Sara KROPF

Asymptotic Analysis of Binary Digit Expansions

MASTER THESIS

written to obtain the academic degree of a Master of Science (MSc)

Master's degree Mathematical Computer Science

Graz University of Technology

Supervisor:
Ao.Univ.-Prof. Dipl.-Ing. Dr.techn. Clemens HEUBERGER

Institut für Optimierung und Diskrete Mathematik (Math B)

Graz, Oktober 2012

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

....

date (signature)

Contents

List of Algorithms

List of Figures

List of Tables

1 Introduction

In many contexts we want to compute a^n in a multiplicative group G for an integer n. For example in public-key cryptography this is a very common operation. One possibility to do this efficient uses the binary expansion of the integer $n = (\varepsilon_L \dots \varepsilon_0)$. It is called *Square-and-Multiply* method. There we compute

$$a^n = \Big(\dots \big((a^{\varepsilon_L})^2 a^{\varepsilon_{L-1}} \big)^2 \dots \Big)^2 a^{\varepsilon_0}.$$

If $\varepsilon_i = 0$ the multiplication by a^{ε_i} is a multiplication by 1 and does not cost anything. So there is a nontrivial multiplication for every one in the binary expansion of n. The number of nonzero digits in an expansion is the Hamming weight of the expansion. Additionally there are L squaring operations.

If we use another digit set $D \subseteq \mathbb{Z}^+$ and radix 2 to represent the integer n, then we have to precompute the values a^ε for $\varepsilon \in D$. Again we have a multiplication for every nonzero digit of the representation. If the representation of an integer is not unique, we can choose a representation with a small Hamming weight to make the computation of a^n more efficiently.

A similar task is to compute nP in an additive group. Here the method is called *Double-and-Add*. Then we have

$$nP = 2\Big(\dots \big(2 \left(2\varepsilon_L P + \varepsilon_{L-1} P \right) + \varepsilon_{L-2} P \big) \dots \Big) + \varepsilon_0 P.$$

Here, every nonzero digit of the representation corresponds to a nontrivial addition and every digit of the expansion leads to a multiplication by 2.

A negative digit ε in the representation corresponds to a division by $a^{|\varepsilon|}$ or a subtraction of $|\varepsilon|P$. If inversion in the group is time consuming, then the use of negative digits yields a long precomputation. However there are examples where inverting an element does not cost a lot of time. One of these examples is the additive group on an elliptic curve, which is also used in public-key cryptography.

Another problem is the computation of a linear combination of group elements like $mP + nQ$ for an additive group or $a^m b^n$ for a multiplicative group. A first approach is to compute mP and nQ separately and add the results. However, there is a more efficient way of computation. We compute a joint integer representation

$$\begin{pmatrix} x_L \dots x_0 \\ y_L \dots y_0 \end{pmatrix}$$

of $(m, n)^T$. Then we can compute

$$\begin{aligned} mP + nQ = 2\Big(\dots \Big(2\big(2(x_L P + y_L Q) + (x_{L-1} P + y_{L-1} Q) \big) + (x_{L-2} P + y_{L-2} Q) \Big) \dots \Big) \\ + (x_0 P + y_0 Q). \end{aligned}$$

Again the Hamming weight, the number of nonzero columns in the representation, is the number of nontrivial additions. Here we have to precompute the values $xP + yQ$ for all x, $y \in D$.

The same computation is done for the linear combination $m_1P_1 + \ldots + m_dP_d$. Here we need the joint integer representation of $(m_1, \ldots, m_d)^T$ with minimal Hamming weight.

Therefore we are interested in the expected value of the minimal Hamming weight when we use a given digit set. In this thesis we present the expected value, the variance and the asymptotic distribution for different digit sets and dimensions.

Section 2 gives definitions and known properties of different representations with minimal Hamming weight. First in Section 2.1 there are definitions of integer representations and joint integer representations. Furthermore the Hamming weight of a representation is defined. Section 2.2 contains properties of the nonadjacent form which is a representation with digit set $\{0, \pm 1\}$. The name of this representation is due to the fact that there are no two adjacent nonzero digits. In Section 2.3 the nonadjacent form is generalized to the width-w nonadjacent form which uses a larger digit set. In Sections 2.4 and 2.5 there are other generalizations of the nonadjacent form. They are called joint sparse form and simple joint sparse form and they are representations of two dimensional vectors of integers. In all previous sections we always considered special digits sets. In Sections 2.6 and 2.7 rather general digit sets are examined, first for the representation of a single integer and later for the representation of an integer vector. All these types of representations and their properties are well known.

In Sections 3, 4 and 5 new results for the distribution of the Hamming weight are presented. The width-w nonadjacent form in Section 3 is asymptotical normally distributed and the expected value and the variance are stated. Section 4 contains the asymptotic distribution for representations in dimension one and a rather general digit set. Again the mean and the variance are computed and the asymptotically normal distribution of the Hamming weight is proved. In Section 5 rather general digit sets in arbitrary dimension is examined. There we give a procedure to compute the expected value and the variance of the Hamming weight for a fixed but rather general digit set. In each section we describe a transducer to compute the Hamming weight of a given integer or integer vector.

The work on this thesis has been partially supported by the Austrian Science Fund (FWF): S9606, that is part of the Austrian National Research Network "Analytic Combinatorics and Probabilistic Number Theory".

2 Definitions and known results

In this section we first define integer representations and the Hamming weight of a representation. Then we present different kinds of representations over different digit sets which have minimal Hamming weight among all representations over the same digit set. We also prove the existence and uniqueness of these representations. All of this is well known.

In Section 2.2 we will start with a small digit set and dimension one. Then we will generalize to larger symmetric digit sets in Section 2.3. The next step in Sections 2.4 and 2.5 again uses a small digit set but in dimension two. Then we will introduce a representation over an asymmetric digit set, first in dimension one in Section 2.6, and in Section 2.7 in arbitrary dimension.

2.1 Integer representation and weight of representations

Now we want to introduce integer representations and give some facts about them.

Definition 2.1. *For a radix r and a digit set $D \subseteq \mathbb{Z}$ a* representation *of an integer n is a word $(\varepsilon_k \dots \varepsilon_0) \in D^*$ with $n = \sum_{i=0}^{k} \varepsilon_i r^i$.*

Thereby D^* is the set of all finite words over the alphabet D. Depending on the digit set D the representation of an integer may be impossible, unique or ambiguous. If the representation of any positive integer is possible, then the digit set D has to contain at least a complete system of residues modulo r. If there exist integers which have more than one representation, then the digit set contains one residue modulo r at least twice. However, there exist digit sets which contain more than a complete system of residues modulo r, but do not allow every integer to be represented. For example with the digit set $\{0, 3, 4\}$ and the radix 2, there is no representation of the integer 1.

We will restrict ourselves to the ambiguous case, since otherwise the representation is unique or impossible and we want to investigate which representation has some minimality property. Since the representation of an integer is not unique, we can require the representation to satisfy some additional properties. This leads to the definition of special types of representations in the following sections.

Furthermore we demand $0 \in D$ to be a digit. Thus we can extend every representation to any length k, by prepending 0 in front of the representation without changing the integer represented. If $(\varepsilon_k \dots \varepsilon_0)$ is a representation of n, then $(0 \dots 0\varepsilon_k \dots \varepsilon_0)$ is one too. We will mainly investigate the case in which the radix $r = 2$.

Definition 2.2. *The radix-r representation with digit set $\{0, 1, \dots, r-1\}$ is called* standard radix-r expansion*. If $r = 2$ it is called binary expansion.*

The standard radix-r expansion exists and is unique for every positive integer.

If we want to represent more than one integer at the same time, we use a joint representation.

Definition 2.3 (Joint representation)**.** *For a radix r, a dimension d and a digit set $D \subseteq \mathbb{Z}$ the dimension-d radix-r* joint representation *of a vector $N \in \mathbb{Z}^d$ is a word $(\varepsilon_k \dots \varepsilon_0)$ with $\varepsilon_i \in D^d$ and $N = \sum_{i=0}^{k} \varepsilon_i r^i$.*

We are mainly interested in the number of nonzero digits of a representation of an integer or vector.

Definition 2.4 (Hamming weight)**.** *The* Hamming weight *$h(\varepsilon_k \dots \varepsilon_0)$ of a representation $(\varepsilon_k \dots \varepsilon_0)$ is the number of $i = 0, \dots, k$ with $\varepsilon_i \neq 0$. If $\varepsilon_i \neq 0$ is a vector, at least one coordinate of ε_i is nonzero.*

The weight of an integer depends on the representation we use. For example (11) and (3) are two different radix-2 representations of 3, but the weight is not the same: $h(11) = 2$ and $h(3) = 1$. If it is clear in the context which representation is meant, we shortly write $h(n)$ for the weight of this representation.

2.2 Nonadjacent form

In this section we introduce a representation, called nonadjacent form, which is a unique integer representation over the digit set $\{0, \pm 1\}$. The uniqueness comes from an additional property.

Definition 2.5 (NAF)**.** *The* nonadjacent form *(short NAF) of an integer n is a radix-2 representation $(\varepsilon_k \dots \varepsilon_0)$ over the digit set $\{0, \pm 1\}$ with the following property:*

$$\text{If } \varepsilon_i \neq 0, \text{ then } \varepsilon_{i+1} = 0.$$

This type of integer representation was first introduced by Reitwiesner in [16]. There he proved the existence, uniqueness and the minimality of the NAF.

Theorem 1. *The NAF representation of an integer n exists and is unique.*

Proof (cf. [16]). To prove uniqueness we assume that there are two different NAF representations $(\varepsilon_L \dots \varepsilon_0)$ and $(\delta_L \dots \delta_0)$ of an integer n. Both have the same length, otherwise we prepend zeros in front of the shorter representation. Without loss of generality we may assume that $\varepsilon_0 \neq \delta_0$. Then we have

$$\varepsilon_0 - \delta_0 = \pm(1 + b)$$

for $b \in \{0, 1\}$. If $b = 1$, then ε_0 and δ_0 are both nonzero. Therefore $\varepsilon_1 = \delta_1 = 0$ because of the NAF condition. Then we have

$$(2\varepsilon_1 + \varepsilon_0) - (2\delta_1 + \delta_0) = \pm 2 = \pm 2^b.$$

If $b = 0$ we have

$$\varepsilon_0 - \delta_0 = \pm 1 = \pm 2^b.$$

Therefore we have

$$0 = \sum_{i=0}^{L} \varepsilon_i 2^i - \sum_{i=0}^{L} \delta_i 2^i = \sum_{i=b+1}^{L} \varepsilon_i 2^i - \sum_{i=b+1}^{L} \delta_i 2^i \pm 2^b$$
$$\Rightarrow \pm 2^b = \sum_{i=b+1}^{L} \varepsilon_i 2^i - \sum_{i=b+1}^{L} \delta_i 2^i,$$

but the difference of these sums has to be a multiple of 2^{b+1}. Therefore our assumption is false and if the NAF representation exists, then it is unique.

To prove existence of such a representation we describe an algorithm to construct it. If $n = 0$, the algorithm terminates. If $n \equiv 0 \mod 2$, we choose $\varepsilon_0 = 0$. If n is odd, we choose $\varepsilon_0 \in \{\pm 1\}$ such that $n \equiv \varepsilon_0 \mod 4$. Then we construct a representation $(\varepsilon_L \dots \varepsilon_1)$ of $\frac{n-\varepsilon_0}{2}$. The representation $(\varepsilon_L \dots \varepsilon_0)$ is the NAF representation of n. First we prove the condition of a NAF representation. If $\varepsilon_0 \neq 0$, then $n - \varepsilon_0 \equiv 0 \mod 4$ and therefore $\frac{n-\varepsilon_0}{2}$ is even and $\varepsilon_1 = 0$. Second we have to prove that the algorithm terminates for $n \neq 0$. We observe that

$$\left| \frac{n - \varepsilon_0}{2} \right| < |n|$$

for $|n| > 1$ and it is obviously true for $|n| = 1$. Therefore the absolute value of the input becomes smaller and the algorithm terminates. □

Theorem 2. *The NAF representation of an integer n has minimal Hamming weight among all representations of n over the digit set $\{0, \pm 1\}$.*

Proof (cf. [16]). If we have any representation $(\varepsilon_L \dots \varepsilon_0)$ of n we describe how to construct a NAF representation without increasing the weight of the representation. Therefore we define the property (1) of a representation $(\varepsilon_L \dots \varepsilon_0)$ and an integer $\lambda \geq 0$

$$\varepsilon_i \varepsilon_{i-1} = 0 \text{ for } \lambda \geq i \geq 0. \tag{1}$$

Thereby we assume $\varepsilon_l = 0$ for $l \geq L$ or $l < 0$. If a representation satisfies property (1) for $\lambda = L$, then the representation is a NAF representation. Every representation satisfies the property for $\lambda = 0$.

Let $(\varepsilon_L \dots \varepsilon_0)$ be a representation of n satisfying property (1) for some λ. We construct a representation $(\delta_{L'} \dots \delta_0)$ of n satisfying property (1) for $\lambda + 1$ without increasing the weight.

We have $\varepsilon_{\lambda+1}\varepsilon_\lambda \in \{0, \pm 1\}$. If $\varepsilon_{\lambda+1}\varepsilon_\lambda = 0$ nothing is to be done, since the representation already satisfies the property for $\lambda+1$. If $\varepsilon_{\lambda+1}\varepsilon_\lambda = 1$, let μ be a index with $\lambda+2 \leq \mu \leq L+1$ and

$$\varepsilon_\mu \neq \varepsilon_{\mu-1} = \dots = \varepsilon_{\lambda+2} = \varepsilon_{\lambda+1} = \varepsilon_\lambda = \pm 1.$$

Then we have

$$\sum_{i=\lambda}^{\mu} \varepsilon_i 2^i = (\varepsilon_\mu + \varepsilon_\lambda) 2^\mu - \varepsilon_\lambda 2^\lambda.$$

Therefore we can replace the digits of the representation without changing the integer represented as follows:

$$\begin{aligned}\delta_\mu &= \varepsilon_\mu + \varepsilon_\lambda,\\ \delta_i &= 0 \quad \text{for } \lambda < i < \mu,\\ \delta_\lambda &= -\varepsilon_\lambda,\\ \delta_i &= \varepsilon_i \quad \text{otherwise.}\end{aligned}$$

Thereby the weight of the representation does not increase. The new representation satisfies property (1) for $\lambda + 1$.

If $\varepsilon_{\lambda+1}\varepsilon_\lambda = -1$, then we have $\mp 1 = \varepsilon_{\lambda+1} \neq \varepsilon_\lambda = \pm 1$ and

$$\varepsilon_{\lambda+1}2^{\lambda+1} + \varepsilon_\lambda 2^\lambda = 0 \cdot 2^{\lambda+1} - \varepsilon_\lambda 2^\lambda.$$

Therefore we can replace the digits of the representation without changing the represented integer:

$$\begin{aligned}\delta_{\lambda+1} &= 0,\\ \delta_\lambda &= -\varepsilon_\lambda,\\ \delta_i &= \varepsilon_i \quad \text{otherwise.}\end{aligned}$$

Thereby the weight of the representation does not increase and the new representation satisfies property (1) for $\lambda + 1$. □

Of course there may be other representations of an integer, which have minimal weight. For example $(10\bar{1})$ and (11) are both representations of 3 with radix 2 and have both minimal weight $h(10\bar{1}) = h(11) = 2$. Note that we write $\bar{1}$ to denote -1 for readability.

Since we are interested in the average number of nonzero digits of a NAF representation, the next theorem proved by Thuswaldner in [19] gives the expected value of the number of nonzero digits.

Theorem 3. *The average number of nonzero digits in a NAF representation is*

$$\frac{1}{N}\sum_{n<N} h(n) = \frac{1}{3}\log_2 N + \phi(\log_4 N) + \mathcal{O}(N^{1-\varepsilon})$$

for $\varepsilon > 0$ small enough. The function $\phi(x)$ has period 1.

In [19], the Fourier coefficients of $\phi(x)$ also are determined.

2.3 Width-w nonadjacent form

In this section we introduce an integer representation similar to the NAF, but with a larger digit set. It is also unique and has minimal Hamming weight.

Definition 2.6 (w-NAF)**.** *The* width-w nonadjacent form *(short w-NAF) of an integer n is a radix-2 representation $(\varepsilon_k \dots \varepsilon_0)$ with the digit set $D_w := \{0, \pm 1, \pm 3, \dots, \pm(2^{w-1} - 3), \pm(2^{w-1} - 1)\}$ and the following property:*

$$\text{If } \varepsilon_i \neq 0, \text{ then } \varepsilon_{i+1} = \dots = \varepsilon_{i+w-1} = 0.$$

This integer representation was introduced in [6] and [17], but without any proofs. Muir and Stinson give proofs of existence, uniqueness and minimality of the w-NAF in [14]. We follow this paper in the first part of this section.

