Denumerants and Their Approximations

Petr Lisoněk
Research Institute for Symbolic Computation,
Johannes Kepler University, Linz
A–4040 Linz, Austria
e-mail: lisonek@risc.uni-linz.ac.at

ABSTRACT. Let a, b, c be fixed, pairwise relatively prime integers. We investigate the number of non-negative integral solutions of the equation $ax + by + cz = n$ as a function of n. We present a new algorithm that computes the "closed form" of this function. This algorithm is simple and its time performance is better than the performance of yet known algorithms. We also recall how to approximate the abovementioned function by a polynomial and we derive bounds on the "error" of this approximation for the case $a = 1$.

1. Definitions

In what follows, $\lfloor x \rfloor$ means the *integer part* of x and $\{x\} = x - \lfloor x \rfloor$ denotes the *fractional part* of x.

Let m be a positive integer and let $(a_1, \dots, a_m)$ be an m-tuple of positive integers. Let n be a non-negative integer. Each m-tuple of non-negative integers $(x_1, \dots, x_m)$ such that

$$\sum_{i=1}^{m} a_i x_i = n$$

is called a *partition* of the number n into parts of size $a_1, \dots, a_m$. For a given n, let $N(n; a_1, \dots, a_m)$ denote the number of all such partitions. Its generating function is

$$\sum_{n=0}^{\infty} N(n; a_1, \dots, a_m) t^n = \prod_{i=1}^{m} \frac{1}{1 - t^{a_i}}. \tag{1}$$

The number $N(n; a_1, \dots, a_m)$ is sometimes called the *denumerant* of n with respect to the sequence $(a_i)_{1 \le i \le m}$. ([1], p. 108.) We deal with the

JCMCC 18 (1995), pp. 225-232

problem of determining N as a function of n for certain sequences (a_i). This issue is also known as the *money changing problem* (when we consider n as an amount to be changed in coins or bills of size a_i).

In the present paper we extend several results from the paper [2] written by Tiberiu Popoviciu in early fifties. This work is quoted in most textbooks on combinatorial enumeration, such as [1] or [3]. While the abovementioned paper is more static (aiming at isolating denumerants with a certain property), we advance its results for dynamic purposes, namely for computing arbitrary denumerants with relatively prime parts.

2. Facts

We begin our investigations with recalling several known facts.

Fact 1. *Let k be a non-negative integer. With the notation as above, we have*

$$N_m(n+ka_m;a_1,\dots,a_m) - N_m(n;a_1,\dots,a_m) = \sum_{i=1}^{k} N_{m-1}(n+ia_m;a_1,\dots,a_{m-1}). \tag{2}$$

Proof: Consider the the following equation with unknowns $x_1,\dots,x_m$

$$a_1x_1+\ldots+a_mx_m = n+ka_m.$$

The solutions of this equation are of two types: *(i)* those with $x_m \geq k$, *(ii)* those with $x_m < k$. Each solution $(x_1,\dots,x_{m-1},x_m)$ of the type *(i)* is in a one-to-one correspondence with the non-negative solution

$$(y_1,\dots,y_m) = (x_1,\dots,x_{m-1},x_m-k)$$

of the equation

$$a_1y_1+\ldots+a_my_m = n.$$

Each solution of the type *(ii)* is in a one-to-one correspondence with the non-negative solution

$$(y_1,\dots,y_{m-1}) = (x_1,\dots,x_{m-1})$$

of the equation

$$a_1y_1+\ldots+a_{m-1}y_{m-1} = n+(k-x_m)a_m.$$

The formula (2) now follows by summation. □

In the present paper we study the denumerants in the case *when the part sizes a_i are pairwise relatively prime.* From the theory of rational generating functions it follows that $N_m(n) := N(n; a_1, \ldots, a_m)$ is then expressible in the nice form

$$N_m(n) = R_m(n) + G_m(n)$$

where R_m is a polynomial of degree $m-1$ in n whose coefficients are symmetric functions in the parameters a_i and G_m is a periodic sequence with the period $\prod_{i=1}^{m} a_i$. The coefficients of R_m for $m \leq 4$ can be found in [1], p. 113.

