ELSEVIER

Available at
www.ElsevierMathematics.com
POWERED BY SCIENCE DIRECT

Discrete Mathematics 275 (2004) 17–41

DISCRETE MATHEMATICS

www.elsevier.com/locate/disc

Algorithmic approach to counting of certain types m-ary partitions

Valentin P. Bakoev

Faculty of Pedagogy, Department of Mathematics and Informatics, University of Veliko Tarnovo, 2 Theodosi Tarnovski Street, Veliko Tarnovo 5000, Bulgaria

Received 6 May 1999; received in revised form 27 January 2003; accepted 17 February 2003

Abstract

Partitions of integers of the type m^n as a sum of powers of m (the so-called m-ary partitions) and their counting is considered in this paper. Two algorithms for counting of m-ary partitions of sums, where each addend is m^n, are developed. On the base of these algorithms some arithmetical and combinatorial properties, and also polynomial form representations of the number of such partitions are derived. An algorithm with a polynomial running time, which produces the coefficients of this polynomial and next computes the number of considered partitions, is proposed. Two applications, concerning counting problems of special types of m-ary trees and partitions of the Boolean cube, are given.

Keywords: m-ary partition algorithm; Recurrence table; Algebraic and combinatorial property; Full m-ary tree; Boolean cube partition

1. Introduction

Churchhouse denotes by $b(n)$ the number of ways of expressing the natural number n as a sum of powers of 2 and he calls this function "*the binary partition function*" [7]. He refers to Euler, Tanturri, Mahler and others, which have investigated this function (formulae (1)–(3) below for $m=2$ are invented by Euler). Churchhouse is one of the first, who uses a computer to calculate $b(n)$ for $0 \leqslant n \leqslant 200$ and to prove some identities and congruence properties of the binary partition function. These are improved, generalized and some new properties are obtained in [2,3,13,17]. Rödseth [17] denotes

E-mail address: v_bakoev@yahoo.com (V.P. Bakoev).

0012-365X/$ - see front matter
doi:10.1016/S0012-365X(03)00096-7

by $t_m(n)$ the number of partitions of the positive integer n into non-decreasing addends, which are positive or zero powers of a given natural number $m > 1$ and he calls $t_m(n)$ "*the m-ary partition function*". He takes $t_m(0) = 1$ and puts $F_m(x) = \sum_{n=0}^{\infty} t_m(n)x^n$ for $|x| < 1$. Then the next four equalities are valid [7,13,17]:

$$F_m(x) = \prod_{k=0}^{\infty} (1 - x^{m^k})^{-1} \tag{1}$$

and therefore $F_m(x)$ satisfies the functional equation

$$(1 - x)F_m(x) = F_m(x^m). \tag{2}$$

Hence

$$t_m(n) = t_m(n-1) + t_m\left(\left\lfloor \frac{n}{m} \right\rfloor\right) \tag{3}$$

and

$$t_m(mn + k) = t_m(mn) \quad \text{for } 0 \leqslant k < m. \tag{4}$$

A few years ago, studying the basic problems SAT and 3-SAT in NP-Completeness theory [10], we reached to the problem for partition of Boolean cube into subcubes with certain dimensions and their counting [5,6]. It turned out that this problem concerns partitions of the integers of the type 2^n as a sum of powers of 2 and their counting [4].

Here we consider partitions of integers of the type

$$m^n = m^{i_1} + m^{i_2} + \cdots + m^{i_k}, \quad i_j \geqslant 0 \text{ for } j = 1, 2, \ldots, k, \tag{5}$$

where m and n are natural numbers, $m > 1, n > 0$. More precisely, we consider m-ary partitions of sums of equal addends, each of them being m^n, into powers of m. In Section 2 we develop two dynamic-programming algorithms for computing the number of partitions of such sums. Some arithmetical and combinatorial properties of the partition function are derived in Section 3. Section 4 is devoted to polynomial form representation of the number of these partitions. The explicit form of such polynomials, for small n ($1 \leqslant n \leqslant 5$), is deduced. Another algorithm with $\Theta(n^3)$ running time, which computes the number of considered partitions (for arbitrary n) by deriving the coefficients of the corresponding polynomial, is proposed. Two applications are given in Section 5.

2. Algorithms for counting

Further m, n and k will always be natural numbers, $n > 0$, $m > 1$.

Definition. Partition of the type (5), which consists of exactly k ($k > 0$) addends is said to be a k-partition [2,15]. When $k = 1$ we call the partition $m^n = m^n$ a trivial partition. All other k-partitions (where $k > 1$) we call non-trivial partitions.

Following [17], we consider each partition of the integer m^n of type (5) as a combination with repetitions, whose addends form monotonically increasing sequence:

$$m^n = m^{i_1} + m^{i_2} + \cdots + m^{i_k}, \quad 0 \leqslant i_1 \leqslant i_2 \leqslant \cdots i_k < n, \quad \text{when } k > 1$$

or $i_1 = n$ when $k = 1$.

We rewrite equality (4) in the form

$$t_m(mn-1) = t_m(mn-2) = \cdots = t_m(m(n-1)) \tag{6}$$

and replace it in (3):

$$t_m(mn) = t_m(m(n-1)) + t_m(n). \tag{7}$$

By repeating the substitution of (3) and (6) in (7) (as for $m = 2$ in [7]) we obtain

$$t_m(mn) = \sum_{i=0}^{n} t_m(i). \tag{8}$$

We could not find or derive the solutions to the last recurrences in a closed form. But these recurrences and the initial condition $t_m(0) = 1$ determine an algorithm for computing the values of $t_m(mn)$ for given m and n. We call it briefly *Algorithm A*. Its Maple code is shown in [18]—for deriving the sequences $A000123$ (partitions of $2n$ into powers of 2) and $A002577$ (partitions of 2^n into powers of 2). For the sequences $A005704$ (partitions of $3n$ into powers of 3), $A005705$ (partitions of $4n$ into powers of 4) and $A005706$ (partitions of $5n$ into powers of 5) generating functions, recurrences or Maple codes are not given.

As it is shown in [9,18] and as we shall see later, the m-ary partition function grows extremely fast. An exact estimation of the time-complexity of Algorithm A and the following algorithms must be based on the *logarithmic cost criterion* [1], because the intermediate results on each step and the final result (output) have exponentially increasing size towards the size of the input m and n. This requires an algorithm for long-integer arithmetic to be implemented and correct estimation of the time-complexity of its operations (as a function of the data-size on each step) to be done. This problem is out of the ideas of this paper. Instead of this we focus our attention only on the number and the type of the arithmetical operations, which the given algorithm performs. So we accept the *uniform cost criterion* [1], which is realistic for small values of the input, such that one computer word is enough to store the output. And for the opposite case we have reasons to consider that the conclusions we made, comparing the time-complexities of the algorithms, will also be valid.

Input of Algorithm A are integers n and m, its output is the value of $t_m(mn)$. We wrote several versions of a non-recursive program, performing Algorithm A and we tried to improve each of them. Finally we obtained a program, which computes the value of $t_m(mn)$ by performing $\lfloor \frac{n}{m} \rfloor$ additions and the same number integer divisions (for computing indices). We accept that this is the minimal possible number of operations for a program, based only on the last recurrences. We assume that the addition, the integer division and the storing of the result require a unit time: respectively $c_1 = \text{const}$, $c_2 = \text{const}$ and $c_3 = \text{const}$, independently of the size of the operands. Then the time-complexity of Algorithm A is $(c_1 + c_2 + c_3)\lfloor \frac{n}{m} \rfloor = \Theta(\lfloor \frac{n}{m} \rfloor)$.

Instead of counting the partitions of type (5), we consider more general problem: *counting all partitions of the sum* $\underbrace{m^n + m^n + \cdots + m^n}_{k}$, $k \geqslant 0$ *into powers of* m, *not exceeding* m^n. Further we use the brief notation $k.m^n$ for this sum. Following the steps of dynamic-programming method in [8], we develop two dynamic-programming algorithms for counting these partitions in dependence of their type.

2.1. First case: all partitions of the sum $k.m^n$ into powers of m, where each addend is less than m^n

In the first case we consider all partitions of the sum $k.m^n$ into powers of m, which do not contain any trivial partitions of the addends. We call all these partitions briefly *non-trivial partitions of the sum* $k.m^n$ *into powers of* m. We denote their number by $\tilde{t}_m(n,k)$, or by $\tilde{t}(n,k)$ when we talk about a fixed integer m.

2.1.1. Characterizing the number of the partitions

It is natural to assume that the addends must be of maximal possible degrees in the first partition and in each of the next partitions—otherwise we shall omit some of them. The following assertion is trivial (and we omit its proof), but we formulate it as a Lemma in order to refer to it easily.

Lemma 1. *The unique non-trivial partition of the integer* m^n *as a sum of addends, where each addend is of maximal possible degree, is the m-partition* $m^n = m.m^{n-1}$.

Following Lemma 1, $k.m^n = (k-1).m^n + m.m^{n-1}$ and hence the solution of the original problem contains within it the solutions of subproblems (with smaller sizes). So the problem exhibits *optimal substructure property*—the first key ingredient for dynamic-programming method to be applicable.

2.1.2. A recursive solution

By the following theorem we determine the values of $\tilde{t}(n,k)$, recursively.

