_{j=1}^{\infty}(-1)^j t^{j(3j-1)/2}(1+t^j).$$

The proof of these two theorems are nowadays a standard material in most books in Number Theory.

It is well known that P. A. MacMahon used the recursive formula in Eq. (1) in 1916 to compute the values of $p(n)$ for n up to $n = 200$. In particular, when one is interested in some arithmetic behavior of the values of partition functions, an efficient recursive formula such as above will be very helpful to us, and it was indeed the case of Ramanujan and Hardy. According to Prof. George Andrew's account in his book (p. 150 in [1]), G. H. Hardy and S. Ramanujan were the first mathematicians who gave an asymptotic formula for $p(n)$ by a *divergent* series in 1917, the first term of which is

$$\frac{1}{2\pi\sqrt{2}}\frac{d}{dn}\left(\frac{\exp(\frac{2\pi}{\sqrt{6}}\sqrt{n-\frac{1}{24}})}{\sqrt{n-\frac{1}{24}}}\right).$$

The first five terms of the Hardy-Ramanujan formula give the "correct" value of $p(200)$, which is $3,972,999,029,388$. But, one should note that this kind of correctness is somewhat based on the assumption that one knows *in advance* the exact values of $p(n)$ for natural numbers n under consideration. Moreover, it is dependent on n that how many first terms be taken to get such a result.

In 1937, H. Rademacher discovered an exact, i.e., complete, formula for $p(n)$ in the form of a *convergent* series, which yields Hardy and Ramanujan's asymptotic formula as a corollary.

THEOREM 3. (Rademacher) *Given a natural number n, one has*

$$(3)\quad p(n) = \frac{1}{\pi\sqrt{2}}\sum_{k=1}^{\infty} A_k(n)\sqrt{k}\frac{d}{dn}\frac{\sinh((\pi/k)\sqrt{(2/3)(n-\frac{1}{24})})}{\sqrt{n-\frac{1}{24}}},$$

where

$$A_k(n) = \sum_{\substack{0\le h<k\\(h,k)=1}} \varpi_{h,k}\exp(-2\pi inh/k),$$

where

$$\varpi_{h,k} = \exp(\pi i s(h,k)),$$

is a 24k-th root of unity and where

$$s(h,k)=\sum_{\mu=1}^{k-1}\left(\frac{\mu}{k}-\left[\frac{\mu}{k}\right]-\frac{1}{2}\right)\left(\frac{h\mu}{k}-\left[\frac{h\mu}{k}\right]-\frac{1}{2}\right)$$

is the so-called Dedekind sum.

It is well known that this theorem is based on Cauchy's theory of contour integration and the theory of modular forms. The proof of this hard-to-understand approximation theorem, yet theoretically very important, to an exact integer value is well-presented, for instance, in Prof. Andrew's book. (pp. 71-87 in [2]).

This is the formula some mathematicians up to now insisted to call an exact "algebraic" formula for $p(n)$, although it is analytic in nature. I humbly request the reader to compare Eq. (3) with my formula Eq. (18) or Eq. (19), which is also valid for the classical unrestricted partition function $p(n)$, in order to see a big difference between them. In particular, my formula is not only complete, i.e., exact, but also purely algebraic, because it terminates after exactly the first $(n+1)$ terms for a given natural number n and there is no trace of analysis in my formula.

WARNING. (The following interesting remarks were added very recently.) Of course, one can develop an **algorithm** based on the Rademacher's formula for the value of $p(n)$ for a *single* input n. However, in developing such an algorithm, there are several things one should keep in mind. Among them, for instance, since truncated fractional real numbers of very much different size are added, subtracted and/or multiplied to become a part of the integer value of $p(n)$ for a given n, one has to be very careful about the problem of adjusting "precision" during actual computation, otherwise, one will never get a reliable correct result at all. Another consideration, for instance, is that in order to make it efficient one is advised to utilize a property of the so-called Dedekind sum. This property is quite useful since one does not know, in advance, how many terms has to be added to obtain a result which is sufficiently close enough to the true integer value of $p(n)$. Because of these approximation-related problems unavoidably associated to Rademacher's formula as well as its non-deterministic characteristic, it is still quite difficult for me to accept the Rademacher's formula as a practically *reliable* means in computing the integer value of $p(n)$ for a given single n, although I believe I myself already developed a very reliable program for it.

If you are only interested in computing the value of $p(n)$ for a *single* natural number n, the Rademacher's formula will answer you very efficiently, *provided that* you really trust reliability of the implementation of the algorithm for it. According to my experience, this is not an easy matter to ignore or to put aside, and it is probably this reason why the well-known Computer Algebra System such as MAPLE does not compute the value of $p(n)$ for a single input n using the Rademacher's formula for it.

A rather disturbing evidence of this is that, the author has recently found a serious error in p. 70 of Prof. George Andrews' book [2], The Theory of Partitions, 2nd printing with revision date in 1981, in which intermediate computational results for the value of $p(200)$ based on Rademacher's formula up to the first eight terms of it are presented as an evidence of the strength and practical usefulness of Rademacher's formula in order to convince other mathematicians. More precisely, the value of the third term, -87.555, presented in his book, turned out to be not correct, according to my computation of the value of $p(200)$ with my own program for Rademacher's formula. It is quite possible that this error was accidentally introduced by the late Prof. Rademacher himself. Immediately after finding the error, I reported it to Prof. Andrews and he promised that it will be soon corrected in the next printing of his book.

I had to spend sometime with a great effort in order to overcome successfully, I believe, the above mentioned serious difficulties associated to the implementation of the Rademacher's formula. In my opinion, reliability of a program should always go first, and then its speed. If one developed somewhat a fast program for a formula which is not very reliable, who will even attempt to use it?

Another disturbing evidence of the difficulties associated with the implementation of the Rademacher's formula is that, the value of $p(10^6)$ which is computed by P. Shiu with his program for it and presented in his recent paper [10] turned out to be wrong. His result is -58 off the true value of it. Since the number of digits taken by the value of $p(10^6)$ is 1108, this is all about the reliability question of the program. With my own program for the Rademacher's formula for $p(n)$, I computed the value of $p(10^6)$ and the result was, to my regret, P. Shiu's result plus 58. After strengthening computational conditions several time for accuracy increase, I computed, at three different PC's, the value of $p(10^6)$ several times and obtained all the time the same result which is bigger than P. Shiu's result by 58. Since this does not mean absolutely that my result for $p(10^6)$ is indeed correct, I had to spend about 3 weeks, using a different method, to further verify that my result is indeed correct with my new Pentium Pro PC running at 200 MHz, and this further verification was definitely quite a pain to me. When I contacted him with my correct result, P. Shiu blamed that the Computer Algebra System, MATHEMATICA, which he used for computing the value of $p(10^6)$ with his own implementation of the Rademacher's formula, caused such a terrible computational error. It seemed to me that he was not aware of seriousness of the problems I mentioned above and he trusted too much that MATHEMATICA will automatically resolve such problems. It will be certainly an interesting subject to discuss the problems associated with the implementation of the Rademacher's formula for the classical partition function $p(n)$ and some appropriate methods to overcome such difficulties. I am planning to write another paper focusing on these problems. The current status of my program for the Rademacher's formula for $p(n)$ is very reliable, but it requires further evolution to speed up its performance.

Also in my later paper, I shall discuss how one can check the validity of the value of $p(n)$ for a given n in a *recursive* and *exact* manner. With this recursive method, I was able to check, spending only less than 3 days with my new Pentium Pro PC running at 200 MHz with 128 Mbytes of RAM, that **all** the values of $p(n)$ which I computed recursively using Euler's formula for n from 1 up to 502090 were indeed correct. This is, to my best opinion, a really efficient and truly reliable method to verify not a single value but all of them up to the last one are all correct, with only one demand that it may need more memory as the last n gets big. I also told this new recursive method to Prof. George E. Andrews in Aug, 1997. Since this new method is completely different from the one successively developed in this paper and its generalization in [14], we are, particularly in the case of the classical partition function $p(n)$, now in a lot better position to compute and verify the values of $p(n)$. But for a general graded partition function $p_{\mathrm{M}}(n)$ over a general multiset M, this new method does not work. (This is the end of Warning).