In the case $m = 1$ we have $R_1(n) = 1/a_1$ and $G_1(n) = -1/a_1 + 1$ or $-1/a_1$ according as a_1 divides or does not divide n.

For relatively prime numbers p, q, let the symbol

$$\left(\frac{n}{q} \mid p\right)$$

denote the unique integer $x \in \{0, \ldots, p-1\}$ such that

$$qx \equiv n \pmod{p}.$$

Fact 2. ([2], pp. 24–25.) *In the case $m = 2$ we have*

$$R_2(n) = \frac{n}{a_1 a_2}, \; G_2(n) = -\frac{1}{a_1}\left(\frac{n}{a_2} \mid a_1\right) - \frac{1}{a_2}\left(\frac{n}{a_1} \mid a_2\right) + 1. \quad (3)$$

□

The interesting cases are $m \geq 3$ where it becomes less trivial to determine the periodic part $G_m(n)$. The rest of this paper deals with the instance $m = 3$. For the sake of brevity, we will use the letters a, b, c instead of a_1, a_2, a_3. The polynomial part of the denumerant can be extracted from the formulas in [1], p. 113:

Fact 3. *Let a, b and c be pairwise relatively prime positive integers and let $N_3(n) := N_3(n; a, b, c)$ be the denumerant of n w.r.t. a, b, c. Then*

$$N_3(n) = R_3(n) + G_3(n)$$

where

$$R_3(n) = \frac{n(n+a+b+c)}{2abc}$$

and $G_3(n)$ is a periodic sequence with period abc. □

Fact 4. ([2], p. 38.) *With the notation introduced in Fact 3, let $r = abc - (a+b+c)$. For each $i = 1, 2, \ldots, a+b+c-1$ we have*

$$G_3(r+i) = \frac{i(a+b+c-i)}{2abc}.$$

□

3. Algorithms for Computing Denumerants

3.1. Known Methods

The traditional methods for computing denumerants are typically based on the partial fraction decomposition ([1], p. 109) which is costly.

In our restricted case when the part sizes a_i are pairwise relatively prime, we may observe that the periodic part G_m is expressible as a sum $\sum_{i=1}^{m} G^{(i)}$ where each $G^{(i)}$ is periodic with period a_i. Then we may set up a linear system for the unknowns $G^{(i)}(j)$, $1 \leq i \leq m$, $0 \leq j \leq a_i - 1$ ([1], p. 114). Solving this system by Gaussian elimination requires $O((\sum_{i=1}^{m} a_i)^3)$ elementary arithmetic operations (addition, subtraction, multiplication and division). Moreover, we need to compute the vector of right-hand sides for this linear system. To this end we must evaluate $N(n)$ at $(\sum_{i=1}^{m} a_i) - m$ contiguous points. This subgoal may further increase the total time complexity.

3.2. The New Algorithm

We now present a new algorithm that in certain situations requires many fewer steps compared to the methods mentioned earlier. We add the fifth "elementary" arithmetic operation in our computational model, namely the binary modulo function (mod). In the complexity analysis we will assume that all five operations are performed at the unit cost.

As before, let a, b, c be three fixed pairwise relatively prime positive integers and let $a \leq b \leq c$. Our goal is to compute $N_3(n; a, b, c)$ as a function of n. It should be noted that this problem actually includes *two* different tasks:

(I) Compute the "closed form" of N, i.e. obtain a representation of N that will allow us to evaluate $N(n)$ for any given n in a constant number of arithmetic operations.

(II) Evaluate $N(n)$ for *one* given n.

For every non-negative integer t we denote

$$g(t) = G_3(t + c) - G_3(t).$$

Lemma 1. *For any two non-negative integers k, l such that $k \equiv l \pmod{ab}$ we have*

$$g(k) = g(l).$$

Proof: Let $t \in \{k, l\}$. We have

$$G_3(t + c) - G_3(t) = N_3(t + c) - N_3(t) + R_3(t) - R_3(t + c).$$

From Facts 1, 2 and equation (3) we obtain

$$N_3(t+c) - N_3(t) = \frac{t+c}{ab} + G_2(t+c)$$
$$R_3(t) - R_3(t+c) = -\frac{t+c}{ab} - \frac{1}{2a} - \frac{1}{2b}.$$

