

What is an answer?—remarks, results and problems on PIO formulas in combinatorial enumeration, part I

Martin Klazar

September 11, 2018

Abstract

For enumerative problems, i.e. computable functions $f : \mathbb{N} \rightarrow \mathbb{Z}$, we define the notion of an *effective (or closed) formula*. It is an algorithm computing $f(n)$ in the number of steps that is polynomial in the combined size of the input n and the output $f(n)$, both written in binary notation. We discuss many examples of enumerative problems for which such closed formulas are, or are not, known. These problems include (i) linear recurrence sequences and holonomic sequences, (ii) integer partitions, (iii) pattern-avoiding permutations, (iv) triangle-free graphs and (v) regular graphs. In part I we discuss problems (i) and (ii) and defer (iii)–(v) to part II. Besides other results, we prove here that every linear recurrence sequence of integers has an effective formula in our sense.

1 Introduction

We define what is an effective (closed, explicit) formula (solution, algorithm) for a problem in enumerative combinatorics or number theory. An *enumerative problem* or a *counting function* is a computable function

$$f : \mathbb{N} \rightarrow \mathbb{Z}$$

(in part II also a computable function $f : \{0, 1\}^* \rightarrow \mathbb{Z}$) given usually via an algorithm computing it. The algorithm, often inefficient, usually follows straightforwardly from the statement of the problem. Our notation: $|X|$ and $\#X$ denote cardinality of a set X , $\mathbb{N} = \{1, 2, \dots\}$, $\mathbb{N}_0 = \{0, 1, \dots\}$, \mathbb{Z} is the ring of integers, \mathbb{Q} and \mathbb{C} are, respectively, the fields of rational and complex numbers and $\{0, 1\}^*$ is the set of finite binary words. The asymptotic symbols $o(\cdot)$, $O(\cdot)$, $\Omega(\cdot)$, $\Theta(\cdot)$, $\cdot \ll \cdot$, $\cdot \sim \cdot$ and $\text{poly}(\cdot)$ have their usual meaning (\ll is synonymous to O and $\text{poly}(x) = O((1 + |x|)^d)$ for some $d \in \mathbb{N}$). We will discuss many enumerative problems and effective formulas.

Here are some examples of enumerative problems $f : \mathbb{N} \rightarrow \mathbb{Z}$.

1. The *Catalan numbers* $f(n) = c_n = \frac{1}{n} \binom{2n-2}{n-1}$ which count planted trees with n vertices, and many other structures.
2. Let $f(n) = c_n$ for even c_n and $f(n) = 1$ for odd c_n .
3. A linear recurrence sequence $f(n+k) = \sum_{i=0}^{k-1} a_i f(n+i)$, given by initial values $f(1), \dots, f(k) \in \mathbb{Z}$ and recurrence coefficients $a_0, \dots, a_{k-1} \in \mathbb{Z}$.
4. We may be interested in $f(n) = \sum_{\lambda \in P(n)} \left(\|\lambda\|^{\|\lambda\|^{\|\lambda\|}} - \lfloor \log \|\lambda\| \rfloor \right)$ where λ runs through all partitions of n and $\|\lambda\|$ is the number of parts.
5. Or in the surplus $f(n)$ of the partitions of n into an even number of distinct parts from $\{1_1, 1_2, 2_1, 2_2, 3_1, \dots\}$ (two-sorted natural numbers) over those with an odd number of distinct parts, that is, $f(n)$ is the coefficient of q^n in the expansion of $\prod_{k=1}^{\infty} (1 - q^k)^2$.
6. Another function $f(n)$ counts 1324-avoiding permutations $a_1 a_2 \dots a_n$ of $[n] = \{1, 2, \dots, n\}$; the avoidance means that no four indices $1 \leq i_1 < i_2 < i_3 < i_4 \leq n$ exist with $a_{i_1} < a_{i_3} < a_{i_2} < a_{i_4}$.
7. Or $f(n)$ may be the number of labeled triangle-free graphs on the vertex set $[n]$.
8. In the binary words setting, if $\lambda = 0^{m_0} 1^{m_1} \dots (n-1)^{m_{n-1}}$, $m_i \in \mathbb{N}_0$, is a multiset with $m_0 + m_1 + \dots + m_{n-1} = n$, then $f(\lambda) \in \mathbb{N}_0$ counts the labeled simple graphs G on the vertex set $[n]$ such that for $i = 0, 1, \dots, n-1$ exactly m_i vertices of G have degree i . We encode the input λ as an element of $\{0, 1\}^*$ in an appropriate way which we will discuss later in part II.

This shows the variety of problems in enumeration one can consider and investigate. We say something on each of them from the perspective of Definition 1.1. Here we consider Examples 1–5 and defer the remaining ones to part II.

We put forward our definition of an effective formula for an enumerative problem. The acronym PIO stands for *polynomial in input and output*.

Definition 1.1 (PIO formula). *For a counting function $f : \mathbb{N} \rightarrow \mathbb{Z}$, a PIO formula is an algorithm, called a PIO algorithm, that for some constants $c, d \in \mathbb{N}$ for every input $n \in \mathbb{N}$ computes the output $f(n) \in \mathbb{Z}$ in at most $c \cdot m(n)^d = O(m(n)^d) = \text{poly}(m(n))$ steps, where*

$$m(n) = m_f(n) := \log(1+n) + \log(2 + |f(n)|)$$

measures the combined complexity of the input and the output. Similarly for counting function $f : X \rightarrow \mathbb{Z}$ defined on a subset $X \subset \mathbb{N}$.

We think this is the precise and definitive notion of a “closed formula”, and the yardstick one should use, possibly with some ramifications or weakenings, to determine if a solution to an enumerative problem is effective. We call counting functions possessing PIO formulas shortly *PIO functions*. Definition 1.1 repeats

(more explicitly) the proposal made already in M. Klazar [88, p. 10] in 2010, the innovation is that meanwhile we learned that the relevant complexity class PIO exists for a long time in the literature. In fact, after submitting this text for publication we found out that J. Shallit mentioned briefly Definition 1.1 as a “very good formula” in [133, slide 3] in 2016.

We comment on the definition. The steps mean steps of the formal specification of an algorithm as a multitape Turing machine. We are primarily interested in the *bit complexity* but sometimes consider also the *algebraic complexity*, the number of required arithmetic operations. We do not consider the *space complexity* which concerns memory requirements. The shifts $1 + n$ and $2 + |f(n)|$ serve for removing arguments 0 and 1, inconvenient for logarithms. The two natural logarithms come from the decadic or binary (but not unary!) encoding of numbers: $\log(2 + |f(n)|) = \Theta(r)$ where r is the number of bits in the binary code for $f(n) \in \mathbb{Z}$. The rationale behind the definition is that any algorithm solving a nontrivial enumerative problem $f : \mathbb{N} \rightarrow \mathbb{Z}$ needs minimum roughly $m(n)$ steps just for reading the input n and printing the output $f(n)$, and an effective algorithm takes only polynomially many steps in this minimum. We include the output in the complexity of the problem because in enumeration typically the output is much larger than the input, and thus considering only the input complexity (and ignoring the time it takes to print the output) is bound to lead to confusion. But we will see, and a moment of reflection reveals it, that not large outputs but on the contrary the unexpectedly small ones pose difficulty—for them one has much less time for effective computation. We reflect such “cancellative” problems by selecting \mathbb{Z} , and not \mathbb{N} or \mathbb{N}_0 , as the codomain of counting functions. In Example 1, $m(n) = \Theta(n)$ because $\log(2 + c_n) = \Theta(n)$, and in Example 2, $m(n) = \Theta(n)$ for even c_n and $m(n) = \Theta(\log(1 + n))$ when c_n is odd.

We state the definition of the complexity class PIO ([149, 65, 153]) which we above specialized to counting functions; $|u|$ denotes the length n of a binary word $u = a_1 a_2 \dots a_n \in \{0, 1\}^*$.

Definition 1.2 (complexity class PIO). *A function $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ is in the complexity class PIO if there is an algorithm that for every input $u \in \{0, 1\}^*$ computes the output $f(u)$ in time polynomial in $\max(|u|, |f(u)|)$.*

PIO belongs to standard complexity classes, see the “complexity zoo” [153], but it appears not to be widely known. It was introduced implicitly by M. Yannakakis [149, Theorem 5.1, also pages 86 and 93] and later explicitly and independently by Y. Gurevich and S. Shelah [65]. Researchers in database theory measure by it complexity of algorithms, see for example S. Cohen, B. Kimelfeld and Y. Sagiv [46], S. Cohen and Y. Sagiv [47], Y. Kanza and Y. Sagiv [85] or M. Vardi [143]. We are not aware of any mention of the class PIO in enumerative combinatorics where we think it has a natural place.

The title alludes to the pioneering work [146] of the late H. S. Wilf who was the first to ponder in the light of computational complexity the question what it precisely means to give an effective, or a nontrivial, solution—an answer—to a problem in enumeration. The first of two of his definitions says that a

nontrivial solution (he actually uses the term *effective solution*) is an algorithm that computes $f(n)$ for a counting function $f : \mathbb{N} \rightarrow \mathbb{N}_0$ in the number of steps that is $o(\text{List}(n))$ where “ $\text{List}(n)$ = the complexity of producing all of the members of the set S_n [where $f(n) = |S_n|$], one at a time, by the speediest known method, and counting them.” ([146, Definition 1 and the preceding sentence]). As it depends on the complexity of the current “speediest known method”, it depends on time and progress of knowledge, and so it is not really a mathematical definition but more a heuristic to measure effectivity of algorithms. In part II we give another specification of this notion. But we have to add here that this is not a bug but a feature of the first definition: “We will see that a corollary of this attitude is that our decision as to what constitutes an answer may be time-dependent: as faster algorithms for listing the objects become available, a proposed formula for counting the objects will have to be comparably faster to evaluate.” ([146, p. 289]).

The second definition of H. S. Wilf says that for superpolynomially growing $f(n)$ (“a problem in the class $\nu\pi$ ”) an *effective solution* (he actually uses the term that a problem is *p-solved*) is an algorithm that computes $f(n)$ in $\text{poly}(n)$ steps ([146, Definition 2]). We also have to mention that he does not restrict only to bit complexity but allows also other measures of complexity, “such as multiplication or division of numbers in a certain size range, or bit operations, or function evaluations, etc.” ([146, p. 290]).

Shortcomings of H. S. Wilf’s second definition, rectified in Definition 1.1, are that it does not take into account complexity of the output and restricts only to functions f with superpolynomial growth. (In the second definition the function $f(n)$ is not bounded from above, but it appears that tacitly it is of at most broadly exponential growth, $\log(1 + f(n)) \ll n^d$.) Thus Example 2 falls outside his framework, which is kind of unsatisfactory, and so we sought better definition. H. S. Wilf illustrates his two definitions with the function (we quote from [146])

$$f(n) = \sum_{\lambda \in P(n)} \frac{2^{g(\lambda)}}{1^{m_1} \cdot m_1! \cdot 2^{m_2} \cdot m_2! \cdot \dots \cdot n^{m_n} \cdot m_n!} \quad \text{where}$$

$$g(\lambda) = \frac{1}{2} \left(\sum_{i,j=1}^n \gcd(i,j) m_i m_j - \sum_{k \geq 1} m_{2k-1} \right)$$

and $\lambda = 1^{m_1} 2^{m_2} \dots n^{m_n}$ runs through the partitions of n — $f(n)$ counts the unlabeled (i.e., nonisomorphic) graphs on the vertex set $[n]$ (G. Pólya [118]). Since $f(n) \sim 2^{n(n-1)/2}/n!$, we have $\text{List}(n) = \Omega(2^{(1+o(1))n^2/2})$, but the formula based on the displayed sum over $P(n)$ computes $f(n)$ in $O(p(n)n^d) = O(\exp(c\sqrt{n}))$ steps (where $c > 0$ is a constant and $p(n) = |P(n)|$ is the number of partitions of n), which is asymptotically much smaller. Thus we have a nontrivial solution. An effective solution is in question because no algorithm is known that would compute $f(n)$ in $O(n^d)$ steps. H. S. Wilf asks if such algorithm exists, but so far his question remains unanswered.

H. S. Wilf’s article [146] is discussed by D. Zeilberger [151] and it gave rise to

the notion of a *Wilfian formula* (also called a *polynomial enumeration scheme*), which is an algorithm working in time polynomial in n , for enumerative problems $n \mapsto f(n) \in \mathbb{N}_0$ of the type $\Omega(n^c) = \log(2 + f(n)) = O(n^d)$ (for some real constants $0 < c < d$). It appears in the works on enumeration of Latin squares by D.S. Stones [140] or permutations with forbidden patterns by V. Vatter [144], B. Nakamura and D. Zeilberger [109], B. Nakamura [108] and others. Recently [146] was invoked by V.S. Miller [104] for counting squares in $F_2^{n \times n}$ by a nontrivial formula making exponentially many steps (an effective solution is not known) or by M. Kauers and D. Zeilberger [86].

Perhaps our proposal in Definition 1.1 has a certain reinventing-the-wheel quality because nowadays, unlike in the times of, say, L. Comtet [48] or J. Riordan [126] when computational complexity did not exist, it is a common knowledge that an effective solution to a problem means, in the first approximation, a polynomial time algorithm (see, for example, B. Edixhoven and J.-M. Couveignes [51] or V. Becher, P. A. Heiber and T. A. Slaman [14]). For this common knowledge we are indebted to A. Cobham [45] and J. Edmonds [52] in 1965. But, then, it seems not to be a common knowledge that one should consistently include the complexity of the output in the complexity of an enumerative problem. Also, one can still read in contemporary literature on enumerative combinatorics statements to the effect that there is no precise definition or determination of a closed formula or answer to an enumerative problem. For example, M. Aigner [3, Introduction, p. 1] writes that “There is no straightforward answer as to what “determining” a counting function means.” or F. Ardila [10, Chapter 1 What is a good answer?] concludes that “**So what is a good answer to an enumerative problem?** [emphasize in original] Not surprisingly, there is no definitive answer to this question.” On the other hand, the text of P. J. Cameron [33, 34, Chapter 1.3] contains an interesting discussion of the complexity matters which does reflect the output complexity of enumerative problems, but does not result in concrete definition of a closed formula. We believe that Definition 1.1 gives the definitive and more or less straightforward answer. Recently (I wrote the main bulk of this article in autumn 2016 and add this in March 2018, and now see that it is even August) the excellent survey [112] of I. Pak appeared that also tackles the question what is an effective formula in enumeration but it changes nothing on our above discussion.

Content and main results. In the following two sections we discuss, from the perspective of Definition 1.1, in their order the eight examples given at the beginning. At least, this we initially intended but as the text started get too long, we decided to split it in two parts. Here we consider Examples 1–5 and defer Examples 6–8 to part II. The length of our text is caused only by the great variety of enumerative problems offering themselves for investigation. Section 2 deals with Examples 1–3. After establishing in Propositions 2.1 and 2.2 PIO formulas for Examples 1 and 2, where Example 2 is chosen to illustrate peculiarity of this notion, we prove in Theorem 2.3 that every integral linear recurrence sequence has a PIO formula (but with a non-effective complexity bound). This seems so far not to be reflected in the literature. Propositions 2.4 and 2.6–2.8 gather tools for proving Theorem 2.3. We present some problems

and results on holonomic sequences which generalize linear recurrence sequences. For example, in Problem 2.10 we ask if every holonomic sequence $f : \mathbb{N} \rightarrow \mathbb{Z}$ is a PIO function.

In Section 3 we consider enumerative problems inspired by Examples 4 and 5. Propositions 3.1 and 3.2 revisit efficient evaluation of the partition function $p(n)$. It still holds from us some secrets, for example, it is not known how to compute efficiently the parity of $p(n)$ (Problem 3.3). In Proposition 3.4 we show that counting functions like Example 4 are PIO functions and in Proposition 3.5 we prove it for partitions with distinct parts. Hence Corollary 3.6: compositions of n with distinct parts are counted by a PIO function. Proposition 3.8 is a general result implying that, for example, the partitions of n whose multiplicities of parts divide n are counted by a PIO function. If parts are required to divide n , PIO formula seems not to be known. Another corollary is Corollary 3.10: if $g : \mathbb{N} \rightarrow \mathbb{N}$ strictly increases, grows only polynomially and is polynomial-time computable then the partitions of n with parts in $\{g(1), g(2), \dots\}$ are counted by a PIO function. Yet another Corollary 3.11 shows that the number of partitions of n into distinct squares is a PIO function. Corollary 3.13 gives PIO formulas for counting functions of partitions with prescribed multiplicities. Problem 3.14, inspired by H. S. Wilf [147], asks if we can effectively count partitions of n with distinct multiplicities of parts. The well known theorem of E. T. Bell [15] (Proposition 3.15) says that partitions of n that take parts from a fixed finite set $A \subset \mathbb{N}$ are counted by a quasipolynomial in n . In Corollaries 3.17 and 3.18 we point out that the argument proving Proposition 3.15 gives with almost no change more general results. Corollary 3.19 returns to Example 4: if the function $g : \mathbb{N} \rightarrow \mathbb{Z}$ has a finite support then $f(n) = \sum_{\lambda \in P(n)} g(\|\lambda\|)$ is a PIO function because it is a quasipolynomial in n . We mention further quasipolynomial enumerative results on partitions, Proposition 3.20 due to D. Zeilberger [152] and Proposition 3.21 due to G. E. Andrews, M. Beck and N. Robbins [8]. Rather general quasipolynomial enumerative result was obtained by T. Bogart, J. Goodricks and K. Woods [19] (Theorem 3.22). Problem 3.23 asks if one can effectively count partitions of n into powers of a fixed integer $m \geq 2$ and Theorem 3.25 quotes a recent positive resolution of this problem by I. Pak and D. Yeliussizov [113, 114]. Proposition 3.26 points out that the two basic cancellative counting problems on partitions, $n \mapsto \sum_{\lambda \in Q(n)} (-1)^{\|\lambda\|}$ and $n \mapsto \sum_{\lambda \in P(n)} (-1)^{\|\lambda\|}$ (where $P(n)$ are all partitions of n , and $Q(n)$ are those with distinct parts), are both PIO functions. The former follows from the pentagonal identity of L. Euler, and the latter from J. W. L. Glaisher's identity [62]. In the former case we have almost complete cancellation but in the latter case only little cancellation. Problem 3.27 asks when such cancellation for $(-1)^{\|\lambda\|}$ -count of partitions occurs. We look at this problem for partitions into squares and for partitions into parts from l -sorted \mathbb{N} (Example 5 is $l = 2$), including the case $l = 24$ that gives the Ramanujan tau function — the last Problem 3.29 concerns computation of coefficients in powers of R. Dedekind's η -function.

When we below state and prove results on PIO functions for enumerative problems, we are not content with just saying that a PIO algorithm for the prob-

lem exists — it would be ironic to refer in such a way to efficient algorithms — but we always indicate if and how the PIO algorithm can be constructed from the given data. See, for example, Theorem 2.3 or Corollary 3.10.

2 The numbers of Catalan and Fibonacci

We start with Example 1. A *planted tree* is a finite tree with a distinguished vertex, called a root, and with every set of children of a vertex linearly ordered. Its size is the number of vertices. (For a long time I used to call this kind of trees *rooted plane trees*, which also some literature uses, but F. Bergeron, G. Labelle and P. Leroux [16] showed me that this terminology is imprecise: embedding in the plane gives to the children of the root only cyclic, not linear, ordering.)

Proposition 2.1. *Let c_n be the n -th Catalan number, the number of (unlabeled) planted trees with size n . Then $n \mapsto c_n$ is a PIO function and the PIO algorithm is given by the recurrence displayed below.*

Proof. We only need to know the recursive structure of planted trees. There is just one planted tree with size 1, and for $n \geq 2$ every planted tree T of size n bijectively decomposes in an ordered pair (U, V) of planted trees with sizes adding to n ; U is the subtree of T rooted in the first child of T 's root and V is the rest of T . Thus the combinatorial recurrence $c_1 = 1$ and, for $n \geq 2$,

$$c_n = \sum_{k=1}^{n-1} c_k c_{n-k} .$$

It implies that $c_n \geq 2c_{n-1}$ for $n \geq 3$ and by induction $c_n \gg 2^n$. On the other hand, by induction $c_n \leq (n-1)! \leq n^n$ for every $n \geq 1$ and $\log(2 + c_n) \ll n^2$ (such crude but easy to obtain bound suffices for our purposes). Hence $n \ll m(n) \ll n^2$ in this enumerative problem and we need to compute c_n in $O(n^d)$ steps.