Theorem 1. *For given natural numbers* n, m *and* k *the next recurrence holds*:

$$\tilde{t}(n,k) = \begin{cases} 1 & \text{if } n = 1 \text{ or } k = 0, \\ \tilde{t}(n,k-1) + \sum_{j=1}^{m} \tilde{t}(n-1,(k-1)m+j) & \text{if } n > 1 \text{ and } k > 0. \end{cases} \tag{9}$$

Proof. (1) For $n = 1$ and for any k the assertion follows directly from Lemma 1. For $n \geqslant 1$, the unique representation $k.m^n = (km).m^{n-1}$ is valid for any k and, formally, for $k = 0$—so we put $\tilde{t}(n,0) = 1$.

(2) Let $n > 1$ and $k > 0$. Lemma 1 implies

$$k.m^n = (k-1).m^n + m^n = (k-1).m^n + m.m^{n-1}.$$

We consider the last m addends. We fix some of them and we partition the rest—consecutively, from left to right, so that each time to obtain a monotonically increasing sequence of addends. The number of the fixed addends varies from m to 0. Depending on this we split all partitions of $k.m^n$ into $m+1$ groups which do not intersect each other.

The first group includes all partitions, whose last m addends are fixed to m^{n-1} and each other addend is less or equal to m^{n-1}. Since $k.m^n=(k-1).m^n+m.m^{n-1}$, the first group contains $\tilde{t}(n,k-1)$ partitions generally. In the second group are all partitions, in which the last $m-1$ addends are exactly m^{n-1} and each other addend is less than m^{n-1}. Since $k.m^n=(k-1).m^n+m.m^{n-1}=((k-1)m+1).m^{n-1}+(m-1).m^{n-1}$, there are $\tilde{t}(n-1,(k-1)m+1)$ partitions in the second group, and so on. The partitions in the last group do not contain any fixed addends, all addends are less than m^{n-1}. Following the equality $k.m^n=((k-1)m+m-1).m^{n-1}+m^{n-1}=(km).m^{n-1}$, their number is $\tilde{t}(n-1,km)$.

So the number of all non-trivial partitions of the sum $k.m^n$ into powers of m is

$$\tilde{t}(n,k)=\tilde{t}(n,k-1)+\tilde{t}(n-1,(k-1)m+1)$$
$$+\tilde{t}(n-1,(k-1)m+2)+\cdots+\tilde{t}(n-1,km). \qquad \square$$

2.1.3. Computing the value of the solution in a bottom-up manner

It is easy to write a recursive algorithm, based on recurrence (9), which computes $\tilde{t}(n,k)$ for given m, n and k. Obviously it will compute first $\tilde{t}(n,k-1)$, after that $\tilde{t}(n-1,(k-1)m+1)$, and so on, finally $\tilde{t}(n-1,km)$. In accordance with recurrence (9), the computing of $\tilde{t}(n-1,km)$ will cause a recomputing of $t(n-1,km-1)$, next a recomputing of $\tilde{t}(n-1,km-2)$, and so on. We observe that the recursive algorithm computes the same subproblems over and over again,[1] i.e. the problem exhibits *overlapping subproblems property*—the second key ingredient to apply a dynamic-programming strategy. That is why we develop our algorithm in a bottom-up approach. We use an auxiliary table $\tilde{T}$ with n rows, numbered $1,2,\ldots,n$. Each integer $\tilde{t}(i,j)$ we store in the cell $\tilde{T}[i,j]$ of the table. Let us compute the number of its columns. Since $\tilde{t}(i,0)=1$ for $1\leqslant i\leqslant n$, the zeroth column will contain ones. Excepting the zeroth cell, we must fill in the values of $\tilde{t}(n,1),\tilde{t}(n,2),\ldots,\tilde{t}(n,k)$ in the last row. The rest of its cells are not essential and they remain empty (or not used). To compute and fill in the numbers in the nth row, we have to know $1+km$ numbers from the corresponding cells of the $(n-1)$st row (the other its cells remain empty again), and so on. In this way we can prove inductively that in the ith row $(1\leqslant i\leqslant n)$ we must fill in the numbers $\tilde{t}(i,j)$, $0\leqslant j\leqslant km^{n-i}$, i.e. $1+km^{n-i}$ numbers generally. So the first row of the table must contain $km^{n-1}+1$ numbers, which are ones, since $\tilde{t}(1,j)=1$ for $j\geqslant 0$. Therefore the table $\tilde{T}$ will have $1+km^{n-1}$ columns and we number them $0,1,\ldots,km^{n-1}$ in correspondence with the values of j.

[1] This is also valid for a similar recursive program, which performs Algorithm A.

Table 1
$\tilde{T}_{2,5,1}$

$i \setminus j$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	3	5	7	9	11	13	15	17								
3	1	9	25	49	81												
4	1	35	165														
5	1	201															

Table 2
A part of $\tilde{T}_{3,5,1}$

$i \setminus j$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	…
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	…
2	1	4	7	10	13	16	19	22	25	28	31	34	37	40	43	…
3	1	22	70	145	247	376	532	715	925	1162						…
4	1	238	1393	4195												…
5	1	5827														…

The following algorithm for computing the value of the solution in a bottom-up manner we call *Algorithm B.*

Algorithm B. Computing the number of all non-trivial partitions of the sum $k.m^n$ into powers of m

Input: m, n and k.

Output: the value of $\tilde{t}(n,k)$.

Procedure: (1) Define a table $\tilde{T}$ with n rows and $km^{n-1}+1$ columns.

(2) Fill in all its cells from the first row and from the zeroth column with ones.

(3) Start with the first cell of the second row, and for each next row, move from the first cell to the rightmost cell to be filled in (it has a number km^{n-i} in the ith row). Compute the serial value of $\tilde{t}(i,j)$ (in accordance with Eq. (9)) and store it in the cell $\tilde{T}[i,j]$.

(4) Return the value of $\tilde{T}[n,k]$. End.

When m, n and k are arbitrary natural numbers, we denote the corresponding table by $\tilde{T}_{m,n,k}$. Table 1 shows $\tilde{T}_{2,5,1}$, Table 2 a part of $\tilde{T}_{3,5,1}$.

2.2. Second case: all partitions of the sum $k.m^n$ into powers of m

This case is more general and it is similar to the first one. We consider all partitions of the sum $k.m^n$ into powers of m, where each addend is less or equal to m^n, i.e. trivial partitions of the addends in the sum are allowed. We denote the number of these partitions by $t_m(n,k)$, or by $t(n,k)$ when m is a fixed integer.

2.2.1. Characterizing the number of the partitions

Analogously to the first case, Lemma 1 implies the existence of the optimal substructure property.

2.2.2. A recursive solution

The recursive solution to the problem is given in the following theorem.

Theorem 2. *For given natural numbers n, m and k the next recurrence holds*:

$$t(n,k)=\begin{cases} k+1 & \text{if } n=1,\\ 1 & \text{if } k=0,\\ t(n,k-1)+t(n-1,km) & \text{if } n>1 \text{ and } k>0. \end{cases} \tag{10}$$

Proof. (1) When $n=1$, the sum $k.m^1$ has one trivial partition and also each addend of the sum can be partitioned only once (Lemma 1). So $t(1,k)=1+k$.

(2) The sum $k.m^n$ has unique trivial partition $k.m^n=k.m^n$ for any n and k. Formally, this equality is valid for $k=0$ and we put $t(n,0)=1$.

(3) Let $n>1$ and $k>0$. We split all partitions of the sum $k.m^n$ into two non-intersecting each other groups.

The partitions in the first group contain some trivial partitions of the addends, i.e. at least one addend is equal to m^n. We consider all partitions of the sum $(k-1).m^n$ into powers of m and we add m^n in the end of each them. Thus we obtain all partitions of the first group and so their number is $t(n,k-1)$.

The partitions in the second group do not contain any trivial partitions of the addends, i.e. the group consists of all partitions from the first case. Lemma 1 implies $k.m^n=(km).m^{n-1}$ and hence the second group contains $t(n-1,km)$ partitions.

Therefore $t(n,k)=t(n,k-1)+t(n-1,km)$. □

2.2.3. Computing the value of the solution in a bottom-up manner

It is obvious that the problem exhibits overlapping subproblems property. On the base of recurrence (10), we develop an algorithm in a bottom-up approach again. We use an auxiliary table T with n rows, numbered $1,2,\ldots,n$ and we store each integer $t(i,j)$ in the cell $T[i,j]$. Analogously to the previous case, we can prove inductively that in the ith row ($1\leqslant i\leqslant n$) of T we must store the values of $t(i,0),t(i,1),\ldots,t(i,km^{n-i})$. So the table T will have $1+km^{n-1}$ columns, numbered $0,1,\ldots,km^{n-1}$.

The following algorithm computes the value of the solution in a bottom-up manner. We call it *Algorithm C*.

Algorithm C. Computing the number of all partitions of the sum $k.m^n$ into powers of m

Input: m, n and k.

Output: the value of $t(n,k)$.

Procedure: (1) Define a table T with n rows and $km^{n-1}+1$ columns.

Table 3
A part of $T_{2,5,2}$

$i \setminus j$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	...
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	...
2	1	4	9	16	25	36	49	64	81	100	121	144	169	196	225	...
3	1	10	35	84	165	286	455	680	969							...
4	1	36	201	656	1625											...
5	1	202	1827													...

Table 4
A part of $T_{3,5,2}$

$i \setminus j$	0	1	2	3	4	5	6	7	8	9	10	11	12	...
1	1	2	3	4	5	6	7	8	9	10	11	12	13	...
2	1	5	12	22	35	51	70	92	117	145	176	210	247	...
3	1	23	93	238	485	861	1393	2108	3033	4195	5621	7338	9373	...
4	1	239	1632	5827	15200	32856	62629							...
5	1	5828	68457											...