Theorem 4. *The w-NAF representation of an integer exists and is unique up to leading zeros.*

Proof (cf. [14]). To prove the uniqueness, we assume that an integer n has two w-NAF representations $(\varepsilon_L \dots \varepsilon_0)$ and $(\delta_{L'} \dots \delta_0)$. We can assume that $L = L'$, otherwise there are leading zeros in one representation. Furthermore we assume that L is minimal. Therefore $\varepsilon_0 \neq \delta_0$. Thus n is odd, since otherwise $\varepsilon_0 = \delta_0 = 0$. Since the representations are w-NAFs, we have $\varepsilon_1 = \dots = \varepsilon_{w-1} = 0 = \delta_0 = \dots = \delta_{w-1}$ and therefore $n \equiv \varepsilon_0 \equiv \delta_0 \mod 2^w$. Since every residue class modulo 2^w has exactly one representative in the digit set, we have $\varepsilon_0 = \delta_0$. This contradicts our assumption, therefore the w-NAF of an integer is unique.

Next we prove the existence of the w-NAF of an integer n. Therefore we describe an algorithm which computes the w-NAF of the input n. This is Algorithm 1. It uses the following two functions

$$f_w(n) = \begin{cases} \frac{n}{2} & \text{if } n \text{ is even,} \\ \frac{n-r}{2^w} & \text{else, with } r \in D_w \text{ and } n \equiv r \bmod 2^w, \end{cases}$$

$$g_w(n) = \begin{cases} 0 & \text{if } n \text{ is even,} \\ 0^{w-1}r & \text{else, with } r \in D_w \text{ and } n \equiv r \bmod 2^w. \end{cases}$$

Algorithm 1 Algorithm to compute the w-NAF of an integer

Input: An integer n
Output: The w-NAF representation α of n
  Set $\alpha = ()$ the empty string
  while $n \neq 0$ **do**
    $\alpha = (g_w(n)\alpha)$ the concatenation of these strings
    $n = f_w(n)$
  end while
  return α

Algorithm 1 terminates because $|f_w(n)| < |n|$: If n is even, then $|f_w(n)| = |\frac{n}{2}| < |n|$. If n is odd, then we have

$$|f_w(n)| = \left|\frac{n-r}{2^w}\right| \leq \left|\frac{n}{2^w}\right| + \left|\frac{r}{2^w}\right| < \left|\frac{n}{2^w}\right| + \left|\frac{1}{2}\right| \leq 2\left|\frac{n}{2^w}\right| < |n|.$$

In the next step we prove that the output of the algorithm is a w-NAF representation of the input n. It is clear that α is a w-NAF, because of the definition of $g_w(n)$. We only have to prove that α is a representation of n. Let $f_w^i(n) = f_w \circ \cdots \circ f_w(n)$ be the i-th application of f_w. Since the algorithm terminates, there exists an $i \geq 0$ with $f_w^i(n) = 0$. We use induction on i to prove the correctness.

If $i = 0$, then $f_w^i(n) = 0$ implies $n = 0$. The output of Algorithm 1 is the empty string and the w-NAF representation of $n = 0$ is (0). Since leading zeros do not matter, the output is a w-NAF representation of n. If $i > 0$, let $n' = f_w(n)$ and α' be the output of the algorithm on the input n'. Then we have $f_w^{i-1}(n') = f_w^i(n) = 0$ and by the induction hypothesis α' is a w-NAF representation of n'. Since the result of the input n' is written in front of the first digits $g_w(n)$ of n, the string α is the concatenation of α' and $g_w(n)$. Let r be the integer represented by the string $g_w(n)$ and l be the length of the string $g_w(n)$. Then α represents the integer $2^l n' + r = n$, since $n' = f_w(n) = \frac{n-r}{2^l}$. Therefore Algorithm 1 is correct and the existence of a w-NAF representation of n follows. □

In [14], Muir and Stinson prove the minimality of the w-NAF. To do this, they construct an algorithm to add a small integer to a representation over the digit set D_w. We first describe this algorithm and prove its correctness. The pseudo code is written in Algorithm 2. Then we prove an inequality for the change of the weight when performing the algorithm. At last we can give a proof for the minimality of the w-NAF.

Let $n := (\varepsilon_L \dots \varepsilon_0) \in D_w^*$ and $\gamma_0 \in \mathbb{Z}$ with $|\gamma_0| < 2^{w-1}$. We want to find a representation $m := (\delta_{L'} \dots \delta_0) \in D_w^*$ with $m = n + \gamma_0$. Furthermore $h(\delta_{L'} \dots \delta_0) \leq h(\varepsilon_L \dots \varepsilon_0) + 1$ should hold. Therefore we present Algorithm 2. The idea of the algorithm is the following:

We define a sequence $\gamma_1, \gamma_2, \dots$ corresponding to the carry of the addition. Let $\varepsilon_l = 0$ for $l > L$. We read ε_i and γ_i and write values for δ_i and γ_{i+1}. We assign the following values to the digits δ_i for $i \geq 0$:

$\varepsilon_i \bmod 2$	$\gamma_i \bmod 2$	δ_i
0	0	ε_i
0	1	γ_i
1	0	ε_i
1	1	0

The other assignment is $\gamma_{i+1} = \dfrac{\varepsilon_i + \gamma_i - \delta_i}{2}$. Then we have the following addition with the carries γ_i.

$$\begin{array}{cccccc} & \dots & \varepsilon_3 & \varepsilon_2 & \varepsilon_1 & \varepsilon_0 \\ + & \dots & \gamma_3 & \gamma_2 & \gamma_1 & \gamma_0 \\ \hline & \dots & \delta_3 & \delta_2 & \delta_1 & \delta_0 \end{array}$$

To prove that the representation of m has digits in D_w, we first examine the carries γ_i. We have $|\varepsilon_i| < 2^{w-1}$ and therefore by induction

$$|\gamma_{i+1}| = \left| \frac{\varepsilon_i + \gamma_i - \delta_i}{2} \right| < 2^{w-1}$$

for every $\delta_i \in \{0, \varepsilon_i, \gamma_i\}$. If $\delta_i = \varepsilon_i$ or $\delta_i = 0$, we clearly have $\delta_i \in D_w$. If $\delta_i = \gamma_i$, then γ_i is odd and therefore $\gamma_i \in D_w$. Therefore the representation of m has only digits in D_w.

This idea is summarized in Algorithm 2. Instead of computing each digit of m separately, we just assign m to be equal to n at the beginning and change digits of this first representation if necessary. Notice that the algorithm does not use any modulo 2^{w-1} operations, in fact only modulo 2 operations, although the digit set is a set of representatives modulo 2^{w-1}.

Algorithm 2 Algorithm to add a small number to a representation over D_w

Input: A representation $n = (\varepsilon_L \dots \varepsilon_0) \in D_w^*$ and an integer γ_0 with $|\gamma_0| < 2^{w-1}$
Output: A representation $m = (\delta_{L'} \dots \delta_0) \in D_w^*$ with $m = n + \gamma_0$

```
δ_i = ε_i for all i
i = 0
while γ_i ≠ 0 do
    ε = ε_i mod 2, γ = γ_i mod 2
    if ε = 0 and γ = 1 then
        δ_i = γ_i
    else if ε = 1 and γ = 1 then
        δ_i = 0
    end if
    γ_{i+1} = (ε_i + γ_i − δ_i)/2
    i = i + 1
end while
```

The next lemma proves the correctness of Algorithm 2.

Lemma 1. *For the output $m = (\delta_{L'} \dots \delta_0)$ of Algorithm 2 and input $n = (\varepsilon_L \dots \varepsilon_0) \in D_w^*$ and γ_0 with $|\gamma_0| < 2^{w-1}$, the relation $m = n + \gamma_0$ is satisfied.*

Proof (cf. [14]). Let $i^* \geq 0$ be the value of i when Algorithm 2 stops. We will prove the lemma by induction on i^*. For $i^* = 0$ we have $\gamma_0 = 0$ and $m = n = n + \gamma_0$.

If $i^* > 0$, let $n' = (\varepsilon_L \dots \varepsilon_1)$ and $m' = (\delta_{L'} \dots \delta_1)$. Then Algorithm 2 with input n' and γ_1 produces the output m'. Furthermore the value of i, when the algorithm stops, is $i^* - 1$. By induction hypothesis we have $m' = n' + \gamma_1$. It follows from $\gamma_1 = \frac{\varepsilon_0 + \gamma_0 - \delta_0}{2}$ that

$$\begin{aligned} 2m' &= 2n' + 2\gamma_1 \\ \Leftrightarrow (\delta_{L'} \dots \delta_1 0) &= (\varepsilon_L \dots \varepsilon_1 0) + 2\gamma_1 \\ \Leftrightarrow (\delta_{L'} \dots \delta_1 \delta_0) &= (\varepsilon_L \dots \varepsilon_1 0) + 2\gamma_1 + \delta_0 \\ \Leftrightarrow (\delta_{L'} \dots \delta_0) &= (\varepsilon_L \dots \varepsilon_1 \varepsilon_0) + \gamma_0 \\ \Leftrightarrow m &= n + \gamma_0, \end{aligned}$$

which proves the lemma. □

For using this algorithm in the proof of minimality of the w-NAF, we need another property of the output of the algorithm. This property is proved in the next lemma.

Lemma 2. *Let $(\delta_{L'} \dots \delta_0)$ be the output of Algorithm 2 with input $(\varepsilon_L \dots \varepsilon_0)$ and γ_0. Then we have $h(\delta_{L'} \dots \delta_0) \leq h(\varepsilon_L \dots \varepsilon_0) + 1$*

Proof (cf. [14]). Let i^* be the value of i when Algorithm 2 terminates. We show that the sequence $t_0, \dots, t_{i^*}$ with

$$t_i = h(\varepsilon_L \dots \varepsilon_i) + h(\gamma_i) + h(\delta_{i-1} \dots \delta_0)$$

is monotonically decreasing. If we have proved this, then we observe that $t_0 = h(\varepsilon_L \dots \varepsilon_0) + h(\gamma_0)$ and $t_{i^*} = h(\delta_{L'} \dots \delta_0)$ since $\gamma_{i^*} = 0$ and for all $i \geq i^*$ we have $\delta_i = \varepsilon_i$. Then from $t_{i^*} \leq t_0$ we obtain

$$h(\delta_{L'} \dots \delta_0) \leq h(\varepsilon_L \dots \varepsilon_0) + h(\gamma_0) \leq h(\varepsilon_L \dots \varepsilon_0) + 1,$$

which proves the lemma.

Now we prove the monotonicity of the sequence $(t_i)_{i \in \mathbb{N}}$. Since $\gamma_{i+1} = \frac{\varepsilon_i + \gamma_i - \delta_i}{2}$, we have

$$h(\gamma_{i+1}) \leq h(\varepsilon_i) + h(\gamma_i) - h(\delta_i).$$

This can be proved by a case analysis for $\delta_i \in \{0, \varepsilon_i, \gamma_i\}$. For example, if $\delta_i = \varepsilon_i$, then $\gamma_{i+1} = \frac{\gamma_i}{2}$. Then the weight of γ_{i+1} and γ_i is the same.

If we compare

$$t_i = h(\varepsilon_L \dots \varepsilon_i) + h(\gamma_i) + h(\delta_{i-1} \dots \delta_0)$$

and

$$t_{i+1} = h(\varepsilon_L \dots \varepsilon_{i+1}) + h(\gamma_{i+1}) + h(\delta_i \dots \delta_0),$$

and cancel every possible digit, we get

$$t_i - t_{i+1} = h(\varepsilon_i) + h(\gamma_i) - h(\gamma_{i+1}) - h(\delta_i) \geq 0.$$

Therefore the sequence is monotonically decreasing. □

Now we are able to prove the following theorem:

Theorem 5. *The w-NAF representation $(\varepsilon_L \dots \varepsilon_0)$ of an integer n has minimal weight among all representations $(\delta_{L'} \dots \delta_0)$ of n with the digit set D_w.*

Proof (cf. [14]). Assume the contrary. Then there is a w-NAF representation $(\varepsilon_L \dots \varepsilon_0)$ of an integer n and a different representation $(\delta_{L'} \dots \delta_0)$ of n with $h(\delta_{L'} \dots \delta_0) < h(\varepsilon_L \dots \varepsilon_0)$. We choose n so that the length of the w-NAF representation is minimal. Hence any w-NAF representation with shorter length has minimal weight.

Because the length of the counterexample n is minimal, we have $\varepsilon_0 \neq \delta_0$. Therefore n is odd and $\varepsilon_0 \neq 0$ and $\delta_0 \neq 0$. Therefore the next $w - 1$ digits of the w-NAF representation have to be zero. Let $n' = (\varepsilon_L \dots \varepsilon_w)$ be the remaining digits and $m' = (\delta_{L'} \dots \delta_w)$.

The representation $(\varepsilon_L \dots \varepsilon_w)$ is a w-NAF representation of n'. Moreover $h(\varepsilon_L \dots \varepsilon_0) = h(\varepsilon_L \dots \varepsilon_w) + 1$.

Next we show that at least two digits of $(\delta_{w-1} \dots \delta_0)$ have to be nonzero. Suppose this is not true. Since $\delta_0 \neq 0$, all other digits have to be zero. Therefore

$$n \equiv \varepsilon_0 \equiv \delta_0 \mod 2^{w-1}$$

and $\varepsilon_0 = \delta_0$. This is a contradiction to the minimal length of the counterexample. Thus $h(\delta_{w-1} \dots \delta_0) \geq 2$ and $h(\delta_{L'} \dots \delta_0) \geq h(\delta_{L'} \dots \delta_w) + 2$.

Now we have

$$n' = m' + \frac{(\delta_{w-1} \dots \delta_0) - (0 \dots 0\varepsilon_0)}{2^w}.$$

We define $\gamma_0 := \frac{(\delta_{w-1} \dots \delta_0) - (0 \dots 0\varepsilon_0)}{2^w}$, which is an integer. We will show that m' and γ_0 are a valid input to Algorithm 2. Therefore we must prove $|\gamma_0| < 2^{w-1}$.

For every digit in D_w we have the upper bound $2^{w-1} - 1$. Therefore

$$|(\delta_{w-1} \dots \delta_0)| \leq \sum_{l=0}^{w-1} (2^{w-1} - 1)2^l = (2^{w-1} - 1)(2^w - 1)$$

and

$$|(0 \dots 0\varepsilon_0)| \leq 2^{w-1} - 1.$$

This gives

$$|\gamma_0| = \frac{1}{2^w} |(\delta_{w-1} \dots \delta_0) - (0 \dots 0\varepsilon_0)| \leq 2^{w-1} - 1.$$

Let $(\delta'_{L''} \dots \delta'_w)$ be the output of Algorithm 2 when the input is $(\delta_{L'} \dots \delta_w)$ and γ_0. Then we have $(\delta'_{L''} \dots \delta'_w) = (\delta_{L'} \dots \delta_w) + \gamma_0 = (\varepsilon_L \dots \varepsilon_w)$ with $h(\delta'_{L''} \dots \delta'_w) \leq h(\delta_{L'} \dots \delta_w) + 1$.

If we summarize the above inequalities, we get

$$\begin{aligned} h(\varepsilon_L \dots \varepsilon_0) &> h(\delta_{L'} \dots \delta_0) \\ h(\varepsilon_L \dots \varepsilon_w) + 1 &> h(\delta_{L'} \dots \delta_w) + 2 \\ h(\varepsilon_L \dots \varepsilon_w) &> h(\delta'_{L''} \dots \delta'_w). \end{aligned}$$

But since $(\varepsilon_L \dots \varepsilon_w)$ is a w-NAF representation of n' with shorter length than the counterexample and $(\delta'_{L''} \dots \delta'_w)$ is a representation of n', we have a contradiction. Our assumption must be wrong and the w-NAF representation has minimal weight. □

A different proof of the minimality of the w-NAF representation is given by Avanzi in [2].

For the w-NAF representation we give the following theorem for the distribution. It is proved by Cohen in [5].

Theorem 6. *The average weight of the w-NAF representation of length N of an integer is*

$$\frac{N}{w+1} - \frac{(w-1)(w+2)}{2(w+1)^2} + \mathcal{O}((1+\varepsilon)^{-N})$$

where $\varepsilon > 0$ is small enough.

2.4 Joint sparse form

Now we want to generalize to higher dimensions. In [18], Solinas introduced a joint integer representation for $d = 2$, which has minimal Hamming weight.