Hence

$$\begin{aligned} g(t) &= G_3(t+c) - G_3(t) \\ &= -\frac{1}{a}\left(\frac{t+c}{b} \mid a\right) - \frac{1}{b}\left(\frac{t+c}{a} \mid b\right) + 1 - \frac{1}{2a} - \frac{1}{2b}. \end{aligned} \tag{4}$$

It is easily seen that $n' \equiv n'' \pmod p$ implies $(n'/q \mid p) = (n''/q \mid p)$, hence $k \equiv l \pmod{ab}$ implies $g(k) = g(l)$. □

Lemma 2. *Let $k \equiv l \pmod{ab}$ and let q be an integer. Then*

$$G_3(k+qc) - G_3(k) = G_3(l+qc) - G_3(l).$$

Proof: This is an easy consequence of Lemma 1. □

Lemma 3. *There are $(ab)^2$ rational numbers Δ_i^j, $0 \le i < ab$, $0 \le j < ab$ such that for any k we have*

$$G_3(i \cdot c + k) = G_3((ab-1) \cdot c + k) + \Delta_i^j \text{ whenever } k \equiv j \pmod{ab}.$$

Proof: Put $\Delta_i^j = G_3(i \cdot c + j) - G_3((ab-1) \cdot c + j)$. Using Lemma 2 we conclude that $k \equiv j \pmod{ab}$ implies $\Delta_i^j = G_3(i \cdot c+k) - G_3((ab-1) \cdot c+k)$. □

Lemma 3 is the basis for the following simple *algorithm* which computes $G_3(n_0)$ for given $0 \le n_0 < abc$:

1. Set $i_0 := \lfloor n_0/c \rfloor$.
2. Set $k_0 := n_0 \bmod c$.
3. Set $j_0 := k_0 \bmod ab$.
4. Evaluate $G_3((ab-1) \cdot c + k_0)$ by Fact 4.
5. Return $G_3(n_0) := G_3((ab-1) \cdot c + k_0) + \Delta_{i_0}^{j_0}$.

Lemma 4. *Let a, b, c be pairwise relatively prime positive integers. If $ab \ge c$ then the task* (I) *can be solved in time $O(abc)$. If $ab \le c$ then the task* (I) *can be solved in time $O((ab)^2)$.*

Proof: The values $\left(\frac{n}{a} \mid b\right)$ and $\left(\frac{n}{b} \mid a\right)$ for all residue classes of n can be identified by computing the values

$$az \bmod b \qquad (0 \leq z < b)$$

and

$$bz \bmod a \qquad (0 \leq z < a)$$

using $O(b)$ arithmetic operations (cf. the description of our computational model).

Then we can compute $g(t)$ for any t in constant time using (4). Now we employ two sets of equations

$$\Delta_0^j = g((ab-1)c+j) \tag{5}$$

and

$$\Delta_i^j = \Delta_{i-1}^j + g((i-1)c+j), \quad 1 \leq i < ab. \tag{6}$$

Using (5) and (6) we determine Δ_i^j for all indices in the range $0 \leq i < ab$ and $0 \leq j < \min(ab, c)$ in constant time per item. If $ab \geq c$ then we compute abc such values, if $ab \leq c$ then we need $(ab)^2$ values. Knowing these Δ_i^j allows us to evaluate $G_3(t)$ and hence also $N_3(t)$ for any t in constant time. □

Lemma 5. *Let a, b, c be pairwise relatively prime positive integers. Then the task* (II) *can be solved in time $O(ab)$.*

Proof: Again we start by computing the values $\left(\frac{n}{a} \mid b\right)$ and $\left(\frac{n}{b} \mid a\right)$ in $O(b)$ time. Now for any given t, we compute $G_3((ab-1)c + (t \bmod c))$ by Fact 4. Then we "jump" to the value $G_3(t)$ in at most $ab-1$ steps described by equation (4), in a constant time per each step. Actually at most $ab/2$ such steps are always sufficient since we can do the steps in both "directions".□

3.3. Comparison with Other Algorithms

From Lemma 4 it follows immediately that our algorithm for task (I) is asymptotically better than the linear system approach described in section 3.1 since the latter one needs at least order of c^3 operations if Gaussian elimination is used.