(2) Filling in the first row: for each j, $1 \leqslant j \leqslant km^{n-1}$ store the number $j+1$ in the cell $T[1,j]$.

(3) Filling in the zeroth column: store ones in all its cells.

(4) Filling in the other essential cells: start with the first cell of the second row, and for each next row, move from the first cell to the rightmost cell to be filled in (it has a number km^{n-i} in the ith row). Compute the serial value of $t(i,j)$ (in accordance with recurrence (10)) and store it in the cell $T[i,j]$.

(5) Return the value of $T[n,k]$. End.

When we talk about arbitrary m, n and k, we denote the corresponding table by $T_{m,n,k}$. Table 3 shows a part of $T_{2,5,2}$, Table 4 a part of $T_{3,5,2}$.

Remark 1. For the tables $T_{m,n,k}$ and $\tilde{T}_{m,n,k}$, the functional relation between the number of the row i, $1 \leqslant i \leqslant n$, and the number of its cells r to be filled in, is $r = km^{n-i}$, i.e. it is of exponentially decreasing type. Namely, the rows of the tables have exponentially decreasing length and time, necessary for their filling in.

We shall compute the time-complexities of the Algorithms B and C (towards the accepted uniform cost criterion) for given input m, n and k.

Algorithm B stores $km^{n-1}+n$ ones in the first row and in the zeroth column of the table $\tilde{T}$. The algorithm has to compute and store generally $k(m^{n-2}+m^{n-3}+\cdots+m^1+m^0) = k(m^{n-1}-1)/(m-1)$ numbers in the rest rows of $\tilde{T}$. Computing each of these numbers requires m additions to be performed and therefore Algorithm B has time-complexity $c_3(km^{n-1}+n)+(c_1+c_3)km(m^{n-1}-1)/(m-1) = \Theta(km^{n-1})$.

Algorithm C stores n ones in the first column of the table T and it also computes and stores $km^{n-1}, km^{n-2}, \ldots, km^0$ numbers, respectively in the first, second, $\ldots, n$th row. Computing each of these numbers requires only one addition. Generally we have $k(m^{n-1}+m^{n-2}+\cdots+m^0)=k(m^n-1)/(m-1)$ additions and hence the time-complexity of Algorithm C is $c_3 n + (c_1 + c_3)km^{n-1} = \Theta(km^{n-1})$.

3. Properties of the numbers from the tables

Filling in the tables $\tilde{T}_{m,n,k}$ and $T_{m,n,k}$ quite resembles the filling in other tables, based on recurrences—Pascal's, Stirling's and Euler's triangles. Similar tables are also the so-called *arithmetical triangles*, *arithmetical squares* and their generalizations, considered in [19]. In the arithmetical square the zeroth row and the zeroth column contain ones and each of the other cells contains a number, which is sum of the numbers in the left cell and in a certain cell above. The numbers from arithmetical triangles and squares are related to:

- various combinatorial problems for moving chess-men or other pieces on a chessboard;
- the so-called "*figurate numbers*" (triangular, quadratic, and so on; pyramidal). The study of them has been set by Pythagoras (Pυθαγορας) and Nickomach (Νικομαχος), and
- the Pascal's triangle in its varieties.

The numbers, stored in the tables $\tilde{T}_{m,n,k}$ and $T_{m,n,k}$, have some interesting properties. They are similar to the properties of the Pascal's, Stirling's, and Euler's numbers [11] and to the properties of the numbers in the arithmetical triangles and squares [19]—as it can be expected. These properties are corollaries of the given definitions and assertions. So we shall formulate them as corollaries. We consider the tables $\tilde{T}$ and T, obtained by a performance of Algorithms B and C for given natural numbers m, n and k.

From considered above cases, it follows immediately:

Corollary 1. *The number of the non-trivial partitions of the number m^n of type* (5) *is equal to $\tilde{t}(n,1)$, defined by recurrence* (9). *The number of all partitions of the same number and of the same type is equal to $t(n,1)$, defined by recurrence* (10).

Algorithms B and C compute the value of $t_m(m^n) = t(n,1) = T[n,1] = \tilde{t}(n,1) + 1 = \tilde{T}[n,1] + 1$ with one and the same time-complexity $\Theta(m^{n-1})$. Algorithm A computes the value of $t_m(m^n)$ with better time-complexity $\Theta(m^{n-2})$.

The following Table 5 is obtained by a performance of Algorithm C (in which the addition is over 12 bytes long computer words) four times: for n=9 and for m=2, 3, 5, 7. It represents the numbers from the first column of the corresponding tables $T_{m,9,1}$, i.e. the values of $t_m(n,1)$ for $1 \leqslant n \leqslant 9$.

As it was defined, the value of the cell $T[i,j]$ represents the number of all partitions of the sum $j.m^i$ for $1 \leqslant i \leqslant n$. Thereto we add:

Table 5

$n \setminus m$	2	3	5	7
1	2	2	2	2
2	4	5	7	9
3	10	23	82	205
4	36	239	3707	24901
5	202	5828	642457	16077987
6	1828	342383	446020582	58169810617
7	27338	50110484	1288155051832	1226373476385199
8	692004	18757984046	15905066118254957	154912862345527456431
9	30251722	18318289003448	856874264098480364332	11967977955077323244243580

Corollary 2. *The value of the cell $T[i,j]$ is equal to the number of all partitions of the sum $j.m^l$ into powers of m, where each addend is greater or equal to m^{l-i}, for $1 \leqslant i \leqslant l \leqslant n$.*

Proof. We consider all partitions of the sum $j.m^i$ and we juxtapose to each of them a partition of the sum $j.m^l$, where each addend is greater or equal to m^{l-i}. The second type partitions we obtain by multiplying the both sides of each equality, representing the first type partition, by m^{l-i}. Obviously this juxtaposing is bijective and therefore the number of these two types partitions is one and the same, namely equal to $T[i,j]$. □

Analogously, Corollary 2 can be formulated for the number of the non-trivial partitions, stored in the cell $\tilde{T}[i,j]$.

We already know that:

$\tilde{T}[1,j] = 1$ for $j = 0,1,2,\ldots,km^{n-1}$,

$\tilde{T}[i,1] = 1$ for $i = 1,2,\ldots,n$,

$\tilde{T}[i,j] = \tilde{T}[i,j-1] + \sum_{l=1}^{m} \tilde{T}[i-1,(j-1)m+l]$ for $1 < i \leqslant n,\ 1 \leqslant j \leqslant km^{n-i}$.

Tables whose cells contain numbers, related between each other with dependencies like given above are called *recurrence tables* in [19]. If $j > 1$, we replace $\tilde{T}[i,j-1]$ by $\tilde{T}[i,j-2] + \sum_{l=1}^{m} \tilde{T}[i-1,(j-2)m+l]$, next we replace $\tilde{T}[i,j-2]$, and so on until we reach to $\tilde{T}[i,0] = 1$. Finally we obtain:

Corollary 3. $\tilde{T}[i,j] = 1 + \sum_{l=1}^{mj} \tilde{T}[i-1,l] = \sum_{l=0}^{mj} \tilde{T}[i-1,l]$ *for* $1 < i \leqslant n$.

Remark 2. The assertion of Corollary 3 for $m = 2$ is proved by Churchhouse [7]. He represents a rectangular table, which absolutely corresponds to the table $\tilde{T}_{2,5,1}$, given above. Generalizing Churchhouse's Theorem 1 [7], Gupta shows [13] that for $m > 2$, $t_m(m^r.n) = \sum_{j=0}^{n} C_r(j) t_m(n-j)$, where the coefficient $C_{r+1}(j) = \sum_{i=0}^{mj} C_r(i)$ and $C_1(j) = 1$ for all $j \geqslant 0$. This is another property of the numbers from the table $\tilde{T}_{m,n,k}$, obtained by using analytical techniques. We clarify the meaning of the coefficients $C_i(j)$—these are the numbers from $\tilde{T}$.

We also know that:

$T[1,j] = j + 1$ for $j = 0,1,2,\ldots,km^{n-1}$,

$T[i,1]=1$ for $i=1,2,\ldots,n$,
$T[i,j]=T[i,j-1]+T[i-1,jm]$ for $1<i\leqslant n,\ 1\leqslant j\leqslant km^{n-i}$.

If $j>1$, we replace $T[i,j-1]$ by $T[i,j-2]+T[i-1,(j-1)m]$, next we replace $T[i,j-2]$, and so on until we reach to $T[i,0]=1$. So we obtain:

Corollary 4. $T[i,j]=1+\sum_{l=1}^{j}T[i-1,lm]=\sum_{l=0}^{j}T[i-1,lm]$ *for* $1<i\leqslant n$.

There is a relation between the tables T and $\tilde{T}$ for given m,n and k, as well as the partitions from the first case are part of the partitions from the second case. From the proof of the Theorem 2 it follows immediately:

Corollary 5 (Relation between the tables $\tilde{T}$ and T). (a) $\tilde{T}[i,j]=T[i-1,jm]$ *and* $T[i,j]=T[i,j-1]+\tilde{T}[i,j]$ *for* $1<i\leqslant n$, *and*

(b) *if we ignore the first row of the table* $\tilde{T}_{m,n,k}$, *then each of its column with number* j $(0\leqslant j\leqslant km^{n-2})$ *coincides with the column with number* jm *of the table* $T_{m,n-1,k}$.