Definition 2.7 (Joint sparse form)**.** *Let x, $y \in \mathbb{Z}$. Then the joint integer representation $\begin{pmatrix} x_L \dots x_0 \\ y_L \dots y_0 \end{pmatrix}$ of $\begin{pmatrix} x \\ y \end{pmatrix}$ over the digit set $\{0, \pm 1\}$ is called* joint sparse form*, if the following conditions are satisfied:*

1. *Of any three consecutive positions, at least one is a double zero.*
2. *Adjacent terms do not have opposite signs, in other words $x_{j+1}x_j \neq -1$ and $y_{j+1}y_j \neq -1$.*
3. *If $x_{j+1}x_j \neq 0$, then $y_{j+1} = \pm 1$ and $y_j = 0$.*
4. *If $y_{j+1}y_j \neq 0$, then $x_{j+1} = \pm 1$ and $x_j = 0$.*

Solinas also proved the following theorem about uniqueness and existence of the joint sparse form in [18]:

Theorem 7. *Every pair of integers $(x, y)^T$ has a unique joint sparse form.*

Proof (cf. [18]). To prove uniqueness we assume that for $(x, y)^T$ there are two different joint sparse forms

$$\begin{pmatrix} x_L \dots x_0 \\ y_L \dots y_0 \end{pmatrix} \text{ and } \begin{pmatrix} x'_L \dots x'_0 \\ y'_L \dots y'_0 \end{pmatrix}.$$

Without loss of generality we can assume that $\min\{|x|, |y|\}$ is minimal. Therefore $(x_0, y_0)^T \neq (x'_0, y'_0)^T$. Furthermore we assume that $x_0 \neq x'_0$. Then both have to be nonzero and we can assume $x_0 = 1$, $x'_0 = -1$.

If $x \equiv 1 \mod 4$, we know that $x_1 = 0$ and x'_1 is odd. Because of Condition 2 we have $x'_1 = -1$. By Condition 3 we get $y'_0 = 0$ and $y'_1 = \pm 1$ and hence $y \equiv 2 \mod 4$. Therefore we have $y_0 = 0$ and $y_1 = \pm 1$. So up to now we have the following two beginnings of the representations:

$$\begin{pmatrix} \dots & 0 & 1 \\ \dots & \pm 1 & 0 \end{pmatrix}, \quad \begin{pmatrix} \dots & \bar{1} & \bar{1} \\ \dots & \pm 1 & 0 \end{pmatrix}.$$

Condition 1 implies that $x_2 = y_2 = x'_2 = y'_2 = 0$. Therefore $x \equiv 1 \mod 8$ due to the left representation and $x \equiv 5 \mod 8$ due to the right representation. This is a contradiction, therefore the assumption must be wrong.

If $x \equiv 3 \mod 4$, by the same reasons as above we know the following two beginnings of the representations

$$\begin{pmatrix} \dots & 0 & 1 & 1 \\ \dots & 0 & \pm 1 & 0 \end{pmatrix}, \quad \begin{pmatrix} \dots & 0 & 0 & \bar{1} \\ \dots & 0 & \pm 1 & 0 \end{pmatrix}.$$

Algorithm 3 Algorithm to compute the joint sparse form of an integer vector

Input: A vector of integers $(x, y)^T$

Output: Joint sparse form of the input vector: $\begin{pmatrix} u_{0,L} \dots u_{0,0} \\ u_{1,L} \dots u_{1,0} \end{pmatrix}$

$k_0 = x$, $k_1 = y$
$j = 0$
$d_0 = d_1 = 0$
while $k_i + d_i > 0$ for $i = 0$ or 1 **do**
    $l_i = d_i + k_i$
    for $i = 0, 1$ **do**
        if l_i is even **then**
            $u = 0$
        else
            $u = -1 + ((l_i + 1) \bmod 4)$
            if $l_i \equiv \pm 3 \mod 8$ **and** $l_{1-i} \equiv 2 \mod 4$ **then**
                $u = -u$
            end if
        end if
        $u_{i,j} = u$
    end for
    for $i = 0, 1$ **do**
        if $2d_i = 1 + u_i$ **then**
            $d_i = 1 - d_i$
        end if
        $k_i = \lfloor \frac{k_i}{2} \rfloor$
    end for
    $j = j + 1$
end while

Therefore $x \equiv 3 \mod 8$ and $x \equiv 7 \mod 8$, which is again a contradiction.

To prove the existence we give Algorithm 3. This algorithm takes two integers x and y as input and gives back their joint sparse form. Note that $n \bmod k$ returns the residue of n modulo k within the set $\{0, 1, \ldots, k-1\}$. The existence of a joint sparse form follows from the correctness of the algorithm.

All operations in the algorithm can be performed if we know the last three bits of the binary expansions of x and y. The modulo operations are trivial and $\lfloor \frac{k_i}{2} \rfloor$ corresponds to cutting off the last digit of the expansion. Therefore we can use these expansions as input to the algorithm instead of the integers themselves.

Algorithm 4 Second algorithm to compute the joint sparse form of an integer vector

Input: A reduced signed joint representation of two integers $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_{L-1} \ldots x_0 \\ y_{L-1} \ldots y_0 \end{pmatrix}$

Output: Joint sparse form of the input vector: $\begin{pmatrix} u_{0,L} \ldots u_{0,0} \\ u_{1,L} \ldots u_{1,0} \end{pmatrix}$

$a_0 = x_0$, $b_0 = x_1$, $c_0 = x_2$, $d_0 = 0$
$a_1 = y_0$, $b_1 = y_1$, $c_1 = y_2$, $d_1 = 0$
for j from 0 to L **do**
    for $i = 0, 1$ **do**
        if $d_i \equiv a_i \mod 2$ **then**
            $u = 0$
        else
            $u = -1 + ((d_i + 2b_i + a_i + 1) \bmod 4)$
            if $d_i + 4c_i + 2b_i + a_i \equiv \pm 3 \mod 8$ **and** $d_{1-i} + 2b_{1-i} + a_{1-i} \equiv 2 \mod 2$ **then**
                $u = -u$
            end if
        end if
        $u_{i,j} = u$
        $R_{i,j} = (d_i, c_i, b_i, a_i)$
    end for
    $S_j = (R_{0,j}, R_{1,j})$
    for $i = 0, 1$ **do**
        $d_i = \frac{d_i + a_i - u_{i,j}}{2}$
        $a_i = b_i$
        $b_i = c_i$
    end for
    $c_0 = x_{j+3}$, $c_1 = y_{j+3}$
end for

To prove the correctness of the algorithm, we have to generalize Algorithm 3 to an algorithm which accepts any reduced signed joint representation as input. Thereby a reduced signed representation is an integer representation with radix 2, digit set $\{0, \pm 1\}$ and any two consecutive digits do not have opposite signs. A reduced signed joint representation

S_j	$R_{0,j}$	$R_{1,j}$
C	$l_{0,j}$ even	$l_{1,j}$ even
D	$l_{0,j} \equiv 0 \mod 4$	$l_{1,j}$ odd
E	$l_{0,j} \equiv 2 \mod 4$	$l_{1,j} \equiv \pm 1 \mod 8$
F	$l_{0,j} \equiv 2 \mod 4$	$l_{1,j} \equiv \pm 3 \mod 8$
G	$l_{0,j}$ odd	$l_{1,j} \equiv 0 \mod 4$
H	$l_{0,j} \equiv \pm 1 \mod 8$	$l_{1,j} \equiv 2 \mod 4$
J	$l_{0,j} \equiv \pm 3 \mod 8$	$l_{1,j} \equiv 2 \mod 4$
K	$l_{0,j}$ odd	$l_{1,j}$ odd

Table 1: A partition of the states S_j into eight sets

consists of two reduced signed representations. This leads to Algorithm 4. For two binary expansions as input both algorithms do the same calculation. The variables $R_{i,j}$ and S_j are not necessary for the algorithm, but we need them for the proof. Digits after the left end of a representation are assumed to be zero.

We examine the variables $R_{i,j}$ and S_i more closely. The variable S_i are called states. The output $(u_{0,j}, u_{1,j})$ is a function of the state S_j. Hence we can describe the algorithm like an automaton: In the j-th iteration of the **for** loop the input is the state S_j and the output is $(u_{0,j}, u_{1,j})$ and we go on to state S_{j+1}.

The next question is which states there are. For the variable $R_{i,j} = (d_i, c_i, b_i, a_i)$ there are $3^4 = 81$ possibilities, but not all of them are allowed because the input is reduced. Only 51 possibilities remain. For each we define

$$k_{i,j} := 4c_i + 2b_i + a_i$$
$$l_{i,j} := d_i + k_i.$$

The values $k_{i,j}$ and $l_{i,j}$ are the same as in Algorithm 3 if the input is an unsigned expansion. With these definitions we group the states S_j into eight different sets as shown in Table 1. From now on we will identify the states with the corresponding sets.

Now we describe which state can follow which. Therefore we check the following conditions for all possible values of $S_j = (R_{0,j}, R_{1,j})$ in Algorithm 4.

- If $l_{i,j} \equiv 0 \mod 4$ and $l_{1-i,j}$ is odd, then $l_{i,j+1}$ is even.
- If $l_{i,j} \equiv 2 \mod 4$ and $l_{1-i,j}$ is odd, then $l_{i,j+1}$ is odd.
- If $l_{i,j}$ is odd and $l_{1-i,j} \not\equiv 2 \mod 4$, then $l_{i,j+1}$ is even.
- If $l_{i,j} \equiv 1 \mod 4$ and $l_{1-i,j} \equiv 2 \mod 4$, then $l_{i,j+1} \equiv 0 \mod 4$.
- If $l_{i,j} \equiv 3 \mod 4$ and $l_{1-i,j} \equiv 2 \mod 4$, then $l_{i,j+1}$ is odd.

S_j	$(u_{0,j}, u_{1,j})$	S_{j+1}
C	$(0,0)$	all
D	$(0,\pm 1)$	C
E	$(0,\pm 1)$	G
F	$(0,\pm 1)$	K
G	$(\pm 1,0)$	C
H	$(\pm 1,0)$	D
J	$(\pm 1,0)$	K
K	$(\pm 1,\pm 1)$	C

Table 2: All possible transitions from S_j to S_{j+1}, together with the output $(u_{0,j}, u_{1,j})$

S_j	consists of	$u_{0,j} = u_{1,j} = 0$?	S_{j+1}
A	E, F, H or J	No	B
B	D, G or K	No	C
C	C	Yes	A, B or C

Table 3: A further grouping of the eight states into three states, together with the successors and the weight of the output

In combination with Table 1 we obtain the transitions between states S_j and S_{j+1} with the output $(u_{0,j}, u_{1,j})$ as listed in Table 2. Except for the state C, the successor of a state S_j is unique. All states can be a successor of the state C.

As a last step we further combine the eight states C to K into three states A, B and C. This is written in Table 3.

Now we are ready to prove that the output of Algorithm 4 is indeed a joint sparse form of the input $(x, y)^T$. It is obvious that the output is a signed joint integer representation of $(x, y)^T$. We now have to prove that it is a joint sparse form.

Condition 1 is satisfied, because for every j at least one of the states S_{j-1}, S_j and S_{j+1} is state C. If $S_{j-1} \neq C$, then it is either A or B. In the first case $S_j = B$ and $S_{j+1} = C$. In the second case $S_j = C$.

If $u_{0,j} \neq 0$ and $u_{0,j+1} \neq 0$, then $S_j = J$ and $S_{j+1} = K$ as we can see in Table 2. If we check all possible cases for $R_{0,j}$, we obtain $u_{0,j} = u_{0,j+1}$. Therefore $u_{0,j}u_{0,j+1} \neq -1$. For the other coordinate we have the same result and hence Condition 2 is satisfied.

To prove Condition 3, we assume that $u_{0,j}u_{0,j+1} \neq 0$. Then $S_j = J$ and $S_{j+1} = K$ and Table 2 shows that $u_{1,j} = 0$ and $u_{1,j+1} = \pm 1$. Condition 4 is proved in the same way. □

Another property of the joint sparse form is proved in [18]:

Theorem 8. *The joint sparse form of $(x, y)^T$ has minimal Hamming weight among all joint integer representations of $(x, y)^T$ over the digit set $\{0, \pm 1\}$.*

Proof (cf. [18]). We will prove that the output of Algorithm 4 has smaller or equal Hamming weight than the input. If we have proved this fact, then we are done: Any signed integer representation is given. If it is reduced, then we take it as input of Algorithm 4 and we get a joint sparse form with less or equal weight. If it is not reduced, then there is a j with, say, $x_j x_{j+1} = -1$. Then we can exchange these two digits $(x_{j+1}x_j)$ with $(0(-x_{j+1}))$. Thereby we get another signed joint representation of $(x, y)^T$ with smaller weight than the first one. After finitely many such steps, we obtain a reduced signed joint representation of $(x, y)^T$ which we use as input of the algorithm. Therefore the joint sparse form has minimal weight among all signed integer representations.

To prove that Algorithm 4 does not increase the weight, we define

$$W_k := \begin{pmatrix} w_{0,L} \dots w_{0,0} \\ w_{1,L} \dots w_{1,0} \end{pmatrix}$$

with

$$w_{i,j} := \begin{cases} u_{i,j} & \text{if } j < k \\ x_j, & \text{if } j \geq k \text{ and } i = 0 \\ y_j, & \text{if } j \geq k \text{ and } i = 1. \end{cases}$$

Then W_0 is the input sequence and W_{L+1} is the output sequence. The j-th step of the algorithm replaces W_j by W_{j+1}.

Now we investigate whether the weight increases or decreases when we replace W_j by W_{j+1}. If one of x_j and y_j is nonzero and $u_{0,j} = u_{1,j} = 0$, then the weight decreases by 1. We say, there is a weight loss at step j. If $u_{0,j}$ or $u_{1,j}$ is nonzero and $x_j = y_j = 0$, then the weight increases by 1. This is a weight gain at step j. In all other cases the weight remains the same.

Next, we have to show that there are not more weight gain steps than weight loss steps. Therefore we use again the states S_j from the proof of Theorem 7. We group the states in another way, but first we have to group the variables $R_{i,j}$ in the following way.

(a) $l_{i,j}$ odd, $d = 0$

(b) $l_{i,j}$ odd, $d = \pm 1$, $b = 0$

(c) $l_{i,j}$ odd, $d = \pm 1$, $b = \pm 1$

(d) $l_{i,j}$ even, $d = 0$

(e) $l_{i,j} \equiv 2 \mod 4$, $d = \pm 1$, $b = 0$

(f) $l_{i,j}$ even, all other cases

The grouping of the states S_j is written in Table 4. There are also the possible successors of a state S_j. These successors can be verified by looking at each of the cases in Algorithm 4.

The state L consists of exactly the states with a weight loss: If, say, $x_j \neq 0$ and $u_{0,j} = u_{1,j} = 0$, then $x_j \equiv d_0 \mod 2$, since $u_{0,j} = 0$, and $l_{0,j} \equiv x_j + d_0 \mod 2$ by

S_j	$R_{0,j}$ and $R_{1,j}$	S_{j+1}
L	both (d), (e) or (f), not both (d)	L, M, N or P
M	both (b), (c) or (d), not both (d)	L or N
N	one (a) and the other (a), (c), (d) or (f), or both (d)	L or N
P	other 12 cases	L, M, N or P

Table 4: A grouping of the states according to the change of weight

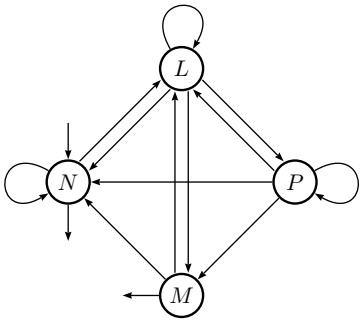

Figure 1: Automaton corresponding to Algorithm 4 with states and transitions listed in Table 4

definition. Then $l_{0,j}$ is even and $d_0 = \pm 1$. These are the cases (e) and (f). In the other coordinate we have $l_{1,j}$ which is even too, but d_1 can be anything. These are the cases (d), (e) and (f).

The state M consists of exactly the states with a weight gain: If, say, $u_{0,j} \neq 1$ and $x_j = y_j = 0$, then $d_0 \not\equiv x_j \mod 2$, since $u_{0,j} \neq 0$, and $l_{0,j} \equiv x_j + d_0 \mod 2$. Then $l_{0,j}$ is odd and $d_0 = \pm 1$. These are the cases (b) and (c). In the second coordinate we have $l_{1,j} \equiv y_j + d_1 \equiv d_1 \mod 2$. These are the cases (b), (c) and (d).

At the beginning of the algorithm, we have $d_0 = d_1 = 0$. These are the cases (a) and (d), therefore N is the initial state . At the end of the algorithm $a_i = b_i = c_i = 0$. These are the cases (b) and (d), which can be verified by looking at each state.

In Figure 1 the states and the transitions are shown. We can see that every time we visit the state M, we first have to visit the state L. Therefore the algorithm does not increase the weight.