We do it on a Turing machine with six tapes T_1, \dots, T_6 . Recall that elementary school algorithms multiply two $O(m)$ -bit numbers in $O(m^2)$ steps and add them in $O(m)$ steps (see comments below). Tape T_1 stores, in this order, the binary codes for c_1, c_2, \dots, c_{n-1} . For $k = 1, 2, \dots, n-1$ we do the following. We find c_k and c_{n-k} on T_1 and write them on the respective tapes T_2 and T_3 . This costs $O(\log(1 + c_1) + \dots + \log(1 + c_{n-1})) = O(n^3)$ steps, say. We compute in $O(n^4)$ steps the product $c_k c_{n-k} =: s$ and write it on T_4 . Tape T_5 stores $\sum_{i=1}^{k-1} c_i c_{n-i} =: t$. In $O(n^2)$ steps we compute the sum $s + t$ and store it on T_6 . We conclude by rewriting in $O(n^2)$ steps the content of T_5 with that of T_6 . After the step $k = n-1$, tape T_5 contains $t = \sum_{i=1}^{n-1} c_i c_{n-i} = c_n$ and we copy this in $O(n^2)$ steps on T_1 . The computation of c_n from c_1, c_2, \dots, c_{n-1} takes $O(n \cdot n^4) = O(n^5)$ steps. The recurrence, implemented by the six-tape Turing machine, computes c_n from beginning in $\sum_{k=2}^n O(k^5) = O(n^6) = O(m(n)^6)$ steps, and $n \mapsto c_n$ is a PIO function. \square

We give some comments. If the Turing machine has only one tape, and the two $O(m)$ -bit numbers to be added are stored on it one after another, it is impossible to add them in $O(m)$ steps as the reading head has to move back and forth between them, and one needs $\Theta(m^2)$ steps for addition. This can be proven similarly as F. C. Hennie [71] proved the $\Theta(m^2)$ lower bound on recognition of m -bit palindromes. Therefore we use multitape Turing machines. The cost of adding or multiplying two numbers is not only the cost of the operation but in practice includes also the cost of recalling both operands from the memory, and therefore we analyzed above the computation of c_n in more details. But these technicalities cause at worst only polynomial slowdown and are not important for our main concern that is a purely qualitative alternative: there is a polynomial time PIO algorithm for the enumerative problem considered or its existence is not known.

The Catalan numbers

$$(c_n)_{n \geq 1} = (1, 1, 2, 5, 14, 42, 132, 429, 1430, 4862, \dots)$$

form sequence A000108 in the database OEIS (The Online Encyclopedia of Integer Sequences) [154]. Using stronger bound $\log(2 + c_n) = \Theta(n)$ and faster integer multiplication (consult D. Harvey, J. van der Hoeven and G. Lecerf [70] and D. Harvey and J. van der Hoeven [69] for the state of art and history of multiplication algorithms) we can evaluate c_n in, say, $O(n^3 \log^d(1 + n))$ steps. It is not a surprise that Catalan numbers can be effectively computed, but it depends on what “effectively” precisely means. The combinatorial recurrence computes c_n in $\text{poly}(n)$ arithmetic operations, and to get $\text{poly}(n)$ steps we need that all numbers involved in the computation have $O(n^d)$ digits for a fixed d . This is ensured by (i) the bound $c_n \leq n^n$ and (ii) the non-negativity of coefficients in the recurrence which entails that the result c_n upper bounds every number arising in evaluating the recurrence. But we also need that each c_n has $\Omega(n^c)$ digits for a fixed real $c > 0$, which is ensured by the bound $c_n \gg 2^n$, so that $\text{poly}(n)$ steps means $\text{poly}(m(n))$ steps and we really have an effective algorithm. If, say, for infinitely many n we had the bound $c_n = O(1)$ then $\text{poly}(n)$ steps would cease to mean an effective algorithm in the sense of Definition 1.1 and we would have to try more, as in Example 2.

To establish qualitatively a PIO formula for c_n , the combinatorial recurrence suffices and one does not need generating functions or “advanced” formulas for c_n like ($n \in \mathbb{N}$)

$$c_n = \frac{1}{n} \binom{2n-2}{n-1} = (-1)^{n+1} \frac{4^n}{2} \binom{\frac{1}{2}}{n} \quad \text{or} \quad c_{n+1} = \frac{4n-2}{n+1} c_n.$$

The last recurrence gives a more efficient PIO formula than the combinatorial recurrence. The Catalan numbers have asymptotics $c_n \sim cn^{-3/2}4^n$ with a constant $c > 0$. For asymptotic methods in enumeration consult P. Flajolet and R. Sedgewick [58], and R. Pemantle and M. C. Wilson [117] for the multivariate case.

We move to Example 2. As we will see shortly, the case $f(n) = 1$ occurs for infinitely many n . Then the reader probably realizes that even though the two functions f_c and f_o , defined as $f_c(n) := c_n$ and $f_o(n) := n$ for even n and $f_o(n) := 1$ for odd n , are PIO functions and compose to the counting function $f(n) = f_o(f_c(n))$ of Example 2, composition of their PIO algorithms is not a PIO algorithm for $f(n)$. It is an algorithm that always does $\Omega(n^d)$ steps, but we need an algorithm that for n with odd c_n makes only $O(\log^d(1+n))$ steps. Naturally, we need to determine effectively which numbers c_n are odd.

Proposition 2.2. *Let c_n be the n -th Catalan number. Then the function $f : \mathbb{N} \rightarrow \mathbb{N}$, $f(n) = c_n$ for even c_n and $f(n) = 1$ for odd c_n , is a PIO function. The PIO algorithm is described below.*

Proof. The combinatorial recurrence for c_n shows that c_n is odd iff $n = 2^m$ for an $m \in \mathbb{N}_0$: $c_1 = 1$ is odd, for odd $n > 1$ the number $c_n = 2(c_1c_{n-1} + \dots + c_{(n-1)/2}c_{(n+1)/2})$ is even and, similarly, for even n the number $c_n = c_{n/2}^2 + 2(c_1c_{n-1} + \dots + c_{(n-2)/2}c_{(n+2)/2})$ has the same parity as $c_{n/2}$. Thus we effectively compute $f(n)$ as follows. For given $n \in \mathbb{N}$ we first in $O(\log^2(1+n))$ steps determine if n is a power of 2. If it is so, we output 1 in $O(1)$ steps. Else we output, using the PIO formula for $n \mapsto c_n$, in $O(n^6)$ or so steps the value c_n . This gives a formula for $f(n)$ that for even c_n makes $O(n^6)$ steps and for odd c_n only $O(\log^2(1+n))$ steps, which is $O(m(n)^6)$ steps for every $n \in \mathbb{N}$. \square

Alternatively, we get a PIO algorithm for the $f(n)$ of Proposition 2.2 or, more generally, we determine effectively if a fixed $m \in \mathbb{N}$ divides c_n (or, more generally, a hypergeometric term), by means of A.-M. Legendre's formula $k = \nu_p(n!) = \sum_{j \geq 1} \lfloor n/p^j \rfloor$ for the largest $k \in \mathbb{N}_0$ for which p^k divides $1 \cdot 2 \cdot \dots \cdot n$ (or by means of a generalization of the formula to products of numbers in arithmetic progressions). At the close of the section we mention other effective formulas for modular reductions of c_n and similar numbers.

Example 2 illustrates the fact that composition of two PIO algorithms need not be a PIO algorithm. In fact, as one expects, composition of two PIO functions need not be a PIO function. An example of such functions is easily constructed by taking a computable function not in PIO, e.g. a function $f : \{0, 1\}^* \rightarrow \{0, 1\}$ in $\text{EXP} \setminus \text{P}$, see Ch. H. Papadimitriou [115, Chapter 7.2]. Y. Gurevich and S. Shelah [65, Lemma 2.1] elaborate such example.

Example 3 concerns the ubiquitous linear recurrence sequences. Best known of them are the *Fibonacci numbers* f_n , given by $f_0 = 0, f_1 = 1$, and $f_{n+2} = f_{n+1} + f_n$ for every $n \in \mathbb{N}_0$. The sequence

$$(f_n) = (f_n)_{n \geq 1} = (1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, \dots)$$

is sequence [154, A000045]. Since $2f_{n-2} \leq f_n \leq 2f_{n-1}$ for $n \geq 3$, we have exponential bounds $(\sqrt{2})^n \ll f_n \ll 2^n$, $n \in \mathbb{N}$. The precise asymptotics is $f_n \sim c\phi^n$ where $c > 0$ is a constant and $\phi = 1.61803\dots$ satisfies $\phi^2 - \phi - 1 = 0$.

In general, a *linear recurrence sequence in \mathbb{Z}* , abbreviated as a *LRS*, is a function $f : \mathbb{N} \rightarrow \mathbb{Z}$ determined by $2k$ integers $a_0, \dots, a_{k-1}, f(1), \dots, f(k)$ with $k \in \mathbb{N}_0$ and $a_0 \neq 0$, and the recurrence relation

$$f(n+k) = a_{k-1}f(n+k-1) + a_{k-2}f(n+k-2) + \dots + a_0f(n), \quad n \in \mathbb{N}.$$

(Later we point out that allowing a_i outside \mathbb{Z} does not give new LRS.) Note that we require $a_0 \neq 0$ and the recurrence to hold from the beginning. By reverting the recurrence every LRS f extends naturally to $f : \mathbb{Z} \rightarrow \mathbb{Q}$, for example $(1, 2, 4, 8, \dots)$ extends to $f(n) = 2^{n-1}$, $n \in \mathbb{Z}$. Thus $(0, 1, 1, 1, \dots)$ is *not* a LRS, as can be seen by reverting the purported recurrence, but it is true that this sequence differs from a LRS in just one term. If $k = 0$ or if $f(1) = f(2) = \dots = f(k) = 0$, we get the *zero sequence* that has $f(n) = 0$ for every $n \in \mathbb{N}$. We say that f (more precisely, the recurrence) has *order k* . Like the Fibonacci numbers, every LRS has an easy exponential upper bound $|f(n)| \leq c^n$ for every $n \in \mathbb{N}$, with the constant $c = k \max_i |a_i| \max_{i \leq k} |f(i)|$. Lower bound is a different story, see Proposition 2.7.

The defining recurrence computes every LRS $f(n)$ in $\Theta(n)$ arithmetic operations. We recall, on the Fibonacci numbers f_n , the well known and beautiful formula (algorithm) based on binary powering that computes $f(n)$ in only $\text{poly}(\log n)$ arithmetic operations. By E. Bach and J. Shallit [11, p. 122] or D.E. Knuth [91, p. 695], it appears first in J.C.P. Miller and D.J. Spencer Brown [103]. For f_n the formula reads: if $n = \sum_{i=0}^k b_i 2^i$ with $b_i \in \{0, 1\}$ is the binary expansion of $n \in \mathbb{N}_0$ (where $0 = 02^0$) then

$$f_n = (0, 1) \cdot \prod_{i=0}^k \underbrace{\left(\dots \left(\left(\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^{b_i} \right)^2 \right)^2 \dots \right)^2}_{i \text{ squarings}} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

Indeed, if M is the stated 2×2 matrix and F_n is the column $(f_{n+1}, f_n)^T$ then, since the first row of M records the recurrence for f_n and matrix multiplication is associative, we have $MF_n = F_{n+1}$ and $F_n = M^n F_0$. The power M^n is then computed by repeated squaring from the b_i s. Since $k = O(\log(n+1))$, the formula computes f_n in only $O(\log(n+1))$ multiplications of two integral 2×2 matrices, so in only $O(\log(n+1))$ arithmetic operations with integers. This easily extends to general LRS. For more information on repeated squaring and computing terms in linear recurrence sequences see J. von zur Gathen and J. Gerhard [61, Chapters 4.3 and 12.3]. Recent contribution to the literature on computing linear recurrence sequences is S. G. Hyun, S. Melczer and C. St-Pierre [77].

For qualitative bit complexity such cleverness seems superfluous, the n -th term $f(n)$ of a LRS is an $O(n)$ digit number (which cannot be printed in fewer steps) and already the defining recurrence computes $f(n)$ in $\text{poly}(n)$ steps, which might be viewed as an efficient computation. But not from the point of view of Definition 1.1. Since $f(n)$ may have as few as $O(1)$ digits, to get a PIO

formula we need to locate effectively these small values (cf. Example 2) and compute them in $\text{poly}(\log n)$ steps. This is not automatic by the above displayed $\text{poly}(\log n)$ arithmetic operations formula because only $\text{poly}(\log n)$ digit numbers may be used, and for general LRS the displayed formula contains negative numbers (the result need not upper bound intermediate values, as in $1 = 2^n - (2^n - 1)$). It can be done, albeit on the verge of non-effectivity, and we prove the following theorem.

Theorem 2.3. *Every linear recurrence sequence $f : \mathbb{N} \rightarrow \mathbb{Z}$ in \mathbb{Z} of order $k \in \mathbb{N}_0$ is a PIO function. In the proof we indicate an algorithm \mathcal{A} with inputs $(a, b, n) \in \mathbb{Z}^k \times \mathbb{Z}^k \times \mathbb{N}$, where $k \in \mathbb{N}_0$, and outputs in \mathbb{Z} such that for every fixed tuple $(a, b) = (a_0, \dots, a_{k-1}, f(1), \dots, f(k))$ with $a_0 \neq 0$, \mathcal{A} is a PIO algorithm computing the LRS $f(n)$ determined by (a, b) .*

We will see that the implicit constant in the $O(m(n)^d)$ complexity bound in these PIO algorithms is currently non-effective — at the present state of knowledge we cannot provide for it any specific value. The theorem follows by combining standard results from the theory of linear recurrence sequences, see the monograph [54] of G. Everest, A. van der Poorten, I. Shparlinski and T. Ward, but we did not find it mentioned in [54] or anywhere else.

We review tools for the proof of Theorem 2.3. For background on linear recurrence sequences see [54], W. M. Schmidt [131] or R. P. Stanley [138, Chapter 4]. By $\overline{\mathbb{Q}} \subset \mathbb{C}$ we denote the field of *algebraic numbers*, consisting of all roots of monic polynomials from $\mathbb{Q}[x]$. The subring of *algebraic integers* is formed by all roots of monic polynomials from $\mathbb{Z}[x]$. A *power sum* is an expression

$$s(x) = \sum_{i=1}^l p_i(x) \alpha_i^x$$

where $l \in \mathbb{N}_0$, $\alpha_i \in \overline{\mathbb{Q}}$ are distinct and nonzero numbers, and $p_i \in \overline{\mathbb{Q}}[x]$ are nonzero polynomials. The numbers α_i are the *roots* of the power sum. A sequence $f : \mathbb{N} \rightarrow \overline{\mathbb{Q}}$ is represented by a power sum $s(x)$ if $f(n) = s(n)$ for every $n \in \mathbb{N}$. The *empty power sum* with $l = 0$ represents the zero sequence. We consider the more general *linear recurrence sequences in \mathbb{Q}* , shortly *LRS in \mathbb{Q}* — they are defined as LRS, only their values and coefficients of defining recurrences lie in the field \mathbb{Q} instead of its subring \mathbb{Z} . If $f : \mathbb{N} \rightarrow \mathbb{Q}$ is a LRS in \mathbb{Q} given by a recurrence $f(n+k) = \sum_{i=0}^{k-1} a_i f(n+i)$, $a_i \in \mathbb{Q}$ and $a_0 \neq 0$, the *recurrence polynomial* $p(x)$ is

$$p(x) = x^k - a_{k-1}x^{k-1} - a_{k-2}x^{k-2} - \dots - a_0 \in \mathbb{Q}[x].$$

If the recurrence for f has minimum order, we denote $p(x)$ by $p_f(x)$ and call it the *characteristic polynomial of f* ; in a moment we show that there is always a unique $p_f(x)$. For example, the characteristic polynomial of the zero sequence is the constant polynomial $p_{\equiv 0}(x) = 1$. For a sequence $f : \mathbb{N} \rightarrow \mathbb{C}$ and a polynomial $p(x) = \sum_{i=0}^k a_i x^i \in \mathbb{C}[x]$ we let $pf : \mathbb{N} \rightarrow \mathbb{C}$ denote the sequence

given by $pf(n) = \sum_{i=0}^k a_i f(n+i)$. We say that p annihilates f if pf is the zero sequence. The set of rational polynomials annihilating f is denoted by $V(f) \subset \mathbb{Q}[x]$. Clearly, $V(f)$ consists exactly of all rational recurrence polynomials for f (with $a_0 = 0$ and non-unit leading coefficient allowed). The following results are well known but we prove them here for reader's convenience and as an workout for the author.

Proposition 2.4. *Power sums and linear recurrence sequences have the following properties.*

1. Every sequence $f : \mathbb{N} \rightarrow \overline{\mathbb{Q}}$ has at most one power sum representation.
2. If $f : \mathbb{N} \rightarrow \mathbb{C}$ is a sequence then $V(f)$ is an ideal in the ring $\mathbb{Q}[x]$.
3. If $f : \mathbb{N} \rightarrow \mathbb{Q}$ is a LRS in \mathbb{Q} then $V(f) = \langle p_f(x) \rangle$ for a unique monic polynomial $p_f \in \mathbb{Q}[x]$ with $p_f(0) \neq 0$. This unique generator p_f of $V(f)$ is called the characteristic polynomial of f and gives the unique minimum order rational recurrence for f .
4. A sequence $f : \mathbb{N} \rightarrow \mathbb{Q}$ is a LRS in \mathbb{Q} if and only if it is represented by a power sum $s(x)$. If it is the case, the roots α_i of $s(x)$ are exactly the roots of $p_f(x)$.

Proof. 1. It suffices to show that no nonempty power sum represents the zero sequence. For a power sum $s(x)$ as above we define $\deg s(x) = \sum_{i=1}^l (\deg p_i(x) + 1) \in \mathbb{N}_0$. Clearly, only the empty power sum has degree 0. For any nonempty $s(x)$ we define the new power sum $\Delta s(x) = s(x+1) - \alpha_1 s(x)$. Note that, crucially, $\deg \Delta s(x) = \deg s(x) - 1$. Also, if $s(x)$ represents the zero sequence then so does $\Delta s(x)$, and no $s(x)$ with $\deg s(x) = 1$ represents the zero sequence. (In fact, if $\deg s(x) = 1$ then $s(n) \neq 0$ for every $n \in \mathbb{N}$.) The last three facts together imply that no nonempty power sum represents the zero sequence.

2 and 3. It is easy to see that for any $p, q \in \mathbb{C}[x]$ and any sequence $f : \mathbb{N} \rightarrow \mathbb{C}$ we have $(p+q)f = pf + qf$ and $(pq)f = p(qf)$. Thus $V(f)$ is an ideal in $\mathbb{Q}[x]$. Every ideal in $\mathbb{Q}[x]$ is principal, is generated by a single element, because the ring $\mathbb{Q}[x]$ is Euclidean. Requiring the generator monic makes it unique because the units of $\mathbb{Q}[x]$ are exactly the nonzero constants. Finally, $p_f(0) \neq 0$ because f being a LRS in \mathbb{Q} implies that $V(f)$ contains a p with $p(0) \neq 0$.

4. Suppose that $f : \mathbb{N} \rightarrow \mathbb{Q}$ is a LRS in \mathbb{Q} with minimum order k and characteristic polynomial $p_f(x)$. Thus in the ring of formal power series $\mathbb{Q}[[x]]$ we have the equality

$$\sum_{n \geq 0} f(n)x^n = \frac{q(x)}{q_f(x)}$$

where $q_f(x) = x^k p_f(1/x)$, $q \in \mathbb{Q}[x]$ has degree $< k$ and $q(x)$ and $q_f(x)$ are coprime (by the minimality of k). The value $f(0)$ is computed from $f(1), f(2), \dots, f(k)$ by the reverted recurrence; now it would be more convenient if f had domain \mathbb{N}_0 or even \mathbb{Z} but counting functions have domain \mathbb{N} . After

decomposing the rational function $q(x)/q_f(x)$ in partial fractions, expanding them in $\overline{\mathbb{Q}}[[x]]$ in generalized geometric series, and comparing coefficients of x^n , we get a representation of $f(n)$ by a power sum $s(x)$. The roots of $s(x)$ are exactly the roots of $p_f(x)$ because of the coprimality of $q(x)$ and $q_f(x)$.