There are fairly many standard results in terms of generating functions of partition functions which I don't have to mention them here even for reviewing purpose, though the method of generating functions is still very useful. For instance, by comparing their generating functions one has

LEMMA 1. $p^{(d)}(n) = p_{\mathbf{O}}(n)$.

Another line of thought is combinatorial methods, especially the theory of graphs. With the help of this, by analyzing the so-called Young diagram or Ferrers diagram, one has

LEMMA 2. $p_{\mathbf{A}}^{(m)}(n) = p_{\mathbf{A},m}(n)$.

For more about elementary historical accounts of the subject, one may refer to [6] [5],[1],[2] or to a section in EDM [8].

Euler, Jacobi, and their immediate successors did not seem to have tried to estimate the size of $p_{\mathbf{A}}(n)$. The construction made in Section 3, when specialized, gives an exact formula for $p_{\mathbf{A}}(n)$.

Before one goes to Section 3 for main results, one may note that in consideration of $p_{\mathbf{A}}(n)$ for a given n, the order of the involved summands is irrelevant by definition; hence rearranging the summands if necessary and grouping together summands that are equal, every partition can be written uniquely in the form

$$n = \nu(1)d(1) + \nu(2)d(2) + \cdots + \nu(i)d(i) + \cdots,$$

where $d(1), d(2), \cdots$, are distinct elements of $\mathbf{A}$ in some fixed order, and where the $\nu(i)$ are positive integers. One also notes that this is a finite sum simply because all the $d(i)$ showing up such a sum are necessarily less than or equal to the given n. As you shall see it later, this is precisely the reason why the method presented in Section 3 is significant in computing $p_{\mathbf{A}}(n)$ for a given n. Each coefficient $\nu(i)$ indicates how often a given summand $d(i) \in \mathbf{A}$ occurs in the partition under consideration, and it is called the *frequency* of $d(i)$ in that partition of n; hence any given partition is completely determined by the set of its frequencies; consequently the number $p_{\mathbf{A}}(n)$ of partitions of n is precisely the number of distinct solutions of the above *Diophantine equation.* It is essentially this reason why the method, which I am going to develop in Section 3, should work with one hundred percent accuracy.

As one can see in Section 3, the above line of thought toward the size of the solution set of certain Diophantine equation is crucial to my construction, only with one major exception that the above $d(i)$ may not have to be all distinct in my construction. One notes that such a restriction of all the participants being distinct in the case of the classical unrestricted partition function $p(n)$ is quite unnatural to assume from the beginning, because, for instance, the same number could be chosen for different $d(i)$'s if one decides each $d(i)$ to have different colors to begin with, i.e., if one is given a multiset.

2. The notion of graded partition functions

After a moment of thought, one should, first of all, recognize that for any partition function introduced in Section 1, one has $p_{\mathbf{A}}(n) = p_{\mathbf{A},m}(n)$ for all $n \leq m$ for a suitably chosen finite subset $\mathbf{A}$ for the given n. Based on the observation made in the last part of Section 1, I introduce a graded partition function $p_{\mathbf{M}}(n)$ as follows:

DEFINITION 1. Let $\mathbf{M} = \cup_{i=1}^{r}\{d(i)_1, d(i)_2, \cdots, d(i)_{\alpha_i}\}$ be a multiset where r a positive integer or $r = 1, 2, 3, \cdots$, each $d(i)_j = d(i)$ a positive integer, and each α_i a positive integer. Given a positive integer n, denote by

$p_{\mathbf{M}}(n) =$ the number of partitions of $n$into parts belonging to $\mathbf{M}$.

This one is clearly unrestricted and is, from now on, called **a graded partition function over M**. One also notes that the elements of $\mathbf{M}$ are equally well described as those $d(i)$ of which some may be the same. Then one manages to obtain

THEOREM 4. *The generating function $F_{\mathbf{M}}(t)$ of $p_{\mathbf{M}}(n)$ is represented as*

$$F_{\mathbf{M}}(t) = \prod_{d(i)\in\mathbf{M}} \frac{1}{(1 - t^{d(i)})^{\alpha_i}}, \tag{4}$$

where $\alpha_i = \#$ of the same $d(i)$ in $\mathbf{M}$ for each i.

Proof. It is Eq. (15) in Section 3. □

In other words, a partition function $p_{\mathbf{M}}(n)$ over a multiset $\mathbf{M}$ is called **graded** if its generating function is a product form of Eq. (4). In the next section, it will be self-clear why I called them "graded".

3. A geometric construction of an exact algebraic formula for graded partition functions

In order to obtain a single exact algebraic formula for graded partition functions introduced in Section 2, one may modify the problem into the problem of a homogeneous complete intersection in Algebraic Geometry and it goes as follows.

Let $A = k[X_1, \cdots, X_m]$ be the polynomial ring in m indeterminates X_i, over a field k and $B = k[f_1, \cdots, f_m]$, where f_i are homogeneous polynomials of positive total degree $d(i)$ for each i. One notes that some of the $d(i)$ may be the same, and hence it reflects the whole situation explained in both Section 1 and Section 2.

Under this set-up, I shall prove:

THEOREM 5. *Suppose that A is a free B-module of finite rank and let $\{\alpha_i\}$ be a B-basis of A. Then the number s_n of those α_i with the same homogeneous degree n, is determined by*

$$\sum_{n=0}^{q} s_n \cdot t^n = \prod_{i=1}^{m} (1 + t + \cdots + t^{d(i)-1}), \tag{5}$$

where $q = \max\{(\deg(\alpha_i)\}$; consequently, the B-rank of A is equal to the degree of the ideal $(f_1, \cdots, f_m)$ in A, namely, the product $\deg(f_1) \cdot \deg(f_2) \cdots \deg(f_m)$.

Thus, if the given m homogeneous polynomials $f_1, \cdots, f_m$ in m variables of positive degree over, for instance, the field of complex numbers $\mathbf{C}$, are such that A is a free B-module of finite rank, then we can easily construct a finite B-basis of A, by utilizing the above theorem and by applying the Buchberger's algorithm for Groebner bases, whenever it is necessary.

EXAMPLE 2. With $f_i = X_i^{d(i)}$ over the field $\mathbf{C}$ of complex numbers for $i = 1, \cdots, m$, it is left to the reader to check that A is a free B-module of finite rank being equal to the product $d(1) \cdot d(2) \cdots d(m)$.

As one shall see below, the algebraic relations established in the proof of this theorem are "locally" a theoretical basis for a diophantine construction of an exact algebraic formula for the graded partition function $p_{\mathrm{M}}(n)$.

Proof of Theorem 5. We first recall some elementary facts about the canonical grading structures on A and B. The polynomial ring $A = \oplus_{n=0}^{\infty} A_n$ is positively graded by total degree with $A_0 = k$. Thus given $n \in \mathbf{N}$, A_n is a finite dimensional k-vector space of which a k-basis consists of all monomials of total degree n and hence we get

$$\dim_k(A_n) = \binom{m+n-1}{n}. \tag{6}$$

Moreover, $B = \oplus_{n=0}^{\infty} B_n$ is the induced positive grading on B with $B_n = B \cap A_n$, where

$$B_n = \left\{ \sum_j^{finite} \rho_i \left(\prod_{i=1}^m f_i^{\nu(i)} \right)_j \in B : n = \sum_{i=1}^m \nu(i)d(i) \text{ for each } j \right\}.$$

Thus, each B_n is also a finite dimensional k-vector space with a k-basis consisting of those products $f_1^{\nu(1)} \cdots f_m^{\nu(m)}$ satisfying

$$n = \sum_{i=1}^m \nu(i)d(i). \tag{7}$$

From this, we thus see that $\dim_k(B_n)$ is not dependent on the f_i themselves but on the total degrees $d(i)$ of the f_i. Moreover, we clearly see that $\dim_k B_n = 0$ if $n < \min_{1\le i\le m}\{d(i)\}$.