□

The joint sparse form is a generalization of the NAF. If we compute the joint sparse form of $(n, 0)^T$, the first row will be the NAF of n. However, if we use two NAF representations and put them together to a joint representation, it will not be a joint representation with minimal weight. For example the NAF representations of $\begin{pmatrix} 11 \\ 2 \end{pmatrix}$ is $\begin{pmatrix} 10\bar{1}0\bar{1} \\ 00010 \end{pmatrix}$. This representation has Hamming weight 4, but the joint sparse form is $\begin{pmatrix} 1011 \\ 0010 \end{pmatrix}$ and has Ham-

ming weight 3.

To compute the distribution of the weight of the joint sparse form, we refer to the next chapter, where we introduce the simple joint sparse form with the same Hamming weight as the joint sparse form. In Theorem 10 we give the mean and the variance of the weight. However, in [18] Solinas also proved that the main term of the density of the Hamming weight is $\frac{1}{2}$.

2.5 Simple joint sparse form

Now we will define another type of integer representation for dimension $d = 2$. It was first introduced by Grabner, Heuberger and Prodinger in [10].

Definition 2.8 (Simple joint sparse form)**.** *The* simple joint sparse form *of a vector* $(x, y)^T \in \mathbb{Z}^2$ *is a dimension-*2 *radix-*2 *joint integer representation* $\begin{pmatrix} x_n \dots x_0 \\ y_n \dots y_0 \end{pmatrix}$ *of* $\begin{pmatrix} x \\ y \end{pmatrix}$ *over the digit set* $\{0, \pm 1\}$ *such that*

1. *If* $|x_i| \neq |y_i|$*, then* $|x_{i+1}| = |y_{i+1}|$.
2. *If* $|x_i| = |y_i| = 1$*, then* $x_{i+1} = y_{i+1} = 0$.

In [10] the following theorem is proved:

Theorem 9. *For any vector* $(x, y)^T \in \mathbb{Z}^2$ *the simple joint sparse form exists and is unique up to leading zeros. Furthermore it has minimal weight among all joint integer expansions of* N *over the digit set* $\{0, \pm 1\}$.

Proof (cf. [10]). For the existence we describe an algorithm to construct the simple joint sparse form. Let $x = (x_L \dots x_0)$ and $y = (y_L \dots y_0)$ with x_i, $y_i \in \{0, \pm 1\}$. If both numbers $(x, y)^T$ are even, then $x_0 = y_0 = 0$. In this case we have a double zero. If both are odd, we choose $(x_0, y_0)^T$ such that both $\frac{x-x_0}{2}$ and $\frac{y-y_0}{2}$ are even. Here the next digit will be a double zero. If x is odd and y is even, or the other way round, then y_0 has to be 0 and we choose x_0 such that $\frac{x-x_0}{2} \equiv \frac{y-y_0}{2} \mod 2$. Then either the next digit is zero if $\frac{y}{2}$ is even, or the digit after the next is a double zero if $\frac{y}{2}$ is odd. Then we go on with $(\frac{x-x_0}{2}, \frac{y-y_0}{2})^T$. Obviously we get a representation of $(x, y)^T$ which is a simple joint sparse form. Observe that of any three consecutive digits at least one is a double zero.

To show uniqueness we assume that $\begin{pmatrix} x_L \dots x_0 \\ y_L \dots y_0 \end{pmatrix}$ and $\begin{pmatrix} x'_L \dots x'_0 \\ y'_L \dots y'_0 \end{pmatrix}$ are two different simple joint sparse forms of $(x, y)^T$. Without loss of generality we can assume that L is minimal. Hence $(x_0, y_0)^T \neq (x'_0, y'_0)^T$. Furthermore we can assume that $x_0 = 1$ and $x'_0 = -1$. Then $x_1 \equiv \frac{x-x_0}{2} \equiv \frac{x-x'_0}{2} - 1 \equiv x'_1 - 1 \not\equiv x'_1 \mod 2$. If y is even, then $y_0 = y'_0 = 0$ and Condition 1 implies $x_1 \equiv y_1 \equiv y'_1 \equiv x'_1 \mod 2$, which is a contradiction. In the other case, if y is odd, then Condition 2 implies $(x_1, y_1)^T = (0, 0)^T$ and $(x'_1, y'_1)^T = (0, 0)^T$, which is again a contradiction. Therefore the simple joint sparse form is unique.

To prove minimality we describe how to convert a simple joint sparse form into a joint sparse form. Thereby the position of the double zeros does not change and as a consequence the weight remains the same. By Theorem 8 we know that the joint sparse form is minimal and therefore the simple joint sparse form is minimal too. The simple joint sparse form satisfies the Conditions 1, 3 and 4 of the joint sparse form. But it need not satisfy Condition 2. If $x_{i+1}x_i = -1$ then in a simple joint sparse form we have $|y_{i+1}| = 1$ and $y_i = 0$. So changing $x_{i+1}x_i$ into $0x_{i+1}$ does not change the weight of the representation and the number represented also stays the same as well. By doing a finite number of these changes, we can convert a simple joint sparse form into a joint sparse form. □

The joint sparse form and the simple joint sparse form are not the same in general, but they have all double zeros at the same positions. Therefore both have the same length and the same Hamming weight. For example the joint sparse form of $\binom{7}{3}$ is

$$\binom{100\bar{1}}{010\bar{1}},$$

but the simple joint sparse form is

$$\binom{100\bar{1}}{1\bar{1}0\bar{1}}.$$

For the same reason as above the simple joint sparse form generalizes the NAF.

In [10], Grabner, Heuberger and Prodinger compute the distribution of the Hamming weight of the simple joint sparse form using exponential sums. The probability space on the pairs of integer is a uniform distribution on all pairs (m, n) with m, $n < N$ for a fixed integer N. Then the following result holds where we have a better error term here than in [10].

Theorem 10. *The weight of the simple joint sparse form is asymptotically normally distributed with mean $\frac{1}{2}\log_2 N$ and variance $\frac{1}{16}\log_2 N$, that is*

$$\mathbb{P}\left(\frac{h(m,n) - \frac{1}{2}\log_2 N}{\frac{1}{4}\sqrt{\log_2 N}} < x\right) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{t^2}{2}}\,dt + \mathcal{O}(\log^{-1/4} N),$$

for all $x \in \mathbb{R}$.

Proof (cf. [10]). The transducer in Figure 2 computes the Hamming weight of the simple joint sparse form of the input $(m, n)^T$. The input is the binary expansion of $(m, n)^T$. This input is read from right to left, from the least significant digit to the most significant one. The output of the transducer is a sequence of zeros and ones and the number of ones is the Hamming weight of the simple joint sparse form of the input.

This transducer is a simplification of the transducer in Figure 1 in [10]. Since we are not interested in the digits of the simple joint sparse form, but rather whether the digit

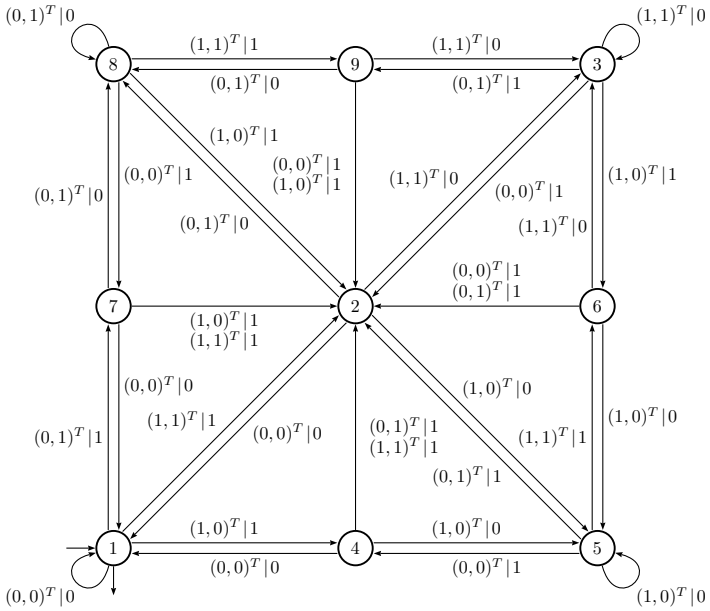

Figure 2: Transducer to compute the Hamming weight of a simple joint sparse form

is $(0,0)^T$ or not, we write a 0 instead of $(0,0)^T$ and a 1 in all other cases. The second simplification comes from the following observation: If we look at state s, it does not matter to which state we go, we always write the same output $\varepsilon \in \{0,1\}$. Therefore we can write this output ε when coming to state s instead of when leaving state s. The resulting transducer is in Figure 2.

We want to calculate $f(m,n) := e^{ith(m,n)}$. Therefore we define the matrices $M_{\delta,\varepsilon}$ for each δ, $\varepsilon \in \{0,1\}$. The (k,l)-th entry of the matrix $M_{\delta,\varepsilon}$ is e^{ith} if we go from state k to state l while reading $(\delta,\varepsilon)^T$ and writing h. If there is no transition from k to l with input label $(\delta,\varepsilon)^T$, then the entry is 0. With this definition we have the matrices (where $z = e^{it}$)

$$M_{0,0} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & z & 0 & 0 & 0 & 0 & 0 \\ 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & z & 0 & 0 \\ 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \qquad M_{0,1} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & z & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & z \\ 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix},$$

$$M_{1,0} = \begin{pmatrix} 0 & 0 & 0 & z & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \qquad M_{1,1} = \begin{pmatrix} 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & z & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & z \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

With these matrices we can rewrite the function $f(m,n)$ in

$$f(m,n) = \vec{v}^T \left(\prod_{l=0}^{L} M_{m_l,n_l} \right) M_{0,0}^2 \vec{v}$$

with $n = \sum_{l=0}^{L} 2^l n_l$, $m = \sum_{l=0}^{L} 2^l m_l$ and $\vec{v}^T = (1,0,0,0,0,0,0,0,0)$. One entry (k,l) of the product of matrices describes the sum of weight of all paths starting in k, ending in l with the binary expansion of $(m,n)^T$ as labels. The factor $M_{0,0}^2$ ensures that we stop at state 1, so that all possible carries are processed. We are interested in the paths starting and ending in state 1, therefore we multiply by $\vec{v}^T$ and $\vec{v}$.

We define $M(m,n) := \prod_{l=0}^{L} M_{m_l,n_l}$. This matrix function is bivariate 2-multiplicative (see, for example, [3]). Next we define the summatory functions

$$E(N) = \sum_{m,n<N} f(m,n)$$

and

$$F(N) = \sum_{m,n<N} M(m,n).$$

Primarily we are interested in $E(N)$, but since $E(N) = \vec{v}^T F(N) M_{0,0}^2 \vec{v}$ we will investigate the function $F(N)$. First we derive a recursion formula for $F(N)$. The following

equations hold for $F(N)$

$$F(2N) = \sum_{\delta,\varepsilon=0}^{1} \sum_{\substack{2m+\delta<2N \\ 2n+\varepsilon<2N}} M(2m+\delta, 2n+\varepsilon) = \sum_{\delta,\varepsilon=0}^{1} M_{\delta,\varepsilon} F(N)$$

and

$$\begin{aligned} F(2N+1) &= \sum_{\delta,\varepsilon=0}^{1} \sum_{\substack{2m+\delta<2N+1 \\ 2n+\varepsilon<2N+1}} M(2m+\delta, 2n+\varepsilon) \\ &= \sum_{\delta,\varepsilon=0}^{1} M_{\delta,\varepsilon} F(N) + \sum_{n<2N} M(2N, n) + \sum_{m<2N} M(m, 2N) + M(2N, 2N). \end{aligned}$$

If we define the matrices

$$\begin{aligned} A &= \sum_{\delta,\varepsilon=0}^{1} M_{\delta,\varepsilon}, \\ B_{1,0} &= M_{0,0} + M_{0,1}, \\ B_{2,0} &= M_{0,0} + M_{1,0} \end{aligned}$$

and the functions

$$\begin{aligned} G_1(N) &= \sum_{n<N} M(N, n), \\ G_2(N) &= \sum_{m<N} M(m, N), \end{aligned}$$

we have

$$\begin{aligned} F(2N) &= AF(N), \\ F(2N+1) &= AF(N) + B_{1,0}G_1(N) + B_{2,0}G_2(N) + M_{0,0}M(N, N). \end{aligned} \tag{2}$$

The functions $G_i(N)$ also satisfy recurrence relations for $i \in \{1, 2\}$, similar to the one of $F(N)$. They are

$$\begin{aligned} G_1(2N) &= \sum_{\varepsilon=0}^{1} \sum_{2n+\varepsilon<2N} M(2N, 2n+\varepsilon) = B_{1,0}G_1(N), \\ G_1(2N+1) &= \sum_{\varepsilon=0}^{1} \sum_{2n+\varepsilon<2N+1} M(2N+1, 2n+\varepsilon) = B_{1,1}G_1(N) + C_1 M(N, N), \end{aligned}$$

and

$$G_2(2N) = \sum_{\varepsilon=0}^{1} \sum_{2n+\varepsilon<2N} M(2n+\varepsilon, 2N) = B_{2,0} G_2(N),$$
$$G_2(2N+1) = \sum_{\varepsilon=0}^{1} \sum_{2n+\varepsilon<2N+1} M(2n+\varepsilon, 2N+1) = B_{2,1} G_2(N) + C_2 M(N,N),$$

with $B_{1,1} = M_{1,0} + M_{1,1}$, $B_{2,1} = M_{0,1} + M_{1,1}$, $C_1 = M_{1,0}$ and $C_2 = M_{0,1}$.

By iterating, we get

$$G_i\left(\sum_{l=0}^{L} \varepsilon_l 2^l\right) = \sum_{l=0}^{L} \varepsilon_l \prod_{j=0}^{l-1} B_{i,\varepsilon_j} C_i \prod_{j=l+1}^{L} M_{\varepsilon_j,\varepsilon_j}.$$

If we insert this last equation into (2) and iterate, we get $F(N) = F_0(N)+F_1(N)+F_2(N)$ with

$$F_0(\sum_{l=0}^{L} \varepsilon_l 2^l) = \sum_{l=0}^{L} \varepsilon_l A^l M_{0,0} \prod_{p=l+1}^{L} M_{\varepsilon_p,\varepsilon_p},$$
$$F_i(\sum_{l=0}^{L} \varepsilon_l 2^l) = \sum_{l=0}^{L} \varepsilon_l A^l B_{i,0} \sum_{j=l+1}^{L} \varepsilon_j \prod_{k=l+1}^{j-1} B_{i,\varepsilon_k} C_i \prod_{k=j+1}^{L} M_{\varepsilon_k,\varepsilon_k} \text{ for } i = 1, 2.$$

Next we want to split these equations into two parts. The main part is contributed by the dominating eigenvalue. The rest will be the error term and is contributed by all other eigenvalues. Therefore we determine the eigenvalues of all occurring matrices. The matrices $M_{\delta,\varepsilon}$ have the characteristic polynomial $-x^8(x-1)$, independent of z. The eigenvalues are therefore 0 and 1. The characteristic polynomial of $B_{i,\varepsilon}$ for $i = 1, 2$ and $\varepsilon \in \{0, 1\}$ is $x^6(x-1)(x^2-x-2z)$. Hence all eigenvalues of $B_{i,\varepsilon}$ have modulus less or equal then 2. The characteristic polynomial of A is

$$x(x-1)(x^2-x-2z)^2(x^3-x^2-8zx-16z^2).$$

At $t = 0$ we have the largest eigenvalue $\mu(0) = 4$, a zero of the last factor. The second largest eigenvalue has modulus $\beta(0) < |\mu(0)|$. Since eigenvalues are continuous, we have $\beta(t) < |\mu(t)|$ around $t = 0$ and $\mu(t)$ is the dominating eigenvalue of A around $t = 0$. The Taylor expansion of $\mu(t)$ around $t = 0$ is

$$\mu(t) = 4 + 2it - \frac{5}{8}t^2 + \mathcal{O}(t^3).$$

Let $J = T^{-1}AT$ be a Jordan decomposition with the eigenvalue $\mu(t)$ in top left entry. Define $\Lambda := T\text{diag}(\mu^{-1}, 0, \ldots, 0)T^{-1}$ and $R := T(J - \text{diag}(\mu, 0, \ldots, 0))T^{-1}$. The matrix R

has a largest eigenvalue with modulus $\beta(t)$. Also define

$$\Lambda_0(x_0, x_1, \ldots) = \sum_{l=0}^{\infty} x_l \Lambda^l M_{0,0} \coprod_{p=0}^{l-1} M_{x_p,x_p},$$
$$\Lambda_i(x_0, x_1, \ldots) = \sum_{l=0}^{\infty} x_l \Lambda^l B_{i,0} \sum_{j=0}^{l-1} x_j \coprod_{k=j+1}^{l-1} B_{i,x_k} C_i \coprod_{k=0}^{j-1} M_{x_k,x_k} \text{ for } i = 1, 2$$

with $\coprod_{l=a}^{b} m_l = m_b \cdot m_{b-1} \cdot \ldots m_a$ and

$$R_0(\varepsilon_L, \varepsilon_{L-1}, \ldots, \varepsilon_0) = \sum_{l=0}^{L} \varepsilon_l R^l M_{0,0} \prod_{p=l+1}^{L} M_{x_p,x_p},$$
$$R_i(\varepsilon_L, \varepsilon_{L-1}, \ldots, \varepsilon_0) = \sum_{l=0}^{L} \varepsilon_l R^l B_{i,0} \sum_{j=l+1}^{L} \varepsilon_j \prod_{k=l+1}^{j-1} B_{i,\varepsilon_k} C_i \prod_{k=j+1}^{L} M_{\varepsilon_k,\varepsilon_k} \text{ for } i = 1, 2.$$

Then we have

$$F_i\left(\sum_{l=0}^{L} \varepsilon_l 2^l\right) = \mu(t)^L \Lambda_i(\varepsilon_L, \varepsilon_{L-1}, \ldots, \varepsilon_0, 0^\omega) + R(\varepsilon_L, \varepsilon_{L-1}, \ldots, \varepsilon_0)$$

for $i = 0, 1, 2$.