Let $f : \mathbb{N} \rightarrow \mathbb{Q}$ be represented by a power sum $s(x) = \sum_{i=1}^l p_i(x)\alpha_i^x$: $f(n) = s(n)$ for every $n \in \mathbb{N}$. We may assume that f is not the zero sequence and so $s(x)$ is nonempty. We show that f is a LRS in \mathbb{Q} . The argument is of interest because of three invocations of Lemma 2.5 below and because it uses negative $n \in \mathbb{Z}$. If $d = \max_i \deg p_i(x)$ then $s(x)$ is a $\overline{\mathbb{Q}}$ -linear combination of the $t = (d+1)l$ expressions $x^j \alpha_i^x$ for $0 \leq j \leq d$ and $1 \leq i \leq l$. So is every shift $s(x+r)$ for $r \in \mathbb{N}$, as can be seen by expanding $(x+r)^j = \sum_{b=0}^j \binom{j}{b} r^{j-b} x^b$ and $\alpha_i^{x+r} = \alpha_i^r \alpha_i^x$. By Lemma 2.5 there exist coefficients $\beta_0, \beta_1, \dots, \beta_t \in \overline{\mathbb{Q}}$, not all zero, such that

$$\beta_0 s(x) + \beta_1 s(x+1) + \dots + \beta_t s(x+t) = 0$$

identically. But we need coefficients not only in $\overline{\mathbb{Q}}$ but in \mathbb{Q} . We set

$$V = \{(f(n), f(n+1), \dots, f(n+t)) \mid n \in \mathbb{N}\} \subset \mathbb{Q}^{t+1} \subset \overline{\mathbb{Q}}^{t+1}$$

and select a maximum subset $B \subset V$ of linearly independent (over $\overline{\mathbb{Q}}$) vectors. By Lemma 2.5, $|B| \leq t+1$. Every vector $z \in V$ is a $\overline{\mathbb{Q}}$ -linear combination of the vectors in B . If $|B| = t+1$, the matrix whose rows are the linearly independent vectors $z \in B$ is a square matrix and thus has linearly independent columns. But the system

$$z \cdot (x_0, x_1, \dots, x_t) = 0, \quad z \in B,$$

has a nontrivial solution $x_i = \beta_i \in \overline{\mathbb{Q}}$ which means that the columns are linearly dependent. Hence $|B| \leq t$. But $B \subset \mathbb{Q}^{t+1}$ and thus by Lemma 2.5 this system has a nontrivial solution $x_i = \gamma_i \in \mathbb{Q}$, with not all γ_i zero. So

$$z \cdot (\gamma_0, \gamma_1, \dots, \gamma_t) = 0 \quad \text{and} \quad \gamma_0 s(n) + \gamma_1 s(n+1) + \dots + \gamma_t s(n+t) = 0$$

for every $z \in V$ and every $n \in \mathbb{N}$. By part 1, then $\gamma_0 s(x) + \gamma_1 s(x+1) + \dots + \gamma_t s(x+t) = 0$ identically (the left side is the empty power sum) and the last displayed equality thus holds for every $n \in \mathbb{Z}$. Let $u \in \mathbb{N}_0$ and $v \in \mathbb{N}_0$ be the respective minimum and maximum index r with $\gamma_r \neq 0$, and let $w = v - u \in \mathbb{N}_0$. Then

$$\begin{aligned} & \gamma_v f(n+w) + \gamma_{v-1} f(n+w-1) + \dots + \gamma_u f(n) \\ &= \gamma_t s(n-u+t) + \gamma_{t-1} s(n-u+t-1) + \dots + \gamma_0 s(n-u) = 0 \end{aligned}$$

for every $n \in \mathbb{N}$ (the arguments of $s(\cdot)$ may be negative). After dividing by γ_v and rearranging we see that f is a LRS in \mathbb{Q} of order w . The roots of $p_f(x)$ and of $s(x)$ coincide by the previously proved opposite implication and by uniqueness of $s(x)$ proved in part 1. \square

Note that nonzero power sums may vanish for infinitely many $n \in \mathbb{N}$, for example $s(x) = 1^x + (-1)(-1)^x$ on $2\mathbb{N}$, and the first part is therefore more subtle result than for polynomials. Recalling the zeros of $\sin(\pi x)$ we have

$$\exp(\pi i x) - \exp(-\pi i x) = \exp(\pi i)^x + (-1) \exp(-\pi i)^x = 0 \text{ for every } x \in \mathbb{Z}$$

but this, of course, is not a counterexample to the first part (why?). Also,

$$f_n = \frac{1}{\sqrt{5}} \left(\frac{1 + \sqrt{5}}{2} \right)^n - \frac{1}{\sqrt{5}} \left(\frac{1 - \sqrt{5}}{2} \right)^n$$

is the familiar power sum representation of the Fibonacci numbers. The next lemma, used several times in the previous proof, is a well known result from linear algebra. Its proof is left to the interested reader as an exercise.

Lemma 2.5. *Let $m, n \in \mathbb{N}$ with $m < n$. Every linear homogeneous system*

$$a_{j,1}x_1 + a_{j,2}x_2 + \cdots + a_{j,n}x_n = 0_K, \quad j = 1, 2, \dots, m,$$

with m equations, n unknowns x_i , and coefficients $a_{j,i}$ in a field K has a non-trivial solution $x_i \in K$ with not all $x_i = 0_K$.

Recall that a *root of unity* is a number $\alpha \in \mathbb{C}$ such that $\alpha^k = 1$ for some $k \in \mathbb{N}$, i.e. α is a root of $x^k - 1$. The minimum such k is the *order* of α . We say that a power sum $s(x)$ is *degenerate* if some root α_i or some ratio α_i/α_j of two roots is a root of unity different from 1, else $s(x)$ is *non-degenerate*. So we allow 1 as a root in a non-degenerate power sum, and empty power sum is non-degenerate. For any sequence $f : \mathbb{N} \rightarrow X$ and numbers $m \in \mathbb{N}$ and $j \in [m]$, the m -section $f_{j,m} : \mathbb{N} \rightarrow X$ of f is the subsequence of values of f on the residue class $j \bmod m$:

$$f_{j,m}(n) = f(j + m(n - 1)), \quad n \in \mathbb{N}.$$

If $f : \mathbb{N} \rightarrow \overline{\mathbb{Q}}$ is represented by a power sum $s(x) = \sum_{i=1}^l p_i(x) \alpha_i^x$, then the m -section $f_{j,m}$ is represented by the power sum

$$s_{j,m}(x) = \sum_{i=1}^l \alpha_i^{j-m} p_i(j - m + mx) (\alpha_i^m)^x = \sum_{i=1}^r q_i(x) \beta_i^x$$

where $r \leq l$ and $\{\beta_i \mid i = 1, \dots, r\} \subset \{\alpha_i^m \mid i = 1, \dots, l\}$ —we collect like terms in the middle expression so that the numbers β_i are distinct and the polynomials $q_i(x)$ nonzero. For example, the degenerate power sum $s(x) = 2^x + (-2)^x$ has 2-sections $s_{1,2}(x) = 0$ (the empty power sum with $r = 0$) and $s_{2,2}(x) = 2 \cdot 4^x$. It is not hard to prove that for any $m \in \mathbb{N}$, $f : \mathbb{N} \rightarrow \mathbb{Q}$ is a LRS in \mathbb{Q} if and only if every m -section $f_{j,m}$ is a LRS in \mathbb{Q} .

Proposition 2.6. *The following holds for roots of unity and power sums.*

1. *If $p \in \mathbb{Z}[x]$ is a monic polynomial with $p(0) \neq 0$ and every root of p has modulus at most 1, then every root of p is a root of unity.*

2. If $s(x) = \sum_{i=1}^l p_i(x)\alpha_i^x$ is a power sum such that every α_i is an algebraic integer, $|\alpha_i| \leq 1$ for every i , and $s(n) \in \mathbb{Q}$ for every $n \in \mathbb{N}$, then every α_i is a root of unity.

Proof. 1. This is called Kronecker's theorem. See U. Zannier [150, Theorem 3.8 and Remark 3.10 (i)] or E. Bombieri and W. Gubler [21, Theorem 1.5.9] or V. V. Prasolov [123, Theorem 4.5.4].

2. By the assumption and part 4 of Proposition 2.4, the sequence $f(n) = s(n)$, $n \in \mathbb{N}$, is a LRS in \mathbb{Q} . By parts 3 and 4 of Proposition 2.4, the numbers α_i are exactly the roots of the characteristic polynomial $p_f(x) \in \mathbb{Q}[x]$. Since all α_i are algebraic integers, so are the coefficients of $p_f(x)$ (by expressing them in terms of the α_i s). But this implies that $p_f(x) \in \mathbb{Z}[x]$. Using part 1 of the present proposition, we get that all α_i are roots of unity. \square

On the Internet or even in paper literature one can encounter the erroneous claim that if $\alpha \in \overline{\mathbb{Q}}$ with $|\alpha| = 1$ then α is a root of unity. The number $\frac{4+3i}{5}$ is a counterexample. Part 1 of Proposition 2.6 shows when arguments of this sort are correct. Concerning exponential lower bounds on growth of linear recurrence sequences, there is the next deep result.

Proposition 2.7. *If $f : \mathbb{N} \rightarrow \overline{\mathbb{Q}}$ is represented by a non-degenerate power sum whose roots have maximum modulus $\beta > 1$, then for every $\varepsilon > 0$ there is an $n_0 \in \mathbb{N}$ such that*

$$|f(n)| > \beta^{(1-\varepsilon)n} \text{ for every } n > n_0 .$$

Proof. This is [54, Theorem 2.3] where the proof is omitted. At [54, p. 32] the result is attributed to J.-H. Evertse [55] and independently A. van der Poorten and H. P. Schlickewei [122]. J.-H. Evertse [55, p. 229] attributes it to A. van der Poorten [120]. See also A. van der Poorten [121]. \square

We deduce the following growth dichotomy for LRS that effectively separates small and large values.

Proposition 2.8. *Suppose $f : \mathbb{N} \rightarrow \mathbb{Z}$ is a LRS of order $k \in \mathbb{N}_0$, represented by a power sum $s(x)$. We let $m \in \mathbb{N}$ be the least common multiple of the orders of the roots of unity among the roots α_i of $s(x)$ and their ratios α_i/α_j , and let $J \subset [m]$ be the set of $j \in \mathbb{N}$ for which the power sum $s_{j,m}(x)$ is empty or has the single root 1. Then there exist a real constant $c > 1$ and an $n_0 \in \mathbb{N}$ such that for every $j \in [m]$ the following holds.*

1. If $j \in J$ then $f_{j,m}(n)$ is a rational polynomial in $n \in \mathbb{N}$ with degree less than k .
2. If $j \notin J$ then $|f_{j,m}(n)| > c^n$ for every $n > n_0$.

Proof. By parts 3 and 4 of Proposition 2.4, all roots α_i of $s(x)$ are algebraic integers. Take a $j \in [m]$ and consider the power sum $s_{j,m}(x) = \sum_{i=1}^r q_i(x)\beta_i^x$ representing $f_{j,m}(n)$. The β_i s are m -th powers of α_i s and are algebraic integers too. Also, $s_{j,m}(x)$ is non-degenerate. Suppose that $|\beta_i| \leq 1$ for every $i = 1, 2, \dots, r$. By part 2 of Proposition 2.6 all β_i are roots of unity. But then non-degeneracy of $s_{j,m}(x)$ implies that either $r = 0$, $s_{j,m}(x)$ is empty and $f_{j,m}$ is the zero sequence, or $r = 1$, $\beta_1 = 1$ and $f_{j,m}(n) = q_1(n) \in \mathbb{Z}$ for every $n \in \mathbb{N}$. It follows that $q_1 \in \mathbb{Q}[x]$ and $\deg q_1 \leq \max_i \deg p_i < k$ (p_i are the polynomials in $s(x)$). Thus we get the first case with $j \in J$. If $|\beta_i| > 1$ for some i , Proposition 2.7 gives the second case with $j \notin J$. \square

Unfortunately, currently no proof of Proposition 2.7 is known giving an explicit upper bound on the threshold n_0 , only its existence is proven. Therefore also the n_0 of Proposition 2.8 is non-effective (we cannot compute it). Effective versions of much weaker inequalities are not known. Already T. Skolem [135] proved that if $f(n)$ is a nonzero LRS represented by a non-degenerate power sum then $|f(n)| \geq 1$ for every $n > n_0$, that is, $f(n) = 0$ has only finitely many solutions $n \in \mathbb{N}$. To obtain an effective version of this result with an explicit upper bound on n_0 , that is, on the sizes of solutions, is a famous open problem, mentioned for example in T. Tao [141, Chapter 3.9] or in B. Poonen [119]. Before we turn to the proof of Theorem 2.3 we state a corollary of Proposition 2.8. Recall that a sequence $f : \mathbb{N} \rightarrow \mathbb{Z}$ is a *quasi-polynomial* if for some $m \in \mathbb{N}$ polynomials $q_1, \dots, q_m \in \mathbb{Q}[x]$ we have $f_{j,m}(n) = q_j(n)$ for every $j \in [m]$ and $n \in \mathbb{N}$.

Corollary 2.9. *If a LRS $f : \mathbb{N} \rightarrow \mathbb{Z}$ has subexponential growth,*

$$\limsup_{n \rightarrow \infty} |f(n)|^{1/n} \leq 1,$$

then $f(n)$ is a quasi-polynomial.

We remark that one can prove Proposition 2.8 and Corollary 2.9 in a conceptually simpler (but probably not much shorter) way without Kronecker's theorem, using incommensurability of the frequencies of the roots of non-degenerate power sums.

Proof of Theorem 2.3. Let $k \in \mathbb{N}_0$ and $2k$ integers $a_0, \dots, a_{k-1}, f(1), \dots, f(k)$, $a_0 \neq 0$, be given. We describe a PIO algorithm for the LRS $f : \mathbb{N} \rightarrow \mathbb{Z}$ defined by

$$f(n+k) = \sum_{i=0}^{k-1} a_i f(n+i).$$

We take the recurrence polynomial $p(x) = x^k - a_{k-1}x^{k-1} - \dots - a_0$ of f , decompose the generating function

$$\sum_{n \geq 0} f(n)x^n = \frac{r(x)}{q(x)} \in \mathbb{Q}(x), \quad q(x) = x^k p(1/x) \quad \text{and} \quad \deg r(x) < k,$$

into partial fractions and as in the proof of the first implication in part 4 of Proposition 2.4 determine from them the power sum $s(x)$ representing $f(n)$. From $s(x)$ we determine the number m and set $J \subset [m]$ as defined in Proposition 2.8. For each $j \in J$ we find the polynomial $q_j \in \mathbb{Q}[x]$ such that $\deg q_j < k$ and $q_j(n) = f_{j,m}(n)$ for $n = 1, 2, \dots, k$. This precomputation can be done algorithmically. Now for an input $n \in \mathbb{N}$ we compute the residue $j \in [m]$ of n modulo m . If $j \in J$, we output $f(n) = q_j((n + m - j)/m)$. If $j \notin J$, we compute $f(n)$ by the defining recurrence.

Correctness of the algorithm follows from Proposition 2.8. We bound its time complexity in terms of $m(n)$. The precomputation takes $O(1)$ steps and determining j takes $\text{poly}(\log n)$ steps. If $j \in J$, computing $q_j((n + m - j)/m) = f(n)$ takes $\text{poly}(\log n)$ steps because we do $O(1)$ arithmetic operations with $O(\log(1+n))$ digit numbers. If $j \notin J$, computing $f(n)$ by the defining recurrence takes $\text{poly}(n)$ steps because $f(n)$ is an $O(n)$ digit number for every $n \in \mathbb{N}$. As for $m(n)$, if $j \in J$ then $f(n)$ is an $O(\log(1+n))$ digit number ($f_{j,m}(n)$ grows only polynomially) and $m(n) = \Theta(\log(1+n))$. If $j \notin J$ then $f(n)$ is an $\Omega(n)$ digit number (by case 2 of Proposition 2.8) and $m(n) = \Theta(n)$. No matter if $j \in J$ or not, for every $n \in \mathbb{N}$ the algorithm does $\text{poly}(m(n))$ steps and is a PIO algorithm. \square

Since the constant in the $\Omega(n)$ lower bound at the end of the proof is non-effective, the complexity bound $\text{poly}(m(n)) = O(m(n)^d)$ involves a non-effective constant as well.

Before we turn to holonomic sequences, we discuss the effect of domain extension for recurrence coefficients of a LRS. In the definition we required them to lie in \mathbb{Z} . Could one get more integer-valued sequences if the coefficients lie in a larger domain than \mathbb{Z} ? The answer is no. We already proved in the proof of the second implication in part 4 of Proposition 2.4 that if $K \subset L$ is an extension of fields and $f : \mathbb{N} \rightarrow K$ is a LRS in L then f is in fact a LRS in K . This folklore result on linear recurrence sequences is mentioned for example in M. Stoll [139, Lemma 3.1]. Recurrence coefficients outside \mathbb{Q} thus give nothing new. One can also prove that if $f : \mathbb{N} \rightarrow \mathbb{Z}$ is a LRS in \mathbb{Q} (recurrence coefficients lie in \mathbb{Q}), then f is in fact a LRS (another recurrence exists with coefficients in \mathbb{Z}). See R. P. Stanley [138, Problem 4.1 (a)] for the proof by generating functions and references for this result, known as the Fatou lemma.

One could also try to extend Theorem 2.3 to linear recurrence sequences in \mathbb{Q} . For this one extends the codomain of counting functions from \mathbb{Z} to \mathbb{Q} and in the definition of $m(n)$ (Definition 1.1) replaces $|f(n)|$ with $\max(|a|, |b|)$ where $f(n) = \frac{a}{b} \in \mathbb{Q}$, $\gcd(a, b) = 1$. We hope to return to this question later.

We conclude the section with some results and problems on computing terms in *holonomic sequences*. These generalize LRS and are also quite common in enumerative combinatorics and number theory. For simplicity we restrict to integer-valued sequences. A sequence $f : \mathbb{N} \rightarrow \mathbb{Z}$ is *holonomic* (synonymous terms in use are *P-recursive* and *polynomially recursive*) if for some k rational

functions $a_0, \dots, a_{k-1} \in \mathbb{Z}(x)$, $k \in \mathbb{N}_0$ and $a_0(x) \neq 0$, we have

$$f(n+k) = a_{k-1}(n)f(n+k-1) + a_{k-2}(n)f(n+k-2) + \dots + a_0(n)f(n)$$

for every $n > n_0$. Now the recurrence cannot hold in general from the beginning because of possible zeros of the denominators in the $a_i(x)$. Examples of such sequences are $f(n) = n!$ or the Catalan numbers $f(n) = c_n$ (see the ‘‘advanced’’ recurrence for c_n). Unfortunately, holonomic sequences lack some analog of the power sum representation; for a form of the matrix exponential representation (used in the matrix formula for the Fibonacci numbers) see Ch. Reutenauer [124]. We propose the following problem.

Problem 2.10. *Is it true that every holonomic sequence $f : \mathbb{N} \rightarrow \mathbb{Z}$ is a PIO function?*

A. Bostan, P. Gaudry and E. Schost [26] give an algorithm computing the n -th term of a holonomic sequence in $O(n^{1/2} \log^d(1+n))$ arithmetic operations.

Since Example 2 and Proposition 2.2 deal with an effective computation of the function $n \mapsto c_n \bmod 2$, we mention a problem and some results on effective computation of modular reductions of holonomic sequences. For a sequence $f : \mathbb{N} \rightarrow \mathbb{Z}$ and $m \in \mathbb{N}$, the *modular reduction* $n \mapsto f(n) \bmod m$ has values in the fixed set of residues $[m]$, and so a PIO formula for it means a computation in $\text{poly}(\log n)$ steps. Trivially, modular reduction of every LRS is eventually periodic and has therefore a PIO formula. As we saw in the proof of Proposition 2.2, c_n modulo 2 is not eventually periodic. We propose the following problem.

Problem 2.11. *Is it true that for every $m \in \mathbb{N}$ and every holonomic sequence $f : \mathbb{N} \rightarrow \mathbb{Z}$ its modular reduction $n \mapsto (f(n) \bmod m) \in [m]$ is a PIO function, that is, can be computed in $O(\log^d(1+n))$ steps?*

The answer is affirmative for *algebraic* f , that is, if the generating series $f(x) = \sum_{n \geq 1} f(n)x^n$ satisfies a polynomial equation, $P(x, f(x)) = 0$ for a nonzero polynomial $P \in \mathbb{Z}[x, y]$ (it is not hard to show that every algebraic sequence is holonomic). This applies to the Catalan numbers as $c(x) = \sum_{n \geq 1} c_n x^n$ satisfies $c(x)^2 - c(x) + x = 0$. See A. Bostan, X. Caruso, G. Christol and P. Dumas [24] for fast algorithm computing modular reduction of algebraic f . In fact, the answer is affirmative for an even larger subclass of holonomic sequences, namely for *rational diagonals*. These are sequences $f : \mathbb{N} \rightarrow \mathbb{Z}$ representable for $n \in \mathbb{N}$ as

$$f(n) = a_{n,n,\dots,n} \quad \text{where} \quad \sum_{n_1, \dots, n_k \geq 1} a_{n_1, \dots, n_k} x_1^{n_1} \dots x_k^{n_k} = \frac{P(x_1, \dots, x_k)}{Q(x_1, \dots, x_k)}$$

for some polynomials $P, Q \in \mathbb{Z}[x_1, \dots, x_k]$, $Q \neq 0$ (one can show that every algebraic sequence is a rational diagonal, and that every rational diagonal is holonomic). See [24], A. Bostan, G. Christol and P. Dumas [25] and E. Rowland

and R. Yassawi [130] (and some of the references therein) for more information on these two results. At the conclusion of [129] E. Rowland mentions that a conjecture of G. Christol [38] implies affirmative answer to Problem 2.11 for any at most exponentially growing holonomic sequence. See C. Krattenthaler and T. W. Müller [96] (and other works of the authors cited therein) for another approach to computation of modular reductions of algebraic sequences.