The numbers from the first row of $\tilde{T}$ form an arithmetic series of order 0. The numbers from the first row of T and these from the second row of $\tilde{T}$ form arithmetic series of order 1. The figurate (or more exactly the polygonal) numbers form an arithmetic series of second order. For a fixed integer $q\geqslant 3$, the nth q-gonal number is given by the well-known formula: $P_n^q=n+(q-2)n(n-1)/2,\ n=1,2,\ldots$.

Corollary 6 (Relation between the tables and the polygonal numbers). (a) *If* $n>2$, *the first* $(m+2)$*-gonal numbers take place in the second row of the table* $T_{m,n,k}$, *i.e.* $T_{m,n,k}[2,j]=P_{j+1}^{m+2}$ *for* $j=0,1,\ldots,km^{n-2}$, *and*

(b) *If* $n>3$, *certain type* $(m+2)$*-gonal numbers take place in the third row of the table* $\tilde{T}_{m,n,k}$, *namely* $\tilde{T}_{m,n,k}[3,j]=P_{mj+1}^{m+2}$ *for* $j=0,1,\ldots,km^{n-3}$.

Proof. (a) $T_{m,n,k}[1,j]=1+j$ for $j=0,1,\ldots,km^{n-1}$. For the second row of the table we have $T_{m,n,k}[2,0]=1$ and for $j=1,2,\ldots,km^{n-2}$, Corollary 4 implies:

$$T_{m,n,k}[2,j]=\sum_{l=0}^{j}T_{m,n,k}[1,lm]=\sum_{l=0}^{j}(1+lm)=j+1+mj(j+1)/2.$$

Therefore $T_{m,n,k}[2,j]=P_{j+1}^{m+2}$ for $j=0,1,\ldots,km^{n-2}$.

(b) The assertion follows from Corollary 5 and (a). □

4. Polynomial form representation of the partitions number

We shall use some of the considered properties to prove and to obtain polynomial form representation of the number of partitions of the sum $k.m^n$ into powers of m.

4.1. Existence of a polynomial, representing the partitions number

Lemma 2. *Let m and n be positive integers. Then the sum* $0^n+1^n+\cdots+m^n=\sum_{k=0}^{m}k^n$ *is a polynomial in m of degree* $n+1$, *where the coefficient of the highest degree is* $1/(n+1)$.

Proof. We put $s_m=0^n+1^n+\cdots+m^n$. Therefore $s_m-s_{m-1}=m^n$. This non-homogeneous recurrence (with an initial condition $s_0=0$) can be solved by the method of the undetermined coefficients [12]. The general solution is of the form $s_m=s_m^{(h)}+s_m^{(p)}$, where $s_m^{(h)}$ is the general solution to the correspondent homogeneous recurrence $s_m-s_{m-1}=0$, and $s_m^{(p)}$ is a partial solution to the same non-homogeneous one. So $s_m^{(h)}=c$, $c=\text{const}$, and $s_m^{(p)}$ is of the polynomial form $s_m^{(p)}=m(a_0+a_1m+\cdots+a_nm^n)$. Replacing $s_m^{(p)}$ in the non-homogeneous recurrence and comparing the coefficients in the both sides of the equality, we obtain $a_i=a_i(n)$, $i=0,1,\ldots,n$ and $a_n=1/(n+1)$. Therefore the solution s_m is a polynomial in m of degree $n+1$. □

Remark 3. The general formula which expresses such sums is the Bernoulli formula $\sum_{k=0}^{m-1}k^n=1/(n+1)\sum_{k=0}^{n}\binom{n+1}{k}B_km^{n+1-k}$, where the coefficients B_k are the Bernoulli numbers [11]. They cannot be represented in a closed form; they are defined by the recurrence $B_0=1$, $\sum_{j=0}^{n}\binom{n+1}{j}B_j=0$ for any $n>0$. Hence $B_0=1$, $B_1=-\frac{1}{2}$, $B_2=\frac{1}{6}$, $B_3=0=B_{2k+1}$ for $k>0$, $B_4=-\frac{1}{30}$, $B_6=\frac{1}{42}$ and so on. In general the sum $S(m,n)=\sum_{k=1}^{m}k^n$ can be expressed by the well-known ζ-function of Riemann.

The main result of the paper are the following two theorems.

Theorem 3. *Let m, n and k be given natural numbers and let i be fixed integer,* $1\leqslant i\leqslant n$. *Then the value of each cell* $T[i,j]$ (*for* $j=0,1,\ldots,km^{n-i}$) *can be expressed by a two-variable polynomial* $P_i(m,j)$, *whose highest monomial is* $\frac{1}{i!}m^{i(i-1)/2}j^i$, *in which the variables m and j are of maximal degrees.*

Proof. We shall prove the assertions of the theorem by mathematical induction in respect to i, using the equality in Corollary 4: $T[i,j]=\sum_{l=0}^{j}T[i-1,lm]$.

(1) We know that $T[1,j]=j+1$ for any $j=0,1,\ldots,km^{n-1}$. The proof of Corollary 6(a) shows that $T[2,j]=1+\frac{1}{2}(2+m)j+\frac{1}{2}mj^2$ for any $j=0,1,\ldots,km^{n-2}$. Therefore for $i=1$ and for $i=2$ the theorem holds.

(2) Let us suppose that for $i=s$, $i<n$ and for any $j=0,1,\ldots,km^{n-s}$, $P_s(m,j)$ are polynomials of the mentioned type, expressing the values of the corresponding cells $T[s,j]$.

(3) Let $i=s+1$ and let us consider the value of the cell $T[s+1,j]$ for an arbitrary j, $0\leqslant j\leqslant km^{n-s-1}$. Then, in accordance with the inductive suggestion,

$$T[s+1,j]=\sum_{l=0}^{j}T[s,lm]=\sum_{l=0}^{j}P_s(m,lm).$$

This is a sum of polynomials which is also a polynomial $P_{s+1}(m,j)$.

(a) Following the inductive suggestion, we consider each $P_s(m,j)$, $0 \leqslant j \leqslant km^{n-s}$, as a polynomial in j of degree s, i.e. $P_s(m,j) = p_0 + p_1 j + p_2 j^2 + \cdots + p_s j^s$. The coefficients $p_l = p_l(m)$ are polynomials in m for $l = 0,1,\ldots,s$ and $p_s(m) = \frac{1}{s!} m^{s(s-1)/2}$. Then:

$$\begin{aligned}
P_{s+1}(m,j) &= \sum_{l=0}^{j} P_s(m,lm) = \sum_{l=0}^{j} (p_0 + p_1.(lm) + \cdots + p_s.(lm)^s) \\
&= p_0 \sum_{l=0}^{j} 1 + p_1 m \sum_{l=0}^{j} l + \cdots + p_s m^s \sum_{l=0}^{j} l^s \\
&= \cdots + p_s m^s \sum_{l=0}^{j} l^s = \cdots + \frac{1}{s!} m^{s(s-1)/2} m^s \sum_{l=0}^{j} l^s \\
&\overset{\text{Lemma 2}}{=} \cdots + \frac{1}{s!} m^{s(s+1)/2} (a_0 + a_1 j + \cdots + a_s j^s + \frac{1}{s+1} j^{s+1}) \\
&= \cdots + \frac{1}{(s+1)!} m^{s(s+1)/2} j^{s+1},
\end{aligned}$$

where $a_l = a_l(s)$, $l = 0,1,\ldots,s$, and so the theorem is valid in the considered case.

(b) Now, in accordance with the inductive suggestion, we consider each $P_s(m,j)$, $0 \leqslant j \leqslant km^{n-s}$ as a polynomial in m of degree $r = s(s-1)/2$, i.e. $P_s(m,j) = q_0 + q_1 m + q_2 m^2 + \cdots + q_r m^r$. The coefficients $q_l = q_l(j)$ are polynomials in j for $l = 1,2,\ldots,r$ and $q_r(j) = (1/s!) j^s$. Hence:

$$\begin{aligned}
P_{s+1}(m,j) &= \sum_{l=0}^{j} P_s(m,lm) = \sum_{l=0}^{j} (q_0(lm) + q_1(lm).m + \cdots + q_r(lm).m^r) \\
&= \sum_{l=0}^{j} q_0(lm) + m \sum_{l=0}^{j} q_1(lm) + \cdots + m^r \sum_{l=0}^{j} q_r(lm) \\
&= \cdots + m^r \sum_{l=0}^{j} \frac{1}{s!} (lm)^s = \cdots + \frac{1}{s!} m^r m^s \sum_{l=0}^{j} l^s \\
&\overset{\text{Lemma 2}}{=} \cdots + \frac{1}{s!} m^{r+s} \left(a_0 + a_1 j + \cdots + a_s j^s + \frac{1}{s+1} j^{s+1} \right) \\
&= \cdots + \frac{1}{(s+1)!} m^{s(s+1)/2} j^{s+1},
\end{aligned}$$

where $a_l = a_l(s)$, $l = 0,1,\ldots,s$, and therefore the theorem is valid in the second case.

So the theorem is proved. □

Theorem 4. *Let m, n and k be given natural numbers and let i be fixed integer, $1 \leqslant i \leqslant n$. Then the value of each cell $\tilde{T}[i,j]$ (for $j = 0,1,\ldots,km^{n-i}$)*

can be expressed by a two-variable polynomial $\tilde{P}_i(m,j)$, *whose highest monomial is* $(1/(i-1)!)m^{i(i-1)/2}j^{i-1}$, *in which the variables* m *and* j *are of maximal degrees.*

The theorem follows from Corollary 5. The proof can also be done as well as the proof of Theorem 3 by using Corollary 3.