First the functions Λ_i are defined on the infinite product space $\{0, 1\}^{\mathbb{N}}$. But we define Λ_i for $i = 0, 1, 2$ to be a function on $[0, 1)$ by

$$\Lambda_i\left(\sum_{l\geq 1} \varepsilon_l 2^{-l}\right) := \Lambda_i(\varepsilon_1, \varepsilon_2, \ldots)$$

where we prefer expansions ending with 0^ω to those ending with 1^ω.

Altogether we have

$$E(N) = \mu^{\log_2 N} \Psi(\log_2 N, t) + \vec{v}^T R(N) M_{0,0}^2 \vec{v}, \tag{3}$$

where $\Psi(\log_2 N, t) = \mu^{-\{\log_2 N\}} \vec{v}^T (\Lambda_0(2^{\{\log_2 N\}}) + \Lambda_1(2^{\{\log_2 N\}}) + \Lambda_2(2^{\{\log_2 N\}})) M_{0,0}^2 \vec{v}$ and $N = \sum_{l=0}^{L} \varepsilon_l 2^l$. The function $\Psi(x, t)$ is periodic in x with period 1 and well defined for every $x \in \mathbb{R}$. To show continuity in x we first observe that for $x \in \mathbb{R}$ such that $x = \log_2 y$ with y not a dyadic rational it is clearly continuous. For $x = \log_2 y$ with $y = \sum_{l=1}^{L} \varepsilon_l 2^{-l}$ a dyadic rational the two one-sided limits exist. Therefore it is enough to compare the limits for two sequences $\log_2 N_k$ and $\log_2 \tilde{N}_k$ with $N_k = y2^{L+k+1} + 2^k$ and $\tilde{N}_k = y2^{L+k+1} + 2^k - 1$. If we insert these two sequences into (3) and subtract, we get

$$\mathcal{O}(N) = E(N_k) - E(\tilde{N}_k) = N_k^2 \Psi(\log_2 N_k, t) - \tilde{N}_k^2 \Psi(\log_2 \tilde{N}_k, t) + \mathcal{O}(N_k^{\log_2 \beta(t)}).$$

Hence $\lim_{k\to\infty}\Psi(\log_2 N_k,t)=\lim_{k\to\infty}\Psi(\log_2\tilde{N}_k,t)$ and the function $\Psi(x,t)$ is continuous in x.

The function $\Psi(x,t)$ is also continuous in t, because eigenvalues are continuous. Furthermore it is arbitrarily often differentiable since it is dominated by a geometric series. The first and the second derivative are continuous and periodic in x too, due to the same argument as above. Therefore $\Psi(x,t)$ and its derivatives with respect to t are $\mathcal{O}(1)$.

The error term $\vec{v}^T R(N) M_{0,0}^2 \vec{v}$ is also differentiable with respect to t because it is dominated by a geometric series.

Since we know that the eigenvalues of $M_{\delta,\varepsilon}$ and $B_{i,\varepsilon}$ have modulus less or equal than 2, we can estimate the error term

$$|\vec{v}^T R(N) M_{0,0}^2 \vec{v}| = \mathcal{O}(N^{\log_2\beta(t)})$$

and so

$$E(N) = N^{2+\frac{i}{2\log 2}t-\frac{1}{32\log 2}t^2+\mathcal{O}(t^3)}\Psi(\log_2 N,t)+\mathcal{O}(N^{\log_2\beta(t)}). \tag{4}$$

If we insert $t=0$ into the last equation, we see that $\Psi(\log_2 N,0)=1+\mathcal{O}(N^{-1})$.

By differentiation of $E(N)$ with respect to t and inserting $t=0$ we obtain the following formula

$$\sum_{m,n<N} h(m,n) = \frac{1}{2}N^2\log_2 N + N^2\Psi_1(\log_2 N)+\mathcal{O}(N\log N)$$

with $\Psi_1(x)=i\frac{\partial}{\partial t}\Psi(x,t)|_{t=0}$.

The second derivative of $E(N)$ with respect to t at $t=0$ gives

$$\sum_{m,n<N} h^2(m,n) = \frac{1}{4}N^2\log_2^2 N + \frac{1}{16}N^2\log_2 N + N^2\log_2 N\Psi_1(\log_2 N) + N^2\Psi_2(\log_2 N) + \mathcal{O}(N\log^2 N)$$

where $\Psi_2(x)=-\frac{\partial^2}{\partial t^2}\Psi(x,t)|_{t=0}$.

We can calculate the expected value and the variance of the random variable $X=h(N)$ now. The expected value is

$$\frac{1}{N^2}\sum_{m,n<N} h(m,n) = \frac{1}{2}\log_2 N + \Psi_1(\log_2 N)+\mathcal{O}(N^{-1}\log N)$$

and the variance is

$$\frac{1}{N^2}\sum_{m,n<N} h^2(m,n) - \left(\frac{1}{N^2}\sum_{m,n<N} h(m,n)\right)^2 = \frac{1}{16}\log_2 N + \Psi_2(\log_2 N) - \Psi_1^2(\log_2 N) + \mathcal{O}(N^{-1}\log N).$$

To compare the distribution of X with the normal distribution, we use the Berry-Esseen inequality (see [4, 7]) in a version proved by Vaaler [20].

Theorem 11 (Berry-Esseen inequality)**.** *Let $f(x)$ and $g(x)$ be probability distribution functions with characteristic functions $\hat{f}(t)$ and $\hat{g}(t)$. Suppose $t^{-1}(\hat{g}(t)-\hat{f}(t))$ is integrable on a neighborhood of zero and $f(x)$ has a density function $f'(x)$ bounded from above by M. Then*

$$|g(x)-f(x)| \leq \int_{-T}^{T} \hat{J}(T^{-1}t)\frac{1}{2\pi t}\left|\hat{g}(t)-\hat{f}(t)\right| dt + \frac{1}{2T}\left(M+\int_{-T}^{T}\hat{K}(T^{-1}t)\left(\hat{g}(t)-\hat{f}(f)\right)dt\right)$$

for all real x, all $T>0$ and

$$\hat{J}(t)=\begin{cases}\pi t(1-|t|)\cot(\pi t)+|t| & \text{for } |t|\leq 1,\\ 0 & \text{else,}\end{cases}$$

$$\hat{K}(t)=\begin{cases}1-|t| & \text{for } |t|\leq 1,\\ 0 & \text{else.}\end{cases}$$

We need the characteristic function $\hat{g}_N(t)$ of the standardized random variable

$$Z=\frac{X-\frac{1}{2}\log_2 N}{\frac{1}{4}\sqrt{\log_2 N}}.$$

Therefore we insert $\frac{t}{\frac{1}{4}\sqrt{\log_2 N}}$ instead of t in (4) and multiply the whole equation by $\frac{1}{N^2}\exp(-it\frac{\frac{1}{2}\log_2 N}{\frac{1}{4}\sqrt{\log_2 N}})$. Then we get

$$\begin{aligned}\hat{g}_N(t) &= \frac{1}{N^2}\sum_{m,n<N}\exp\left(it\frac{h(m,n)-\frac{1}{2}\log_2 N}{\frac{1}{4}\sqrt{\log_2 N}}\right)\\ &= e^{-\frac{t^2}{2}}\left(1+\mathcal{O}\left(\frac{t^3}{\log^{3/2} N}\right)\right)\Psi\left(\log_2 N,\frac{t}{\frac{1}{4}\sqrt{\log_2 N}}\right)\\ &\quad+\tilde{R}\left(N,\frac{t}{\frac{1}{4}\sqrt{\log_2 N}}\right)e^{-it2\sqrt{\log_2 N}}.\end{aligned}$$

Here we have $\tilde{R}(N,t):=\frac{1}{N^2}\vec{v}^T R(N) M_{0,0}^2\vec{v}$. Since $\hat{g}_N(t)$ is a characteristic function, we have

$$1=\hat{g}_N(0)=\psi_0+r_0 \tag{5}$$

with $\psi_0:=\Psi(\log_2 N,0)$ and $r_0:=\tilde{R}(N,0)$. Since we have a dominating eigenvalue around $t=0$ we get $\tilde{R}(N,t)=\mathcal{O}(N^{-\varepsilon})$ for a small $\varepsilon>0$. Next we have to estimate the difference between $\hat{g}_N(t)$ and the characteristic function of the normal distribution with mean 0 and variance 1, which is $\hat{f}(t):=e^{-\frac{t^2}{2}}$. For $t=o(\sqrt{\log N})$ we have

$$\begin{aligned}\left|\hat{g}_N(t)-\hat{f}(t)\right| &= \left|e^{-\frac{t^2}{2}}\left(1+\mathcal{O}\left(\frac{t^3}{\log^{3/2} N}\right)\right)\left(\psi_0+\mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right)\right)\right.\\ &\qquad \left.+\tilde{R}\left(N,\frac{t}{\frac{1}{4}\sqrt{\log_2 N}}\right)e^{-it2\sqrt{\log_2 N}}-(\psi_0+r_0)e^{-\frac{t^2}{2}}\right|\\ &\le \left|e^{-\frac{t^2}{2}}\mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right)+\left(r_0+\mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right)\right)\left(1+\mathcal{O}\left(t\sqrt{\log N}\right)\right)-r_0\right|\\ &\le \mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right)+\mathcal{O}\left(\frac{t\sqrt{\log N}}{N^{\varepsilon}}\right)=\mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right).\end{aligned}$$

Here we used $e^{-\frac{t^2}{2}} \le 1$, (5) and $\tilde{R}(N,t) = r_0 + \mathcal{O}(t) = \mathcal{O}(N^{-\varepsilon})$ around $t = 0$.

Let $g_N(x) := \mathbb{P}(Z < x)$ be the distribution function of the random variable Z and $f(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{y^2}{2}}dy$ the distribution function of the normal distribution with mean 0 and variance 1. By the Berry-Esseen inequality, Theorem 11, we have

$$\begin{aligned}|g_N(x)-f(x)| &\le \int_{-T}^{T}\hat{J}(T^{-1}t)\frac{1}{2\pi t}\left|\hat{g}_N(t)-e^{-\frac{t^2}{2}}\right|dt\\ &\qquad +\frac{1}{2T}\left(1+\int_{-T}^{T}\hat{K}(T^{-1}t)\left(\hat{g}_N(t)-e^{-\frac{t^2}{2}}\right)dt\right)\\ &\le \int_{-T}^{T}\mathcal{O}\left(\frac{1}{\sqrt{\log N}}\right)dt+\frac{1}{2T}\left(1+\int_{-T}^{T}(1-|T^{-1}t|)\mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right)dt\right)\\ &= \mathcal{O}\left(\frac{T}{\sqrt{\log N}}\right)+\frac{1}{2T},\end{aligned}$$

because the density function of the normal distribution is bounded from above by 1. If we choose $T = \sqrt[4]{\log N}$ we get

$$\mathbb{P}\left(\frac{h(n)-\frac{1}{2}\log_2 N}{\frac{1}{4}\sqrt{\log_2 N}} < x\right) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{y^2}{2}}dy+\mathcal{O}\left(\frac{1}{\sqrt[4]{\log_N}}\right).$$

□

In [10], Grabner, Heuberger and Prodinger also describe a generalization of the simple joint sparse form for higher dimensions. But we present an other generalization in Section 2.7.

2.6 Asymmetric digit sets

Up to now we have only considered digits sets which are symmetric around 0, that is if $\varepsilon \in D$, then also $-\varepsilon \in D$. In [15], Phillips and Burgess present minimal weight representations over an asymmetric digit set satisfying the following definition.

Definition 2.9. *Let r be a radix, l and u integers and $w \geq 2$ an integer. Let $D = \{a \in \mathbb{Z} \mid l \leq a \leq u \text{ and } a \not\equiv 0 \mod r\} \cup \{0\}$ be the digit set. Further following conditions have to be satisfied:*

1. $1 - r^{w-1} \leq l \leq 0$
2. $1 \leq u \leq r^{w-1} - 1$
3. $|D| \geq r^{w-1} - r^{w-2} + 1$
4. $u \equiv -1 \mod r$
5. *If $l < 0$, then $l \equiv 1 \mod r$.*

Conditions 1 and 2 imply $1 \in D$. Furthermore together with Condition 3 we have for any $n = (\varepsilon_{w-2} \dots \varepsilon_0)$ over the digit set $\{0, 1, \dots, r-1\}$ and a carry $c \in \{0, 1\}$ with $\varepsilon_0 + c \not\equiv 0 \mod r$ at least one of $n + c$ or $n + c - r^{w-1}$ is contained in D. This also implies that $u - l \geq r^{w-1}$. Conditions 4 and 5 are used to show minimality of the output of Algorithm 5.

In [15], Phillips and Burgess give Algorithm 5 which computes a representation over D from a standard radix-r expansion of an integer. They further show that the output has minimal weight among all representations over the digit set D.

It is clear that the algorithm produces a representation over the digit set D of the input n. To prove that the output is a minimal weight representation we first investigate the carry propagation in this digit set in the next lemmas.

Lemma 3. *Let $(\varepsilon_L \dots \varepsilon_0)$ be a representation over the digit set D, $i \in [0, L]$ and $c \in [\frac{l}{r-1}, \frac{u}{r-1}]$ integers. Then there exists a representation $(\delta_{L'} \dots \delta_0)$ over the digit set D with*

$$(\delta_{L'} \dots \delta_0) = (\varepsilon_L \dots \varepsilon_0) + cr^i$$

and

$$h(\delta_{L'} \dots \delta_0) \leq h(\varepsilon_L \dots \varepsilon_0) + 1.$$

Proof (cf. [15]). Algorithm 6 computes a representation $(\delta_{L'} \dots \delta_0)$ of $(\varepsilon_L \dots \varepsilon_0) + cr^i$. It is clearly a representation over the digit set D. To prove the bound for the weight change we examine the four cases of the **if** statement.

1. If $\delta_i \equiv 0 \mod r$, then we will set $\delta_i = 0$. Therefore the weight will be reduced by one. The carry which propagates to the next digit is $\frac{\delta_i}{r}$. Since $l \leq \delta_i \leq u$ and $\frac{l}{r-1} \leq c \leq \frac{u}{r-1}$, we know that the new carry c satisfies $\frac{l}{r-1} \leq c \leq \frac{u}{r-1}$ too. Therefore the new carry c and $i + 1$ is a valid input to the algorithm.

2. If $\delta_i > u$ and $\delta_i \not\equiv 0 \mod r$, then we have $u < \delta_i \leq u + \frac{u}{r-1}$. Therefore we have $l < \delta_i - r^{w-1} < u$. Thus $\delta_i - r^{w-1}$ is a valid digit. There is a carry propagation of 1 to the digit δ_{i+m}. Since $\varepsilon_i > 0$ because $\delta_i > u$, there is no change of weight at this position.

Algorithm 5 Algorithm to compute a minimal weight representation over an asymmetric digit set

Input: Standard radix-r expansion $(\varepsilon_{L-1} \dots \varepsilon_0)$ of an integer n
Output: Minimal weight representation $(\delta_L \dots \delta_0)$ of n over the digit set D

$s = 0$, $c = 0$
for $i = 0$ to $L - 1$ **do**
  if $s = 0$ **then**
    if $\varepsilon_i + c \bmod r = 0$ **then**
      $\delta_i = 0$
    else
      $x = \sum_{j=0}^{w-2} \varepsilon_{i+j} r^j$
      if $x + c > u$ **then**
        $\delta_i = x + c - r^{w-1}$
        $c = 1$
      else if $\varepsilon_{i+m} = r - 1$ **and** $x + c \geq l + r^{w-1}$ **then**
        $\delta_i = x + c - r^{w-1}$
        $c = 1$
      else
        $\delta_i = x + c$
        $c = 0$
      end if
      $s = w - 2$
    end if
  else
    $\delta_i = 0$
    $s = s - 1$
  end if
end for
$\delta_L = c$

3. If $\delta_i < l$ and $\delta_i \not\equiv 0 \mod r$, then we have $l + \frac{l}{r-1} \leq \delta_i < u$. Therefore we have $l < \delta_i + r^{w-1} < u$. Thus $\delta_i + r^{w-1}$ is a valid digit. There is a carry propagation of -1 to the digit δ_{i+m}. Since $\varepsilon_i < 0$ because $\delta_i < l$, there is no change of weight at this position.