3 Integer partitions

Let us discuss PIO formulas for enumerative problems related to and motivated by the initial Examples 4 and 5. A *partition* λ of a number $n \in \mathbb{N}_0$ is a multiset of natural numbers summing to n . We write partitions in two formats,

$$\lambda = 1^{m_1} 2^{m_2} \dots n^{m_n}, m_i \in \mathbb{N}_0, \text{ and } \lambda = (\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_k), \lambda_i \in \mathbb{N}, k \in \mathbb{N}_0.$$

So $|\lambda| := n = \sum_i m_i i = \sum_i \lambda_i$. The numbers $1, 2, \dots, n$ and $\lambda_1, \lambda_2, \dots, \lambda_k$ are the *parts* of λ and the m_i are their *multiplicities*. We denote the number of parts in λ by $\|\lambda\|$, so $\|\lambda\| = m_1 + m_2 + \dots + m_n = k$. The set of all partitions of n is $P(n)$, their number is $p(n) = |P(n)|$, and $P := \bigcup_{n \geq 0} P(n)$. For the empty partition $\emptyset = ()$ we have $|\emptyset| = \|\emptyset\| = 0$, $P(0) = \{()\}$ and $p(0) = 1$. Similar quantities are $Q(n)$, $q(n)$, and Q , defined for partitions with distinct parts (all $m_i \leq 1$, i.e. $\lambda_i > \lambda_{i+1}$). $P_k(n)$ are the partitions of n with k parts, and similarly $Q_k(n)$ are those with k distinct parts. The sequence $p : \mathbb{N} \rightarrow \mathbb{N}$ is [154, A000041] and begins

$$(p(n))_{n \geq 1} = (1, 2, 3, 5, 7, 11, 15, 22, 30, 42, 56, 77, 101, 135, 176, \dots);$$

$(q(n))_{n \geq 1} = (1, 1, 2, 2, 3, 4, 5, \dots)$ is [154, A000009]. We give two proofs for the well known fact that $p(n)$ can be efficiently computed and is a PIO function in our parlance, and so is $q(n)$. What would be a non-efficient computation? For example, the ‘‘cave man formula’’ in [151]: $p(n) = \sum_{\lambda \in P(n)} 1$. By the multiplicity format, $q(n) \leq p(n) \leq (n+1)^n$. To get a lower bound, for a given $n \in \mathbb{N}$, $n \geq 4$, consider the maximum $m \in \mathbb{N}$ with $1+2+\dots+m = \binom{m+1}{2} \leq \frac{n}{2} - 1$. Then $m = \Theta(n^{1/2})$ and

$$n = \sum_{i \in X} i + \left(n - \sum_{i \in X} i \right), X \subset [m],$$

are 2^m different partitions in $Q(n)$. So $p(n) \geq q(n) \geq 2^m \gg \exp(\Omega(n^{1/2}))$ for $n \in \mathbb{N}$. Thus for the *partition function* $p(n)$ we have $n^{1/2} \ll m(n) \ll n^2$ and need to compute $p(n)$ in $\text{poly}(n)$ steps. Asymptotically,

$$p(n) \sim \frac{\exp(\pi\sqrt{2n/3})}{4 \cdot 3^{1/2} \cdot n} \text{ and } q(n) \sim \frac{\exp(\pi\sqrt{n/3})}{4 \cdot 3^{1/4} \cdot n^{3/4}} \text{ as } n \rightarrow \infty$$

(G. H. Hardy and S. Ramanujan [67], G. Meinardus [102], G. E. Andrews [6], V. Kotěšovec [94]). Thus, more precisely, $p(n)$ and $q(n)$ have $\Theta(n^{1/2})$ digits

and $m(n) = \Theta(n^{1/2})$. For the sake of brevity we treat the PIO algorithms and their complexity in Propositions 3.1 and 3.2 below more schematically and do not discuss their implementation by multitape Turing machines as we did in Proposition 2.1. These omitted details could be easily filled in, and it is easy to see that the deduced polynomiality of algorithms holds true.

Proposition 3.1. *For $k, n \in \mathbb{N}$ with $1 \leq k \leq n$ let $p_k(n) = |P_k(n)|$ be the number of partitions of n with k parts, and let $p_k(n) = 0$ and $P_k(n) = \emptyset$ if $k > n$. Then for every $n \geq 1$ we have $p_n(n) = p_1(n) = 1$, and for every $n \geq 2$ and every k with $1 < k < n$ we have*

$$p_k(n) = p_k(n - k) + p_{k-1}(n - 1).$$

Consequently, $p(n) = p_1(n) + p_2(n) + \dots + p_n(n)$ is a PIO function.

Proof. The values $p_n(n) = p_1(n) = 1$ are trivial. The displayed recurrence mirrors the set partition $P_k(n) = A \cup B$ where A are the partitions of n with all k parts at least 2 and B are the remaining partitions with at least one part 1. Decreasing each part in every $\lambda \in A$ by 1 gives the bijection $A \rightarrow P_k(n - k)$ and removing one part 1 from every $\lambda \in B$ gives the bijection $B \rightarrow P_{k-1}(n - 1)$, whence the recurrence.

For given input $n \in \mathbb{N}$ we use the recurrence and the initial and border values and generate the array of $O(n^2)$ numbers $(p_k(m) \mid 1 \leq k \leq m \leq n)$ in $O(n^2)$ additions. Another $n - 1$ additions produce $p(n)$. Every number involved in the computation has $O(n^2)$ digits (in fact, $O(n^{1/2})$ digits), and therefore the algorithm makes $O(n^4)$ steps (in fact, $O(n^{5/2})$ steps), which is $O(m(n)^8)$ steps (in fact, $O(m(n)^5)$ steps) as $m(n) \gg n^{1/2}$. Therefore the stated recurrence schema is a PIO formula for $p(n)$. \square

The interested reader will find in M. Bodirsky, C. Gröpl and M. Kang [17] a recurrence schema, in its principle similar to the previous one but much more involved in details, that computes in polynomial time the number of labeled planar graphs on the vertex set $[n]$; computation of this number for $n = 50$ in one hour is reported.

The second proof shows that L. Euler's generating function formula

$$\sum_{n \geq 0} p(n)q^n = \prod_{k \geq 1} \frac{1}{1 - q^k} = \prod_{k \geq 1} (1 + q^k + q^{2k} + \dots)$$

also gives a PIO formula for $p(n)$. Let $[x^n]a(x) := a_n$ if $a(x) = \sum_{n \geq 0} a_n x^n$.

Proposition 3.2. *Let $m, n \in \mathbb{N}$. The product ab of two polynomials $a, b \in \mathbb{Z}[x]$ such that $\deg a, \deg b \leq n$ and each coefficient in them has at most m digits can be computed in the obvious way in $O(m^2 n^3)$ steps. Thus*

$$p(n) = [q^n] \prod_{k=1}^n (1 + q^k + q^{2k} + \dots + q^{\lfloor n/k \rfloor k})$$

is a PIO function.

Proof. Each of the $1 + \deg a + \deg b = O(n)$ sums $[x^k]ab = \sum_{i+j=k} [x^i]a \cdot [x^j]b$, $k \leq \deg a + \deg b$, has $O(n)$ summands, multiplication in each summand takes $O(m^2)$ steps, and each addition costs $O(m+n)$ steps (we add two numbers with $\ll m + \log(1+n) \ll m+n$ digits). The list of coefficients of ab is thus computed in

$$\ll n(m^2n + (m+n)n) = O(m^2n^3)$$

steps.

The value $p(n)$ is a coefficient in the product of n polynomials with degrees at most n and coefficients 0 and 1. We apply the lemma about the product of two polynomials $n-1$ times and each time we multiply two polynomials with degrees at most n^2 and with coefficients of size $\leq p(n^2)$ that have $\ll n^4$ digits. Thus we compute the product of the n polynomials in $O(n(n^4)^2(n^2)^3) = O(n^{15}) = O(m(n)^{30})$ steps (recall that $m(n) \gg n^{1/2}$) and see that $p(n)$ is a PIO function. \square

Often less elementary recurrences are invoked to efficiently compute $p(n)$:

$$p(n) = \sum_{i \geq 1} (-1)^{i+1} (p(n-a_i) + p(n-b_i)) \quad \text{or} \quad p(n) = \frac{1}{n} \sum_{i=1}^n \sigma(i) p(n-i)$$

where $n = 1, 2, \dots$, $a_i = \frac{i(3i-1)}{2}$ and $b_i = \frac{i(3i+1)}{2}$ are so called (*generalized*) *pentagonal numbers*, $p(0) = 1$, $p(n) := 0$ for $n < 0$, and $\sigma(n) := \sum_{d|n} d$ is the sum of divisors function (G. E. Andrews [6]). For more recurrences for $p(n)$ see Y. Choliy, L. W. Kolitsch and A. V. Sills [37]. The pentagonal recurrence yields an algorithm computing the list of values $(p(m) \mid 1 \leq m \leq n)$ in $O(n^2)$ steps (D. E. Knuth [92, Chapter 7.2.1.4, exercise 20]) while the algorithm of Proposition 3.1 makes $O(n^{5/2})$ steps. N. Calkin, J. Davis, K. James, E. Perez and C. Swannack [32, Corollary 3.1] give an algorithm producing this list in $O(n^{3/2} \log^2 n)$ steps, which is close to optimum complexity (because it takes $\Omega(n^{3/2})$ steps just to print it).

The exponent 30 in the proof of Proposition 3.2 is hilarious but the reader understands that we do not optimize bounds and instead focus on simplicity of arguments. We can decrease it by computing the product $a(x)b(x)$ more quickly but in a less elementary way in $O((mn)^{1+o(1)})$ steps by [61, Chapter 8.4]. A clever implementation of the Hardy–Ramanujan–Rademacher analytic formula for $p(n)$ ([6]) by F. Johansson [80, Theorem 5] computes $p(n)$ in $O(n^{1/2} \log^{4+o(1)} n) = O(m(n)^{1+o(1)})$ steps, again in close to optimum complexity. F. Johansson reports computing $p(10^6)$ by his algorithm in milliseconds and $p(10^{19})$ in less than 100 hours; see [81] for his computation of $p(10^{20})$. Proposition 3.2 represents $p(n)$ as a coefficient in a polynomial from $\mathbb{Z}[x]$. J. H. Bruiner and K. Ono [30] recently found another (more complicated) representation in this spirit, which has $(1-24n)p(n)$ as the next to leading coefficient in a monic polynomial from $\mathbb{Q}[x]$, see also J. H. Bruiner, K. Ono and A. V. Sutherland [31]. Computation-wise for $p(n)$ it lags far behind the H–R–R formula ([31]).

There is an extensive literature on modular properties of $p(n)$ (for both meanings of “modular”), see K. Ono [110, 111] and the references therein. N. Calkin et al. [32, Theorem 3.1] can generate the list $(p(k) \bmod m \mid 1 \leq k \leq n)$ for special prime moduli m depending on n in $O(n^{1+o(1)})$ steps. But unlike for the Catalan numbers and algebraic sequences, so far we do not know an efficient way to determine the parity of individual numbers $p(n)$.

Problem 3.3. *Is the parity of $p(n)$, the function $n \mapsto (p(n) \bmod 2) \in [2]$, a PIO function? That is, can one compute it in $O(\log^d(1+n))$ steps (bit operations)?*

By [81], “With current technology, the most efficient way to determine $p(n)$ modulo a small integer is to compute the full value $p(n)$ and then reduce it.” (cf. the discussion of Example 2). Cannot we do better? The parity of $p(n)$ was investigated by T.R. Parkin and D. Shanks [116] already in 1967. At the end of their article they ask if it can be computed in $O(n)$ steps.

We generalize Proposition 3.1 by replacing 1 in $p(n) = \sum_{\lambda \in P(n)} 1$ with a positive PIO function of the number of parts. This includes Example 4.

Proposition 3.4. *If $g : \mathbb{N} \rightarrow \mathbb{N}$ is a PIO function then $f : \mathbb{N} \rightarrow \mathbb{N}$,*

$$f(n) = \sum_{\lambda \in P(n)} g(\|\lambda\|),$$

is a PIO function too. Construction of the PIO algorithm for f from that for g is described in the proof.

Proof. We set $G(n) = \max(g(1), g(2), \dots, g(n))$. Clearly,

$$f(n) = \sum_{k=1}^n g(k)p_k(n),$$

and so

$$f(n) \geq p(n) + G(n) - 1.$$

Thus the combined input and output complexity of $f(n)$ satisfies

$$m(n) = \log(1+n) + \log(2+f(n)) \gg \log(p(n) + G(n)) \gg n^{1/2} + \log(2+G(n)).$$

By the assumption on g we compute the list $(g(k) \mid 1 \leq k \leq n)$ in

$$\ll n(\log(1+n) + \log(2+G(n)))^d$$

steps, for a fixed $d \in \mathbb{N}$. By Proposition 3.1 we compute the list $(p_k(n) \mid 1 \leq k \leq n)$ in $O(n^{5/2})$ steps. The product $g(k)p_k(n)$ is computed by elementary school multiplication in

$$\ll \log^2 p(n) + \log^2(2+G(n)) \ll (n^{1/2} + \log(2+G(n)))^2$$

steps, and this also bounds the cost of each addition in the sum. The displayed sum therefore computes $f(n)$ in

$$\begin{aligned} &\ll n(\log(1+n) + \log(2+G(n)))^d + n^{5/2} + n(n^{1/2} + \log(2+G(n)))^2 \\ &\ll (n^{1/2} + \log(2+G(n)))^{d+4} \ll m(n)^{d+4} \end{aligned}$$

steps. □

Functions covered by the proposition include $f(n) = p(n)$ for $g(n) = 1$ and the contrived counting function $f(n)$ of Example 4. For $g(n) = n$ we get the total number of parts in all partitions of n , so

$$f(n) = \sum_{\lambda \in P(n)} \|\lambda\| = \sum_{i=1}^n \tau(i)p(n-i)$$

is a PIO function. Here $\tau(i)$ denotes the number of divisors of i . One can deduce the last sum (which itself is a PIO formula for $f(n)$, given one for $p(n)$, no matter that we cannot compute effectively $i \mapsto \tau(i)$) by differentiating the generating function $\sum_{\lambda \in P} y^{|\lambda|} x^{|\lambda|} = \frac{1}{(1-yx)(1-yx^2)\dots}$ by y and then setting $y = 1$.

For partitions with distinct parts we have similar results.

Proposition 3.5. *If $g : \mathbb{N} \rightarrow \mathbb{N}$ is a PIO function then $f : \mathbb{N} \rightarrow \mathbb{N}$,*

$$f(n) = \sum_{\lambda \in Q(n)} g(\|\lambda\|),$$

is a PIO function too. Construction of the PIO algorithm for f from that for g is described in the proof.

Proof. Now $f(n) = \sum_{k=1}^n g(k)q_k(n)$ where $q_k(n) = |Q_k(n)|$ is the number of partitions of n with k distinct parts. For $q_k(n)$ we have the recurrence schema $q_1(n) = q_n(n) = 1$ for every $n \geq 1$, $q_k(n) = 0$ for $k > n$, and

$$q_k(n) = q_k(n-k) + q_{k-1}(n-k)$$

for $n \geq 2$ and $1 < k < n$. Compared to Proposition 3.1, this differs in the last summand: now the set of $\lambda \in Q_k(n)$ with one part 1 bijectively corresponds to $Q_{k-1}(n-k)$, delete the 1 and decrease each of the remaining $k-1$ parts by 1. We only replace $p(n)$ with $q(n)$ and $p_k(n)$ with $q_k(n)$ and argue as in the proof of Proposition 3.4. □

Now for $g(n) = n$ we get that the total number of parts in all $\lambda \in Q(n)$,

$$f(n) = \sum_{\lambda \in Q(n)} \|\lambda\| = \sum_{i=1}^n \tau^\pm(i)q(n-i),$$

is a PIO function. Here $\tau^\pm(i) := \sum_{d|i} (-1)^{d+1}$ is the surplus of the odd divisors of i over the even ones. The last sum (again by itself a PIO formula for

$f(n)$, given one for $q(n)$) follows by the same manipulation with the generating function $\sum_{\lambda \in Q} y^{|\lambda|} x^{|\lambda|} = (1 + yx)(1 + yx^2) \dots$ as before. We remark that

$$\begin{aligned} (\sum_{\lambda \in P(n)} \|\lambda\|)_{n \geq 1} &= (1, 3, 6, 12, 20, 35, 54, 86, 128, 192, \dots) \text{ and} \\ (\sum_{\lambda \in Q(n)} \|\lambda\|)_{n \geq 1} &= (1, 3, 3, 5, 8, 10, 13, 18, 25, 30, \dots) \end{aligned}$$

are respective sequences [154, A006128] and [154, A015723]. The first one was investigated by “Miss S. M. Luthra, University of Delhi” [99] (see p. 485 for the formula with $\tau(n)$), and the second by A. Knopfmacher and N. Robbins [89] (they deduce the formula with $\tau^\pm(n)$).

Recall that a *composition* c of $n \in \mathbb{N}$ is an “ordered partition” of n , that is, a tuple $c = (c_1, c_2, \dots, c_k) \in \mathbb{N}^k$ with $c_1 + c_2 + \dots + c_k = n$. It is well known and easy to show that there are 2^{n-1} compositions of n . What is the number $f_{cdp}(n)$ of compositions of n with distinct parts?

Corollary 3.6. *The number $f_{cdp}(n)$ of compositions of n with no part repeated is a PIO function. The PIO algorithm for f_{cdp} is described in the proof.*

Proof. The mapping $(c_1, c_2, \dots, c_k) \mapsto (c_{i_1} > c_{i_2} > \dots > c_{i_k})$ sending a composition of n with distinct parts to its decreasing reordering is a $k!$ -to-1 mapping from the set of compositions of n with k distinct parts onto $Q_k(n)$. Thus $f_{cdp}(n) = \sum_{k=1}^n k! q_k(n)$ and the result is an instance of Proposition 3.5 for $g(n) = n!$ (clearly, $n \mapsto n!$ is a PIO function, also see P. B. Borwein [23]). \square

The sequence $(f_{cdp}(n))_{n \geq 1} = (1, 1, 3, 3, 5, 11, 13, 19, 27, \dots)$ is [154, A032020]. B. Richmond and A. Knopfmacher [125] note that

$$f_{cdp}(n) = \exp((1 + o(1))(2n)^{1/2} \log n)$$

and obtain a more precise asymptotics. See the book of S. Heubach and T. Mansour [72] for many more enumeration problems for compositions and words, especially with forbidden patterns.

We pose the following problem.

Problem 3.7. *Give general sufficient conditions on functions $g : \mathbb{N} \rightarrow \mathbb{Z}$ ensuring that*

$$f(n) = \sum_{\lambda \in P(n)} g(\|\lambda\|) \text{ and } f(n) = \sum_{\lambda \in Q(n)} g(\|\lambda\|)$$

are PIO functions.

Propositions 3.4 and 3.5 say that it suffices when g is a positive PIO function, but it would be more interesting to have general sufficient conditions allowing negative values of g . Corollary 3.19 and Proposition 3.26 are motivated by this problem too.

We generalize Proposition 3.2. Many enumerative problems on partition, but of course not all, fit in the general schema of counting partitions with prescribed

parts and multiplicities: for every triple $(n, i, j) \in \mathbb{N}^2 \times \mathbb{N}_0$ we say if i^j , part i with multiplicity j , may or may not appear in the counted partitions of n . If $m(n) \gg n^c$ with $c > 0$ for the counting problem, the simple algorithm of Proposition 3.2 gives a PIO formula. We spell it out explicitly.

Proposition 3.8. *Suppose that $X \subset \mathbb{N}$ is a set, $g(n, i, j) \in \{0, 1\}$ is a function defined for $n, i \in \mathbb{N}$ and $j \in \mathbb{N}_0$ with $i, j \leq n$ and computable in $\text{poly}(n)$ steps, and that the function $f : \mathbb{N} \rightarrow \mathbb{N}_0$, defined by*

$$\begin{aligned} f(n) &= |\{\lambda = 1^{j_1} 2^{j_2} \dots n^{j_n} \in P(n) \mid g(n, i, j_i) = 1 \text{ for every } i \in [n]\}| \\ &= [q^n] \prod_{i=1}^n \sum_{j=0}^n g(n, i, j) q^{ij}, \end{aligned}$$

grows for $n \in X$ as $f(n) \gg \exp(n^c)$ with a constant $c > 0$. Then the restriction $f : X \rightarrow \mathbb{N}_0$ is a PIO function. The proof shows that the algorithm for g constructively gives the PIO algorithm for f , provided that the function f gets as inputs only elements of X .