Task (II) is treated in [2], p. 27 with a formula which has time complexity $O(c)$. If $ab > c$ then this formula more effective while our approach (Lemma 5) is asymptotically better in the case $ab < c$.

We also have to emphasize the *simplicity* of our algorithms as they do not use any procedure other than the basic arithmetic (no linear systems, no partial fraction decompositions etc.).

4. Approximations

The main goal of the paper [2] was to determine all pairwise relatively prime triples (a, b, c) such that the denumerant $N(n) = N_3(n; a, b, c)$ is expressible as the floor of some polynomial $P(n)$, i.e. $N(n) = \lfloor P(n) \rfloor$. This is possible exactly if

$$\max_{0 \le i < abc} G(i) - \min_{0 \le i < abc} G(i) < 1. \tag{7}$$

For the sake of completeness we mention that there are 18 such triples (a, b, c) and all of them are listed in [2]. The equality $a = 1$ turns out to be a necessary condition for (7) to hold.

In our paper we extend these investigations by giving bounds on the values of $G(n)$ for all cases with $a = 1$, $(b, c) = 1$. Hence we give bounds on the "error" that may occur if the denumerant $N_3(n)$ is approximated by the polynomial $R_3(n)$.

Lemma 6. *Let b and c be pairwise relatively prime positive integers, $b < c$. For any non-negative n we have*

$$\frac{b+c+1}{2bc} - \frac{b}{8} \le N_3(n; 1, b, c) - R_3(n; 1, b, c) \le \frac{((b+c+1)/2)^2}{2bc} + \frac{b}{8}.$$

Proof: Recall that $\{x\}$ means the fractional part of x. From equation [4] it follows that

$$G_3(n + c; 1, b, c) - G_3(n; 1, b, c) = \frac{b-1}{2b} - \left\{\frac{n+c}{b}\right\}$$

for any n. For the rest of the paper, let $G(n)$ denote $G_3(n; 1, b, c)$. For any $1 \le k \le b$ we have

$$G(n + kc) - G(n) = k \cdot \frac{b-1}{2b} - \sum_{j=1}^{k} \left\{\frac{n+jc}{b}\right\}.$$

Let us examine the function

$$F(k) = k(b-1)/2 - \sum_{j=1}^{k} (n + jc) \bmod b.$$

One can write $F(k) = \sum_{j=1}^{k} f_j$ where

$$f_j = (b-1)/2 - (n + jc) \bmod b.$$

From $(b, c) = 1$ it follows that

$$\{f_j \mid 1 \le j \le b\} = \{-(b-1)/2, -(b-3)/2, \ldots, (b-3)/2, (b-1)/2\}.$$

Denote

$$f_- = \{f_j \mid f_j < 0\}, \quad f_+ = \{f_j \mid f_j > 0\}.$$

For any value of b we have

$$-\frac{b^2}{8} \leq \sum_{x\in f_-} x, \ \sum_{x\in f_+} x \leq \frac{b^2}{8}.$$

Hence,

$$-\frac{b^2}{8} \leq F(k) \leq \frac{b^2}{8}.$$

Incidentally, these bounds are indeed achieved for certain choices of n, b, c and k. (The proof is left as an exercise.) Coming back to the definition of $F(k)$, we see that

$$-\frac{b}{8} \leq \Delta_i^j \leq \frac{b}{8}$$

for all i, j. By Fact 4 we have

$$\frac{b+c+1}{2bc} \leq G(n) \leq \frac{((b+c+1)/2)^2}{2bc}$$

for all $bc-(b+c) \leq n \leq bc-1$. The rest follows from Lemma 3. □

Remark: A slight refinement of the last lemma can be achieved by splitting it in three statements according to the parity of b and c.

References

[1] L. Comtet, *Advanced Combinatorics.* D.Riedel, Dodrecht, 1974.

[2] T. Popoviciu, Asupra unei probleme de partiţie a numerelor, *Acad. R.P.R., Filiala Cluj, Studie şi cercetari ştiintifice* 4 (1953), 7–58.

[3] R.P. Stanley, *Enumerative Combinatorics. Volume I.* Wadsworth, Monterey, 1986.