Remark 4. (a) From algebraic point of view, Theorem 4 states the existence of a family of two-variable polynomials $P_i(m,j)$, $i=1,\ldots,n$, describing the rows of the table $T_{m,n,k}$.

(b) From geometric point of view, this theorem implies the existence of a family of algebraic surfaces $S_i : z = P_i(m,j)$, $i=1,\ldots,n$. Their sections with a plane $m=\text{const}$ are curves, containing the values of the corresponding rows of the table $T_{m,n,k}$. In the same way, the sections with the planes $j=\text{const}$ discretely describe the values of one and the same cell $T[i,j]$ of the different tables $T_{m,n,k}$—when i is fixed and m varies.

Theorem 4 implies validity of (a) and (b) for the tables $\tilde{T}_{m,n,k}$ and for the corresponding to them polynomials $\tilde{P}_i(m,j)$.

4.2. Computing the partitions number by deriving and evaluation of the corresponding polynomial

Theorems 3 and 4 show some characteristics of the polynomials, expressing the number of partitions of the sum $k.m^n$, but they do not give the explicit form of these polynomials, because a good closed form of the polynomials from Lemma 2 is not known [11]. Using Lemma 2 and the proof of Theorem 3, we show how to obtain the polynomials $P_n(m,j)$ for the first several values of n. We already know that $P_1(m,j)=j+1$ and $P_2(m,j)=1+\frac{1}{2}(2+m)j+\frac{1}{2}mj^2$. Lemma 2 implies: $\sum_{l=0}^{j} l^2 = j/6+j^2/2+j^3/3$, and we derive

$$\begin{aligned} P_3(m,j) &= \sum_{l=0}^{j} P_2(m,ml) = \sum_{l=0}^{j}\left(1+\frac{2+m}{2}ml+\frac{m}{2}(ml)^2\right) \\ &= \sum_{l=0}^{j} 1+\frac{2m+m^2}{2}\sum_{l=0}^{j} l+\frac{m^3}{2}\sum_{l=0}^{j} l^2 \\ &= \cdots = 1+\frac{1}{12}(12+6m+3m^2+m^3)j+\frac{1}{4}m(2+m+m^2)j^2+\frac{1}{6}m^3j^3. \end{aligned}$$

The polynomials $\tilde{P}_n(m,j)$ can be obtained in the same way, or more easily—only by using Corollary 5. For example, we replace j by mj in the last expression and we obtain the polynomial, representing the fourth row of the table $\tilde{T}_{m,n,k}$:

$$\tilde{P}_4(m,j) = 1+\tfrac{1}{12}m(12+6m+3m^2+m^3)j+\tfrac{1}{4}m^3(2+m+m^2)j^2+\tfrac{1}{6}m^6j^3.$$

For $m=2$ we obtain

$$P_3(2,j)=1+\frac{11}{3}\,j+4j^2+\frac{4}{3}\,j^3=\frac{(2j+1)(2j+2)(2j+3)}{6},$$

$$\tilde{P}_4(2,j)=1+\frac{22}{3}\,j+16j^2+\frac{32}{3}\,j^3=\frac{(4j+1)(4j+2)(4j+3)}{6}.$$

The lth tetrahedral number is expressed by the formula $l(l+1)(l+2)/6,\ l=1,2,\ldots$. Hence we extend the relations between the tables and the figurate numbers.

Corollary 7 (Relation between the tables and the tetrahedral numbers). *If $n>3$, the odd tetrahedral numbers take place in the third row of the table $T_{2,n,k}$.*

If $n>4$, odd tetrahedral numbers, such that $l=4j+1,\ j=0,1,\ldots,2^{n-4}$, take place in the fourth row of the table $\tilde{T}_{2,n,k}$.

The following two polynomials (for $n=4$ and 5) are derived by using the Mathematica 2.2 package:

$$\begin{aligned}P_4(m,j)&=1+\tfrac{1}{24}\,(24+12m+6m^2+5m^3+2m^4+m^5)j\\&\quad+\tfrac{1}{24}\,m(12+6m+9m^2+4m^3+3m^4+m^5)j^2\\&\quad+\tfrac{1}{12}\,m^3(2+m+m^2+m^3)j^3+\tfrac{1}{24}\,m^6j^4,\\P_5(m,j)&=1+\tfrac{1}{720}\,(720+360m+180m^2+150m^3\\&\quad+105m^4+75m^5+36m^6+15m^7+5m^8-m^{10})j\\&\quad+\tfrac{1}{48}\,m(24+12m+18m^2+11m^3+11m^4+7m^5+4m^6+2m^7+m^8)j^2\\&\quad+\tfrac{1}{72}\,m^3(12+6m+9m^2+10m^3+6m^4+4m^5+3m^6+m^7)j^3\\&\quad+\tfrac{1}{48}\,m^6(2+m+m^2+m^3+m^4)j^4+\tfrac{1}{120}\,m^{10}j^5.\end{aligned}$$

We put $j=1$ in the polynomials above and we obtain polynomials in m, which represent the number of all partitions of the number m^n of type (5):

$P_1(m,1)=2,$
$P_2(m,1)=2+m,$
$P_3(m,1)=\frac{1}{2}\,(4+2m+m^2+m^3),$
$P_4(m,1)=\frac{1}{12}\,(24+12m+6m^2+9m^3+4m^4+3m^5+2m^6),$
$P_5(m,1)=\frac{1}{24}\,(48+24m+12m^2+18m^3+11m^4+11m^5+9m^6+5m^7+3m^8+2m^9+m^{10}).$

We can obtain the next polynomial $P_6(m,1)$ without having the polynomial $P_6(m,j)$. Recurrence (10) implies $T_{m,n,k}[6,1]=1+T_{m,n,k}[5,m]$. So we replace j by m in $P_5(m,j)$, then we add 1 and finally we obtain

$$\begin{aligned}P_6(m,1)&=\tfrac{1}{720}(1440+720m+360m^2+540m^3+330m^4+375m^5+360m^6\\&\quad+260m^7+210m^8+165m^9+120m^{10}+69m^{11}+45m^{12}\\&\quad+25m^{13}+15m^{14}+6m^{15}).\end{aligned}$$

Remark 5. Theorem 3 implies the existence of a polynomial $P_n(m,k)$, expressing the value of $t_m(n,k)$ for a fixed n. Every such polynomial, whose explicit form is known, can be evaluated (for given m and k) by appropriate applying the Horner's rule

$$Q(x) = a_0 + x(a_1 + x(a_2 + \cdots + x(a_{n-1} + xa_n))\cdots)$$

to each of the polynomial variables m and k. Thus the evaluation takes time $\Theta(n.n(n-1)/2) = \Theta(n^3)$. This is quite better time-complexity in comparison with the time-complexities of Algorithms B and C for solving the same problem. The common problem, which can be solved by Algorithms A–C, is computing the number of all partitions of the number m^n of type (5) for given input m and n. There exist a polynomial in m of degree $n(n-1)/2$, which express this number. When its explicit form is known, the evaluation of the value of $t_m(m^n) = t_m(n,1) = P_n(m,1)$ takes polynomial time $\Theta(n^2)$, which is quite better than the exponential time-complexity $\Theta(m^{n-2})$ of Algorithm A for computing the same value.

Algorithm C (also Algorithm B) can be accelerated by applying a mixed strategy. For example, if $n > 5$ we use the polynomial $P_5(m,j)$ to compute and fill in the cells in the fifth row of T. We suppose that the coefficients of the powers of j in $P_5(m,j)$ are known for given m (otherwise they can be computed in time $\Theta(n^3)$). We apply the Horner's rule to compute the value of each cell $T[5,j]$, $j = 1,2,\ldots,km^{n-5}$. So the filling in the fifth row costs $\Theta(km^{n-5})$ units time. The filling in the next rows of T continues in the manner, which was described in the procedure of Algorithm C and it needs $\Theta(km^{n-6})$ units time. Therefore, for given input m,n $(n > 5)$ and k, the modified Algorithm C computes the value of $t_n(n,k)$ with time-complexity $\Theta(km^{n-5})$. In particular, it computes the value of $t_m(m^n)$ with time-complexity $\Theta(m^{n-5})$, which is better than the time-complexity of Algorithm A for computing the same value.

We note, that in the general case the mentioned improvement of Algorithm C reduces its exponential time-complexity to a better, but exponential one again. We shall show, that in the general case it is possible to compute the number of considered partitions in a polynomial time. For that purpose we propose the following algorithm, called *Algorithm D*.

Algorithm D. Deriving of the polynomial, expressing the number of all partitions of the sum $k.m^n$ into powers of m and computing this number

Input: m,n and k.

Output: the coefficients of the corresponding polynomial $P_n(m,j)$, computed for given m and n, and its value for $j = k$.

Procedure: (1) Filling in the Pascal triangle: compute and store the binomial coefficients in the array c, so that $c[i,j] = \binom{i}{j}$, for $0 \leqslant i \leqslant n+1$, $0 \leqslant j \leqslant i$.

(2) Filling in the Bernoulli numbers: compute and store the Bernoulli numbers in the array b, so that $b[i] = B_i$, $0 \leqslant i \leqslant n$ (in accordance with Remark 3).