Proof. In the formula for $f(n)$ we have a product of n polynomials with degrees at most n^2 each and with the coefficients 0 and 1 that can be computed in $\text{poly}(n)$ steps. We argue as in the proof of Proposition 3.2 and deduce that $f(n)$ can be computed in $\text{poly}(n)$ steps, which means $\text{poly}(m(n))$ steps for $n \in X$ by the assumption on growth of f . \square

This applies to partitions of n into distinct parts, odd parts, squares, etc. but first we illustrate the proposition with two problems where the condition defining counted partitions $\lambda \in P(n)$ depends on n —then our reflex to write a formula for $\sum_{n \geq 0} f(n)q^n$, often an infinite product of simple factors, fails us as it cannot be done. They are the functions $f_m, f_p : \mathbb{N} \rightarrow \mathbb{N}$ where $f_m(n)$ (resp. $f_p(n)$) counts partitions of n such that every nonzero multiplicity (resp. every part with nonzero multiplicity) divides n . Then

$$\begin{aligned} (f_m(n))_{n \geq 1} &= (1, 2, 3, 5, 4, 10, 6, 17, 14, 26, 13, 66, 19, 63, 60, \dots) \text{ and} \\ (f_p(n))_{n \geq 1} &= (1, 2, 2, 4, 2, 8, 2, 10, 5, 11, 2, 45, 2, 14, 14, \dots) \end{aligned}$$

are respective sequences [154, A100932] and [154, A018818]. By Proposition 3.8, applied with $X = \mathbb{N}$ and $g(n, i, j) = 1$ if j divides n and $g(n, i, j) = 0$ else, the function $f_m(n)$ is a PIO function (with PIO algorithm given in Proposition 3.8) because $g(n, i, j)$ is computable even in $\text{poly}(\log n)$ steps and $f_m(n) \geq q(n) \gg \exp(\Omega(n^{1/2}))$ (by the above lower bound on $q(n)$). The second function $f_p(n)$ was investigated by D. Bowman, P. Erdős and A. Odlyzko [27] who proved that

$$(1 + o(1)) \left(\frac{\tau(n)}{2} - 1 \right) \log n < \log f_p(n) < (1 + o(1)) \frac{\tau(n)}{2} \log n,$$

in fact with stronger bounds in place of the $o(1)$ terms. A PIO formula for $f_p(n)$ is apparently not known (in contrast to $f_m(n)$, the growth condition is

not satisfied and small values occur). After M. Agrawal, N. Kayal and N. Saxena [2] we however know a PIO function $f : \mathbb{N} \rightarrow \{0, 2\}$ with the property that $f(n) = 2 \iff f_p(n) = 2$ for every $n \in \mathbb{N}$ —we can efficiently compute $f_p(n)$ for infinitely many n , namely the prime numbers.

We turn to the more standard situation when $g(n, i, j)$ does not depend on n . Then we easily obtain Corollary 3.10 below. For its proof we need the next lemma which is also used in the proof of Corollary 3.13.

Lemma 3.9. *Let $a_1, a_2, \dots, a_k \in \mathbb{N}$ be distinct numbers such that if $d \in \mathbb{N}$ divides each a_i then $d = 1$.*

1. *There is an n_0 , specified in the proof, such that for every $n \in \mathbb{N}$ with $n > n_0$ the equation*

$$n = a_1x_1 + \dots + a_kx_k$$

has a solution $x_1, \dots, x_k \in \mathbb{N}_0$.

2. *The same holds even if the k numbers x_i are required be distinct.*

Proof. 1. The ideal $\langle a_1, \dots, a_k \rangle$ in the ring \mathbb{Z} shows that $1 = a_1b_1 + \dots + a_kb_k$ for some $b_i \in \mathbb{Z}$. It is not hard to see that one may take all b_i with $|b_i| \leq A^{k-1}$ if $|a_i| \leq A$ for every i . We set $c = a_1 \max_i |b_i|$ and $d = \sum_i a_i c$. It follows that $n_0 = d - 1$ works because any $n \geq d$ is expressed by the nonnegative solution $x_1 = c + l + jb_1$, $x_i = c + jb_i$ if $i > 1$, for appropriate $l \in \mathbb{N}_0$ and $j \in \{0, 1, \dots, a_1 - 1\}$.

2. Now we set $c = 2a_1 \max_i |b_i|$ and $d = \sum_i a_i c(k + 1 - i)$. Then $n_0 = d - 1$ works, because any $n \geq d$ is expressed by the nonnegative solution with distinct coordinates $x_1 = ck + l + jb_1$, $x_i = c(k + 1 - i) + jb_i$ if $i > 1$, again for appropriate $l \in \mathbb{N}_0$ and $j \in \{0, 1, \dots, a_1 - 1\}$. \square

The first part of the lemma is well known and is the simplest version of the Frobenius problem, see the book [4] of J. L. R. Alfonsín for more information.

We look at *restricted partitions* with parts in a prescribed set $A \subset \mathbb{N}$; let $P_A(n) \subset P(n)$ be their set and $p_A(n) := |P_A(n)|$. We show that we can count them efficiently if the elements of A can be efficiently recognized and A is not too sparse. To be precise, in general we probably only “can” count them efficiently because the proof relies on quantities $d \in \mathbb{N}$ and $B \subset \mathbb{N}$ that in general appear not to be computable.

Corollary 3.10. *Suppose that the function $g = g(n) : \mathbb{N} \rightarrow \mathbb{N}$ increases, is computable in $\text{poly}(n)$ steps and grows only polynomially, $g(n) < (1 + n)^d$ for every $n \in \mathbb{N}$ and a constant $d \in \mathbb{N}$, and define $f(n) = p_A(n)$ for $A = \{g(1), g(2), \dots\}$,*

$$\sum_{n \geq 0} f(n)q^n = \prod_{i \geq 1} \left(1 + \sum_{j=1}^{\infty} q^{g(i)j} \right) = \prod_{i \geq 1} \frac{1}{1 - q^{g(i)}}.$$

Then $f(n)$ is a PIO function. We show how to compute the PIO algorithm for $f(n)$ from the algorithm for $g(n)$ if we are given the number $d = \gcd(A) = \gcd(g(1), g(2), \dots) \in \mathbb{N}$ and a finite set $B \subset A$ with $\gcd(B) = d$.

Proof. Let $g(n)$, A , d and B be as stated (it is easy to see from prime factorizations that such finite subset B exists) and let

$$n_0 = \max(\{0\} \cup \{n \in d\mathbb{N} \mid p_B(n) = 0\}) \in \mathbb{N}_0 .$$

Then $n_0 < \infty$ by part 1 of Lemma 3.9 applied to the numbers $\frac{1}{d}B$, and in fact we can compute n_0 from the given B . So $f(n) = 0$ if $n \notin d\mathbb{N}$ and for $n \in \mathbb{N}$ we can decide in $\text{poly}(\log n)$ steps if $n \in d\mathbb{N}$. Subsets $S \subset \{g(1), g(2), \dots, g(m)\} \setminus B$, where $m \in \mathbb{N}$ is maximum with

$$g(1) + g(2) + \dots + g(m) \leq n - n_0 - d ,$$

prove that for $n \in d\mathbb{N}$ with $n \geq n_0 + d$ one has $f(n) \gg \exp(\Omega(n^{1/(d+1)}))$ (for each S we complete the sum of its elements by an appropriate partition with parts in B to a partition of n).

We compute $f(n)$ effectively as follows. For the input $n \in \mathbb{N}$ we check in $\text{poly}(\log n)$ steps if $n \notin d\mathbb{N}$ and if $n \leq n_0$. In the former case we output $f(n) = 0$ and in the latter case we compute $f(n)$ by brute force. If neither case occurs, we have $n \in d\mathbb{N}$ and $n > n_0$ and compute $f(n)$ by Proposition 3.8, applied with $X = d\mathbb{N} \setminus [n_0]$ and $g(n, i, j)$ defined as $g(n, i, j) = 0$ if $j \geq 1$ and $i \notin A$, and $g(n, i, j) = 1$ else (it is clear that the assumptions are satisfied, we can check if $i \notin A$ in $\text{poly}(i)$ steps). It follows that this is a PIO algorithm for $f(n)$. Also, we have constructed it explicitly from the algorithm for $g(n)$ and the knowledge of d and B . \square

In general the quantities d and B probably are not computable from the algorithm for the function $g(n)$. Hence, probably, the PIO algorithm for $f(n)$ cannot be computed given only the algorithm for $g(n)$. For example, we may take $g(n) = n^2$ (so $d = 1$ and $B = \{1\}$) and compute the number $f_{sq}(n)$ of partitions of n into squares, $\sum_{n \geq 0} f_{sq}(n)q^n = \prod_{k \geq 1} (1 - q^{k^2})^{-1}$. By Corollary 3.10 we get a PIO function

$$(f_{sq}(n))_{n \geq 1} = (1, 1, 1, 2, 2, 2, 2, 3, 4, 4, 4, 5, 6, 6, 6, 8, \dots) ,$$

[154, A001156]. The function $f_{sq}(n)$ was investigated by J. Bohman, C.-E. Fröberg and H. Riesel [20]. As we showed in the proof, $f_{sq}(n) \gg \exp(\Omega(n^{1/3}))$. More generally, already in 1934 E.M. Wright found in [148] the asymptotics for the number $p_{S_k}(n)$ of partitions of n into k -th powers $S_k = \{n^k \mid n \in \mathbb{N}\}$ ($k \in \mathbb{N}$):

$$p_{S_k}(n) \sim \frac{\Delta}{(2\pi)^{(k+1)/2}} \cdot \frac{k^{1/2}}{(k+1)^{3/2}} \cdot n^{\frac{1}{k+1} - \frac{3}{2}} \cdot \exp(\Delta n^{1/(k+1)})$$

where

$$\Delta = (k + 1) \cdot ((1/k) \cdot \Gamma(1 + 1/k) \cdot \zeta(1 + 1/k))^{1-1/(k+1)} .$$

More recently the asymptotics for $p_{S_k}(n)$ with $k = 2$ (i.e., $f_{sq}(n)$) was treated by R. C. Vaughan [145], for general k by A. Gafni [60], and for $p_A(n)$ with A formed by values of an integral polynomial by A. Dunn and N. Robles [49].

Why not partition n into *distinct* squares, $f_{dsq}(n) := [q^n] \prod_{k \geq 1} (1 + q^{k^2})$? The initially somewhat dull sequence

$$(f_{dsq}(n))_{n \geq 1} = (1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, \dots) ,$$

[154, A033461], eventually takes off (the first n with $f(n) \geq 2$ is $n = 25$) and R. Sprague [137] proved in 1948 that $n = 128$ is the last number with $f_{dsq}(n) = 0$. See M. D. Hirschhorn [74] for “an almost complete proof” that

$$f_{dsq}(n) \sim c_2 n^{-5/6} \exp(c_1 n^{1/3})$$

where $c_1 = 3c_3^{2/3}$, $c_2 = c_3^{1/3}/\sqrt{6\pi}$, and $c_3 = \sqrt{\pi}(2 - \sqrt{2})\zeta(3/2)/8$. We have

Corollary 3.11. *The function $f_{dsq}(n) : \mathbb{N} \rightarrow \mathbb{N}_0$ counting partitions of n into distinct squares, which is the same as counting partitions of n such that each part $i \in \mathbb{N}$ has multiplicity either 0 or i , is a PIO function. The PIO algorithm is described in the proof.*

Proof. This follows from Proposition 3.8, applied with $X = \{129, 130, \dots\}$ and $g(n, i, j) = 1$ iff $(i = k^2 \& j = 1)$ or $j = 0$, if we show that $f_{dsq}(n) \gg \exp(n^c)$ on X for some $c > 0$.

To obtain such lower bound for $f_{dsq}(n)$ we begin with a lemma: for every $j \in [4]$ and $n \geq 4$ we have $|\{A \subset [n] \mid |A| \equiv j \pmod{4}\}| \gg 2^n$ because any $B \subset [n-3]$ can be enlarged by adding one of $n-2, n-1, n$ to have cardinality j modulo 4. Now for given $n \in \mathbb{N}$ with $n \equiv j \pmod{4}$, $j \in [4]$ and $n > 1000$, consider the maximum $m \in \mathbb{N}$ with $1^2 + 3^2 + 5^2 + \dots + (2m-1)^2 \leq n-4 \cdot 129$. Clearly, $m = \Theta(n^{1/3})$. For every subset $C \subset [m]$ with $|C| \equiv j \pmod{4}$ the sum $S_C = \sum_{i \in C} (2i-1)^2$ is also j modulo 4. By R. Sprague’s theorem mentioned above and our selection of m we may partition $\frac{n-S_C}{4} \in \mathbb{N}$ into distinct squares, say $\frac{n-S_C}{4} = x_1^2 + \dots + x_k^2$. But then $n = S_C + (2x_1)^2 + \dots + (2x_k)^2$ is a partition of n into distinct squares, and distinct subsets C yield distinct such partitions. Using the lemma we get that $f_{dsq}(n) \gg 2^m = \exp(\Omega(n^{1/3}))$ for $n \in X$. \square

As M. D. Hirschhorn [74] himself admits, also in [75], his proof of the asymptotics for $f_{dsq}(n)$ is not complete.

Problem 3.12. *Derive rigorously asymptotics for $f_{dsq}(n)$.*

Could not theta functions tell us something about $f_{sq}(n)$ or $f_{dsq}(n)$? It transpires that $f_{dsq}(n)$ (with other partition counting functions) comes up in quantum statistical physics. M. N. Tran, M. V. N. Murty and R. K. Bhaduri [142]

and, recently, M. V. N. Murthy, M. Brack, R. K. Bhaduri and J. Bartel [106] investigate the asymptotics of $f_{dsq}(n)$. The main term is derived, with some further terms in the asymptotic expansion, but it is not clear to us whether these arguments are rigorous, and so we leave Problem 3.12 as it is.

We partitioned n into squares with arbitrary multiplicities, why not partition n into arbitrary parts but with square multiplicities, $f_{sm}(n) := [q^n] \prod_{i \geq 1} (1 + \sum_{j \geq 1} q^{ij^2})$? In general setting this leads to the next counterpart to Corollary 3.10. Now we have no restriction on the growth of the set of allowed multiplicities, it may very well be finite.

Corollary 3.13. *Suppose that $1 \leq g_1 < g_2 < \dots$ is a finite or infinite nonempty increasing sequence of natural numbers such that $n \mapsto g_n$ is computable in $\text{poly}(n)$ steps and define $f(n) = f_A(n) \in \mathbb{N}_0$ to be the number of partitions $\lambda \in P(n)$ with all nonzero multiplicities in $A = \{g_1, g_2, \dots\}$,*

$$\sum_{n \geq 0} f(n)q^n = \prod_{i \geq 1} \left(1 + \sum_{j \geq 1} q^{ig_j} \right).$$

Then $f(n)$ is a PIO function. We show how to compute the PIO algorithm for $f(n)$ from the algorithm for g_n if we are given the number $d = \gcd(A) = \gcd(g(1), g(2), \dots) \in \mathbb{N}$ and a finite set $B \subset A$ with $\gcd(B) = d$.

Proof. Let g_n , A , d and B be as stated and let

$$n_0 = \max(\{0\} \cup \{n \in d\mathbb{N} \mid f_B(n) = 0\}) \in \mathbb{N}_0.$$

Then $n_0 < \infty$ by the second part of Lemma 3.9 and we can compute n_0 from B . So $f(n) = 0$ if $n \notin d\mathbb{N}$, and for any $n \in \mathbb{N}$ the membership of n in $d\mathbb{N}$ is decidable in $\text{poly}(\log n)$ steps. We obtain a lower bound for $f(n)$ with large $n \in d\mathbb{N}$ as in the proof of Corollary 3.11. Namely, we may assume that $\frac{g_1}{d}$ is odd (there is certainly a g_k with odd $\frac{g_k}{d}$) and take arbitrary $n \in d\mathbb{N}$ with $n > 2n_0 + 1$ and $\frac{n}{d} \equiv r$ modulo 2, $r \in [2]$. Let $m \in \mathbb{N}$ be maximum with $(1 + 3 + 5 + \dots + (2m - 1))g_1 \leq n - 2(n_0 + 1)$. Then $m = \Theta(n^{1/2})$ and there are $\gg 2^m$ subsets $C \subset [m]$ such that $|C| \frac{g_1}{d} \equiv r$ modulo 2. For each C we have $(n - g_1 \sum_{i \in C} (2i - 1))/2 \in d\mathbb{N} \setminus [n_0]$ and this number equals $\sum_{g \in D} i_g g$ for some $D \subset B$ and $|D|$ distinct numbers $i_g \in \mathbb{N}$. But then

$$n = \sum_{i \in C} (2i - 1)g_1 + \sum_{g \in D} (2i_g)g$$

is a partition of n into parts $\{2i - 1 \mid i \in C\}$, each with multiplicity g_1 , and parts $\{2i_g \mid g \in D\}$, with respective multiplicities $\{g \mid g \in D\}$. All multiplicities are in A and distinct subsets C give distinct such partitions. Thus for $n \in d\mathbb{N}$ with $n > n_0$ one has $f(n) \gg 2^m = \exp(\Omega(n^{1/2}))$.

We compute $f(n)$ effectively as follows. For the input $n \in \mathbb{N}$ we check in $\text{poly}(\log n)$ steps if $n \notin d\mathbb{N}$ and if $n \leq n_0$. In the former case we output $f(n) = 0$

and in the latter case we compute $f(n)$ by brute force. If neither case occurs, we have $n \in d\mathbb{N}$ and $n > n_0$ and compute $f(n)$ by Proposition 3.8, applied with $X = d\mathbb{N} \setminus [n_0]$ and $g(n, i, j)$ defined for $j \geq 1$ and $j \notin A$ as $g(n, i, j) = 0$ and $g(n, i, j) = 1$ else (clearly the assumption is satisfied, this $g(n, i, j)$ is computable in $\text{poly}(n)$ steps). We have for $f(n)$ a PIO algorithm which we have constructed explicitly from the algorithm for g_n and the knowledge of d and B . \square

Similarly to Corollary 3.10, the PIO algorithm for $f(n)$ probably cannot be computed given only the algorithm for g_n . The sequence

$$(f_{sm}(n))_{n \geq 1} = (1, 1, 2, 3, 3, 5, 6, 8, 12, 12, 17, 23, 27, 32, 41, 52, \dots)$$

corresponding to $g_n = n^2$ ($d = 1$ and $B = \{1\}$) was for a long time we were preparing this article absent in OEIS, but checking it once more in August 2018 we found out that S. Manyama had added it as [154, A300446] in May 2018, thank you.

An interesting counting problem for partitions outside the framework of Proposition 3.8 is the number $f_{dm}(n)$ of partitions of n into parts with distinct nonzero multiplicities, so

$$f_{dm}(n) = [q^n]^* \prod_{i=1}^n \left(1 + \sum_{j=1}^n q^{ij} \right)$$

where the “star extraction” of the coefficient means that in the product we only accept monomials $q^{i_1 j_1} q^{i_2 j_2} \dots q^{i_k j_k}$, $1 \leq i_1 < \dots < i_k \leq n$, with $|\{j_1, \dots, j_k\}| = k$. The sequence

$$(f_{dm}(n))_{n \geq 1} = (1, 2, 2, 4, 5, 7, 10, 13, 15, 21, 28, 31, 45, 55, 62, \dots)$$

is [154, A098859]. The problem to investigate $f_{dm}(n)$ was posed by H. S. Wilf [147] in 2010. To get a lower bound on $f_{dm}(n)$, for given $n \in \mathbb{N}$ consider the maximum $m \in \mathbb{N}$ such that $1m + 2(m-1) + \dots + m1 = (m+1) \sum_{i=1}^m i - \sum_{i=1}^m i^2 \leq n$. Then $m = \Theta(n^{1/3})$ and the subsets of $\{2^{m-1}, 3^{m-2}, \dots, m^1\}$, where the exponents are used in the multiplicities sense, show that $f_{dm}(n) \geq 2^{m-1} \gg \exp(\Omega(n^{1/3}))$ (we can always add at least m 1s to get a distinct multiplicities partition of n). Thus $f_{dm}(n)$ still has a broadly exponential growth but it is not clear how to compute it efficiently.