4. In all other cases, $\delta_i \in D$ and the carry propagation stops. Therefore the algorithm terminates. If $\delta_i = 0$, then the weight decreases by one. If $\delta_i \neq 0$ and $\varepsilon_i \neq 0$, then the weight remains the same. If $\delta_i \neq 0$ and $\varepsilon_i = 0$, then the weight increases by one.

In all cases the weight of the output only increases by one if the algorithm terminates. Therefore the inequality

$$h(\delta_{L'} \dots \delta_0) \leq h(\varepsilon_L \dots \varepsilon_0) + 1$$

holds. □

Algorithm 6 Algorithm to add a digit to a representation over an asymmetric digit set D

Input: Any representation $(\varepsilon_L \dots \varepsilon_0)$ of an integer n over the digit set D, $c \in [\frac{l}{r-1}, \frac{u}{r-1}]$
Output: Representation of $(\varepsilon_L \dots \varepsilon_0) + cr^i$ over the digit set D

```
δ_j = ε_j for all j
while c ≠ 0 do
    δ_i = ε_i + c
    if δ_i ≡ 0 mod r then
        c = δ_i / r
        δ_i = 0
        i = i + 1
    else if δ_i > u then
        c = 1
        δ_i = δ_i − r^m
        i = i + m
    else if δ_i < l then
        c = −1
        δ_i = δ_i + r^m
        i = i + m
    else
        c = 0
    end if
end while
```

Lemma 4. *Let $(\varepsilon_L \dots \varepsilon_0)$ be a representation over the digit set D with $\varepsilon_i \neq 0$ for an $i \in [0, L]$ and $c = \pm 1$. Then there exists a representation $(\delta_L \dots \delta_0)$ over the digit set D with*

$$(\delta_L \dots \delta_0) = (\varepsilon_L \dots \varepsilon_0) + cr^i$$

and

$$h(\delta_L \dots \delta_0) \leq h(\varepsilon_L \dots \varepsilon_0).$$

Proof (cf. [15]). The proof of this lemma follows along the proof of Lemma 3. The difference to Lemma 3 is that because of Conditions 4 and 5 of our digit set D we are either in case 4 and the carry does not propagate, or we are in case 1 and set $\delta_i = 0$. The other two cases do not occur because if $\delta_i \notin D$, then $\delta_i \equiv 0 \mod r$.

In case 4 the weight is unchanged because $\varepsilon_i \neq 0$. In case 1 the weight is first decreased by one and later at the end of the algorithm it may be increased by one. Thus the weight remains the same too. □

Lemma 5. *Let $z \in \mathbb{Z}$ be such that $l + lr^i \leq z \leq u + ur^i$ for an $i \in \{1, \dots, w-2\}$. Then there exists an integer α with $\frac{l}{r-1} \leq \alpha \leq \frac{u}{r-1}$ and $l \leq z - \alpha r^{w-1} \leq u$.*

Proof. If $l \leq z \leq u$, then $\alpha = 0$ is a solution. If $u < z \leq u + ur^i$, then we must have $\alpha > 0$. If we can choose an α in the interval $\left[\frac{l}{r-1}, \frac{u}{r-1}\right]$ with $u + ur^i \leq u + \alpha r^{w-1}$, then this α satisfies $z - \alpha r^{w-1} \leq u$. Thus we need an α satisfying

$$\frac{u}{r^{w-1-i}} \leq \alpha \leq \frac{u}{r-1}.$$

By Condition 4 on the digit set D, there exists an integer $c \geq 1$ with $u = cr - 1$. Therefore we have

$$\frac{u}{r^{w-1-i}} \leq \frac{u}{r} \leq c \leq \frac{u}{r-1},$$

and we can choose $\alpha = c$.

If α also satisfies $l \leq z - \alpha r^{w-1}$, then it is the solution stated in the lemma. Otherwise we know that $u - l \geq r^{w-1}$, therefore there exists an integer $d \geq 0$ with $l \leq z - dr^{w-1} \leq u$ for $z > 0$. Then we have $\alpha \geq d$. If we use d instead of α, then $\frac{l}{r-1} \leq 0 \leq d \leq \alpha \leq \frac{u}{r-1}$.

If $l + lr^i \leq z < l$ and $l < 0$ holds, we can find a solution α as in the case $u < z \leq u + ur^i$. If $l = 0$, nothing has to be proved. □

Now we can prove the minimality of the output of Algorithm 5 like in [15].

Theorem 12. *The output of Algorithm 5 is a minimal weight representation of the input n over the digit set D.*

Proof (cf. [15]). We assume that the output is not always a minimal weight representation. Let $(\varepsilon_L \dots \varepsilon_0)$ be the output of Algorithm 5 with input n. Let $(\delta_{L'} \dots \delta_0)$ be a minimal weight representation of n over the digit set D with $h(\delta_{L'} \dots \delta_0) < h(\varepsilon_L \dots \varepsilon_0)$. We assume L to be minimal. Then we have the following five cases. In each case we either find a contradiction or give a construction for a shorter counterexample.

1. If $\varepsilon_0 = \delta_0$, then we can use $(\varepsilon_L \dots \varepsilon_1)$ and $(\delta_{L'} \dots \delta_1)$ instead of $(\varepsilon_L \dots \varepsilon_0)$ and $(\delta_{L'} \dots \delta_0)$.

2. If $\delta_0 = 0$ and $\varepsilon_0 \neq 0$, $\varepsilon_0 \not\equiv 0 \mod r$, but $\delta_0 \equiv \varepsilon_0 \mod r$. This is a contradiction.

3. If $\delta_0 = \varepsilon_0 - r^{w-1}$ and $\delta_1 = \ldots = \delta_{w-2} = 0$, then we have $\delta_{w-1}r^{w-1} + \delta_0 = \varepsilon_{w-1}r^{w-1} + \varepsilon_0 + ar^w$ for an integer a and therefore

$$\delta_{w-1} = \varepsilon_{w-1} + ar + 1. \tag{6}$$

By Condition 1 on the digit set D, we have $-r^{w-1}+1 \leq \delta_0 = \varepsilon_0 - r^{w-1}$ and therefore $\varepsilon_0 \geq 0$. In Algorithm 5, ε_0 was not chosen to be negative, therefore the w-th digit γ_{w-1} of the standard radix-r expansion of n is not $r-1$. Since $\varepsilon_{w-1} \equiv \gamma_{w-1} \mod r$ in the algorithm, we have $\varepsilon_{w-1} \not\equiv -1 \mod r$. Hence (6) implies $\delta_{w-1} \neq 0$.

Therefore we can change the least significant digit to $\delta'_0 = \delta_0 + r^{w-1} = \varepsilon_0$. Then we have to find a representation $(\delta'_{L''} \ldots \delta'_1)$ for $(\delta_{L'} \ldots \delta_1) - r^{w-1}$. This is done by Lemma 4 without increasing the weight. Then we can consider the representations $(\delta'_{L''} \ldots \delta'_1)$ and $(\varepsilon_L \ldots \varepsilon_1)$ instead of the original ones.

4. If $\delta_0 = \varepsilon_0 + r^{w-1}$ and $\delta_1 = \ldots = \delta_{w-2} = 0$, then we have $\delta_{w-1}r^{w-1} + \delta_0 = \varepsilon_{w-1}r^{w-1} + \varepsilon_0 + ar^w$ for an integer a and therefore

$$\delta_{w-1} = \varepsilon_{w-1} + ar - 1. \tag{7}$$

By Condition 2 on the digit set D, we have $r^{w-1} - 1 \geq \delta_0 = \varepsilon_0 + r^{w-1}$ and therefore $\varepsilon_0 \leq 0$. In Algorithm 5, ε_0 was chosen to be negative, therefore the w-th digit γ_{w-1} of the standard radix-r expansion of n is $r-1$. Then we will choose $\varepsilon_{w-1} = 0$ in the algorithm. Hence (7) implies $\delta_{w-1} \neq 0$.

Therefore we can change the least significant digit to $\delta'_0 = \delta_0 - r^{w-1} = \varepsilon_0$. Then we have to find a representation $(\delta'_{L''} \ldots \delta'_1)$ for $(\delta_{L'} \ldots \delta_1) + r^{w-1}$. This is done by Lemma 4 without increasing the weight. Then we can consider the representations $(\delta'_{L''} \ldots \delta'_1)$ and $(\varepsilon_L \ldots \varepsilon_1)$ instead of the original ones.

5. In all other cases we have $\delta_0 \neq 0$ and $\delta_0 \neq \varepsilon_0$. We will construct a representation $(\delta'_{L''} \ldots \delta'_0)$ of n without increasing the weight and $\delta'_0 = \varepsilon_0$ or $\delta'_0 = \varepsilon_0 \pm r^{w-1}$ and $\delta_1 = \ldots = \delta_{w-2} = 0$. Then with one of the previous cases we can construct a shorter counterexample.

Since $(\varepsilon_L \ldots \varepsilon_0)$ is the output of Algorithm 5, we have $\varepsilon_1 = \ldots = \varepsilon_{w-2} = 0$ and therefore

$$\sum_{i=0}^{w-2} \delta_i r^i = \varepsilon_0 \bmod r^{w-1}.$$

Thus there is an $a \in \mathbb{Z}$ with

$$\delta_0 - \varepsilon_0 = ar^{w-1} - \delta_{w-2}r^{w-1} - \ldots - \delta_1 r.$$

To construct the representation $(\delta'_{L''} \ldots \delta'_0)$, we first assign $\delta'_i = \delta_i$ for all i. We start at $i = 1$ and stop at $i = w-2$. If $\delta'_i = 0$, then we do nothing. If $\delta'_i \neq 0$, then $\delta_0 - \varepsilon_0$

is a multiple of r^i. We change the digits $\delta'_0 = \delta'_0 + r^i\delta'_i$ and $\delta'_i = 0$. Then we have reduced the weight by one. Since $l + lr^i \leq \delta'_0 \leq u + ur^i$ is not a valid digit, we have to find an $\alpha \in \mathbb{Z}$ with $l \leq \delta'_0 - \alpha r^{w-1} \leq u$. By Lemma 5 we can choose $\alpha \in \left[\frac{l}{r-1}, \frac{u}{r-1}\right]$. Then we have a carry α to the digit δ'_{w-1}. By Lemma 3 we can process this carry with increasing the weight by at most one. So after both steps of changing δ'_0 the weight has decreased by one or is unchanged. Then we go on with the next $i = i + 1$.

At the end of this algorithm when $i = w - 1$, we have $\delta_1 = \ldots = \delta_{w-2} = 0$ and $\delta'_0 - \varepsilon_0 = ar^m$. Since δ'_0, $\varepsilon_0 \in [-r^{w-1} + 1, r^{w-1} - 1]$, we have $a \in \{0, \pm 1\}$.

So for every counterexample we have constructed a shorter counterexample, which is a contradiction. Therefore the output of Algorithm 5 is minimal. □

In [15], Phillips and Burgess also present the expected value of the weight of a minimal representation over the digit set D. They use the probability space of all representations of length k with equal distribution.

Theorem 13. *The expected value of the weight of the minimal representation over the digit set D of length k is $\dfrac{k}{w - 1 + \frac{pr}{r-1}}(1 + o(1))$ with $p = \dfrac{u - l - \left\lfloor \frac{-l}{r} \right\rfloor - \left\lfloor \frac{u}{r} \right\rfloor}{r^{w-1}(r-1)}$.*

2.7 Joint integer representations with asymmetric digit sets

In the last section we introduced representations over asymmetric digit sets in dimension one. Now we will generalize to an arbitrary dimension d, and an asymmetric digit set

$$D_{l,u} = \{a \in \mathbb{Z} \mid l \leq a \leq u\}$$

for $l \leq 0$ and $u \geq 1$, as in [11] by Heuberger and Muir. Obviously $0, 1 \in D_{l,u}$ for any l and u. We look at dimension-d radix-2 representations. If $l = 0$, $D_{l,u}$ only consists of positive digits. Therefore we can only represent positive integers. Otherwise if $l \leq -1$ we also can represent negative integers, for example by exchanging every 1 for $\bar{1}$ in the standard binary expansion.

The digit set $D_{l,u}$ contains $u - l + 1$ digits. We define w to be the unique integer such that

$$2^{w-1} \leq u - l + 1 < 2^w.$$

We know that $w \geq 2$, since $D_{l,u}$ has at least two elements. Without loss of generality we can restrict l to be greater than -2^{w-1}. Otherwise we would take the digit set $D_{-u,-l}$ where we have $-2^{w-1} < -u \leq -1$. Then every representation of a vector N of integers with digit set $D_{l,u}$ would correspond to a representation of $-N$ with digit set $D_{-u,-l}$ by changing the sign of each digit. Thereby the weight of the representation does not change.

The digit set $D_{l,u}$ contains a complete system of residues modulo 2^{w-1}. Two possibilities are the sets $\mathbf{lower}(D_{l,u}) := \{l, l + 1, \ldots, l + 2^{w-1} - 1\}$ and $\mathbf{upper}(D_{l,u}) := \{u - 2^{w-1} + 1, \ldots, u - 1, u\}$. If $u - l + 1 = 2^{w-1}$ these two sets will coincide. Then $D_{l,u}$ contains every residue modulo 2^{w-1} exactly once. Otherwise there will be digits which are unique modulo

2^{w-1}, these are $\mathbf{unique}(D_{l,u}) := \{a \in D_{l,u} | u - 2^{w-1} < a < l + 2^{w-1}\}$, and digits which are not unique, these are $\mathbf{nonunique}(D_{l,u}) := \{a \in D_{l,u} | a \leq u - 2^{w-1} \text{ or } l + 2^{w-1} \leq a\}$. If we have a number $n \in \mathbb{Z}$ and we want to compute a digit $a \in D_{l,u}$ such that $n \equiv a \mod 2^{w-1}$, we have two choices, namely

$$a = l + \left(n - l \bmod 2^{w-1}\right) \text{ or}$$
$$a = u - \left(u - n \bmod 2^{w-1}\right).$$

Maybe these two possibilities are the same. If they are different, however, then the difference is 2^{w-1}. No matter which one we choose, the next $w - 2$ digits are 0 since $n - a \equiv 0 \mod 2^{w-1}$.

Next we will define an order on the words over the alphabet $\{0, 1\}$. It will be similar to the lexicographic order, but reading words from right to left.

Definition 2.10 (Colexicographic order)**.**

1. *Let $(a_n \dots a_0), (b_n \dots b_0) \in \{0,1\}^*$ be two words of the same length. Then*

$$(a_n \dots a_0) \prec (b_n \dots b_0)$$

if $\exists 0 \leq m \leq n\, (\forall i < m : \quad a_i = b_i) \wedge a_m < b_m$ and

$$(a_n \dots a_0) \preceq (b_n \dots b_0)$$

if $(a_n \dots a_0) \prec (b_n \dots b_0)$ or $(a_n \dots a_0) = (b_n \dots b_0)$. If the words are not of the same length, just prepend zeros in front of the shorter word.

2. *For a joint integer representation $(\varepsilon_n \dots \varepsilon_0)$ of a vector of integers over a digit set D we define a string $(a_n \dots a_0) \in \{0,1\}^*$ with*

$$a_i = \begin{cases} 0 & \text{if } \varepsilon_i = 0, \\ 1 & \text{else.} \end{cases}$$

For two integer representations $(\varepsilon_n \dots \varepsilon_0)$ and $(\delta_n \dots \delta_0)$ of two vectors of integers over the digit set D we say $(\varepsilon_n \dots \varepsilon_0) \preceq (\delta_n \dots \delta_0)$ if $(a_n \dots a_0) \preceq (b_n \dots b_0)$ is true for the corresponding strings $(a_n \dots a_0)$ and $(b_n \dots b_0)$.

For example these words are colexicographically ordered:

$$0 \preceq 10100 \preceq 10 \preceq 1 \preceq 101 \preceq 11.$$

The following integer representations are colexicographically ordered as well:

$$\binom{10}{20} \preceq \binom{310}{100} \preceq \binom{430}{110} \preceq \binom{21}{10} \preceq \binom{112}{020} \preceq \binom{032}{113}.$$

As in the previous sections we are interested in integer representations with minimal weight. In [11], Heuberger and Muir present an algorithm which computes the colexicographically minimal integer representation. At the same time the output has minimal weight. An integer representation $(\varepsilon_n \dots \varepsilon_0)$ of an integer vector N over a digit set $D_{l,u}$ is colexicographically minimal, if for all integer representations $(\delta_m \dots \delta_0)$ over $D_{l,u}$ with $\sum_{i=0}^{n} \delta_i 2^i = N$ we have $(\varepsilon_n \dots \varepsilon_0) \preceq (\delta_m \dots \delta_0)$. Algorithm 7 is this algorithm. For simplicity we write $n + a$, for a vector n and an integer a, and mean that we add a to every coordinate of the vector n. Furthermore we have the vector $M = (m_1, \dots, m_d)^T$.