Now, A is a free B-module of finite rank by hypothesis, and hence we can write A as

$$A = \oplus_{i=1}^r B\alpha_i$$

for some homogeneous generators α_i of A over B, each of which is necessarily a component, i.e., a term, i.e., a monic monomial, of positive total degree such that $\deg(\alpha_1) \le \cdots \le \deg(\alpha_r)$. Then since A consists of all the homogeneous polynomials of every degree ≥ 0, we must have, for instance, that, for each j with $0 \le j < \min_{1\le i\le m}\{d(i)\}$, there exist more than or equal to one number of α_i such that $\deg(\alpha_i) = j$. Taking these into consideration we must have that for each $n \ge 0$,

$$A_n = \oplus_{i=1}^r B_{n-\deg(\alpha_i)}\alpha_i.$$

Thus,

$$\dim_k(A_n) = \sum_{i=1}^{r} \dim_k(B_{n-\deg(\alpha_i)}\alpha_i) = \sum_{i=1}^{r} \dim_k(B_{n-\deg(\alpha_i)}), \tag{8}$$

where $\dim_k(B_j) = 0$ if $j < 0$. Since some α_i may have the same total degree, we then look at the right side more carefully:

DEFINITION 2. Under this situation, we define a function $s : \mathbf{N} \to \mathbf{N}$ by sending $n \mapsto s_n =$ the number of those α_i such that $\deg(\alpha_i) = n$, where $\mathbf{N} = \{0, 1, 2, 3, \cdots\}$.

Thus in particular, $s_n = 0$ if $n > q = \max\{\deg(\alpha_i)\}$; hence we clearly have,

$$rank_B(A) = \sum_{n=0}^{q} s_n. \tag{9}$$

From now on, I shall call the list $(s_0, ..., s_q)$ of finitely many non-zero s_n's the *arithmetic invariants* of the rank. One notes that it only depends on the given positive integers m and n.

Since $s_j =$ the number of those α_i such that $\deg(\alpha_i) = j$, we obtain the following equality from Eq. (8):

$$\dim_k(A_n) = \sum_{j \in \mathbf{N}} \dim_k(B_{n-j}) \cdot s_j, \tag{10}$$

where $s_j = 0$ if $j \geq q = \max(\deg(\alpha_i))$ and by Eq. (7), each $\dim_k(B_n)$ is only dependent on the total degrees $d(i)$ of the f_i. Thus when n runs through $\mathbf{N}$, we see that the s_j are also only dependent on the total degrees $d(i)$ of the f_i for fixed m. In particular, this indicates that one can work with the monomials $X_1^{d(1)}, \cdots, X_m^{d(m)}$, instead of $f_1, \cdots, f_m$, in order to compute both the s_j and the B-rank of A.

Now the basic question is how one can "compute" each of the s_n. Once they are computed then Eq. (9) determines the B-rank of A. But we can pursue more than that.

We shall see below that we can avoid the intermediate computations of $\dim_k A_n$ and $\dim_k B_n$ for various n if we are only interested in determining the rank of A over B as is described in the theorem. Moreover, we shall also see, at some stage below, how the s_n can be explicitly computed for each n.

For this purpose, we introduce

DEFINITION 3. We *define* the following power series in an undeterminate t,

$$a(t) = \sum_{n=0}^{\infty} \dim_k(A_n) \cdot t^n, \qquad b(t) = \sum_{n=0}^{\infty} \dim_k(B_n) \cdot t^n.$$

Then one can easily see that Eq. (10) becomes

$$a(t) = b(t) \cdot \left(\sum_{n=0}^{q} s_n \cdot t^n \right), \tag{11}$$

where $\sum_{n=0}^{q} s_n \cdot t^n$ is a polynomial in t with coefficients in $\mathbf{N}$ where s_n is defined in Definition 2. Since

$$\prod_{i=1}^{m} \frac{1}{1-t} = \prod_{i=1}^{m} (1 + t + \cdots + t^{\nu(i)} + \cdots) = \sum_{n=0}^{\infty} a_n \cdot t^n,$$

where $a_n = \#\{n = \sum_{i=1}^{m} \nu(i) : \nu(i) \geq 0\}$, and hence

$$a_n = \dim_k(A_n). \tag{12}$$

Thus we have

$$a(t) = \prod_{i=1}^{m} \frac{1}{1-t}. \tag{13}$$

Also since

$$\prod_{i=1}^{m} \frac{1}{1-t^{d(i)}} = \prod_{i=1}^{m} (1 + t^{d(i)} + \cdots + t^{\nu(i)d(i)} + \cdots) = \sum_{n=0}^{\infty} b_n \cdot t^n,$$

where $b_n = \#\{n = \sum_{i=1}^{m} \nu(i)d(i) : \nu(i) \geq 0\}$, and hence

$$b_n = \dim_k(B_n). \tag{14}$$

Thus we have

$$b(t) = \prod_{i=1}^{m} \frac{1}{1-t^{d(i)}}. \tag{15}$$

Thus, Eq. (11) becomes

(16) $$\sum_{n=0}^{q} s_n \cdot t^n = \frac{a(t)}{b(t)} = \prod_{i=1}^{m}(1 + t + ... + t^{d(i)-1}),$$

where $q = \max\{\deg(\alpha_i)\}$.

In particular, taking $t = 1$, we obtain from Eq. (16) that

(17) $$rank_B(A) = \prod_{i=1}^{m} d(i),$$

and this finishes the proof of the theorem. □

REMARK 1. The above proof also shows the following. By expanding the right side of Eq. (16) for a given m and the $d(i)$, we can explicitly compute the arithmetic invariant $(s_0, \cdots, s_q)$ of the $rank_B A$. As we have seen before, Eq. (16) also tells us that each component s_n of the arithmetic invariant is only dependent on total degrees $d(i)$ of homogeneous polynomials f_i with which one started. Thus, if one is interested in the computation of the arithmetic invariants and hence the $rank_B A$, then one can safely replace the f_i given at the start with the $X_i^{d(i)}$, $i = 1, \cdots, m$.

REMARK 2. On the other hand, one notes that Eq. (7) above tells us that $\dim_k B_n$ is equal to the value of the graded partition function $p_{\mathbf{M}}(n)$ for a given positive number n and for those possibly non-distinct m summands $d(i)$ used in representation of n, namely

$$n = \nu(1)d(1) + \nu(2)d(2) + \cdots + \nu(m)d(m)$$

where $d(i)$ are elements of the multiset $\mathbf{M}$ which are actually used as summands to represent the given n. In particular, the first two **Lemmas 1 and 2** tell us which of them are in fact the same as other computable ones.

Based on these remarks, I obtain the main result of this paper as a corollary of the above theorem:

THEOREM 6. *Given a non-empty multiset* $\mathbf{M}$, *there exists a list of positive integers* $(s_0, s_1, \cdots, s_q)$ *(cf. Eq. (16)), called the* arithmetic invariants *of* M, *such that the graded partition function* $p_{\mathbf{M}}(n)$ *is represented as*

$$(18)\quad p_{\mathbf{M}}(n) = \sum_{i=0}^{n} (-1)^i Q_i(s_0, s_1, \cdots, s_i) \binom{m + (n-i) - 1}{n-i}, \quad n \in \mathbf{N}$$

where m *is the number of those* $d(i)$ (*of which some may be the same*) *belonging to* M, *and where each* $Q_i(s_0, \cdots, s_i)$ *is a uniquely determined polynomial in* $s_0, s_1, \cdots, s_i$ *with integer coefficients* (*cf.* Corollary 1 *below*).

REMARK 3. One first notes that Eq. (18) is the reciprocal of Eq. (10), that is,

$$\dim_k(B_n) = \sum_{i=0}^{n} (-1)^i Q_i(s_0, s_1, ..., s_i) \dim_k(A_{n-i}),$$

where

$$\binom{m + (n-i) - 1}{n-i} = \dim_k(A_{n-i}) = 0$$

for $n < i$, simply because of the canonical positive grading on $A = k[X_1, X_2, ..., X_n]$. Thus Eq. (18) can be safely rewritten as

$$(19)\qquad p_{\mathbf{M}}(n) = \sum_{i=0}^{\infty} (-1)^i Q_i(s_0, s_1, ..., s_i) \binom{m + (n-i) - 1}{n-i},$$

although it is a finite sum of the first $(n+1)$ alternating terms.