(3) Following Lemma 2 and Remark 3, compute the coefficients of the polynomials $Q_i(j) = q_{i,0} + q_{i,1}j + \cdots + q_{i,i+1}j^{i+1}$, representing the sum $0^i + 1^i + \cdots + j^i$, and store them in the ith row of the array q, for $i = 0,1,\ldots,n-1$.

(4) Following the proof of Theorem 3, compute the coefficients of the polynomials $P_i(m,j) = p_{i,0} + p_{i,1}j + \cdots + p_{i,i}j^i$ (where $p_{i,l} = p_{i,l}(m)$, $0 \leqslant l \leqslant i$), and store them in the ith row of the array p, for $i = 1,2,\ldots,n$.

(5) Applying the Horner's rule, compute the value of $P_n(m,j)$ for $j = k$ and return it. End.

Now we clarify and analyze the steps of Algorithm D.

Step 1 can be realized by the known formula $\binom{i}{j} = 1$ if $j = 0$ or $j = i$, and $\binom{i}{j} = \binom{i-1}{j-1} + \binom{i-1}{j}$ for $0 < j < i$. So computing and filling in the binomial coefficients in the array c costs $\Theta(n^2)$ time.

In step 2 (following Remark 3), we fill in $b[0] = 1$, $b[1] = -\frac{1}{2}$ and $b[i] = 0$ for all odd i, $3 \leqslant i \leqslant n$. For all even i, $2 \leqslant i \leqslant n$, we compute $B_i = (-1/(i+1))\sum_{j=0}^{i-1}\binom{i+1}{j}B_j$ and store it in the cell $b[i]$. All binomial coefficients and Bernoulli numbers, necessary for computing of B_i, are already stored in the arrays c and b, correspondingly. The entire performance of step 2 costs $\Theta(n^2)$ time and again.

In step 3 we use the Bernoulli formula

$$0^i + 1^i + \cdots + j^i = \sum_{r=0}^{j} r^i = \frac{1}{i+1}\sum_{r=0}^{i}\binom{i+1}{r}B_r(j+1)^{i+1-r}$$

to obtain the explicit form of the polynomials from Lemma 2, for $i = 0,1,\ldots,n-1$. The results, i.e. the coefficients of the powers of j, we store in the array q with n rows, corresponding to the values of i, and $n+1$ columns, corresponding to the powers of j from 0 to n. So $q[i,l]$ will contain the coefficient of j^l in the polynomial, representing the sum $0^i + 1^i + \cdots + j^i$. Since $0^0 + 1^0 + \cdots + j^0 = 1 + j$, we put $q[0,0] = 1, q[0,1] = 1$ and $q[0,l] = 0$ for $l = 2,3,\ldots,n$, initially. We define the working array w with n rows (enumerated from 0 to $n-1$) and $n+2$ columns (enumerated from 0 to $n+1$), the last column is auxiliary. For each $i = 1,2,\ldots,n-1$ we repeat the following:

(a) initially we put $w[r,l] = 0$ for $0 \leqslant r \leqslant i$, $0 \leqslant l \leqslant i+1$,
(b) for each $r = 0,1,\ldots,i$ we compute the value of $\binom{i+1}{r}B_r$ and store it in the cell $w[r,n+1]$ (by using the corresponding numbers from the arrays c and b),
(c) for each $r = 0,1,\ldots,i$ we check the value of $w[r,n+1]$. If it is not zero, we take the coefficients of the binomial expansion of $(1+j)^{i+1-r}$ from the $(i+1-r)$th row of the array c. We multiply each of them by $w[r,n+1]$ and store the result in the corresponding cell of the rth row of the array w,
(d) for each $l = 0,1,\ldots,i+1$ we sum the numbers from lth column (i.e. $w[0,l] + w[1,l] + \cdots + w[i,l]$) and divide this sum to $(i+1)$—so we obtain the coefficient $q_{i,l}$ in the polynomial $Q_i(j)$ and store it in the cell $q[i,l]$.

Obviously, all operations from (a) to (d) can be performed in $\Theta(i^2)$ time for given i. The algorithm repeats them for each $i = 1,\ldots,n-1$, and so the time-complexity of step 3 is $\Theta(1^2) + \Theta(2^2) + \cdots + \Theta((n-1)^2) = \Theta(1^2 + 2^2 + \cdots + (n-1)^2) = \Theta(n^3)$.

In step 4 we use the proof of Theorem 3 and the given examples. Step 4 is analogous to step 3 and it is based on the statement, that if all coefficients of the polynomial $P_i(m,j) = a_0 + a_1 j + \cdots + a_i j^i$ are known, then the coefficients of the next polynomial $P_{i+1}(m,j)$ can be computed by the formula

$$P_{i+1}(m,j) = \sum_{l=0}^{j} P_i(m,ml) = \sum_{l=0}^{j} (a_0 + a_1(ml) + \cdots + a_i(ml)^i)$$

$$= \sum_{l=0}^{j} a_0 + a_1 m \sum_{l=0}^{j} l + \cdots + a_i m^i \sum_{l=0}^{j} l^i.$$

By the polynomials, obtained in step 3, we expand the last equality and compute the coefficients of $P_{i+1}(m,j)$, for $i = 1,\ldots,n-1$. We store them in the array p with n rows (corresponding to the values of i) and $n+1$ columns (corresponding to the powers of j), so that the cell $p[i,l]$ contains the coefficient of j^l in the polynomial $P_i(m,j)$. Since $P_1(m,j) = 1 + j$, we put $p[1,0] = 1$, $p[1,1] = 1$ and $p[1,l] = 0$ for $l = 2,3,\ldots,n$, initially. We compute and store the powers of m in the array u, so that $u[i] = m^i$ for $i = 0,\ldots,n-1$. Also we use the same working array w from step 3. For each $i = 1,2,\ldots,n-1$ we determine the coefficients of $P_{i+1}(m,j)$ by repeating of:

(a) initially we put $w[r,l] = 0$ for $0 \leqslant r \leqslant i$, $0 \leqslant l \leqslant i+1$,
(b) for each $r = 0,1,\ldots,i$ we multiply a_r by m^r and store the result in the cell $w[r,n+1]$ (using the coefficients from the ith row of the array p and the value of $u[r]$),
(c) for each $r = 0,1,\ldots,i$ we take the coefficients from the rth row of the array q, we multiply each of them by $w[r,n+1]$ and store the result in the corresponding cell of the rth row of the array w,
(d) for each $l = 0,1,\ldots,i+1$ we sum the numbers from lth column (i.e. $w[0,l] + w[1,l] + \cdots + w[i,l]$) and the result is the coefficient $p_{i+1,l}$ in the polynomial $P_{i+1}(m,j)$—we store it in the cell $p[i+1,l]$.

Analysis of the time-complexity of step 4 is identical to the one in step 3 and for the time-complexity of step 4 we obtain $\Theta(n^3)$ again.

Step 5 applies the Horner's rule using the coefficients from the nth row of the array p. This step can be executed in time $\Theta(n)$.

The correctness of Algorithm D follows directly from the assertions and the explanations, given in this section. The general time-complexity of Algorithm D (towards the accepted uniform cost criterion) is a sum of the time-complexities of its five steps and therefore it is $\Theta(n^3)$. As in the cases, when the explicit form of the polynomials $P_n(m,j)$ is known, we obtain the same polynomial time-complexity, which does not depend on m. The computing of the value of $t_m(m^n) = t_m(n,1) = P_n(m,1)$, in the general case, can be done by applying Algorithm D in time $\Theta(n^3)$. Finally we note, that we omitted some details in the description of Algorithm D—for example computations over rational numbers. They are included in our program, realizing Algorithm D, because the coefficients of the polynomials are rational numbers and the use of real numbers

can cause round-off errors and their accumulation. It is easy to see, that operating with rational numbers slows the performance of Algorithm D, but its time-complexity remains the same.

In future we intend to investigate more general problem than the one, considered here. Namely, to count all partitions of sums of the type $a_1m^{n_1}+a_2m^{n_2}+\cdots+a_km^{n_k}$ (the coefficients a_i, $i=1,\ldots,k$ are positive integers) into powers of m. And next to look for answers of the questions: "What is the relation between the partitions of such sums and the partitions of sums, considered in this paper?" and "If $n=c_1m^{j_1}+c_2m^{j_2}+\cdots+c_rm^{j_r}$ is the representation of a given natural number n in m-ary counting system, then what is the relation between $t_m(n)$ and the number of all partitions of the sum above?".

5. Applications

There are various combinatorial problems, whose solutions are related to the considered partitions. For example, binary partition problems arise at a partition of the computer memory and of the hard disk space. m-ary partition problems arise in many other cases—for example in chain-reaction processes (chemical, nuclear, radioactive decay in cosmic rays, cell-division and so on) during a given period of time. Their solutions can be illustrated by trees, having certain properties.

Donald Knuth [14] pays special attention to the trees as informational structures and sets and/or solves series of counting problems, concerning certain trees. He notes that one of the most immediate applications of the mathematical theory of the trees into investigation of algorithms is related to formulas for counting of different types of trees.

5.1. Counting of some full m-ary trees

We start with an inductive (and constructive) definition of a *full m-ary rooted tree*, similar to the definitions of rooted trees, given in [16].

Definition. (0) The graph $G_0=(\{v_0\},\emptyset)$ is full m-ary rooted tree, having only a root v_0 and no edges. G_0 is said to be a trivial tree.

(1) The graph $G_1=(V_1,E_1)$, where $V_1=\{v_0,v_1,\ldots,v_m\}$ and $E_1=\{(v_0,v_1),(v_0,v_2),\ldots,(v_0,v_m)\}$, is full m-ary tree with root v_0.