Problem 3.14. *Is the number $f_{dm}(n)$ of distinct-multiplicity partitions of n a PIO function? That is, can we compute it in $O(n^d)$ steps for a constant $d \in \mathbb{N}$?*

D. Zeilberger [152] says: “I conjecture that the fastest algorithm takes exponential time, but I have no idea how to prove that claim.” See [154, A098859] for the first 500 or so terms of $(f_{dm}(n))_{n \geq 1}$. J. A. Fill, S. Janson and M. D. Ward [56] proved that $f_{dm}(n) = \exp((1 + o(1)) \frac{1}{3} (6n)^{1/3} \log n)$ and D. Kane and R. C. Rhoades [84] obtained an even more precise asymptotics.

So far we mostly considered partition counting functions $f(n)$ of broadly exponential growth, satisfying $\log(2 + f(n)) \gg n^c$ for a constant $c > 0$. We

turn to broadly polynomial growth when $\log(2 + f(n)) \ll \log^d(1 + n)$ for a constant $d \in \mathbb{N}$. In Corollary 3.10 we effectively computed $p_A(n)$ for infinite and not too sparse sets $A \subset \mathbb{N}$. At the opposite extreme lie finite sets $A \subset \mathbb{N}$, for them $p_A(n) \ll n^{|A|}$. By the classical result of E. T. Bell [15] (going back to J. Sylvester in 1857, as E. T. Bell himself acknowledges in [15]), then $p_A(n)$ can also be effectively computed.

Proposition 3.15 (E. T. Bell, 1943). *For any finite set $A \subset \mathbb{N}$, the number $p_A(n)$ of partitions $\lambda = (\lambda_1 \geq \dots \geq \lambda_k) \in P(n)$ with all $\lambda_i \in A$ is a rational quasipolynomial in n .*

Hence for every finite A , $p_A(n)$ is a PIO function. Below we extend Proposition 3.15 and indicate how to obtain the PIO algorithm. Recall from the previous section that a quasipolynomial $f : \mathbb{Z} \rightarrow \mathbb{C}$ is determined by a modulus $m \in \mathbb{N}$ and m polynomials $p_i \in \mathbb{C}[x]$, $i \in [m]$, such that $f(n) = p_i(n)$ if $n \equiv i$ modulo m . If $d \geq \deg p_i(x)$ for every $i \in [m]$, we say that the quasipolynomial f has class (m, d) . Replacing in the definition \mathbb{C} with \mathbb{Q} , we get *rational quasipolynomials*. Equivalently, f is a quasipolynomial if and only if

$$f(n) = a_k(n)n^k + \dots + a_1(n)n + a_0(n)$$

for some periodic functions $a_i : \mathbb{Z} \rightarrow \mathbb{C}, \mathbb{Q}$. If $f(n) = p_i(n)$, $n \equiv i$ modulo m , only holds for every $n \geq N$ for some N , we speak of *eventual quasipolynomials*. Particular cases are (eventually) periodic sequences $f : \mathbb{N}_0 \rightarrow \mathbb{C}, \mathbb{Q}$ which are the constant (eventual) quasipolynomials, each polynomial $p_i(x)$ is (eventually) constant.

There are many treatments of E. T. Bell's result in the literature. We mention only R. P. Stanley [138, Chapter 4.4], Ø. J. Rødseth and J. A. Sellers [128], M. Cimpoeaş and F. Nicolae [41, 42], the inductive proof of R. Jakimczuk [78] or the recent S. Robins and Ch. Vignat [127]. The last reference gives a very nice, simple and short proof of Proposition 3.15 by generating functions, which moreover presents a PIO formula for $p_A(n)$, $A = \{a_1, a_2, \dots, a_k\}$, explicitly:

$$p_A(n) = \sum_{j \in J, a_1 j_1 + \dots + a_k j_k \equiv n \pmod{D}} \binom{\frac{1}{D}(n - a_1 j_1 - \dots - a_k j_k) + k - 1}{k - 1}$$

where D is any common multiple of a_1, \dots, a_k and $J = [0, \frac{D}{a_1} - 1] \times \dots \times [0, \frac{D}{a_k} - 1]$. We give yet another proof via a closure property. We prove that the family of rational quasipolynomials is closed under convolution. The (Cauchy) convolution of functions $f, g : \mathbb{N}_0 \rightarrow \mathbb{C}$ is the function $f * g : \mathbb{N}_0 \rightarrow \mathbb{C}$,

$$(f * g)(n) = \sum_{i=0}^n f(i)g(n-i) = \sum_{i+j=n} f(i)g(j).$$

Convolution gives coefficients in products of power series, if $a, b \in \mathbb{C}[[x]]$ then $[x^k]ab = ([x^n]a) * ([x^n]b)(k)$. If we can compute a rational quasipolynomial $f(n)$

and know its class (m, d) , we effectively have a PIO algorithm for $f(n)$. Namely, we compute the first (in fact, any) $d + 1$ values $f(n)$ in each congruence class $n \equiv i$ modulo m and find by Lagrange interpolation the m polynomials $p_i \in \mathbb{Q}[x]$ fitting them. The $p_i(x)$ then constitute the PIO algorithm for $f(n)$. Thus it is useful to know how classes of quasipolynomials transform under convolution (and linear combination). For other closure properties of sequences under the operation of convolution see S. A. Abramov, M. Petkovšek and H. Zakrajšek [1].

Proposition 3.16. *Let $f, g : \mathbb{N}_0 \rightarrow \mathbb{Q}$, $\alpha, \beta \in \mathbb{Q}$ and $N, N' \in \mathbb{N}_0$.*

1. *If f and g are rational quasipolynomials then so is $f * g$. If f and g have classes, respectively, (m, d) and (m', d') then $f * g$ has class $(M, d + d' + 1)$ where M is any common multiple of m and m' .*
2. *If f and g are eventual rational quasipolynomials then so is $\alpha f + \beta g$. If f and g have classes, respectively, (m, d) for $n \geq N$ and (m', d') for $n \geq N'$ then $\alpha f + \beta g$ has class $(M, \max(d, d'))$ for $n \geq \max(N, N')$ where M is any common multiple of m and m' .*
3. *If f and g are eventual rational quasipolynomials then so is $f * g$. If f and g have classes, respectively, (m, d) for $n \geq N$ and (m', d') for $n \geq N'$ then $f * g$ has class $(M, d + d' + 1)$ for $n \geq N + N'$ where M is any common multiple of m and m' .*

Proof. 1. By [138, Chapter 4.4] we have

$$\sum_{n \geq 0} f(n)q^n = \frac{a(q)}{(1 - q^m)^{d+1}} \quad \text{and} \quad \sum_{n \geq 0} g(n)q^n = \frac{b(q)}{(1 - q^{m'})^{d'+1}}$$

for some polynomials $a, b \in \mathbb{Q}[q]$ with degree smaller than that of the denominator. Using the identity

$$1 - q^{ke} = (1 - q^e)(1 + q^e + q^{2e} + \cdots + q^{(k-1)e}), \quad e, k \in \mathbb{N},$$

which is the main trick in [127], we can write for any common multiple M of m and m' the product in the same form

$$\sum_{n \geq 0} f(n)q^n \cdot \sum_{n \geq 0} g(n)q^n = \frac{a(q)}{(1 - q^m)^{d+1}} \cdot \frac{b(q)}{(1 - q^{m'})^{d'+1}} = \frac{c(q)}{(1 - q^M)^{d+d'+2}}$$

where $c \in \mathbb{Q}[q]$ with $\deg c < M(d + d' + 2)$. By expanding the denominator in generalized geometric series we get the result.

2. This is immediate to see.

3. Let $\sum_{n \geq 0} f(n)q^n = a(q) + b(q)$ and $\sum_{n \geq 0} g(n)q^n = c(q) + d(q)$ where $a, c \in \mathbb{Q}[[q]]$ have sequences of coefficients that are rational quasipolynomials with the stated classes for $n \geq 0$ and $b, d \in \mathbb{Q}[q]$ are polynomials with degrees

$\deg b < N$ and $\deg d < N'$. Let M be any common multiple of m and m' . Then $(f * g)(n)$ is the sequence of coefficients in

$$(a + b)(c + d) = ac + ad + bc + bd.$$

By parts 1 and 2, the sequences of coefficients in the last four summands are eventual rational quasipolynomials with classes, respectively, $(M, d + d' + 1)$ for $n \geq 0$, (m, d) for $n \geq N'$ (the sequence of coefficients in ad is a linear combination of $\deg d + 1$ shifts, by numbers $< N'$, of that in a), (m', d') for $n \geq N$, and $(1, 0)$ for $n \geq N + N' - 1$. The result for $f * g$ follows by applying part 2 thrice. \square

We generalize Proposition 3.15 as follows.

Corollary 3.17. *Let $k, m \in \mathbb{N}$ and*

$$g : \mathbb{N} \times \mathbb{N}_0 \rightarrow \{0, 1\}$$

be a function such that $g(i, 0) = 1$ for $i > k$, $g(i, j) = 0$ for $i > k$ and $j > 0$, and for $i \leq k$ each of the k 0-1 sequences $(g(i, j))_{j \geq 0}$ is m -periodic. Then the function $f : \mathbb{N} \rightarrow \mathbb{N}_0$,

$$\begin{aligned} f(n) &= \#\{\lambda = 1^{j_1} 2^{j_2} \dots n^{j_n} \in P(n) \mid g(i, j_i) = 1 \text{ for every } i = 1, 2, \dots, n\} \\ &= [q^n] \prod_{i=1}^k \sum_{j=0}^{\infty} g(i, j) q^{ij}, \end{aligned}$$

is a rational quasipolynomial with class $(m \cdot k!, k - 1)$. The PIO algorithm for $f(n)$ follows constructively from k, m and g .

We state the “eventual” version. Let k, m and g be as before, with the modification that each sequence $(g(i, j))_{j \geq 0}$, $i \leq k$, is m -periodic only for $j \geq N$, for some given $N \in \mathbb{N}_0$. Then the function $f(n)$, defined as above, is an eventual rational quasipolynomial of class $(m \cdot k!, k - 1)$ for $n \geq \binom{k+1}{2} N$. The PIO algorithm for $f(n)$ follows constructively from k, m, N and g .

Proof. For each $i \leq k$ we set $G_i(q) = \sum_{j=0}^{\infty} g(i, j) q^{ij}$. So $G_i \in \{0, 1\}[[q]]$ has im -periodic sequence of coefficients (i.e., with class $(im, 0)$). By $k - 1$ applications of part 1 of Proposition 3.16,

$$f(n) = [q^n] \prod_{i=1}^k G_i(q)$$

is a rational quasipolynomial with class $(m \cdot k!, k - 1)$. In the “eventual” version, $G_i \in \{0, 1\}[[q]]$ has im -periodic sequence of coefficients for $n \geq iN$. The result follows by $k - 1$ applications of part 3 of Proposition 3.16. \square

For finite $A \subset \mathbb{N}$, Proposition 3.15 is the instance with $k = \max(A)$ and $g(i, j) = 1$ if and only if $i \in A$ or $j = 0$. More generally Corollary 3.17 implies, for

example, eventual quasipolynomiality of the numbers of partitions of n with parts in $A = \{3, 4, 27\}$ and such that 3 appears an even number of times, except that multiplicity 2018 is not allowed, and the multiplicity of 27 equals 2 or 7 modulo 11. Not much changes in the proof, using again Proposition 3.16, of the next generalization, and we leave it as an exercise.

Corollary 3.18. *Let $k, m, d \in \mathbb{N}$ and*

$$g : \mathbb{N} \times \mathbb{N}_0 \rightarrow \mathbb{Z}$$

be a function such that $g(i, 0) = 1$ for $i > k$, $g(i, j) = 0$ for $i > k$ and $j > 0$, and for every $i \leq k$ the function $j \mapsto g(i, j)$ is a rational quasipolynomial of class (m, d) . Then $(\lambda = 1^{j_1} 2^{j_2} \dots n^{j_n})$

$$f(n) := \sum_{\lambda \in P(n)} \prod_{i=1}^n g(i, j_i) \in \mathbb{Z}$$

is a rational quasipolynomial with class $(m \cdot k!, k(d+1) - 1)$. The PIO algorithm for $f(n)$ follows constructively from k, m, d and g .

If each function $g(i, j)$, $i \leq k$, is an eventual rational quasipolynomial of class (m, d) for $j \geq N$, for some given $N \in \mathbb{N}_0$, then $f(n)$ is an eventual rational quasipolynomial with class $(m \cdot k!, k(d+1) - 1)$ for $n \geq \binom{k+1}{2} N$. The PIO algorithm for $f(n)$ follows constructively from k, m, d, N and g .

For example, the weighted number of partitions $\lambda \in P_A(n)$, $A = \{3, 4, 27\}$, with the weight of λ equal to $(-1)^m m^5 + 3m$ where m is the multiplicity of 4, is a rational quasipolynomial (we leave determination of its class as an exercise for the reader).

Support of a function is the set of arguments where it attains nonzero values. In Propositions 3.4 and 3.5 we made our life easy by positivity of the function $g(n)$. Another easy case occurs when $g(n)$ is almost always zero.

Corollary 3.19. *If $g : \mathbb{N} \rightarrow \mathbb{Z}$ has finite support $S \subset \mathbb{N}$ with $s = \max(S)$ then both functions $f_1, f_2 : \mathbb{N} \rightarrow \mathbb{Z}$,*

$$f_1(n) = \sum_{\lambda \in P(n)} g(\|\lambda\|) \quad \text{and} \quad f_2(n) = \sum_{\lambda \in Q(n)} g(\|\lambda\|),$$

are rational quasipolynomials with class $(s!, s - 1)$. The PIO algorithm for $f_i(n)$ follows constructively from s and g . (Recall that $Q(n)$ are the partitions of n with no part repeated.)

Proof. Using conjugation of partitions, which is the involution

$$(\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_k) \leftrightarrow 1^{\lambda_1 - \lambda_2} 2^{\lambda_2 - \lambda_3} \dots (k-1)^{\lambda_{k-1} - \lambda_k} k^{\lambda_k},$$

we get the well known identity

$$p_k(n) = p_{[k]}(n) - p_{[k-1]}(n)$$

—the number of partitions of n with k parts equals the number of those partitions of n with parts in $[k]$ that use part k . Thus

$$f_1(n) = \sum_{\lambda \in P(n)} g(\|\lambda\|) = \sum_{k \in S} g(k)(p_{[k]}(n) - p_{[k-1]}(n)).$$

By part 1 of Proposition 3.16, $p_{[k]}(n)$ is a rational quasipolynomial of class $(k!, k-1)$. The result follows by part 2 of Proposition 3.16.

As for $f_2(n)$, we have $f_2(n) = \sum_{k \in S} g(k)q_k(n)$. Conjugation of partitions yields the identity

$$q_k(n) = p_{[k]}(n) - \left| \bigcup_{i=1}^k P_{[k] \setminus \{i\}}(n) \right|$$

—the number of partitions of n into k distinct parts equals the number of those partitions of n with parts in $[k]$ that use each part $1, 2, \dots, k$. Applying the principle of inclusion and exclusion, we express $f_2(n)$ as an integral linear combination of the functions $p_A(n)$ with $A \subset [s]$. The result follows by part 2 of Proposition 3.16. \square

In the literature one can find several interesting enumerative results on partitions involving quasipolynomials. We state the results of D. Zeilberger [152] and of G. E. Andrews, M. Beck and N. Robbins [8], for other quasipolynomial results see A. D. Christopher and M. D. Christopher [39] and V. Jelínek and M. Klazar [79].

Proposition 3.20 (D. Zeilberger, 2012). *For every finite set $A \subset \mathbb{N}$ the number $f(n)$ of the partitions $\lambda \in P_A(n)$ that have distinct nonzero multiplicities is a quasipolynomial in n (and so a PIO function).*

Proposition 3.21 (G. E. Andrews, M. Beck and N. Robbins, 2015). *Let $t \in \mathbb{N}_0$ with $t \geq 2$. The number $f(n)$ of $\lambda = (\lambda_1 \geq \dots \geq \lambda_k) \in P(n)$ with $\lambda_1 - \lambda_k = t$ is a quasipolynomial in n (and so a PIO function).*

For $t = 0$ and 1 the reader can check that, respectively, $f(n) = \tau(n)$ and $f(n) = n - \tau(n)$ where $\tau(n)$ is the number of divisors of n (and not the Ramanujan function which we will discuss too). Sadly, despite their simplicity, we do not know if these are PIO formulas because we do not know how to efficiently factorize numbers. In fact, [8] contains a more general result for prescribed differences between parts.

Quite general result in enumeration and logic involving quasipolynomials was achieved by T. Bogart, J. Goodrick and K. Woods [19]. In the statement we extend in the obvious way the notion of eventual quasipolynomial to function defined on an eventually periodic subset of \mathbb{N}_0 .

Theorem 3.22 (T. Bogart, J. Goodrick and K. Woods, 2017). *Let $d \in \mathbb{N}$ and*

$$t \mapsto X_t \subset \mathbb{Z}^d, t \in \mathbb{N},$$

be a sequence of sets X_t that is defined by a formula in 1-parametric Pressburger arithmetic. Then the set $Y \subset \mathbb{N}$ of $t \in \mathbb{N}$ for which $|X_t| < \infty$ is eventually periodic and $f : Y \rightarrow \mathbb{N}_0$, $f(t) = |X_t|$, is an eventual quasipolynomial (and so a PIO function).

The way of definition of the sequence $t \mapsto X_t$ is that the membership

$$(x_1, x_2, \dots, x_d) \in X_t$$

is defined, for some $k \in \mathbb{N}$, by a formula built by logical connectives and quantification of integer variables from atomic inequalities of the form

$$a_1 y_1 + a_2 y_2 + \dots + a_k y_k \leq b \quad \text{where } a_i, b \in \mathbb{Z}[t]$$

(here enters the single parameter t in the problem) and the y_i are integer variables including the x_i . Consult [19] for details and examples and, of course, for the proof. T. Bogart, J. Goodrick, D. Nguyen and K. Woods prove in [18] that for more than one parameter, polynomial-time computability disappears (assuming $P \neq NP$). We state Propositions 3.20 and 3.21 and Theorem 3.22 in their original form and so do not indicate how to get PIO algorithms from the given data, but with some effort such extensions probably could be obtained from the proofs.

In Corollary 3.17, for $i \leq k$ each 0-1 sequence $(g(i, j))_{j \geq 0}$ recording allowed multiplicities of part i follows a linear (periodic or eventually periodic) pattern. What happens for, say, quadratic patterns? What is the number $f_{x^2+2y^2}(n)$ of partitions $\lambda = 1^{x^2} 2^{y^2} \in P(n)$, $x, y \in \mathbb{N}_0$, that is, partitions of n into parts 1 and 2 with square multiplicities? A nice formula exists:

$$f_{x^2+2y^2}(n) = \frac{\tau_{1,8}(n) + \tau_{3,8}(n) - \tau_{5,8}(n) - \tau_{7,8}(n) + \delta}{2}$$

where $\tau_{i,m}(n)$ counts divisors of n that are i modulo m , $\delta = 1$ if n is a square or twice a square and $\delta = 0$ else. This goes back to P. Dirichlet in 1840, see M.D. Hirschhorn [73] for a proof. From the reason we already mentioned we do not know if it is a PIO formula. We do not know how to count efficiently solutions of equations like $n = x^2 + 2y^2$ or $n = x^2 + y^2$ ($x, y \in \mathbb{N}_0$ or, more classically, $x, y \in \mathbb{Z}$). But if only one of the patterns is quadratic, we can again count efficiently. For example, we can count efficiently solutions of $n = x + 2y^2$:

$$f_{x+2y^2}(n) := \#\{\lambda = 1^{j_1} 2^{j_2} \in P(n) \mid j_i \in \mathbb{N}_0\} = \lfloor \sqrt{n/2} \rfloor + 1$$

is a PIO function (but not a quasipolynomial). We compute it in $\text{poly}(\log n)$ steps as follows. To compute integral square root $n \mapsto \lfloor \sqrt{n} \rfloor$ in $\text{poly}(\log n)$ steps, initialize $m := 0$, add to m in $m := m + 2^r$ the largest power of two such that $m^2 \leq n$, and repeat. When m cannot be increased by adding a power of two, $m = \lfloor \sqrt{n} \rfloor$. See [61, Chapter 9.5 and Exercise 9.43] for faster algorithms. We hope to treat generalizations of $f_{x+2y^2}(n)$ elsewhere.