Secondly, one shall see in the proof of this theorem that each partition function mentioned in Section 1 is represented by this "single" exact algebraic formula, Eq. (18), only with the different $s_0, s_1, ..., s_q$ which are dependent on **M**, whereas the polynomials $Q_i(s_0, s_1, ..., s_i)$ in the s_i themselves are not dependent on the given m, i.e., the size of the given **M**, but are only dependent on i. In this sense, one may say that Eq. (18) is a **universal representation** of a certain class of partition functions which includes those mentioned in Section 1.

Since the s_i in Eq. (16) are generally dependent on **M**, one may have to write them as $s_{i,m}$ if one needs to indicate them in a more precise manner. Such a situation will show up later.

Proof of Theorem 6. One notes that since $p_{\mathbf{M}}(n) = \dim_k(B_n)$ in Eq. (10) with $\dim_k(B_j) = 0$ if j is a negative integer, Eq. (10) is a system of linear equations for multiple unknowns $p_M(n)$ when n runs trough from 1 up to a given n. As $\alpha_0 = 1 = y_0$; hence defining $p_M(0) = 1$ (which is usually the case), Eq. (10) is obviously rewritten as

$$Ay = \alpha,$$

where $y = (y_0, y_1, ..., y_n)^t$, $\alpha = (\alpha_0, \alpha_1, ..., \alpha_n)^t$, and A is the $(n+1) \times (n+1)$ invertible matrix:

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdot & 0 & 0 \\ s_1 & 1 & 0 & 0 & \cdot & 0 & 0 \\ s_2 & s_1 & 1 & 0 & \cdot & 0 & 0 \\ \cdot & s_2 & s_1 & 1 & \cdot & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 0 \\ s_n & s_{n-1} & s_{n-2} & \cdot & \cdot & s_1 & 1 \end{pmatrix}.$$

Thus, one obtains each $p_M(j)$ for $j = 1, 2, ..., n$, as solutions of this system of linear equations:

$$(20) \qquad \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdot & 0 & 0 \\ s_1 & 1 & 0 & 0 & \cdot & 0 & 0 \\ s_2 & s_1 & 1 & 0 & \cdot & 0 & 0 \\ \cdot & s_2 & s_1 & 1 & \cdot & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 0 \\ s_n & s_{n-1} & s_{n-2} & \cdot & \cdot & s_1 & 1 \end{pmatrix}^{-1} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \cdot \\ \cdot \\ \cdot \\ \alpha_n \end{pmatrix},$$

where the s_i and the $\alpha_i = \binom{m+i-1}{i}$ are uniquely determined by the given M. (cf. Eq. (16)). In particular, one obtains Eq. (18) by equating the $(n+1)^{st}$ rows of both sides, where y_i is $p_M(i)$ for each $i = 0, 1, ..., n$. This should be the end of the proof, if one could recognize that the polynomials in $s_0, s_1, ..., s_n$ one obtains from the above $(n+1) \times (n+1)$ inverted matrix can be expressed as those with $(-1)^i$ in Eq. (18).

If one cannot see why the summands in Eq. (18) are alternating in sign, then one may have to look at it a little bit more carefully from a different angle and it goes as follows, where one writes $y_n = \dim_k B_n$ as before and $Q_i = Q_i(s_0, s_1, \cdots, s_i)$ only for convenience.

This is in fact a second proof of the theorem. If $n = 1$, Eq. (10) says that $y_1 = \binom{m}{1} - s_1 = (-1)^0 Q_0 \binom{m+1-1}{1} + (-1)^1 Q_1 \binom{m+(1-1)-1}{(1-1)}$ with $Q_0 = 1$ and $Q_1 = s_1$, since $s_0 = 1$ and $y_0 = 1$. Then with the hypothesis that $y_n = \sum_{i=0}^{n} (-1)^i Q_i \binom{m+(n-i)-1}{(n-i)}$holds for n with $Q_0 = 1$ for all $n < n+1$, one is left to show that the equality

$$y_{n+1} = \sum_{i=0}^{n+1} (-1)^i Q_i \binom{m + (n+1) - i - 1}{(n+1) - i}$$

holds for $n + 1$. Then one finishes the second proof the theorem by induction on n. But, one easily sees again from Eq. (10) that

$$\begin{aligned}
y_{n+1} &= \binom{m + (n+1) - 1}{(n+1)} - s_1 y_n - s_2 y_{n-1} - \cdots - s_n y_1 - s_{n+1} \\
&= \binom{m + (n+1) - 1}{(n+1)} - \textstyle\sum_{i=0}^{n} (-1)^i s_1 Q_i \binom{m + (n-i) - 1}{(n-i)} \\
&\quad - \textstyle\sum_{i=0}^{n-1} (-1)^i s_2 Q_i \binom{m + (n-1) - i - 1}{(n-1) - i} \\
&\quad \vdots \\
&\quad -(s_n Q_0 - s_n Q_1) \binom{m + (n - (n-1)) - i - 1}{(n - (n-1)) - i} \\
&\quad - s_{n+1} Q_0 \binom{m-1}{0},
\end{aligned}$$

where one puts $Q_0 = 1$. Thus rewriting this one obtains

$$
\begin{aligned}
y_{n+1} \;=\;& 1\cdot\binom{m+(n+1)-1}{(n+1)} \\
& -s_1Q_0\binom{m+n-1}{n} \\
& +(s_1Q_1-s_2Q_0)\binom{m+(n-1)-1}{(n-1)} \\
& -(s_1Q_2-s_2Q_1+s_3Q_0)\binom{m+(n-2)-1}{(n-2)} \\
& +(s_1Q_3-s_2Q_2+s_3Q_1-s_4Q_0)\binom{m+(n-3)-1}{(n-3)} \\
& \vdots \\
& +(-1)^{n+1}(s_1Q_n-s_2Q_{n-1}+s_3Q_{n-2}-\cdots+(-1)^n s_{n+1}Q_0) \\
=\;& \textstyle\sum_{i=0}^{n+1}(-1)^iQ_i\binom{m+(n+1)-i-1}{(n+1)-i}
\end{aligned}
$$

where each $Q_0(j)=1$ for $j=1,2,\cdots,n$, and this finishes the proof. □

From the second proof of the theorem, one obtains the following corollaries of which the first tells us the polynomials in Eq. (18) are recursively related to the previous ones and the second tells what they actually look like.

COROLLARY 1. *The $Q_i(s_0,s_1,\cdots,s_i)$ are determined recursively by*

$$
\begin{aligned}
(21)\quad Q_n(s_0,s_1,\cdots,s_n) &= \sum_{j=0}^{n-1}(-1)^j s_{j+1}Q_{n-1-j}(s_0,s_1,\cdots,s_{n-1-j}), \\
n &= 1,2,3,\cdots.
\end{aligned}
$$

where $Q_0=1$ by definition.