(2) Let $G_i=(V_i,E_i)$ be full m-ary tree with a root $r\in V_i$ and let $V_i\cap V_1=\emptyset$. Then the graph $G_{i+1}=(V_{i+1},E_{i+1})$, where $V_{i+1}=V_1\cup V_i\setminus\{v_j\}$, $v_j\in V_i$, $E_{i+1}=E_1\cup(E_i\setminus\{(v_k,v_j)\})\cup\{(v_k,v_0)\}$, is also full m-ary tree with root r. G_{i+1} is said to be obtained by the operation "replacement of an arbitrary leaf v_j of G_i by the tree G_1."

(3) There are not other full m-ary rooted trees, excepting G_0,G_1 and all trees, obtained by applying (2).

It is well known how to represent any arithmetical expression by a binary tree. The considered partitions are also arithmetical expressions, having only operation addition. Instead of binary trees we choose full m-ary rooted trees to represent the partitions of m^n. In other words we juxtapose such a tree to each partition of m^n of type (5). Each

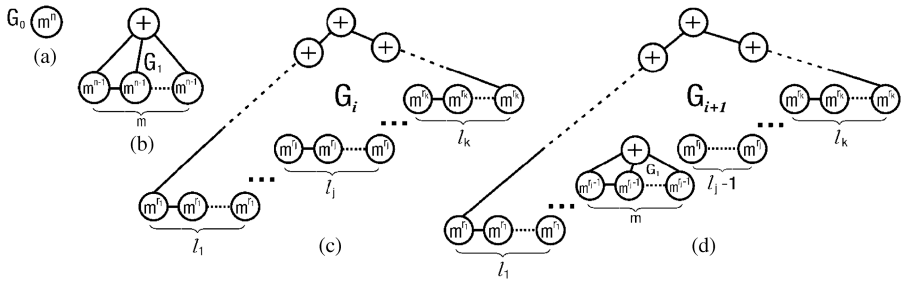

Fig. 1. Illustration of the juxtaposing-procedure.

leaf of the tree we label with an addend and each internal node we label with a "+" sign, understanding it as a sign of m-argumented summation (because each partition of an addend always replaces it by a sum of m new addends).

The first partition of the number m^n is the trivial partition. Using Lemma 1, the second one is $m^n = \underbrace{m^{n-1} + \cdots + m^{n-1}}_{m}$. We can obtain each next partition consecutively, step by step, applying Lemma 1 again. After the second partition we continue with a partition of some addend in each step. When there are equal addends we choose to partition the leftmost one—so each time we obtain a monotonically increasing sequence of addends. Following the given definition and explanations we define a procedure for juxtaposing a full m-ary rooted tree to each partition of m^n of type (5).

Procedure for juxtaposing. (0) *We juxtapose the trivial tree G_0 to the first (trivial) partition $m^n = m^n$. We label the root v_0 with m^n, as it is shown in* Fig. 1(a).

(1) *We juxtapose the tree G_1 to the second partition $m^n = \underbrace{m^{n-1} + \cdots + m^{n-1}}_{m}$. The root v_0 we label with a "+" sign and the leaves we label with m^{n-1}—Fig.* 1(b).

(2) *Let $G_i = (V_i, E_i)$ be full m-ary rooted tree, which is juxtaposed to the serial partition $m^n = \underbrace{m^{r_1} + \cdots + m^{r_1}}_{l_1} + \cdots + \underbrace{m^{r_j} + \cdots + m^{r_j}}_{l_j} + \cdots + \underbrace{m^{r_k} + \cdots + m^{r_k}}_{l_k}$, i. e. the internal nodes of G_i are labeled with a "+" sign and the leaves of G_i are labeled (from left to right) with $m^{r_1}, \ldots, m^{r_1}, \ldots, m^{r_j}, \ldots, m^{r_j}, \ldots, m^{r_k}, \ldots, m^{r_k}$, in correspondence with their order in the partition (as in Fig.* 1(c)).

(3) *We juxtapose the tree G_{i+1} to the next partition $m^n = \underbrace{m^{r_1} + \cdots + m^{r_1}}_{l_1} + \cdots + \underbrace{m^{r_j - 1} + \cdots + m^{r_j - 1}}_{m} + \underbrace{m^{r_j} + \cdots + m^{r_j}}_{l_j - 1} + \cdots + \underbrace{m^{r_k} + \cdots + m^{r_k}}_{l_k}$, which is obtained by replacement the leftmost addend m^{r_j} by the sum $\underbrace{m^{r_j - 1} + \cdots + m^{r_j - 1}}_{m}$. The tree G_{i+1} is obtained from G_i by replacement the corresponding to m^{r_j} leftmost leaf by the tree G_1, whose root is labeled by "+" again, but now its leaves are labeled by $m^{r_j - 1}$ (as in Fig.* 1(d)).

We denote by $\mathscr{T}$ the set of all trees which are a result of the juxtaposing-procedure for given m and n. They are characterized by the following three restrictions:

(1) the height of each tree does not exceed n, since each addend (inclusive m^n) can be partitioned up to n times;
(2) the number of the leaves (addends) is of the form $k(m-1)+1$ for $k=0,1,\ldots,(m^n-1)/(m-1)$, because each time a leaf is converted to an internal node and the tree grows with m new leaves; 1 is the minimal and m^n is the maximal number of the leaves;
(3) the distances between the root and the leaves, considered from left to right, form a monotonically decreasing sequence of integers (between n and 0), because the addends in each partition are ordered in a monotonically increasing sequence.

Theorem 5. *Let m and n be natural numbers, $m>1$, $n>0$. Then $t_m(m^n)=|\mathscr{T}|$.*

Proof. We denote by $\mathscr{P}$ the set of all partitions of m^n of type (5) and by $f:\mathscr{P}\to\mathscr{T}$ the function defined by the juxtaposing procedure. We consider two different partitions from $\mathscr{P}$ and let us suppose that the deduction the first of them (in the chain of consecutive partitions) ends before the deduction the second one. Hence, when the tree, corresponding to the first partition is already built, the tree, corresponding to the second partition continues to grow—so these trees are different. Therefore f is an injection. Any tree from $\mathscr{T}$ is a result from the juxtaposing-procedure and so there exists a partition from $\mathscr{P}$, which corresponds to this tree. Hence f is a surjection. Therefore f is a bijection and $|\mathscr{P}|=|\mathscr{T}|$. □

For example, the so-called *recursion trees* [8] are trees from $\mathscr{T}$. They are used for visualization the iterations of a recurrence, describing a divide-and-conquer algorithm. Full *m-ary* rooted trees, having only the first two restrictions, can represent the compositions[2] of m^n. It is not hard to prove the equivalence between the number of all compositions and the number of all such trees.

5.2. Counting the partitions of the Boolean cube

This was the problem, whose generalizations yielded to the writing of this paper.

The Boolean cube is one of the most famous and basic notions in Discrete Mathematics. There are many definitions of it. We give an inductive definition, based on the lexicographic order of the vectors (it looks like the definition, given in [16]).

Definition. (1) We call a one-dimensional Boolean cube the set $\{0,1\}=\{(0),(1)\}$, and its elements (0) and (1)—binary vectors of it.

(2) We call a two-dimensional Boolean cube the set of binary vectors $\{0,1\}^2=\{(00),(01),(10),(11)\}$, which is obtained from $\{0,1\}^1$, firstly by adding 0 in the beginning of all its vectors and next by adding 1.

[2] Compositions are partitions, where the order of the addends is significant; ordered partitions [3].

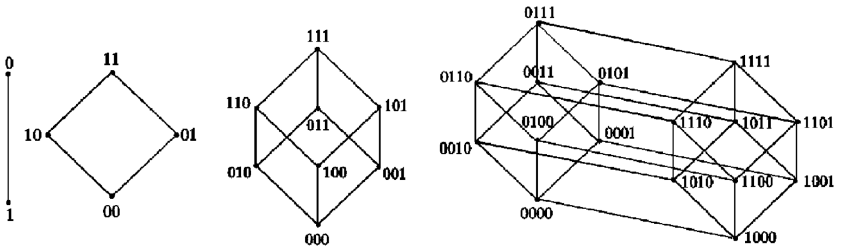

Fig. 2. $\{0,1\}^n$ for $n=1,2,3,4$.

(3) Let $\{0,1\}^{n-1}=\{\alpha_0,\alpha_1,\ldots,\alpha_{2^{n-1}-1}\}$ be the $(n-1)$-dimensional Boolean cube, where $\alpha_0,\alpha_1,\ldots,\alpha_{2^{n-1}-1}$ are its $(n-1)$-dimensional binary vectors. We build the n-dimensional Boolean cube $\{0,1\}^n$ from $\{0,1\}^{n-1}$, firstly by adding 0 in the beginning of all its vectors and next by adding 1, i.e.

$$\{0,1\}^n=\{(0\alpha_0),(0\alpha_1),\ldots,(0\alpha_{2^{n-1}-1}),(1\alpha_0),(1\alpha_1),\ldots,(1\alpha_{2^{n-1}-1})\}.$$

From the definition it follows that $\{0,1\}^n$ contains exactly 2^n vectors.

Boolean cubes with dimensions 1–4 are represented in Fig. 2 as graphs whose vertices are the vectors and their edges connect each two neighboring vectors.