In the setup of Corollary 3.10, we get for $f(n)$ broadly polynomial growth if $g(n)$ grows at least exponentially. The classical example is for $m \in \mathbb{N}$ with $m \geq 2$ the counting function

$$f_{mp}(n) = f_{mp}(n, m) := p_{\{1, m, m^2, m^3, \dots\}}(n)$$

counting the partitions of n in powers of m , so called *m-ary partitions*. For $m = 2$ we get the *binary partitions*. Binary partitions with distinct parts are easy to count as $\prod_{k=0}^{\infty} (1 + q^{2^k}) = \sum_{n=0}^{\infty} q^n$ (partition theorist's joke). But it appears not easy to count effectively general binary partitions or *m-ary partitions*. “Effectively” here means, of course, in $\text{poly}(\log n)$ steps: $m(n) = \log(1+n) + \log(2 + f_{mp}(n)) = \Theta(\log^2(1+n))$ because K. Mahler [100] proved that $f_{mp}(n) = \exp((1 + o(1))(1/2 \log m) \log^2 n)$. More precise asymptotic relations were derived by N. G. de Bruijn [29], C.-E. Fröberg [59] and others.

Problem 3.23. *Let $m \in \mathbb{N}$ with $m \geq 2$. Is the function*

$$f_{mp}(n) = \#\{(x_i)_{i \geq 0} \subset \mathbb{N}_0 \mid \sum_{i \geq 0} x_i m^i = n\}$$

counting m-ary partitions of n a PIO function? That is, can we compute it in $O(\log^d(1+n))$ steps (bit operations) for a fixed $d \in \mathbb{N}$?

An interesting algorithm of V. P. Bakoev [12] suggests that the answer might be positive.

Proposition 3.24 (V. P. Bakoev, 2004). *Let $m \in \mathbb{N}$ with $m \geq 2$ be given. There is an algorithm computing $f_{mp}(m^n)$ for every $n \in \mathbb{N}$ in $O(n^3)$ arithmetic operations.*

From the literature on *m-ary partitions* we further mention T. Edgar [50], M. D. Hirschhorn and J. A. Sellers [76] and D. Krenn and S. Wagner [97] (which deals mostly with *m-ary compositions*). It is easy to see that the number $f_{bp}(n) := f_{mp}(n, 2)$ of binary partitions of n follows the recurrence $f_{bp}(0) = 1$ and $f_{bp}(n) = f_{bp}(n-1) + f_{bp}(n/2)$ for $n \geq 1$ (where $f_{bp}(n/2) = 0$ if $n/2 \notin \mathbb{N}_0$). The reduction $f'_{bp}(n) := f_{bp}(2n)$ follows the recurrence $f'_{bp}(0) = 1$ and $(n \geq 1) f'_{bp}(n) = f'_{bp}(n-1) + f'_{bp}(\lfloor n/2 \rfloor)$ and forms the sequence [154, A000123],

$$(f_{bp}(2n))_{n \geq 0} = (f'_{bp}(n))_{n \geq 0} = (1, 2, 4, 6, 10, 14, 20, 26, 36, 46, 60, 74, \dots),$$

investigated by D. E. Knuth [90] fifty years ago.

Recently, I. Pak [112, Theorem 2.19] has announced positive resolution of Problem 3.23 in I. Pak and D. Yeliussizov [113, 114]: (we quote verbatim from [112])

Theorem 3.25 (I. Pak and D. Yeliussizov). *Let $\mathcal{A} = \{a_1, a_2, \dots\}$, and suppose a_k/a_{k-1} is an integer ≥ 2 , for all $k > 1$. Suppose also that membership $x \in \mathcal{A}$ can be decided in $\text{poly}(\log x)$ time. Then $\{p_{\mathcal{A}}(n)\}$ can be computed in time $\text{poly}(\log n)$.*

In conclusion of Section 3 and of our article we turn to cancellative problems related to the initial Example 5. Sums of integers with large absolute values still may be small, even 0. In enumeration it means that a formula, effective for nonnegative summands, may no longer be effective (in the sense of Definition 1.1) for integral summands, if cancellations occur. In the next proposition we give both an example and a non-example of such cancellation. The former is a classic but the latter may be not so well known.

Proposition 3.26. *Both functions*

$$q^\pm(n) := \sum_{\lambda \in Q(n)} (-1)^{\|\lambda\|} \quad \text{and} \quad p^\pm(n) := \sum_{\lambda \in P(n)} (-1)^{\|\lambda\|}$$

are PIO functions. Concretely,

$$\sum_{n=0}^{\infty} q^\pm(n) q^n = \prod_{k=1}^{\infty} (1 - q^k) = 1 + \sum_{n=1}^{\infty} (-1)^n (q^{n(3n-1)/2} + q^{n(3n+1)/2})$$

and

$$\sum_{n=0}^{\infty} p^\pm(n) q^n = \prod_{k=1}^{\infty} \frac{1}{1 + q^k} = \prod_{k=1}^{\infty} (1 + (-q)^{2k-1}) = 1 + \sum_{n=1}^{\infty} (-1)^n q_o(n) q^n$$

where $q_o(n) := \#\{\lambda \in Q(n) \mid \lambda_i \equiv 1 \pmod{2}\}$.

Proof. The first identity is the famous *pentagonal identity* of L. Euler [53] (or [6, 68]). Replacing $q(n)$ with $q^\pm(n)$ leads to almost complete cancellation to values just 0 and ± 1 , and $m(n) = \Theta(\log(1+n))$ for $q^\pm(n)$. The algorithms of Propositions 3.5 (recurrence schema) and 3.2 (coefficient extraction from a generating polynomial) still work but do $\text{poly}(n)$ steps and are not effective for computing $n \mapsto q^\pm(n)$. For a PIO formula more efficient algorithm is needed. Fortunately, the pentagonal identity provides it. We easily determine in $\text{poly}(\log n)$ steps the existence of a solution $i \in \mathbb{N}$ to the equation $n = \frac{i(3i \pm 1)}{2}$ and its parity, simply by computing the integral square root as discussed above in connection with $f_{x+2y^2}(n)$.

The second identity, more precisely the middle equality, follows at once from $\frac{1}{1+q^k} = \frac{1-q^k}{1-q^{2k}}$. Now the replacement of $p(n)$ with $p^\pm(n)$ leads to almost no cancellation because

$$|p^\pm(n)| = q_o(n) = [q^n] \prod_{k=1}^{\infty} (1 + q^{2k-1}) \sim \frac{\exp(\pi \sqrt{n/6})}{2^{3/2} \cdot 6^{1/4} \cdot n^{3/4}}$$

([154, A000700]; V. Kotěšovec [94, p. 9]; G. Meinardus [102, p. 301]) remains of broadly exponential growth. Thus for $p^\pm(n)$ we have $m(n) \gg n^{1/2}$ and both algorithms for $p(n)$ remain effective for $p^\pm(n)$. Hence, more easily than for $q^\pm(n)$, $p^\pm(n)$ is a PIO function. \square

We had derived the second identity and then we learned in A. Ciolan [43] that it is in fact due to J.W.L. Glaisher [62]. The above examples lead us to the following question.

Problem 3.27. *Find general sufficient conditions on the functions*

$$a = a_i : \mathbb{N} \rightarrow \mathbb{N}_0 \quad \text{and} \quad b = b_{i,j,k} : \mathbb{N} \times \mathbb{N}_0 \times \mathbb{N} \rightarrow \{0, 1\}$$

ensuring that

$$f(n) = [q^n] \prod_{i=1}^{\infty} \prod_{k=1}^{a_i} \sum_{j=0}^{\infty} b_{i,j,k} (-1)^j q^{ij}$$

is a PIO function. Find asymptotics of $f(n)$.

Thus $f(n)$ is the $(-1)^{\|\lambda\|}$ -count of the partitions λ of n into parts $i \in \mathbb{N}$ coming in a_i sorts $(i, 1), (i, 2), \dots, (i, a_i)$ such that the part $(i, k)^j$ may appear if and only if $b_{i,j,k} = 1$. For $a_i = 1$, $b_{i,0,1} = b_{i,1,1} = 1$, and $b_{i,j,k} = 0$ else we get $q^{\pm}(n)$, and for $a_i = 1$, $b_{i,j,1} = 1$, and $b_{i,j,k} = 0$ else we get $p^{\pm}(n)$. For $a_i = 2$, $b_{i,0,1} = b_{i,1,1} = b_{i,0,2} = b_{i,1,2} = 1$, and $b_{i,j,k} = 0$ else we get the function $f(n)$ of Example 5. More generally, for $l \in \mathbb{N}$ the counting functions

$$\sum_{n=0}^{\infty} q^{\pm, l}(n) q^n := \prod_{n=1}^{\infty} (1 - q^n)^l$$

correspond to $a_i = l$, $b_{i,0,1} = b_{i,1,1} = b_{i,0,2} = b_{i,1,2} = \dots = b_{i,0,l} = b_{i,1,l} = 1$, and $b_{i,j,k} = 0$ else; $q^{\pm}(n) = q^{\pm, 1}(n)$ and Example 5 is $q^{\pm, 2}(n)$. A related open problem, due to D. Newman, is mentioned in G.E. Andrews and D. Newman [9]: In

$$\sum_{n \geq 0} p(n) q^n = \prod_{k=1}^{\infty} (1 + p^k + p^{2k} + \dots),$$

can one change some signs in the last product so that on the left side the $p(n)$ turn to coefficients 0 and ± 1 ?

Another example of non-cancellation in Problem 3.27 is the result of A. Ciolan [43, (22)]: if $S_2 := \{1, 4, 9, 16, \dots\}$, $B := \Gamma(3/2)\zeta(3/2)/2\sqrt{2}$ and

$$t_n := \sum_{\lambda \in P(n), \lambda_i \in S_2} (-1)^{\|\lambda\|} = [q^n] \prod_{k \geq 1} \frac{1}{1 + q^{k^2}}$$

then

$$t_n \sim (-1)^n \frac{\exp(3(B/2)^{2/3} n^{1/3})}{(3\pi)^{1/2} (2n)^{5/6} / B^{1/3}}.$$

In [154, A292520], for example

$$(t_n)_{n=32}^{49} = (1, -2, 3, -4, 3, -2, 1, 0, 1, -2, 3, -4, 3, -2, 1, 0, 0, -2),$$

V. Kotěšovec gave this asymptotic formula as well, without proof. If

$$s_n := [q^n] \prod_{k \geq 1} (1 - q^{k^2})$$

is the corresponding number for distinct squares, with the help of MAPLE we get the values

$$(s_n)_{n=0}^{15} = (1, -1, 0, 0, -1, 1, 0, 0, 0, -1, 1, 0, 0, 1, -1, 1, 0)$$

or

$$(s_n)_{n=2990}^{3000} = (111, -112, 61, 46, -114, 116, -21, 11, -30, -17, 37)$$

and $\max_{n \leq 3000} |s_n| = 319$. It is [154, A276516].

Problem 3.28. *Is (s_n) unbounded?*

We finish with the generalization of Example 5 to the numbers $q^{\pm, l}(n)$, $l \in \mathbb{N}$. These result from an almost complete cancellation because by the pentagonal identity,

$$|q^{\pm, l}(n)| \leq [q^n] \left(\sum_{n=0}^{\infty} |q^{\pm, 1}(n)| q^n \right)^l \leq [q^n] (1 - q)^{-l} = \binom{n + l - 1}{l - 1}.$$

So $q^{\pm, l}(n) = O(n^{l-1})$, $m(n) = \Theta_l(\log(1 + n))$ for this counting problem and effective computation of $q^{\pm, l}(n)$ means computation in $\text{poly}(\log n)$ steps. Besides the pentagonal identity for $l = 1$, another nice identity occurs for $l = 3$:

$$\prod_{n \geq 1} (1 - q^n)^3 = \sum_{n \geq 0} (-1)^n (2n + 1) q^{n(n+1)/2},$$

due to C. Jacobi (G. H. Hardy and E. M. Wright [68, Theorem 357]). Thus also $q^{\pm, 3}(n)$ is a PIO function. For $l = 2$, Example 5, we get

$$(q^{\pm, 2}(n))_{n \geq 0} = (1, -2, -1, 2, 1, 2, -2, 0, -2, -2, 1, 0, 0, 2, 3, -2, 2, 0, \dots)$$

or

$$(q^{\pm, 2}(n))_{n=58}^{75} = (0, -2, 0, -2, 0, -2, 2, 0, -4, 0, 0, -2, -1, 2, 0, 2, 0, 0),$$

[154, A002107], not showing any clear pattern. J.W.L. Glaisher [63, p. 183] writes: “I had no hope that these coefficients would follow any simple law, as in the Eulerian or Jacobian series; for, if such a law existed, it could not fail to have been discovered long ago by observation.” Let us see how we advanced in 130 years. In August 2018 the “links” section of the entry [154, A002107] (author N. J. A. Sloane) lists these references: table of first 10000 values by S. Manyama, G.E. Andrews [7], M. Boylan [28], S. Finch [57], J.W.L. Glaisher [63], J. T. Joichi [82], V. G. Kač and D. H. Peterson [83], M. Koike [93], V. Kotěšovec [95],

Y. Martin [101], T. Silverman [134], index to 74 sequences in [101] by M. Somos, M. Somos [136], and article Ramanujan Theta Functions in E. Weisstein [156]. J. W. L. Glaisher [63] did discover and prove a kind of simple pattern for this sequence, which we state in the elegant form given in [57] (the other references seem not relevant for computation of $q^{\pm,2}(n)$): $q^{\pm,2}(n) = G(12n + 1)$ where $G : \mathbb{N} \rightarrow \mathbb{Z}$ is a multiplicative function, which means that $G(ab) = G(a)G(b)$ whenever $a, b \in \mathbb{Z}$ are coprime numbers, defined on prime powers p^r , $r \in \mathbb{N}$, by

$$G(p^r) = \begin{cases} 1 & \dots \quad p \equiv 7, 11 \pmod{12}, r \equiv 0 \pmod{2}, \\ (-1)^{r/2} & \dots \quad p \equiv 5 \pmod{12}, r \equiv 0 \pmod{2}, \\ r + 1 & \dots \quad p \equiv 1 \pmod{12}, (-3)^{(p-1)/4} \equiv 1 \pmod{p}, \\ (-1)^r(r + 1) & \dots \quad p \equiv 1 \pmod{12}, (-3)^{(p-1)/4} \equiv -1 \pmod{p}, \\ 0 & \dots \quad \text{otherwise.} \end{cases}$$

This is a PIO formula for $G(n) = G(p^r)$ if n is a known prime power. But in general we do not know if $q^{\pm,2}(n)$ is a PIO function because we do not know how to effectively factorize numbers. For example, $q^{\pm,2}(58) = G(697) = G(17)G(41) = 0 \times \dots = 0$ and $q^{\pm,2}(59) = G(709) = (-1)^1(1 + 1) = -2$ because 709 is a prime that is 1 modulo 12 and $(-3)^{177} = -3^{2^7} 3^{2^5} 3^{2^4} 3 \equiv -1$ modulo 709 as $3^{16} \equiv 495$, $3^{32} \equiv 420$ and $3^{128} \equiv 29$.

For repeated parts,

$$\sum_{n \geq 0} p^{\pm,2}(n)q^n := \prod_{n \geq 1} \frac{1}{(1 + q^n)^2} = \prod_{n \geq 1} (1 + (-q)^{2n-1})^2,$$

we get

$$(p^{\pm,2}(n))_{n \geq 0} = (1, -2, 1, -2, 4, -4, 5, -6, 9, -12, 13, -16, 21, -26, \dots),$$

[154, A022597], and see, like before, that $p^{\pm,2}(n)$ is also $(-1)^n$ times the number of partitions of n into 2-sorted distinct odd numbers and that we have almost no cancellation.

For $l = 24$ a shift of $q^{\pm,24}(n)$ gives the *Ramanujan tau function* $\tau(n)$,

$$\sum_{n \geq 0} \tau(n)q^n := q \prod_{n \geq 1} (1 - q^n)^{24}.$$

So

$$(\tau(n))_{n \geq 1} = (1, -24, 252, -1472, 4830, -6048, -16744, 84480, -113643, \dots),$$

[154, A000594]. Combinatorially, $\tau(n)$ is the $(-1)^{|\lambda|}$ -count of partitions λ of $n - 1$ into parts in 24-sorted \mathbb{N} (not that this would really help for deriving properties of $\tau(n)$). Is $\tau(n)$ a PIO function, can we compute it effectively, in poly($\log n$) steps (as we noted above, $\tau(n) = O(n^{23})$)? The Wikipedia article [155] on tau function is silent about this fundamental aspect but the simple and unsatisfactory answer is again that we do not know. If we could effectively factorize numbers, we could besides decoding secret messages also compute $\tau(n)$

effectively: (i) $\tau(mn) = \tau(m)\tau(n)$ if m and n are coprime, (ii) $\tau(p^{k+2}) = \tau(p)\tau(p^{k+1}) - p^{11}\tau(p^k)$ for every prime number p and every $k \in \mathbb{N}_0$ and (iii) $\tau(p)$ can be computed in $\text{poly}(\log p)$ steps for every prime p . The first two properties, conjectured by S. Ramanujan, were proved by L. J. Mordell [105] and the whole book [51], edited and mostly written by B. Edixhoven and J.-M. Couveignes, is devoted to exposition of an algorithm proving (iii).

Problem 3.29. *Are, in the current state of knowledge, the known PIO functions $q^{\pm,l}(n)$ only those for $l = 1$ and 3 ? For which $l \in \mathbb{N}$ can one compute $q^{\pm,l}(n)$ in $\text{poly}(\log n)$ steps with the help of an oracle that can factorize integers efficiently?*

From the extensive literature on the numbers $q^{\pm,l}(n)$ we further mention only H. H. Chan, S. Cooper and P. Ch. Toh [35, 36] (check the former for $l = 26$), E. Clader, Y. Kemper and M. Wage [44] and J.-P. Serre [132].

Acknowledgments. The OEIS database [154] was very helpful. I thank I. Pak for valuable comments and references.

References

- [1] S. A. Abramov, M. Petkovšek and H. Zakrajšek, Convolutions of Liouvillian sequences, arXiv:1803.08747v1, 2018, 24 pages.
- [2] M. Agrawal, N. Kayal and N. Saxena, PRIMES is in P, *Ann. Math.* **160** (2004), 781–793.
- [3] M. Aigner, *A Course in Enumeration*, Springer, Berlin, 2007.
- [4] J. L. R. Alfonsín, *The Diophantine Frobenius Problem*, Oxford University Press, Oxford, 2005.
- [5] F. Amoroso and U. Zannier (editors), *Diophantine Approximation. Lectures given at the C.I.M.E. Summer School held in Cetraro, Italy, June 28–July 6, 2000*, Lecture Notes in Mathematics 1819, Springer, Berlin, 2003.
- [6] G. E. Andrews, *The Theory of Partitions*, Addison-Wesley, Reading, MA, 1976.
- [7] G. E. Andrews, Advanced problem 6562, *Amer. Math. Monthly* **94** (1987), 1011.
- [8] G. E. Andrews, M. Beck and N. Robbins, Partitions with fixed differences between largest and smallest parts, *Proc. Amer. Math. Soc.* **143** (2015), 4283–4289.
- [9] G. E. Andrews and D. Newman, Binary representations of theta functions, *Integers* **18** (2018), #A34, 8 pages.
- [10] F. Ardila, *Algebraic and Geometric Methods in Enumerative Combinatorics*, arXiv:1409.2562v2, 2014, 144 pages (also Chapter 1 in [22]).
- [11] E. Bach and J. Shallit, *Algorithmic Number Theory, Vol. 1: Efficient Algorithms*, The MIT Press, Cambridge, MA, 1996.
- [12] V. P. Bakoev, Algorithmic approach to counting of certain types m -ary partitions, *Discrete Math.* **275** (2004), 17–41.
- [13] Y. Bar-Hillel (editor), *Logic, Methodology and Philosophy of Science. Proceedings of the 1964 International Congress*, North-Holland, Amsterdam, 1965.