EXAMPLE 3. One can easily compute the polynomial $Q_i(s_0, s_1, \cdots, s_i)$ using this formula. For example, when $n = 5$, one gets

$$
\begin{aligned}
Q_0 &= 1 \\
Q_1 &= s_1 \\
Q_2 &= -s_2 + s_1^2 \\
Q_3 &= s_3 - 2s_2s_1 + s_1^3 \\
Q_4 &= -s_4 + 2s_3s_1 + s_2^2 - 3s_2s_1^2 + s_1^4 \\
Q_5 &= s_5 - 2s_4s_1 - 2s_3s_2 + 3s_3s_1^2 + 3s_2^2s_1 - 4s_2s_1^3 + s_1^5
\end{aligned} \tag{22}
$$

COROLLARY 2. 1. *The number of alternating terms in each polynomial* $Q_i(s_0, s_1, \cdots, s_i)$ *is equal to the value of* $p(i)$ = *the classical unrestricted partition function.*

2. *Each* $Q_i(s_0, s_1, \cdots, s_i)$ *is homogeneous of total degree* i *with respect to a grading on the* $s_0, s_1, \cdots, s_i$ *defined by* $\deg(s_i) = i$ *for each* i.
3. *Each* $Q_i(s_0, s_1, \cdots, s_i)$ *is monic in* s_1 *with* $Q_0(s_0) = s_0 = 1$ *and* $Q_1(s_0, s_1) = s_1$ = *the number of non-constant polynomial factors of the form* $1 + t + t^2 + \cdots + t^{d(i)-1}$ *shown up in the right side of Eq. (16).*

Again one notes that these polynomials in $s_0, \cdots, s_n$ are not dependent on m, but only dependent on n, and since they are theoretically (but perhaps not practically) determined from the $(n+1)^{st}$ row of the inverted matrix in Eq. (20), it is well qualified to call Eq. (18) an **"exact" algebraic formula** for graded partition functions. In particular, if one compares Eq. (19) with the non-algebraically exact formula, Eq. (3), for the case of the classical unrestricted partition function $p(n)$, one may easily recognize the obvious reason why Eq. (19) should be called an exact algebraic formula. Its first corollary then tells us that one has a recursive formula for the polynomials $Q_i(s_0, s_1, \cdots, s_i)$.

REMARK 4. Now one learned from the above proof how to obtain polynomials $Q_i(s_0, s_1, \cdots, s_i)$. When the actual computation is involved, since Eq. (10) above is equivalently expressed for n unknowns $y_1, \cdots, y_n$

as a system of linear equations

$$
(23) \qquad \begin{cases} \alpha_1 & = & y_1 \cdot s_0 + y_0 \cdot s_1 \\ \alpha_2 & = & y_2 \cdot s_0 + y_1 \cdot s_1 + y_0 \cdot s_2 \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \alpha_{n-1} & = & y_{n-1} \cdot s_0 + y_{n-2} \cdot s_1 + \cdots + y_0 \cdot s_{n-1} \\ \alpha_n & = & y_n \cdot s_0 + y_{n-1} \cdot s_1 + \cdots + y_1 \cdot s_{n-1} + y_0 \cdot s_n \end{cases}
$$

where $\alpha_i = \binom{m+i-1}{i}$ for each i with $y_0 = 1$, and $s_0 = 1$, the process of solving this system to get the desired solution $(p_{\mathrm{M}}(1), p_{\mathrm{M}}(2), \cdots, p_{\mathrm{M}}(n))$ of these n diophantine equations for n unknowns y_i successively for already given unknown parameters $s_0, s_1, \cdots, s_n$, and unknown $\alpha_1, \cdots, \alpha_n$, in turn produces each polynomial $Q_i(s_0, s_1, \cdots, s_i)$ in a simultaneous manner.

WARNING. One should note that Eq. (23) is not a linear recurrence of finite order.

One also notes that since the role of the y_i and that of the s_i are exactly the same in it, Eq. (23) can be equally used to compute each s_i from the α_i recursively if one knows a priori the value $p_{\mathrm{M}}(i)$ for each $i = 1, \cdots, n$ for given positive integers n and m.

REMARK 5. Eq. (23) also tells us the following. One starts with $p_{\mathrm{M}}(0) = \alpha_0$, where $\alpha_0 = \binom{m+0-1}{0} = 1$; then one extends this partial solution $(p_{\mathrm{M}}(0))$ to the solution $(p_{\mathrm{M}}(0), p_{\mathrm{M}}(1))$ of the first equation of Eq. (23); then one extends this partial solution to the next solution $(p_{\mathrm{M}}(0), p_{\mathrm{M}}(1), p_{\mathrm{M}}(2))$ of the second equation in Eq. (23); and so on, until one obtains the unique total solution $(p_{\mathrm{M}}(0), p_{\mathrm{M}}(1), \cdots, p_{\mathrm{M}}(n))$ of the n-th equation in Eq. (23).

Therefore, one sees that the number $p_{\mathrm{M}}(n)$ of distinct solutions of a diophantine equation

$$n = \nu(1)d(1) + \nu(2)d(2) + \cdots + \nu(m)d(m),$$

where $\nu(i) \geq 0$ for each i, is again the solution of the n-th diophantine equation in Eq. (23); hence the word "diophantine" in this paper has a double meaning.

4. Built-in error-checking ability of the formula in theorem 6

As was mentioned in Section 1, P. A. MacMahon used the following recursive formula in Eq. (1) to compute the values of $p(n)$ up to $n = 200$,

$$p(n) = p(n-1)+p(n-2)-p(n-5)-p(n-7)+\cdots+(-1)^{j+1}p(n-n_j)+\cdots$$

where the sum is taken for up to the largest j such that $n - n_j \geq 0$ and the $n_j = j(3j \pm 1)/2$ are the so-called "pentagonal numbers".

But the problem with computation of the values of $p(n)$ based on Theorem 1 is that one may not be quite sure whether or not the value for $p(n)$ one obtains from this recursive formula is free of computational errors, including "**soft-errors**", which may be accidentally introduced during computation in an undetectable manner, unless one knows in advance the correct values of $p\,(n)$. In particular, obtaining such a knowledge is obviously impossible when the values of $p(n)$ are ever first computed for large numbers n. To make it worse, once a computational error has been introduced to the value of $p(n)$, the values of $p(m)$, if computed based on this recursive formula of Euler, for $m > n$ are all going to be wrong. This is not a silly worry, since I personally had such a bad experience at least one time. Personally, I do not know by what means mathematicians at around 1915 believed the results were absolutely free of computational errors when P.A. MacMahon first computed the value of

$$p(200) = 3,972,999,029,388$$

based on Theorem 1. Of course, one can utilize Rademacher's formula to verify the results, but as I indicated before in the previous Section, this approach introduces another kinds of difficulties in an actual implementation of it.

An important feature or an extra ordinary strength of my construction of an exact algebraic formula made in Section 3 in general is that when one develops an algorithm based on Eq. (23), the algorithm itself has a built-in ability to check correctness of the value of $p_{\mathrm{M}}(n)$ at each step, and of which the reason goes as follows. Please note that this feature is due to the geometric idea of the graded structure I put into the whole construction process, as is explained in **Remark 5**.

Suppose one has computed the lattice point $(p_{\mathbf{M}}(0), p_{\mathbf{M}}(1), \cdots, p_{\mathbf{M}}(n))$ for some n, each $p_{\mathbf{M}}(j)$ is determined by the s_i and the α_i each of which is uniquely determined by the given m and n. To check the correctness of the $p_{\mathbf{M}}(j)$ for $j = 1, 2, \cdots, n$, one simply adds, for instance, the following integer

$$\max(n+1, 1 + \max_{1 \le i \le m}(d(i)))$$

(in fact, you can add any positive integer strictly bigger than n) to the old set $\mathbf{M}$ to obtain a new set $\mathbf{M}$; then computes the s_i and the α_i again with respect to this new $\mathbf{M}$. With these numbers, one then computes the lattice point $(p_{\mathbf{M}}(0), p_{\mathbf{M}}(1), \cdots, p_{\mathbf{M}}(n))$ again using Eq. (23). If the result of this computation is not the same as the previous one, it simply means that some computational error has been introduced during the computation of these numbers.

The reason may be easily explained from a geometric point of view. To make the explanation very clear, one only considers the following situation. Assuming that the lattice point $(p_{\mathbf{M}}(0), p_{\mathbf{M}}(1), \cdots, p_{\mathbf{M}}(n-1))$ is correctly computed, one considers the following system of two linear equations for unknown y_n and y_{n-1} with definitely different positive integer slopes $s_{1,m}$ and $s_{1,m+1}$:

$$(24) \qquad \begin{cases} \alpha_{n,m} &= y_n + s_{1,m} y_{n-1} + C_{n,m} \\ \alpha_{n,m+1} &= y_n + s_{1,m+1} y_{n-1} + C_{n,m+1} \end{cases},$$

where the $C_{n,m}$ and $C_{n,m+1}$ are well-determined integers from the lattice point $(p_{\mathbf{M}}(0), \cdots, p_{\mathbf{M}}(n-2))$, and the $s_{i,m}$ and the $s_{i,m+1}$, respectively. Then since the lattice point $(p_{\mathbf{M}}(n-1), p_{\mathbf{M}}(n))$ is defined as the intersection of these two lines by Eq. (10) and since one already knows the exact value of $p_{\mathbf{M}}(n-1)$ by assumption, the two computations for $p_{\mathbf{M}}(n)$ from the two different $\mathbf{M}$, the old one and the new one, should yield the same answer for both, unless some computation errors have been introduced during computation of $p_{\mathbf{M}}(n)$ from either the old $\mathbf{M}$ or the new $\mathbf{M}$. If this happens, one should stop computation and try to find out why it fails to compute the correct value of $p_{\mathbf{M}}(n)$, although it successfully computes the exact values for those $p_{\mathbf{M}}(j)$ for $j = 1, 2, \cdots, n-1$.