Definition. We call a k-dimensional subcube $(0\leqslant k\leqslant n)$ of $\{0,1\}^n$ the subset of all vectors of $\{0,1\}^n$, which have equal values in $(n-k)$ fixed positions and their values in the remaining k positions (considered as k-dimensional vectors) form a k-dimensional Boolean cube.

We assume that for $k=0$ it comes out a subcube of minimal possible dimension $\{0,1\}^0=\{\alpha\}$, where α is its unique n-dimensional vector.

We can determine a concrete subcube among all possible k-dimensional subcubes of $\{0,1\}^n$ by enumerating the fixed positions of the vectors and the fixed values in these positions—for example like this:

$$\begin{pmatrix} i_1,i_2,\ldots,i_{n-k}\\ \sigma_1,\sigma_2,\ldots,\sigma_{n-k}\end{pmatrix},\quad 1\leqslant i_1<i_2<\cdots<i_{n-k}\leqslant n,\quad \sigma_j\in\{0,1\},$$

where σ_j is the fixed value of each vector in a position i_j for $j=1,2,\ldots,n-k$.

We obtain the number of all possible k-dimensional subcubes of $\{0,1\}^n$ by multiplying the number of ways for choosing $(n-k)$ fixed positions and the number of ways the values in these fixed positions can vary, i.e.

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

Definition. We call a partition or a k-partition of the n-dimensional Boolean cube into subcubes the representation $\{0,1\}^n=\{0,1\}^{r_1}\cup\{0,1\}^{r_2}\cup\cdots\cup\{0,1\}^{r_k}$, $0\leqslant r_i<n$, $i=$

$1,2,\ldots,k$, so that $\{0,1\}^{r_i} \cap \{0,1\}^{r_j} = \emptyset$ for $1 \leqslant i < j \leqslant n$. The partition $\{0,1\}^n = \{0,1\}^n$ is said to be trivial partition of the Boolean cube.

Obviously the subcubes in a fixed k-partition can be chosen (or formed) in many different ways without changing their dimensions.

Definition. Two or more k-partitions of the Boolean cube are said to be equivalent or k-equivalent if the dimensions of their subcubes are one and the same (they can differ from each other only in the vectors which their subcubes consist of). And vice versa: two k-partitions are non-equivalent if the dimensions of their subcubes, considered as non-ordered k-tuples, are different.

Further we discuss only the non-equivalent partitions of the Boolean cube.

Lemma 3. *For any integer $n > 0$ there exists unique 2-partition of $\{0,1\}^n$ into two subcubes of dimensions $n-1$.*

Proof. The definition of the Boolean cube yields to the existence of such a partition. The uniqueness of this partition follows directly from the definition of the partition and Lemma 1 (for $m = 2$). □

We denote this 2-partition of $\{0,1\}^n$ by $\{0,1\}^n = \{0,1\}_0^{n-1} \cup \{0,1\}_1^{n-1}$, where $\{0,1\}_0^{n-1}$ and $\{0,1\}_1^{n-1}$ denote both $(n-1)$-dimensional subcubes, the first of them containing all vectors, which begin with 0, and the second—all vectors, which begin with 1. When we have a 2-partition of a subcube of dimension r, $\{0,1\}^r = \{0,1\}_0^{r-1} \cup \{0,1\}_1^{r-1}$, $1 \leqslant r < n$, then $\{0,1\}_0^{r-1}$ denotes all vectors from $\{0,1\}^r$, in which the first non-fixed position is 0, and $\{0,1\}_1^{r-1}$—all vectors from $\{0,1\}^r$, in which this position is 1.

Following the given definitions and assertions, we define a procedure for juxtaposing a k-partition of $\{0,1\}^n$ to each k-partition of the integer 2^n of type (5). This procedure is similar to that, given above.

Procedure for juxtaposing. (0) *We juxtapose the trivial partition of the Boolean cube to the trivial partition of the number 2^n.*

(1) *We juxtapose the 2-partition $\{0,1\}^n = \{0,1\}_0^{n-1} \cup \{0,1\}_1^{n-1}$ (from Lemma 3) to the 2-partition $2^n = 2^{n-1} + 2^{n-1}$ (from Lemma 1).*

(2) *Let $\{0,1\}^n = \{0,1\}^{r_1} \cup \cdots \cup \{0,1\}^{r_i} \cup \cdots \cup \{0,1\}^{r_k}$ be a k-partition of $\{0,1\}^n$, which is juxtaposed to the serial k-partition $2^n = 2^{r_1} + \cdots + 2^{r_i} + \cdots + 2^{r_k}$ of the integer 2^n, so that each r_i-dimensional subcube corresponds to an addend 2^{r_i}, $i = 1,2,\ldots,k$. The dimensions of the subcubes (as well as the addends) form monotonically increasing sequence.*

(3) *We juxtapose the $(k+1)$-partition $\{0,1\}^n = \{0,1\}^{r_1} \cup \cdots \cup \{0,1\}_0^{r_i-1} \cup \{0,1\}_1^{r_i-1} \cup \cdots \cup \{0,1\}^{r_k}$ to the next $(k+1)$-partition $2^n = 2^{r_1} + \cdots + 2^{r_i-1} + 2^{r_i-1} + \cdots + 2^{r_k}$, obtained by applying Lemma 1 to the leftmost addend 2^{r_i} in (2). The $(k+1)$-partition of $\{0,1\}^n$ is obtained by applying Lemma 3 to the leftmost subcube $\{0,1\}^{r_i}$ in (2).*

This procedure can be illustrated by binary trees as well as the partitions of the integer 2^n. This procedure is also a constructive one—it provides the obtaining of all partitions of $\{0,1\}^n$

Theorem 6. *For any integer $n > 0$ the number of all partitions of the n-dimensional Boolean cube is $t_2(2^n)$.*

Proof. We denote by g the function, defined by the last juxtaposing-procedure. A domain of g is the set of all possible partitions of $\{0,1\}^n$ and a codomain of g is the set of all partitions of 2^n. The proof that g is a bijection is identical with the proof that f is a bijection in Theorem 5, and so the theorem holds. □

We omit the question for counting the partitions of the Boolean cube, which are equivalent to the given one [4], because it stands apart from the main idea of this paper. We hope that the considered partitions will find applications in other counting problems.

Acknowledgements

We are very grateful to our anonymous referees for their valuable remarks and suggestions. They gave us hints regarding the two main ideas in Section 4, and also in Corollaries 6 and 7. We thank to our colleagues Dr. Milen Hristov and Professor Stefka Bouyuklieva, who supported us so far and contributed to improve this paper.

References

[1] A.V. Aho, J.E. Hopcroft, J.D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, MA, 1974.
[2] G.E. Andrews, Congruence properties of the *m*-ary partition function, J. Number Theory 3 (1971) 104–110.
[3] G.E. Andrews, The Theory of Partitions, Addison-Wesley, Reading, MA, 1976.
[4] V.P. Bakoev, Partitions of the *n*-dimensional Boolean cube into subcubes, in: Annual of the University of V. Tarnovo "St. St. Cyrill and Methodius", Faculty of Mathematics and Informatics, Vol. 1, University Publication, 1998, pp. 55-67 (in Bulgarian).
[5] V.P. Bakoev, K.N. Manev, Irreducible conjunctive normal forms of the zero function II, in: Mathematics and Education in Mathematics, Proceedings of the 26th Spring Conference of the Union of Bulgarian Mathematicians, Sofia, 1997, pp. 209–215.
[6] V.P. Bakoev, K.N. Manev, Some necessary and some sufficient conditions about the 3-satisfiability problem, in: Mathematics and Education in Mathematics, Proceedings of the 29th Spring Conference of the Union of Bulgarian Mathematicians, Lovetch, 2000, pp. 233–239.
[7] R.F. Churchhouse, Congruence properties of the binary partition function, Proc. Cambridge Philos. Soc. 66 (1969) 371–376.
[8] T.H. Cormen, Ch.E. Leiserson, R.L. Rivest, Introduction to Algorithms, The MIT Press, New York, 1990.
[9] C.-E. Fröberg, Accurate estimation of the number of binary partitions, BIT 17 (1977) 386–391.
[10] M.R. Garey, D.S. Johnson, Computers and Intractability. A Guide to the Theory of NP-Completeness, Freeman, San Francisco, 1979.

[11] R.L. Graham, D.E. Knuth, O. Patashnik, Concrete Mathematics. A Foundation for Computer Science, 2nd Edition, Addison-Wesley, Reading, MA, 1998.
[12] R.P. Grimaldi, Discrete and Combinatorial Mathematics. An Applied Introduction, 4th Edition, Addison-Wesley, Reading, MA, 1999.
[13] H. Gupta, On *m*-ary partitions, Proc. Cambridge Philos. Soc. 71 (1972) 343–345.
[14] D.E. Knuth, Fundamental algorithms, The Art of Computer Programming, 2nd Edition, Vol. 1, Addison-Wesley, Reading, MA, 1973.
[15] V. Lipski, Combinatorics for Programmers, Mir, Moskow, 1988 (in Russian).
[16] K.N. Manev, Introduction to Discrete Mathematics, 2nd Edition, NBU, Sofia, 1998 (in Bulgarian).
[17] Ö. Rödseth, Some arithmetical properties of m-ary partitions, Proc. Cambridge Philos. Soc. 68 (1970) 447–453.
[18] N.J.A. Sloane, The on-line encyclopedia of integer sequences, 2000, Published electronically at http://www.research.att.com/∼njas/sequences/.
[19] N.J. Vilenkin, Combinatorics, Nauka, Moskow, 1969 (in Russian).