- [14] V. Becher, P. A. Heiber and T. A. Slaman, A polynomial-time algorithm for computing absolutely normal numbers, *Information and Computation* **232** (2013), 1–9.
- [15] E. T. Bell, Interpolated denumerants and Lambert series, *Amer. J. Math.* **65** (1943), 382–386.
- [16] F. Bergeron, G. Labelle and P. Leroux, *Combinatorial Species and Tree-like Structures*, Cambridge University Press, Cambridge, 1998.
- [17] M. Bodirsky, C. Gröpl and M. Kang, Generating labeled planar graphs uniformly at random, *Theoretical Comp. Sci.* **379** (2007), 377–386.
- [18] T. Bogart, J. Goodrick, D. Nguyen and K. Woods, Parametric Presburger arithmetic: complexity of counting and quantifier elimination, arXiv:1802.00974v1, 2018, 14 pages.
- [19] T. Bogart, J. Goodrick and K. Woods, Parametric Presburger arithmetic: logic, combinatorics, and quasipolynomial behavior, *Discrete Analysis*, 2017, 34 pages.
- [20] J. Bohman, C.-E. Fröberg and H. Riesel, Partitions in squares, *BIT* **19** (1979), 297–301.
- [21] E. Bombieri and W. Gubler, *Heights in Diophantine Geometry*, Cambridge University Press, Cambridge, 2006.
- [22] M. Bóna (editor), *Handbook of Enumerative Combinatorics*, CRC Press, Boca Raton, FL, 2015.
- [23] P. B. Borwein, On the complexity of calculating factorials, *J. of Algorithms* **6** (1985), 376–380.
- [24] A. Bostan, X. Caruso, G. Christol and P. Dumas, Fast coefficient computation for algebraic power series in positive characteristic, arXiv:1806.06543v1, 2018, 16 pages.
- [25] A. Bostan, G. Christol and P. Dumas, Fast computation of the N th term of an algebraic series in positive characteristic, in: *ISSAC'16 Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation (2016)*, 119–126.
- [26] A. Bostan, P. Gaudry and E. Schost, Linear recurrences with polynomial coefficients and application to integer factorization and Cartier–Manin operator, *SIAM J. Comput.* **36** (2007), 1777–1806.
- [27] D. Bowman, P. Erdős and A. Odlyzko, Partitions of n into parts which are divisors of n , *Amer. Math. Monthly* **99** (1992), 276–277.
- [28] M. Boylan, Exceptional congruences for the coefficients of certain eta-product newforms, *J. Number Theory* **98** (2003), 377–389.
- [29] N. G. de Bruijn, On Mahler’s partition problem, *Proc. Kon. Ned. Akad. v. Wet. Amsterdam* **51** (1948), 659–669.
- [30] J. H. Bruiner and K. Ono, Algebraic formulas for the coefficients of half-integral weight harmonic weak Maas forms, *Adv. Math.* **246** (2013), 198–219.
- [31] J. H. Bruiner, K. Ono and A. V. Sutherland, Class polynomials for nonholomorphic modular functions, *J. Number Theory* **161** (2016), 204–229.
- [32] N. Calkin, J. Davis, K. James, E. Perez and C. Swannack, Computing the integer partition function, *Math. Comp.* **76** (2007), 1619–1638.

- [33] P. J. Cameron, *Notes on Counting: An Introduction to Enumerative Combinatorics*, 2010, 217 pages (available from P. J. Cameron's homepage).
- [34] P. J. Cameron, *Notes on Counting: An Introduction to Enumerative Combinatorics*, Cambridge University Press, Cambridge, 2017.
- [35] H. H. Chan, S. Cooper and P. Ch. Toh, The 26th power of Dedekind's η -function, *Adv. in Math.* **207** (2006), 532–543.
- [36] H. H. Chan, S. Cooper and P. Ch. Toh, Ramanujan's Eisenstein series and powers of Dedekind's η -function, *J. London Math. Soc.* **75** (2007), 225–242.
- [37] Y. Choliy, L. W. Kolitsch and A. V. Sills, Partition recurrences, *Integers* **18B** (2018), article A1, 15 pages.
- [38] G. Christol, Globally bounded solutions of differential equations, in: [107], 45–64.
- [39] A. D. Christopher and M. D. Christober, Estimates of five restricted partition functions that are quasi polynomials, *Bull. Math. Sci.* **5** (2015), 1–11.
- [40] W. Chu, G. Gardin, S. Ohsuga and Y. Kambayashi (editors), *Proc. 7th International Conference on Very Large Data Bases*, Morgan Kaufmann, Cannes, 1981.
- [41] M. Cimpoeaş and F. Nicolae, On the restricted partition function, arXiv:1609.06090v1, 2016, 20 pages (to appear in *The Ramanujan J.*).
- [42] M. Cimpoeaş and F. Nicolae, On the restricted partition function II, arXiv:1611.00256v4, 2018, 12 pages.
- [43] A. Ciolan, Asymptotics and inequalities for partitions into squares, arXiv:1806.00708v1, 2018, 16 pages.
- [44] E. Clader, Y. Kemper and M. Wage, Lacunarity of certain partition-theoretic generating functions, *Trans. Amer. Math. Soc.* **137** (2009), 2959–2968.
- [45] A. Cobham, The intrinsic computational difficulty of functions, in: [13], 24–30.
- [46] S. Cohen, B. Kimelfeld and Y. Sagiv, Generating all maximal induced subgraphs for hereditary and connected-hereditary graph properties, *J. Computer and System Sci.* **74** (2008), 1147–1159.
- [47] S. Cohen and Y. Sagiv, An incremental algorithm for computing ranked full disjunction, *J. Computer and System Sci.* **73** (2007), 648–668.
- [48] L. Comtet, *Advanced Combinatorics: The Art of Finite and Infinite Expansions*, D. Reidel, Dordrecht, 1974 (first published in French in 1970).
- [49] A. Dunn and N. Robles, Polynomial partition asymptotics, arXiv:1705.00384v1, 2017, 22 pages.
- [50] T. Edgar, On the number of hyper m -ary partitions, *Integers* **18** (2018), article A47.
- [51] B. Edixhoven and J.-M. Couveignes (editors), *Computational aspects of modular forms and Galois representations. How one can compute in polynomial time the values of Ramanujan's tau at a prime*, Princeton University Press, Princeton, NJ, 2011.
- [52] J. Edmonds, Path, trees, and flowers, *Canad. J. Math.* **17** (1965), 449–467.

- [53] L. Euler, Evolutio producti infiniti $(1-x)(1-xx)(1-x^3)(1-x^4)(1-x^4)(1-x^5)(1-x^6)$ etc. in seriem simplicem, *Acta Academiae Scientiarum Imperialis Petropolitanae (1780)* (1783), 47–55.
- [54] G. Everest, A. van der Poorten, I. Shparlinski and T. Ward, *Recurrence Sequences*, AMS, Providence, RI, 2003.
- [55] J.-H. Evertse, On sums of S -units and linear recurrences, *Compositio Math.* **53** (1984), 225–244.
- [56] J. A. Fill, S. Janson and M. D. Ward, Partitions with distinct multiplicities of parts: On an “Unsolved Problem” posed by Herbert Wilf, *Electronic J. Combinatorics* **19** (2012), Paper #P18, 6 pages.
- [57] S. Finch, Powers of Euler’s q -series, ArXiv:math/0701251v2, 2007, 17 pages.
- [58] P. Flajolet and R. Sedgewick, *Analytic Combinatorics*, Cambridge University Press, Cambridge, 2009.
- [59] C.-E. Fröberg, Accurate estimation of the number of binary partitions, *BIT* **17** (1977), 386–391.
- [60] A. Gafni, Power partitions, *J. Number Theory* **163** (2016), 19–42.
- [61] J. von zur Gathen and J. Gerhard, *Modern Computer Algebra*, Cambridge University Press, Cambridge, 2013 (3rd edition).
- [62] J. W. L. Glaisher, On formulae of verification in the partition of numbers, *Proc. Royal Soc. London* **24** (1876), 250–259.
- [63] J. W. L. Glaisher, On the square of Euler’s series, *Proc. London Math. Soc.* **21** (1889), 182–215.
- [64] T. Gowers, J. Barrow-Green and I. Leader (editors), *The Princeton Companion to Mathematics*, Princeton University Press, Princeton, NJ, 2008.
- [65] Y. Gurevich and S. Shelah, Time polynomial in input or output, *J. Symb. Logic* **54** (1989), 1083–1088.
- [66] G. Halász (editor), *Topics in Classical Number Theory. Vol. II (Colloq. Math. Soc. J. Bolyai 34)*, North-Holland, Amsterdam, 1984.
- [67] G. H. Hardy and S. Ramanujan, Asymptotic formulae in combinatory analysis, *Proc. London Math. Soc.* **2** (1918), 75–115.
- [68] G. H. Hardy and E. M. Wright, *An Introduction to the Theory of Numbers*, Clarendon Press, Oxford, 1979 (fifth edition, first published in 1938).
- [69] D. Harvey and J. van der Hoeven, Faster integer multiplication using short lattice vectors, arXiv:1802.07932v1, 2018, 16 pages.
- [70] D. Harvey, J. van der Hoeven and G. Lecerf, Even faster integer multiplication, *J. Complexity* **36** (2016), 1–30.
- [71] F. C. Hennie, One-tape, off-line Turing machine computations, *Information and Control* **8** (1965), 553–578.
- [72] S. Heubach and T. Mansour, *Combinatorics of Compositions and Words*, Chapman and Hall/CRC, Boca Raton, FL, 2009.
- [73] M. D. Hirschhorn, Partial fractions and four classical theorems of number theory, *Amer. Math. Monthly* **107** (2000), 260–264.

- [74] M. D. Hirschhorn, The number of partitions of a number into distinct squares, *Math. Gazette* **90** (2006), 80–87.
- [75] M. D. Hirschhorn, My contact with Ramanujan, *J. Indian Math. Soc. (N. S.)*, (2013), 33–43 (special volume to commemorate the 125th birth anniversary of Srinivasa Ramanujan).
- [76] M. D. Hirschhorn and J. A. Sellers, A different view of m -ary partitions, *Australasian J. Combinatorics* **30** (2004), 193–196.
- [77] S. G. Hyun, S. Melczer and C. St-Pierre, A fast algorithm for solving linearly recurrent sequences, arXiv:1806.03554v1, 2018, 4 pages.
- [78] R. Jakimczuk, Restricted partitions, *Internat. J. of Mathem. and Mathem. Sciences* **36** (2004), 1893–1896.
- [79] V. Jelínek and M. Klazar, Generalizations of Khovanskii’s theorem on the growth of sumsets in Abelian semigroups, *Adv. in Appl. Math.* **41** (2008), 115–132.
- [80] F. Johansson, Efficient implementation of the Hardy–Ramanujan–Rademacher formula, *LMS J. Comput. Math.* **15** (2012), 341–359.
- [81] F. Johansson, New partition function record: $p(10^{20})$ computed, fredrikj.net/blog/2014/new-partition-function-record/, 2014 (viewed in August 2016).
- [82] J. T. Joichi, Hecke–Rogers, Andrews identities; combinatorial proofs, *Discrete Math.* **84** (1990), 255–259.
- [83] V. G. Kač and D. H. Peterson, Infinite-dimensional Lie algebras, theta functions and modular forms, *Adv. in Math.* **53** (1984), 125–264.
- [84] D. Kane and R. C. Rhoades, Asymptotics for Wilf’s partitions with distinct multiplicities, preprint, 2012, 8 pages.
- [85] Y. Kanza and Y. Sagiv, Computing full disjunctions, in: *Proceedings of the 22nd ACM SIGMOND-SIGACT-SIGART Symposium on Principles of Database Systems*, ACM Press, San Diego, CA, 2003, 78–89.
- [86] M. Kauers and D. Zeilberger, A simple re-derivation of Onsager’s solution of the 2D Ising model using experimental mathematics, arXiv:1805.09057v1, 2018, 10 pages.
- [87] J. Kennedy (editor), *Interpreting Gödel. Critical Essays*, Cambridge University Press, Cambridge, 2014.
- [88] M. Klazar, Overview of some general results in combinatorial enumeration, in: [98], 3–40.
- [89] A. Knopfmacher and N. Robbins, Identities for the total number of parts in partitions of integers, *Util. Math.* **67** (2005), 9–18.
- [90] D. E. Knuth, An almost linear recurrence, *Fibonacci Q.* **4** (1966), 117–128.
- [91] D. E. Knuth, *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*, Addison-Wesley, Reading, MA, 1997.
- [92] D. E. Knuth, *The Art of Computer Programming, Pre-fascicle 3B, A Draft of Sections 7.2.1.4–5: Generating all Partitions*, preprint, 2004.
- [93] M. Koike, On McKay conjecture, *Nagoya Math. J.* **95** (1984), 85–89.
- [94] V. Kotěšovec, A method of finding the asymptotics of q -series based on the convolution of generating functions, arXiv:1509.08708v3, 2016, 28 pages.

- [95] V. Kotěšovec, The integration of q -series, oeis.org/A258232/a258232_2.pdf, 2015, 3 pages (viewed in August 2018).
- [96] C. Krattenthaler and T. W. Müller, Motzkin numbers and related sequences modulo powers of 2, arXiv:1608.05657v1, 2016, 28 pages.
- [97] D. Krenn and S. Wagner, Compositions into powers of b : asymptotic enumeration and parameters, *Algorithmica* **75** (2016), 606–631.
- [98] S. Linton, N. Ruskuc and V. Vatter (editors), *Permutation Patterns. St. Andrews 2007*, vol. 376 of London Mathematical Society Lecture Note Series, Cambridge University Press, Cambridge, 2010.
- [99] S. M. Luthra, On the average number of summands in partitions of n , *Proc. Nat. Inst. Sci. India, Part A* **23** (1957), 483–498.
- [100] K. Mahler, On a special functional equation, *J. London Math. Soc.* **68** (1940), 115–123.
- [101] Y. Martin, Multiplicative η -quotients, *Trans. of the Amer. Math. Soc.* **348** (1996), 4825–4856.
- [102] G. Meinardus, Über Partitionen mit Differenzenbedingungen, *Math. Zeit.* **61** (1954), 289–302.
- [103] J. C. P. Miller and D. J. Spencer Brown, An algorithm for evaluation of remote terms in a linear recurrence sequence, *Comput. J.* **9** (1966), 188–190.
- [104] V. S. Miller, Counting matrices that are squares, arXiv:1606.09299v1, 2016, 37 pages.
- [105] L. J. Mordell, On Mr. Ramanujan’s empirical expansions of modular functions, *Proc. Cambridge Phil. Soc.* **19** (1917), 117–124.
- [106] M. V. N. Murthy, M. Brack, R. K. Bhaduri and J. Bartel, Semi-classical analysis of distinct square partitions, arXiv:1808.05146v1, 2018, 38 pages.
- [107] K. Nagasaka and E. Fouvry (editors), *Analytic Number Theory*, Lectures Notes in Mathematics 1434, Springer, Berlin, 1990.
- [108] B. Nakamura, Approaches for enumerating permutations with a prescribed number of occurrences of patterns, *Pure Math. Appl.* **24** (2013), 179–194.
- [109] B. Nakamura and D. Zeilberger, Using Noonan–Zeilberger functional equation to enumerate (in polynomial time!) generalized Wilf classes, *Adv. in Appl. Math.* **50** (2013), 356–366.
- [110] K. Ono, The distribution of the partition function modulo m , *Ann. of Math.* **151** (2000), 293–307.
- [111] K. Ono, *The Web of Modularity: Arithmetic of the Coefficients of Modular Forms and q -series*, AMS, Providence, RI, 2004.
- [112] I. Pak, Complexity problems in enumerative combinatorics, arXiv:1803.06636v2, 2018, 31 pages.
- [113] I. Pak and D. Yeliussizov, in preparation.
- [114] I. Pak and D. Yeliussizov, Ehrhart polynomial of some Schläfli simplices, 2018 (talk at Joint Mathematics Meeting, San Diego, Ca, January 2018).
- [115] Ch. H. Papadimitriou, *Computational Complexity*, Addison-Wesley, Reading, MA, 1994.

- [116] T. R. Parkin and D. Shanks, On the distribution of parity in the partition function, *Math. Comput.* **21** (1967), 466–480.
- [117] R. Pemantle and M. C. Wilson, *Analytic Combinatorics in Several Variables*, Cambridge University Press, Cambridge, 2013.
- [118] G. Pólya, Kombinatorische Anzahlbestimmungen für Gruppen, Graphen and chemische Verbindungen, *Acta Math.* **68** (1937), 145–254.
- [119] B. Poonen, Undecidable problems: a sampler, in: [87], 211–241.
- [120] A. van der Poorten, Some problems of recurrent interest, *Macquarie Math. Reports*, Macquarie University, Northridge, Australia, 81-0037 (1981).
- [121] A. van der Poorten, Some problems of recurrent interest, in: [66], 1265–1294.
- [122] A. van der Poorten and H. P. Schlickewei, Additive relations in fields, *J. Austral. Math. Soc. Ser. A* **51** (1991), 154–170.
- [123] V. V. Prasolov, *Polynomials*, Springer, Berlin, 2004 (translated from the Russian by Dimitry Leites).
- [124] Ch. Reutenauer, On a matrix representation for polynomially recursive sequences, *Electr. J. Combin.* **19** (2012), #P36, 26 pages.
- [125] B. Richmond and A. Knopfmacher, Compositions with distinct parts, *Aequat. Math.* **49** (1995), 86–97.
- [126] J. Riordan, *Introduction to Combinatorial Analysis*, John Wiley, New York, 1958.
- [127] S. Robins and Ch. Vignat, Simple proofs and expressions for the restricted partition function and its polynomial part, arXiv:1802.07310v1, 2018, 5 pages.
- [128] Ø. J. Rødseth and J. A. Sellers, Partitions with parts in a finite set, *Int. J. Number Theory* **2** (2006), 455–468.
- [129] E. Rowland, What is . . . an automatic sequence?, *Notices AMS* **62** (2015), 274–276.
- [130] E. Rowland and R. Yassawi, Automatic congruences for diagonals of rational functions, *J. Théor. Nombres Bordeaux* **27** (2015), 245–288.
- [131] W. M. Schmidt, Linear recurrence sequences, in: [5], 171–247.
- [132] J.-P. Serre, Sur la lacunarité des puissances de η , *Glasgow Math. J.* **27** (1985), 203–221.
- [133] J. Shallit, The Logical Approach to Automatic Sequences. Part 4: Enumeration and Automatic Sequences, 2016,
<https://cs.uwaterloo.ca/~Shallit/Talks/linz4a.pdf>
(viewed September 2018)
- [134] T. Silverman, Counting cliques in finite distant graphs, arXiv:1612.08085v1, 2016, 16 pages.
- [135] T. Skolem, Ein Verfahren zur Behandlung gewisser exponentialer Gleichungen, in: *Comptes rendus du congrès des mathématiciens scandinaves*, Stockholm, 1934 (1935), 163–188.
- [136] M. Somos, Introduction to Ramanujan theta functions,
<https://somas.crg4.com/multiq.pdf> (viewed in August 2018).

- [137] R. Sprague, Über Zerlegungen in ungleiche Quadratzahlen, *Math. Z.* **51** (1948), 289–290.
- [138] R. P. Stanley, *Enumerative Combinatorics. Volume I*, Wadsworth & Brooks/Cole, Monterey, CA, 1986.
- [139] M. Stoll, Rational and transcendental growth series for the higher Heisenberg groups, *Invent. Math.* **126** (1996), 85–109.
- [140] D. S. Stones, The many formulae for the number of Latin squares, *Electron. J. Combin.* **17** (2010), #A1, 46 pages.
- [141] T. Tao, *Structure and Randomness: pages from year one of a mathematical blog*, AMS, Providence, NJ, 2008.
- [142] M. N. Tran, M. V. N. Murty and R. K. Bhaduri, On the quantum density of states and partitioning an integer, *Ann. Phys.* **311** (2004), 204–219.
- [143] M. Vardi, On the complexity of bounded-variable queries, in: *Proc. 14th Symposium on Principles of Database Systems*, ACM Press, San Jose, CA, 1995, 266–276.
- [144] V. Vatter, Enumeration schemes for restricted permutations, *Combinatorics, Probability and Computing* **17** (2008), 137–159.
- [145] R. C. Vaughan, Squares: additive questions and partitions, *Int. J. Number Theory* **11** (2015), 1367–1409.
- [146] H. S. Wilf, What is an answer?, *Amer. Math. Monthly* **89** (1982), 289–292.
- [147] H. S. Wilf, Some unsolved problems, 3 pages, www.math.upenn.edu/~wilf/website/UnsolvedProblems.pdf, posted Dec. 13, 2010 (viewed in August 2018).
- [148] E. M. Wright, Asymptotic partition formulae, III. Partitions into k -th powers, *Acta Math* **63** (1934), 143–191.
- [149] M. Yanakakis, Algorithms for acyclic database schemes, in: [40], 82–94.
- [150] U. Zannier, *Lecture Notes on Diophantine Analysis*, Scuola Normale Superiore, Pisa, 2009.
- [151] D. Zeilberger, Enumerative and Algebraic Combinatorics, in: [64], 550–561.
- [152] D. Zeilberger, Using GENERATINGFUNCTIONOLOGY to enumerate distinct-multiplicity partitions, ArXiv:1201.493v1, 2012, 6 pages.
- [153] Complexity Zoo, <http://complexityzoo.uwaterloo.ca> (zoo keeper: Scott Aaronson).
- [154] The On-line Encyclopedia of Integer Sequences, <https://oeis.org> (founded in 1964 by N. J. A. Sloane).
- [155] https://en.wikipedia.org/wiki/Ramanujan_tau_function (viewed in August 2018).
- [156] mathworld.wolfram.com/RamanujanThetaFunctions.html (WolframMathWorld, created by E. Weisstein).

Martin Klazar
Department of Applied Mathematics

Charles University, Faculty of Mathematics and Physics
Malostranské náměstí 25
11800 Praha
Czechia
`klazar@kam.mff.cuni.cz`