The worst situation would be the case when one gets wrong y_n's from the first and the second equations, respectively, but it just happened that these two wrong solutions are accidentally the same. Fortunately, the possibility of having such a worst situation is going to be extremely rare for sufficiently large numbers n. In fact, one easily convinces oneself that the more gets n bigger, the less chance gets one of having the *same* wrong solution for both equations, since the number of digits needed to represent the number $p_{\mathbf{M}}(n)$ increases very rapidly as n increases.

5. Applications

As mentioned before, once a natural number n is given to us for which $p_{\mathbf{M}}(n)$ is to be computed, only those elements d of $\mathbf{M}$ which are less than or equal to n are used to represent the number n as a finite sum of positive integers.

5.1. Graded partition functions

Here is a general situation. Since there has been no formula like Eq. (23) for a general multiset $\mathbf{M}$, one may expect that there should be lots of situations for which this formula could be applied. Example 4 below is a simple one, but Example 5 below shows some other type of them.

EXAMPLE 4. Let $\mathbf{M} = \{1_{red}, 1_{blue}, 1_{green}, 2, 3\}$ be a finite multiset. One wants to find the number $p_{\mathbf{M}}(n)$ of partitions of n where the 1's have different colors as is indicated as indices. For this, one first computes the arithmetic invariant, the s_i, as follows:

$$1 \cdot 1 \cdot 1 \cdot (1+t) \cdot (1+t+t^2) \quad = \quad 1 + 2t + 2t^2 + t^3 \; .$$

Thus, $s_0 = 1, s_1 = 2, s_2 = 2,\ s_3 = 1$, and $s_4 = s_5 = \cdots = 0$. One notes that

$$1 \cdot 1 \cdot 1 \cdot 2 \cdot 3 = 1 + 2 + 2 + 1 = s_0 + s_1 + s_2 + s_3.$$

Then by Eq. (23) above with $m = 5$, one obtains with $n = 1,\ 5 = y_1 + 2$, where $\alpha_{1,5} = \binom{5+1-1}{1} = 5$, hence $y_1 = 3$; with $n = 2,\ 15 = y_2 + 3 \cdot 2 + 1 \cdot 2$, where $\alpha_{2,5} = \binom{5+2-1}{2} = 15$, hence $y_2 = 7$; with $n = 3,\ 35 = y_3 + 7 \cdot 2 + 3 \cdot 2 + 1 \cdot 1$, where $\alpha_{3,5} = \binom{5+3-1}{3} = 35$, hence $y_3 = 14$; with $n = 4,\ 70 = y_4 + 14 \cdot 2 + 7 \cdot 2 + 3 \cdot 1 + 1 \cdot 0$, where $\alpha_{4,5} = \binom{5+4-1}{4} = 70$, hence $y_4 = 25$; with $n = 5,\ 126 = y_5 + 25 \cdot 2 + 14 \cdot 2 + 7 \cdot 1 + 3 \cdot 0 + 1 \cdot 0$, where $\alpha_{5,5} = \binom{5+5-1}{5} = 126$, hence $y_5 = 41$. Thus, $p_{\mathbf{M}}(1) = 3,\ p_{\mathbf{M}}(2) = 7,\ p_{\mathbf{M}}(3) = 14,\ p_{\mathbf{M}}(4) = 25$, and $p_{\mathbf{M}}(5) = 41$, etc.

Based on the explanation in Section 4, one easily check whether or not these computations are correct: Let us add, say $6 = 5 + 1$, which is strictly bigger than $n = 5$, to the old set $\mathbf{M}$ so that the new $\mathbf{M} =$

$\{1_{red}, 1_{blue}, 1_{green}, 2, 3, 6)$ and compute the s_i and the α_i. The expansion of the following polynomial in t

$$1 \cdot 1 \cdot 1 \cdot (1+t) \cdot (1+t+t^2) \cdot (1+t+t^2+t^3+t^4+t^5)$$

gives $s_0 = 1, s_1 = 3, s_2 = 5, s_3 = 6 = s_4 = s_5$. Thus, with $m = 6$, one obtains with $n = 1$, $\alpha_{1,6} = \binom{6+1-1}{1} = 6$, $6 = y_1 + 3$, hence $y_1 = 3$; with $n = 2$, $\alpha_{2,6} = \binom{6+2-1}{2} = 21$, $21 = y_2 + 3 \cdot 3 + 5$, hence $y_2 = 7$; with $n = 3$, $\alpha_{3,6} = \binom{6+3-1}{3} = 56$, $56 = y_3 + 3 \cdot 7 + 5 \cdot 3 + 6$, hence $y_3 = 14$; with $n = 4$, $\alpha_{4,6} = \binom{6+4-1}{4} = 126$, $126 = y_4 + 3 \cdot 14 + 5 \cdot 7 + 6 \cdot 3 + 6$; hence $y_4 = 25$; with $n = 5$, $\alpha_{5,6} = \binom{6+5-1}{5} = 252$, $252 = y_5 + 3 \cdot 25 + 5 \cdot 14 + 6 \cdot 7 + 6 \cdot 3 + 6$; hence $y_5 = 41$. Since one obtains the same value for $p_{\mathbf{M}}(j)$ as before, for $j = 1, 2, 3, 4, 5$, this time with a new $\mathbf{M}$, one concludes by Section 4 that no computational error has been introduced during computation of these values and consequently the results are all correct.

EXAMPLE 5. One notes that, for instance, the number of different ways of representing a given natural number n as a sum of squares of others is equal to $p_{\mathbf{M}}(n)$ with $\mathbf{M}$ consisting of those numbers of which squares are less than or equal to n. And all kinds of similar counting-related questions may be answered by my formula in 23 for $p_{\mathbf{M}}(n)$ for appropriate choices of $\mathbf{M}$. It is left to the reader to have his own examples in this direction.

5.2. The classical unrestricted partition function $p(n)$

Here is a very special but well-known situation with $\mathbf{M} = \{1, 2, 3, 4, \cdots\}$, where all numbers are required to have the same color, say, black. Although it still remains the possibility of having some computational errors in the values of the $p(n)$, it is nowadays not a big deal to compute the values of $p(n)$ based on Euler's formula, Eq. (1), for quite large integers n, thanks to rapid development in technology in recent three decades. In my later paper, I will discuss in detail how one can verify correctness of the values of $p(n)$ computed based on Euler's formula in a purely recursive manner.

On the other hand, this is the case which may be, in particular, thought as a worst situation in computation of $p_{\mathbf{M}}(n)$ using Eq. (23) when n gets very big, since the arithmetic invariants s_i become super big as n gets very big. This suggests there should be a further development of the theory under consideration. For more development in this direction, the reader may look at the last section of this paper.

For comparison purpose with the result of Example 4, one now considers the following easy example.