Algorithm 7 Algorithm to compute the colexicographically minimal and minimal weight representation

Input: $N \in \mathbb{Z}^d$, $l \leq 0$, $u \geq 1$, N has to be nonnegative if $l = 0$
Output: $(\varepsilon_{L-1} \dots \varepsilon_0)$ a colexicographically minimal, minimal weight representation of N

1: $L = 0$
2: **while** $N \neq 0$ **do**
3: **if** $N \equiv 0 \mod 2$ **then**
4: $A = 0$
5: **else**
6: $A = l + ((N - l) \bmod 2^{w-1})$
7: $I_{\text{unique}} = \{j \in \{1, 2, \dots, d\} \mid a_j \in \mathbf{unique}(D_{l,u})\}$
8: $I_{\text{nonunique}} = \{j \in \{1, 2, \dots, d\} \mid a_j \in \mathbf{nonunique}(D_{l,u})\}$
9: $M = (N - A)/2^{w-1}$
10: **if** $m_j \equiv 0 \mod 2$ for all $j \in I_{\text{unique}}$ **then**
11: **for** $j \in I_{\text{nonunique}}$ such that m_j is odd **do**
12: $a_j = a_j + 2^{w-1}$
13: **end for**
14: **else**
15: **for** $j \in I_{\text{nonunique}}$ such that $m_j \equiv u + 1 \mod 2^{w-1}$ **do**
16: $a_j = a_j + 2^{w-1}$
17: **end for**
18: **end if**
19: **end if**
20: $\varepsilon_L = A$
21: $N = (N - A)/2$
22: $L = L + 1$
23: **end while**
24: **return** $\varepsilon_{L-1} \dots \varepsilon_0$

The **if** branch in line 3 of Algorithm 7 makes the digit at position L a zero column if possible. If this is not possible, the **else** branch in line 5 chooses a smallest digit which is congruent to the input. In the inner **if** and **else** branches the algorithm checks if we should change any non-unique digits. In the **if** statement in line 10 we check whether we can make the $(w - 1)$-st digit after the current digit zero. If this is not possible we check

in the **else** statement in line 14 whether we can increase the redundancy at the $(w-1)$-st digit after the current digit by changing any non-unique digits at the current position.

In the following lemma we prove that Algorithm 7 terminates.

Lemma 6. *For an input vector $N \in \mathbb{Z}^d$, where N is nonnegative if $l = 0$, Algorithm 7 terminates.*

Proof (cf. [11]). First let $l < 0$. Then $\max\{u, |l|\} \leq u - l - 1 < 2^{w-1} - 2$. We show that $\|N\|_\infty$ is overall decreasing. Here $\|N\|_\infty = \max\{|n_j|\}$ is the maximum norm of N.

If $A = 0$, then $\|(N - A)/2\|_\infty = \|N/2\|_\infty < \|N\|_\infty$. If $A \neq 0$, $\|N\|_\infty$ does not decrease in the next step, but it is decreased after $w-1$ steps. In the steps in between we have zero digits, since $N \equiv A \mod 2^{w-1}$. If $\|N\|_\infty \geq 2$, the weight decrease is because of

$$\begin{aligned}\|(N - A)/2^{w-1}\|_\infty &\leq (\|N\|_\infty + \max\{u, |l|\})/2^{w-1} \\ &< (\|N\|_\infty + 2^w - 2)/2^{w-1} \\ &\leq \|N\|_\infty.\end{aligned}$$

If $\|N\|_\infty = 1$, the algorithm chooses $A = N$, because all entries of N belong to the digit set $D_{l,u}$. Therefore A is a colexicographically minimal representation of N, since we can not choose the least significant digit of the representation to be 0. Hence Algorithm 7 terminates for $l < 0$.

If $l = 0$, N has to be nonnegative at the beginning. We will show that N will never be negative during the runtime of the algorithm. Since all digits are nonnegative $(N - A)/2$ will decrease and the algorithm terminates.

To show that N is never negative, we observe that this could only happen if $n_j < u$, that is n_j is a digit. If the algorithm chooses n_j as a digit of the representation, the j-th coordinate will become zero, which does not matter. But if $n_j + 2^{w-1}$ is also a digit and the algorithm chooses this digit, then the j-th coordinate would become negative. In this case $n_j \in \mathbf{nonunique}(D_{l,u})$ and $n_j \in \mathbf{lower}(D_{l,u})$. Since we first choose the digit from the lower part, we have $m_j = 0$ in line 9 of Algorithm 7. Since $u \geq 2^{w-1}$, we do not have $0 \equiv m_j \equiv u + 1 \mod 2^{w-1}$. Therefore we do not change anything in the **if** branch in line 10 nor in the **else** branch in line 14 and the algorithm chooses the digit n_j. □

Since the output of Algorithm 7 obviously is a representation of N over the digit set $D_{l,u}$, we have to show that the output is a colexicographically minimal representation. Therefore there are the next lemmas from [11].

Lemma 7.

1. *Let $(\varepsilon_L \dots \varepsilon_0)$ be a colexicographically minimal representation of a vector $N \in \mathbb{Z}^d$, then $(\varepsilon_L \dots \varepsilon_1)$ is a colexicographically minimal representation of $(N - \varepsilon_0)/2$.*

2. *Let $(\varepsilon_L \dots \varepsilon_0)$ be a minimal weight representation of a vector $N \in \mathbb{Z}^d$, then $(\varepsilon_L \dots \varepsilon_1)$ is a minimal weight representation of $(N - \varepsilon_0)/2$.*

Proof (cf. [11]).

1. Suppose $(\varepsilon_L \dots \varepsilon_1)$ is not a colexicographically minimal representation of $(N-\varepsilon_0)/2$. Let $(\delta_{L'} \dots \delta_1)$ be a colexicographically minimal representation of $(N-\varepsilon_0)/2$. Then we have $(\delta_{L'} \dots \delta_1) \prec (\varepsilon_L \dots \varepsilon_1)$ and therefore $(\delta_{L'} \dots \delta_1\varepsilon_0) \prec (\varepsilon_L \dots \varepsilon_1\varepsilon_0)$. Since both $(\delta_{L'} \dots \delta_1\varepsilon_0)$ and $(\varepsilon_L \dots \varepsilon_1\varepsilon_0)$ are representations of N, this contradicts the minimality of $(\varepsilon_L \dots \varepsilon_0)$.

2. Suppose $(\varepsilon_L \dots \varepsilon_1)$ is not a minimal weight representation of $(N-\varepsilon_0)/2$. Let $(\delta_{L'} \dots \delta_1)$ be a minimal weight representation of $(N-\varepsilon_0)/2$. Then we have $h(\delta_{L'} \dots \delta_1) < h(\varepsilon_L \dots \varepsilon_1)$ and therefore $h(\delta_{L'} \dots \delta_1\varepsilon_0) < h(\varepsilon_L \dots \varepsilon_1\varepsilon_0)$. Since both $(\delta_{L'} \dots \delta_1\varepsilon_0)$ and $(\varepsilon_L \dots \varepsilon_1\varepsilon_0)$ are representations of N, this contradicts the minimality of $(\varepsilon_L \dots \varepsilon_0)$.

□

This first lemma does not make use of any properties of $D_{l,u}$. Therefore it is true for any digit set D, in contrast to the next lemmas.

Lemma 8. *For any representation $(\varepsilon_{w-2} \dots \varepsilon_0)$ with $\varepsilon_j \in D_{l,u}^d$, the equation*

$$(\varepsilon_{w-2} \dots \varepsilon_0) = x2^{w-1} + y$$

has an integer solution (x, y) with x, $y \in D_{l,u}^d$.

Proof (cf. [11]). If the result is true for $d = 1$, we can solve the equation for each coordinate separately. Therefore we only look at one coordinate. All integers between $l2^{w-1} + l$ and $u2^{w-1} + u$ can be expressed as $x2^{w-1} + y$ with x, $y \in D_{l,u}$. Since $l \le 0$ and $u \ge 1$, we have

$$l2^{w-1} + l \le \sum_{i=0}^{w-2} l2^i \le (\varepsilon_{w-2} \dots \varepsilon_0) \le \sum_{i=0}^{w-2} u2^i \le u2^{w-1} + u,$$

and thus $(\varepsilon_{w-2} \dots \varepsilon_0)$ can be expressed in this a way. □

Lemma 9. *For any $a \in D_{l,u}^d$ and any representation $(0\varepsilon_{L-1} \dots \varepsilon_1\varepsilon_0)$ with $\varepsilon_j \in D_{l,u}^d$, there exists a representation $(\delta_L \dots \delta_0)$ with $\delta_j \in D_{l,u}^d$ such that*

$$(\delta_L \dots \delta_0) = (0\varepsilon_{L-1} \dots \varepsilon_0) + a$$

and

$$h(\delta_L \dots \delta_0) \le h(0\varepsilon_{L-1} \dots \varepsilon_0) + 1.$$

Proof (cf. [11]). We just use the classical algorithm for addition in each coordinate. There can be a carry which is bounded by l and u. Therefore the carry propagation stops if we reach the first zero from the right. Thus there is at most one zero column of $(0\varepsilon_{L-1} \dots \varepsilon_0)$ changed into a nonzero column. This is the first zero column from the right. Hence the weight increases by at most one. □

Lemma 10.

1. *Let $(\varepsilon_L \dots \varepsilon_0)$ be a colexicographically minimal representation of $N \in \mathbb{Z}^d$ over the digit set $D_{l,u}$, then every nonzero column of this representation must contain an odd digit.*

2. *Every vector $N \in \mathbb{Z}^d$ has a minimal weight representation over digit set $D_{l,u}$, where each nonzero column contains an odd digit.*

Proof (cf. [11]).

1. Assume the representation $(\varepsilon_L \dots \varepsilon_0)$ contains a nonzero column with only even digits. We can assume that ε_0 is this specific column. Let the $(t+1)$-st column be the first zero column from the right, then we have the representation $(0\varepsilon_{t-1} \dots \varepsilon_0)$ with $\varepsilon_i \neq 0$ for $i < t$. By Lemma 9 we know that there exists a representation $(\delta_t \dots \delta_1)$ with

$$(\delta_t \dots \delta_1) = (0\varepsilon_{t-1} \dots \varepsilon_1) + \frac{\varepsilon_0}{2}.$$

   Now we can replace $(0\varepsilon_{t-1} \dots \varepsilon_0)$ by $(\delta_t \dots \delta_1 0)$ in the representation $(\varepsilon_L \dots \varepsilon_0)$ of N and we get a colexicographically strictly smaller representation of N which contradicts our assumptions.

2. Let $(\varepsilon_L \dots \varepsilon_0)$ be a minimal weight representation of N with a nonzero column consisting of only even digits. We can assume that $\varepsilon_0 \neq 0$. Let ε_t be the first zero column from the right. So we have $(0\varepsilon_{t-1} \dots \varepsilon_0)$. By Lemma 9 there is a representation

$$(\delta_t \dots \delta_1) = (0\varepsilon_{t-1} \dots \varepsilon_1) + \frac{\varepsilon_0}{2}$$

   and

$$h(\delta_t \dots \delta_1) \leq h(0\varepsilon_{t-1} \dots \varepsilon_1) + 1.$$

   Therefore we have

$$h(\delta_t \dots \delta_1 0) \leq h(0\varepsilon_{t-1} \dots \varepsilon_1 \varepsilon_0)$$

   and when we exchange these two representations, we again obtain a minimal weight representation.

□

Lemma 11.

1. *Let $(\varepsilon_L \dots \varepsilon_0)$ be a colexicographically minimal representation of a vector $N \in \mathbb{Z}^d$ over the digit set $D_{l,u}$. If $\varepsilon_j \neq 0$, then $\varepsilon_{j+1} = \dots = \varepsilon_{j+w-2} = 0$.*

2. *Every vector $N \in \mathbb{Z}^d$ has a minimal weight representation over the digit set $D_{l,u}$ where there are at least $w-2$ zero columns in front of each nonzero column.*

Proof (cf. [11]).

1. Assume the result is false for N. Let $(\varepsilon_L \dots \varepsilon_0)$ be a representation of N. We can assume that $\varepsilon_0 \neq 0$ and one of the digits $\varepsilon_{w-2}, \dots, \varepsilon_1$ is nonzero too. By Lemma 8 there exist x, $y \in D^d_{l,u}$ such that

$$(\varepsilon_{w-2} \dots \varepsilon_0) = x2^{w-1} + y.$$

With this solution x, y we can construct a representation of N which is colexicographically less than the original one. This is a contradiction to the fact that $(\varepsilon_L \dots \varepsilon_0)$ is colexicographically minimal.

We have

$$\begin{aligned} N &= (\varepsilon_L \dots \varepsilon_{w-1}\varepsilon_{w-2} \dots \varepsilon_0) \\ &= (\varepsilon_L \dots \varepsilon_{w-1})2^{w-1} + (\varepsilon_{w-2} \dots \varepsilon_0) \\ &= ((\varepsilon_L \dots \varepsilon_{w-1}) + x)\, 2^{w-1} + y \\ &= (\delta_{L+1} \dots \delta_{w-1}0 \dots 0y), \end{aligned}$$

where the addition $(0\varepsilon_L \dots \varepsilon_{w-1}) + x = (\delta_{L+1} \dots \delta_{w-1})$ is done by Lemma 9.

2. We construct the same new representation as above:

$$N = (\varepsilon_L \dots \varepsilon_0) = (\delta_{L+1} \dots \delta_{w-1}0 \dots 0y).$$

By Lemma 9 we further know that

$$h(\delta_{L+1} \dots \delta_{w-1}) \leq h(0\varepsilon_L \dots \varepsilon_{w-1}) + 1$$

and

$$h(0 \dots 0y) \leq h(\varepsilon_{w-2} \dots \varepsilon_0) - 1$$

since on the right hand side there are at least two nonzero digits in the representation. Therefore the new representation has weight

$$h(\delta_{L+1} \dots \delta_{w-1}0 \dots 0y) \leq h(\varepsilon_L \dots \varepsilon_0).$$

Thus it is a minimal weight representation again.

□

Now we have all prerequisites to prove the minimality of the output of Algorithm 7.

Theorem 14.

1. *For any input $N \in \mathbb{Z}^d$, the output of Algorithm 7 has minimal weight.*
2. *For any input $N \in \mathbb{Z}^d$, the output of Algorithm 7 is a colexicographically minimal representation.*

Proof (cf. [11]).

1. Let N be a vector of integers for which the output of Algorithm 7 has not minimal weight. We choose N such that the output $(\varepsilon_L \dots \varepsilon_0)$ of Algorithm 7 has minimal length. Let $(\delta_{L'} \dots \delta_0)$ be a minimal weight representation of N. We can assume that $\varepsilon_0 \neq \delta_0$, since otherwise $\frac{N-\varepsilon_0}{2}$ would be a smaller counterexample to the algorithm. Furthermore by Lemmas 10 and 11 we can assume that $(\delta_{L'} \dots \delta_0)$ has at least one odd coordinate in every nonzero digit and that any nonzero digit is preceded by at least $w-2$ zero columns. If $N \equiv 0 \mod 2$, then $\varepsilon_0 = \delta_0 = 0$, since every nonzero digit of the minimal weight representation contains at least one odd coordinate and the algorithm chooses $\varepsilon_0 = 0$. But this contradicts our choice of N, therefore $N \not\equiv 0 \mod 2$. Because in both representations any nonzero column is preceded by $w-2$ zero columns we have the two representations

$$(\varepsilon_L \dots \varepsilon_{w-1} \underbrace{0 \dots 0}_{w-2} \varepsilon_0) \text{ and } (\delta_{L'} \dots \delta_{w-1} \underbrace{0 \dots 0}_{w-2} \delta_0).$$

   Therefore we have $\varepsilon_0 \equiv \delta_0 \mod 2^{w-1}$ but $\varepsilon_0 \neq \delta_0$. If $D_{l,u}$ consists of exactly 2^{w-1} digits, then $\mathbf{nonunique}(D_{l,u})$ is empty and this contradicts these conditions on ε_0 and δ_0. In this case we have proved that the output of the algorithm has minimal weight.