EXAMPLE 6. Let $\mathbf{M} = \mathbf{N} - \{0\} = \{1, 2, \cdots\}$ and one want to compute the value of the classical partition function $p(n)$ for n up to, say 3 first. Since every element of $\mathbf{M}$ is distinct, $p_{\mathbf{M}}(n) = p(n)$ this time. As no numbers bigger than 3 are used to represent the number 3, we may safely assume that $\mathbf{M} = \{1, 2, 3\}$; hence $m = 3$. Thus

$$1 \cdot (1 + t) \cdot (1 + t + t^2)$$

gives $s_0 = 1, s_1 = 2, s_2 = 2, s_3 = 1$, which happens to be the same as the first case of the previous Example. By Eq. (23) above with $m = 3$, one obtains with $n = 1$, $\alpha_{1,3} = \binom{3+1-1}{1} = 3$, $3 = y_1 + 2$, hence $y_1 = 1$; with $n = 2$, $\alpha_{2,3} = \binom{3+2-1}{2} = 6$, $6 = y_2 + 2 \cdot 1 + 2$, hence $y_2 = 2$; with $n = 3$, $\alpha_{3,3} = \binom{3+3-1}{3} = 10$, $10 = y_3 + 2 \cdot 2 + 2 \cdot 1 + 1$, hence $y_3 = 3$.

Now, in order to compute $p_{\mathbf{M}}(4)$, one adds the number 4 to the previous $\mathbf{M}$ so that the new $\mathbf{M} = \{1, 2, 3, 4\}$ and this time $m = 4$. With these, one expands

$$1 \cdot (1 + t) \cdot (1 + t + t^2) \cdot (1 + t + t^2 + t^3)$$

to obtain $s_0 = 1, s_1 = 3, s_2 = 5, s_3 = 6, s_4 = 5$. By Eq. (23) above with $m = 4$, one obtains with $n = 1$, $\alpha_{1,4} = \binom{4+1-1}{1} = 4$, $4 = y_1 + 3$, hence $y_1 = 1$; with $n = 2, \alpha_{2,4} = \binom{4+2-1}{2} = 10$, $10 = y_2 + 3 \cdot 1 + 5$, hence $y_2 = 2$; with $n = 3$, $\alpha_{3,4} = \binom{4+3-1}{3} = 20$, $20 = y_3 + 3 \cdot 2 + 5 \cdot 1 + 6$, hence $y_3 = 3$; with $n = 4$, $\alpha_{4,4} = \binom{4+4-1}{4} = 35$, $35 = y_4 + 3 \cdot 3 + 5 \cdot 2 + 6 \cdot 1 + 5$, hence $y_4 = 5$. Thus by Section 4, one concludes that no computational error has been introduced during computation of $p_{\mathbf{M}}(n)$ for n up to $n = 3$.

In order to check the correctness of $p_{\mathbf{M}}(4)$ with $\mathbf{M} =\{1, 2, 3, 4\}$, one simply adds 5 to the second $\mathbf{M}$ to obtain a new $\mathbf{M} =\{1, 2, 3, 4, 5\}$ and hence this time $m = 5$; with these one expands

$$1 \cdot (1 + t) \cdot (1 + t + t^2) \cdot (1 + t + t^2 + t^3)(1 + t + t^2 + t^3 + t^4)$$

to obtain $s_0 = 1, s_1 = 4, s_3 = 9, s_4 = 15, s_5 = 20, s_6 = 22$. Thus, $\alpha_{4,5} = \binom{5+4-1}{4} = 70$. Then by Eq. (24) above one obtains a system of linear equations in two unknowns with different positive integer slopes 3 and 4:

$$\begin{cases} 35 = y_4 + 3 \cdot y_3 + 5 \cdot 2 + 6 \cdot 1 + 5 \\ 70 = y_4 + 4 \cdot y_3 + 9 \cdot 2 + 15 \cdot 1 + 20 \end{cases},$$

where one already knows that $y_1 = 1, y_2 = 2$, and $y_3 = 3$. By the previous computation, one knows that $(3,5)$ is a solution of the first equation of this system of which correctness is now in question. Then Section 4 tells us that if $(3,5)$ does not satisfy the second equation of this system, the previously computed value $y_4 = 5$ is not correct. But in our situation, $(3,5)$ clearly satisfies the second equation, so that one concludes that the computation of $p_{\mathbf{M}}(4) = 5$ with $\mathbf{M} = \{1,2,3,4\}$ is correct.

Repeating the above process, one can not only compute the values of $p(n)$ but also checks correctness of its results up to n, which can be done without spending any other extra time and labor, based on Section 4.

5.3. Reverse applications

From these examples the reader should recognize that the only limit to the computation of $p_{\mathbf{M}}(n)$ based on my formula in Eq. (23) is computer's hardware and software ability to compute the first $(n+1)$ arithmetic invariants $s_0, s_1, \cdots, s_n$ of the set $\mathbf{M} = \{1, 2, \cdots, n\}$ for a given natural number n. Unfortunately, the reader may easily recognize by doing some computational experiments that this direction is not quite good from computational point of view, since computation of the first $n+1$ coefficients s_i of the polynomial is not that easy as n gets big, since direct expansion of the involved polynomial

$$1 \cdot (1+x) \cdot (1+x+x^2) \cdot \cdots \cdot (1+x+x^2+\cdots+x^{n-1})$$

requires time efficiency of $O(n^3)$ but with a very bad space efficiency. One has an easy

LEMMA 3. *For each i, the i-th coefficient s_i of the above polynomial is equal to the number of elements of the set*

$$\{i : i = \nu_1 + \nu_2 + \cdots + \nu_n,\ 0 \le \nu_i < i\}.$$

The proof of this lemma can be easily obtained by counting those terms in the polynomial above contributing to each i-th coefficient s_i of the polynomial.

In my later paper [15], a new elementary method of constructing a complete formula for a single s_i for $0 \le i \le n$, is presented. The method presented there is completely different from the well-known one which gives the same formula for s_i as my construction. One may apply this complete formula $n+1$ times separately to compute each s_i for $0 \le i \le n$, but since it is not recursive it takes a lot of time to compute all of them.

Another direction of application of Eq. (23) is the following one. Very recently the author has learned that R. P. Stanley proved

THEOREM 7. *The i-th coefficient s_i (which I call arithmetic invariant) of the above polynomial is equal to the number of permutations π in the group S_n of $n \times n$ permutation matrices having precisely i inversions.*

For the proof of the theorem, one may refer to Corollary 1.3.10 of [11].

On the other hand, the author of this paper proved (originally in [13]), independently and without knowing the Stanley's result, Theorem 5 of which a special version, the following corollary, is related to the Stanley's result, with a particular $\mathbf{M} = \{1, 2, 3, \cdots, n\}$:

COROLLARY 3. *For each i, the i-th coefficient s_i of the above polynomial is equal to the number of homogeneous basis elements of degree i of a basis of the free module $A = k[X_1, X_2, \cdots, X_n]$ of finite rank, $n!$, over its subring $B = k[f_1, f_2, \cdots, f_n]$, where A is the polynomial ring in n variables over an algebraically closed field k and the f_i are any homogeneous polynomials of positive degree i for each i such that the origin is their only common zero.*

In particular, if one replaces each f_r in the Corollary with the elementary symmetric polynomial

$$\sigma_i = \sum_{m_1 < m_2 < \cdots < m_i} X_{m_1} X_{m_2} \cdots X_{m_i},$$

one obtains a result upon which the above Stanley's result holds. For more about this, the reader may refer to Theorem 2.7.6 and around it in pp. 72-73 of [12]. What is missing in Stanley's result is about how to compute *all* the coefficients s_i of the above polynomial.

It is in the proof of this theorem that the author discovered another new means (both complete (Eq. (18) and recursive (Eq. (23)) to compute the values of the partition function $p(n)$ over a particular $\mathbf{M} = \{1, 2, \cdots, n\}$. As is slightly explained in the Appendix below, recursive computation of values of $p(n)$ utilizing Eq. (23) cannot be as efficient as the well-known Euler's recursive formula, because computation of the needed coefficients s_i for $0 \le i \le n$ of the above polynomial is not that easy. But, one should note that one can compute the coefficients s_i for $0 \leq i \leq n$, which appear in Stanley's theorem (Theorem 7), in a *recursive* manner, utilizing first

1. Euler's recursive formula for $p(n)$ and then
2. my formula in Eq. (23) in a reverse direction successively.

I call this direction of application of my formula, the **reverse application** of it. If one intends to examine the data pattern of the first $n+1$ of the s_i, then this way of *recursive* computation of them is very efficient. Since Euler's recursive formula only requires $O\ (n^{3/2})$ operations and my formula requires $O(n^2)$ operations, this provides an **efficient algorithm** with a polynomial complexity $O(n^2)$ to compute *all* those s_i for $0 \leq i \leq n$, showing up in the above Stanley's result. I am personally wondering if a more efficient algorithm for them (i.e., not just a single one) is even known to us.

The reader may try to compute those s_i, for instance, for $0 \leq i \leq n$ when $n = 5000$, with his computer (in my case, a Pentium PC with 16 Mbyte of main memory), utilizing any algorithms for it available to him, and compare their efficiency with computation based on the above reverse application of my recursive formula in Eq. (23), to see a big difference both in space and time efficiency between them.

The obvious problem with the above reverse application is that one cannot compute recursively, applying the above method, any of those s_i for which $i > n$, since Euler's formula is only available up to n, whereas one has to compute not only the first $n+1$ but *all* the coefficients of the above polynomial. To resolve this problem, one has to consider how one can efficiently compute all the values of the partition function $p_{\mathbf{M}}(j)$ of which components all belongs to the *fixed* set $\mathbf{M} = \{1, 2, \cdots, n\}$. In my later paper [15] based on [14], which is a natural generalization of this paper, this line of thought is very successfully examined from a more general point of view.

6. Problems

Finally, I would like to raise two problems related to graded partition functions $p_{\mathbf{M}}(n)$ with $\mathbf{M}$ finite or not.

PROBLEM 1. Establish an exact non-algebraic formula for $p_{\mathrm{M}}(n)$, like the one, Eq. (3), for the classical $p(n)$.

PROBLEM 2. Establish another exact algebraic formula for $p_{\mathrm{M}}(n)$ which is different from the one in Eq. (18).

The generating function of $p_{\mathrm{M}}(n)$ for the graded partition function is given by Eq. (4). For obvious reasons, answers to Problems 1 and 2 are important as it has been in the case of the classical unrestricted partition function $p(n)$.

7. Appendix (announcement of further results)

It is explained in this paper how one can relate a given graded partition function of which components belong to a multiset with the dimensions of subspaces of certain vector spaces, so that the nature of this article is rather theoretic. In this direction, one should note that my formulas Eq. (18) and Eq. (23) are suitable for the graded partition function over *any* multiset, and the classical unrestricted partition function $p(n)$ is just one of them. The built-in error-checking ability of my formula, Eq. (23), is theoretically quite remarkable since it is questionable if any other formulas in mathematics do have such a feature at all. Such an ability will be practically important if an algorithm based on my formula, Eq. (23), would turn out to be quite efficient (which is unfortunately not the case, though, when n gets big, for an obvious reason).

In the case of the classical unrestricted partition function $p(n)$, as Prof. George E. Andrews pointed out after reading out the first preprint of this paper, the efficiency of an **algorithm** based on the recursive formula, Eq. (23), cannot be as good as the one based on Euler's formula, Eq. (1), since expansion of the polynomial to obtain the arithmetic invariants s_i becomes a quite difficult job as n gets big.

In general, the efficiency question is less important when one originally establishes a formula like mine in Eq. (23). Nevertheless, once established, such a concern is very important for practical implementation of it, because the time efficiency, which is a main issue of the complexity theory, is an intrinsic property of a formula in general. This way of thinking is clearly a right motivation for a further investigation, which Prof. George E. Andrews also strongly suggested, to work on efficiency matters of my formula in Eq. (23). The result, [14], is a natural generalization of this paper from this point of view.

In my next paper [14], it is presented how one can develop *relative* formulas for a graded partition function with respect to another such one, which are generalization of the formulas in this paper, and the space and time efficiency of these generalized formulas are discussed especially in the case of the classical unrestricted partition function $p(n)$. In particular, it is also shown in there that the "direct enumeration" of the values of $p(n)$ from its generating function is nothing but a badly deformed version of my generalized formula with an unwisely chosen control parameter (meaning that it takes unnecessarily too much time with that number as a parameter). This will show that a version of my quasi-recursive formulas in [14] is an efficient generalization of the well-known method of direct enumeration of its generating function (Cf. Example 1). It is also fully explained in that paper how one can use the built-in error checking and/or correcting ability of my formulas of relative version.

It turned out that this generalized version is quite efficient. For instance, it took about 12.2 minutes to compute the values of $p(n)$ for $0 \leq n \leq 5001$, using Euler's formula Eq. (1). Then I was, using my formula which I decided to call quasi-recursive, able to check correctness of the value of the single $p(5001)$ only within 13.2 minutes, with a very reasonable assumption that the *correct* values of $p(n)$, for $0 \leq n \leq [\frac{5001}{2}]$, are given to us. Then by the nature of Euler's formula, this means that all the values of $p(n)$ for n up to 5001 are *all* correct. On the other hand, the method of "direct enumeration" as in Example 1 took about 80 minutes to compute the values of $p(n)$ for $0 \leq n \leq 5001$.

Computational results such as above suggest, especially among others, that it is not wise, unless n is very small, to use the method of "direct enumeration" of its generating function for a graded partition function $p_{\mathbf{M}}(n)$ for the purpose of error-checking and correction, when one *recursively* computes the values of $p_{\mathbf{M}}(n)$. My formula in [14], called *quasi-recursive*, should provide a much better means for this purpose. One should note that unless the multiset $\mathbf{M}$ is very special, for instance, $\mathbf{M} = \{1, 2, 3, \cdots\}$ or $\{1, 2, \cdots, n\}$, it is not quite possible to establish a complete, or exact formula for $p_{\mathrm{M}}(n)$ for a *single* input n, which is different from my formula in Eq. (18).

The above computational results were obtained under the following hardware and software limitations: a Computer Algebra System, a personal REDUCE 3.5, under Microsoft Windows 3.1 on my Pentium PC (a Gateway product of 1993) running at 60 MHz with 16 Mbyte of total main memory (with "memory size" 12.5 Mbyte and "stack size" 1 Mbyte, chosen at REDUCE's beginning prompt).

References

[1] G. E. Andrews, *Number Theory*, Sunders Co., 1975, pp. 149-198.
[2] ———, *The Theory of Partitions*, Addison-Wesley, Reading, 2nd printing, 1981.
[3] B. C. Berndt, *Ramanujan's Notebooks*, Part II, Springer-Verlag, 1989, pp. 305
[4] W. Fulton and J. Harris, *Representation Theory, A First Course,* GTM 129, Springer-Verlag, 1991.
[5] E. Grosswald, *Topics from the Theory of Numbers*, Birkhauser, 1984, pp. 109-140.
[6] G. H. Hardy and E. M. Wright, *An Introduction to the Theory of Numbers*, 5th Ed., Oxford, 1983, pp. 272-295.
[7] I. N. Herstein, *Topics in Algebra*, 2nd Ed., 1975, pp. 88-90.
[8] Math. Soc. of Japan, *Encyclopedic Dictionary of Mathematics* (English), 2nd Ed., 1987, Vol. 2, pp.1230-1232.
[9] I. Niven et al., *An Introduction to the Theory of Numbers*, 5th Ed., John Wiley & Sons, 1991, pp. 446-471.
[10] P. Shiu, *Computations of the partition function*, The Mathematical Gazette, Vol. 81, J. Math. Assoc., U.K., 1997, No. 490, pp. 45-52.
[11] R. P. Stanley, *Enumerative Combinatorics,* vol. I. Wadsworth & Brooks/Cole, Monterey, California, 1986.
[12] B. Sturmfels, *Algorithms in Invariant Theory*, Texts and Monographs in Symbolic Computation, Springer-Verlag, 1993.
[13] Sun T. Soh, *On a Complete Intersection,* RIM-GARC Preprint Series 95-15, Research Institute of Math., Global Analysis Research Center, Dept. of Math., Seoul National Univ., 1995.
[14] ———, *(Quasi-) Recursive Formulas for Graded Partition Functions,* preprint
[15] ———, *Efficient Formulas for the Coefficients of the Gaussian Polynomials* $[\frac{n}{m}]$, preprint

Department of Mathematics
MyongJi University
YongIn 449-728, Korea