   From now on we assume that $\mathbf{nonunique}(D_{l,u})$ is not empty. Next we show that the digits ε_{w-1} and δ_{w-1} are nonzero. If $\delta_{w-1} = 0$, then $N \equiv \delta_0 \mod 2^w$ and Algorithm 7 would choose $\varepsilon_0 = \delta_0$ since there is no other possibility. Therefore $\delta_{w-1} \neq 0$ and $\delta_w = \dots = \delta_{2w-3} = 0$ and we have the representation

$$(\delta_{L'} \dots \delta_{2w-2} \underbrace{0 \dots 0}_{w-2} \delta_{w-1} \underbrace{0 \dots 0}_{w-2} \delta_0)$$

   of N.

   We write $a_j = (a_{1,j}, \dots, a_{d,j})^T$ to denote the coordinates of a_j.

   Now assume $\varepsilon_{w-1} = 0$. Then $\varepsilon_0 \equiv \delta_{w-1} 2^{w-1} + \delta_0 \mod 2^w$ and thus

$$\delta'_w := \frac{\delta_{w-1}}{2} + \frac{\delta_0 - \varepsilon_0}{2^w}$$

   is a vector of integers. Furthermore for all i the entry $\delta'_{i,w}$ of δ'_w is in $D_{l,u}$ since

$$l - 1 \leq \frac{l}{2} - 1 < \frac{\delta_{i,w-1}}{2} + \frac{\delta_{i,0} - \varepsilon_{i,0}}{2^w} < \frac{u}{2} + 1 \leq u + 1.$$

   The strict inequalities follow from the fact that $\delta_{i,0} - \varepsilon_{i,0} = \pm 2^{w-1}$ if they are not equal. If we set $\delta'_{w-1} = 0$, $\delta'_0 = \varepsilon_0$ and for all other digits $\delta'_j = \delta_j$, then we get the representation

$$(\delta'_{L'} \dots \delta'_{w+1} \delta'_w \delta'_{w-1} \delta'_{w-2} \dots \delta'_1 \delta'_0) = (\delta_{L'} \dots \delta_{2w-2} \underbrace{0 \dots 0}_{w-3} \delta'_w \underbrace{0 \dots 0}_{w-1} \varepsilon_0)$$

of N. Since the weight of this new representation and of the minimal weight representation is the same, the new representation is a minimal weight representation of N too. Therefore we have a shorter counterexample $\frac{N-\varepsilon_0}{2}$ which contradicts our choice of N. Therefore $\varepsilon_{w-1} \neq 0$ and $\varepsilon_w = \ldots = \varepsilon_{2w-3} = 0$.

Now we examine the integer vector $\frac{N-\varepsilon_0}{2^{w-1}}$. We have

$$\begin{aligned}\frac{N-\varepsilon_0}{2^{w-1}} &= (\varepsilon_L \ldots \varepsilon_{2w-2} \underbrace{0 \ldots 0}_{w-2} \varepsilon_{w-1}) \\ &= (\delta_{L'} \ldots \delta_{2w-2} \underbrace{0 \ldots 0}_{w-2} \delta_{w-1}) + \frac{\delta_0 - \varepsilon_0}{2^{w-1}}.\end{aligned}$$

Every coordinate of the vector $\frac{\delta_0-\varepsilon_0}{2^{w-1}}$ is in $\{0, \pm 1\}$. Now we have to perform the addition $(\delta_{L'} \ldots \delta_{2w-2} 0 \ldots 0 \delta_{w-1}) + \frac{\delta_0-\varepsilon_0}{2^{w-1}}$. Thereby we only have to change the digit δ_{w-1} as we will prove. The result $(\delta_{L'} \ldots \delta_{2w-2} 0 \ldots 0 \delta''_{w-1})$ of the addition has the same weight as $(\delta_{L'} \ldots \delta_{2w-2} 0 \ldots 0 \delta_{w-1})$ and is therefore a minimal weight representation of $\frac{N-\varepsilon_0}{2^{w-1}}$. But we have

$$h(\varepsilon_L \ldots \varepsilon_{2w-2} \underbrace{0 \ldots 0}_{w-2} \varepsilon_{w-1}) < h(\delta_{L'} \ldots \delta_{2w-2} \underbrace{0 \ldots 0}_{w-2} \delta''_{w-1}) \tag{8}$$

which is a contradiction to our choice of N.

To prove that we only have to change the digit δ_{w-1} in the addition

$$(\delta_{L'} \ldots \delta_{2w-2} 0 \ldots 0 \delta_{w-1}) + \frac{\delta_0 - \varepsilon_0}{2^{w-1}},$$

we consider three cases. In each case we only look at one coordinate i. If $\frac{\delta_{i,0}-\varepsilon_{i,0}}{2^{w-1}} = 0$, nothing changes in the addition. If $\frac{\delta_{i,0}-\varepsilon_{i,0}}{2^{w-1}} = 1$, we only have a problem if $\delta_{i,w-1} = u$. Then we would have a carry since $u + 1 \notin D_{l,u}$. We have

$$\begin{aligned}\delta_{i,w-1} + 1 &\equiv \varepsilon_{i,w-1} \mod 2^{w-1} \\ \Rightarrow u + 1 &\equiv \varepsilon_{i,w-1} \mod 2^{w-1} \\ \Rightarrow \varepsilon_{i,w-1} &= u - 2^{w-1} + 1.\end{aligned}$$

Thus $\varepsilon_{i,w-1}$ is at the border of the unique digits. Additionally we have $\varepsilon_{i,0} \in$ **nonunique**$(D_{l,u})$ since $\varepsilon_{i,0} \neq \delta_{i,0}$ and $\varepsilon_{i,0} \equiv \delta_{i,0} \mod 2^{w-1}$. But this combination is impossible in Algorithm 7, since it would change a non-unique digit to increase redundancy at the next nonzero digit. Therefore we will never have to add 1 to the digit u.

If $\frac{\delta_{i,0}-\varepsilon_{i,0}}{2^{w-1}} = -1$ we only have a carry if $\delta_{i,w-1} = l$. Since the difference is -1, we have $\varepsilon_{i,0} = 2^{w-1} + \delta_{i,0}$, thus $\varepsilon_{i,0} \in$ **upper**$(D_{l,u})$. We further have $\varepsilon_{i,w-1} = l + 2^{w-1} - 1$ since $\varepsilon_{i,w-1} \equiv \delta_{i,w-1} - 1 \equiv l - 1 \mod 2^{w-1}$. But Algorithm 7 would choose $\varepsilon_{i,0} \in$ **lower**$(D_{l,u})$ and only change it if the column ε_{w-1} can be made 0 or the redundancy can be increased. These conditions do not occur since $\varepsilon_{w-1} \neq 0$ and $\varepsilon_{i,w-1} \neq u - 2^{w-1}$.

2. The proof of 2 is the same as the proof of 1, only (8) has to be changed. There we argue as follows: The two representations of $\frac{N-\varepsilon_0}{2^{w-1}}$ satisfy

$$(\delta_{L'} \dots \delta_{2w-2} \underbrace{0 \dots 0}_{w-2} \delta''_{w-1}) \prec (\varepsilon_L \dots \varepsilon_{2w-2} \underbrace{0 \dots 0}_{w-2} \varepsilon_{w-1}).$$

Therefore we have constructed a shorter counterexample which contradicts our choice of N.

□

Now we know that every colexicographically minimal representation has minimal weight, since all have the same weight and the output of Algorithm 7 is at the same time colexicographically minimal and has minimal weight.

The combinatorial characterization of the colexicographically minimal integer representation will be presented in the following theorem (which is Theorem 6.1 in [11]).

Theorem 15. *Let $N \in \mathbb{Z}^d$. Then there is exactly one representation $(\varepsilon_n \dots \varepsilon_0)$ (up to leading zeros) of N over the digit set $D_{l,u}$ such that the following conditions are satisfied:*

1. *Each column ε_j is zero or contains an odd digit.*
2. *If $\varepsilon_j \neq 0$ for some j, then $\varepsilon_{j+w-2} = \dots = \varepsilon_{j+1} = 0$.*
3. *If $\varepsilon_j \neq 0$ and $\varepsilon_{j+w-1} \neq 0$ for some j, then*
   (a) *there is an $i \in \{1, \dots, d\}$ such that $\varepsilon_{i,(j+w-1)}$ is odd and $\varepsilon_{i,j} \in \boldsymbol{unique}(D_{l,u})$,*
   (b) *if $\varepsilon_{i,j} \in \boldsymbol{nonunique}(D_{l,u})$, then $\varepsilon_{i,(j+w-1)} \not\equiv u+1 \mod 2^{w-1}$,*
   (c) *if $\varepsilon_{i,j} \in \boldsymbol{upper}(D_{l,u}) \cap \boldsymbol{nonunique}(D_{l,u})$, then $\varepsilon_{i,(j+w-1)} \equiv u \mod 2^{w-1}$.*

Moreover, this representation is the output of Algorithm 7 on input N.

Proof (cf. [11]). We can see that the output of Algorithm 7 satisfies these conditions because of the decisions made in the algorithm. From this and the correctness of the algorithm the existence of such a representation follows.

To show uniqueness, we assume that for an integer vector N there exist two different representations $(\varepsilon_L \dots \varepsilon_0)$ and $(\delta_{L'} \dots \delta_0)$ which satisfy the conditions of the theorem. We choose N such that the minimum of the lengths of the representations is minimal. Thus $\varepsilon_0 \neq \delta_0$. If $\varepsilon_0 = 0$, then Condition 1 implies that $\delta_0 = 0$. Therefore ε_0 and δ_0 are both nonzero. Condition 2 implies $\varepsilon_1 = \dots = \varepsilon_{w-2} = \delta_1 = \dots = \delta_{w-2} = 0$. Hence $\varepsilon_0 \equiv \delta_0 \mod 2^{w-1}$. Therefore for all coordinates i with $\varepsilon_{i,0} \in \mathbf{unique}(D_{l,u})$ we have $\varepsilon_{i,0} = \delta_{i,0}$.

The digits ε_{w-1} and δ_{w-1} cannot both be zero, otherwise $\varepsilon_0 \equiv \delta_0 \mod 2^w$ which implies $\varepsilon_0 = \delta_0$. Without loss of generality $\varepsilon_{w-1} \neq 0$. Let i be the index described in Condition 3a. Then we have $\varepsilon_{i,0} = \delta_{i,0} \in \mathbf{unique}(D_{l,u})$ and $\varepsilon_{i,w-1} \equiv \delta_{i,w-1} \equiv 1 \mod 2$. Hence, $\delta_{w-1} \neq 0$ either. Condition 2 implies $\varepsilon_w = \dots = \varepsilon_{2w-3} = \delta_w = \dots = \delta_{2w-3} = 0$. Then we have

$$\delta_{w-1} \equiv \varepsilon_{w-1} + \frac{\varepsilon_0 - \delta_0}{2^{w-1}} \mod 2^{w-1}. \tag{9}$$

Since $\varepsilon_0 \neq \delta_0$ there is a coordinate i with $\varepsilon_{i,0} \in \mathbf{upper}(D_{l,u}) \cap \mathbf{nonunique}(D_{l,u})$ and $\delta_{i,0} = \varepsilon_{i,0} - 2^{w-1}$. Then Condition 3c implies $\varepsilon_{i,w-1} \equiv u \mod 2^{w-1}$. With (9) we get $\delta_{i,w-1} \equiv u + 1 \mod 2^{w-1}$. But this is a contradiction of Condition 3b since $\delta_{i,0} \in \mathbf{nonunique}(D_{l,u})$. □

For $d = 1$ we can simplify our digit set $D_{l,u}$: Because of Theorem 15, Condition 1 we do not have any even $\varepsilon_i \neq 0$ in the representation of an integer. Therefore we can assume l and u to be odd, and we can use the digit set $D := \{a \in D_{l,u} \mid a = 0 \vee a \text{ odd}\}$. Now we see that this digit set is slightly more general than the digit set in Section 2.6, since for example $u = 2^w - 3$ and $l = -1$ are also allowed.

This type of joint integer representation generalizes the w-NAF. For $d = 1$, $l = -2^{w-1} + 1$ and $u = 2^{w-1} - 1$ we have the simplified digit set $D = \{0, \pm 1, \ldots, \pm(2^{w-1} - 3), \pm(2^{w-1} - 1)\}$ and Conditions 2 and 3a guarantee that there is only one nonzero digit in every block of length w.

It also generalizes the simple joint sparse form. For $d = 2$, $l = -1$ and $u = 1$ we have $w = 2$. Condition 3a implies, if $x_i = \pm 1$ and $y_i = \pm 1$, then $x_{i+1} = y_{i+1} = 0$. If $|x_i| \neq |y_i|$ and $\varepsilon_{i+1} = 0$, then we have $|x_{i+1}| = |y_{i+1}|$. If $|x_i| \neq |y_i|$ and $\varepsilon_{i+1} \neq 0$, then $\varepsilon_i \neq 0$ and $\varepsilon_{i+1} \neq 0$. Therefore Conditions 3a and 3b and $\varepsilon_i = (0, \pm 1)^T$ (or the other way round) imply $\varepsilon_{i+1} = (\pm 1, \pm 1)^T$, hence $|x_{i+1}| = |y_{i+1}|$.

But the colexicographically minimal integer representation presented in this section does not generalize the joint sparse form, since the joint sparse form is different from the simple joint sparse form and this colexicographically minimal integer representation is unique. Nevertheless the joint sparse form is also colexicographically minimal, since it has all double zeros at the same positions as the simple joint sparse form.

In [11], Heuberger and Muir also derived a limit law for the distribution of $h(n)$. They consider the probability space over all representations of length N with uniform distribution.

Theorem 16. *Let the random variable X_N be the weight of the colexicographically minimal representation over the digit set $D_{l,u}$ of length at most N. Then there are constants $e_{l,u,d}$ and $v_{l,u,d}$ such that the expected value and the variance of X_N are*

$$e_{l,u,d} N + \mathcal{O}(1) \text{ and } v_{l,u,d} N + \mathcal{O}(1).$$

For $d = 1$ we have $e_{l,u,1} = \dfrac{1}{w - 1 + \lambda}$ and $v_{l,u,1} = \dfrac{(3 - \lambda)\lambda}{(w - 1 + \lambda)^3}$, where

$$\lambda = \frac{2(u - l + 1) - (-1)^l - (-1)^u}{2^w}.$$

Furthermore the random variable X_N satisfies the central limit law

$$\lim_{n \to \infty} \mathbb{P}\left(\frac{X_N - e_{l,u,d} N}{\sqrt{v_{l,u,d} N}} \leq x \right) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{t^2}{2}} \, dt.$$

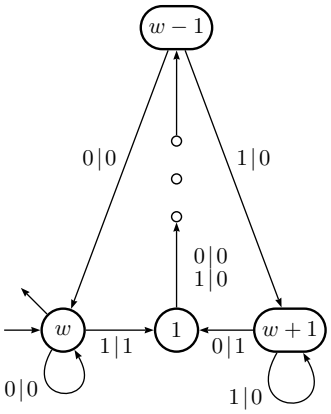

Figure 3: Transducer to compute the Hamming weight of a w-NAF representation

3 Asymptotic distribution of the weight of a w-NAF

In this section we will present the asymptotic distribution of the Hamming weight of the w-NAF representation. Therefore we construct a transducer which computes the Hamming weight of the w-NAF of the input.

Lemma 12. *The transducer in Figure 3 calculates the weight $h(n)$ of the w-NAF of a number n given in binary expansion. Thereby the output is a series of 0 and 1 and the weight $h(n)$ is the number of 1.*

Proof. Let $n = \sum_{j=0}^{L} 2^j n_j$ with $n_j \in \{0,1\}$ be the standard binary expansion of n and $(\varepsilon_k \dots \varepsilon_0)$ be the w-NAF representation of n. If $n \equiv 0 \mod 2$, then $\varepsilon_0 = 0$ and we start the transducer again with input $\frac{n}{2}$. Otherwise $\varepsilon_0 \neq 0$ and the weight $h(\varepsilon_0) = 1$. Since we have a w-NAF representation, the next $w-1$ digits $\varepsilon_1 = \varepsilon_2 = \ldots = \varepsilon_{w-1} = 0$, no matter what the corresponding n_j, $j = 1, \dots, w-1$ are. The sign of the digit ε_0 depends on $n_{w-1} \bmod 2$. If $n_{w-1} \equiv 0 \mod 2$, then $\varepsilon_0 > 0$ and we can just restart the transducer with $\frac{n-\varepsilon_0}{2^w}$. If $n_{w-1} \equiv 1 \mod 2$, then $\varepsilon_0 < 0$ and we therefore have a carry of 1. As long as we read a 1, we pretend to have read a 0, and the carry stays the same. When we read a 0, we pretend to have read a 1 and the carry vanishes. □

We want to compute the average weight of w-NAF representation. Therefore we assume the following probabilistic model on the discrete space $\{n \in \mathbb{Z} \mid 0 \leq n < N\}$, where $N \in \mathbb{Z}$ is a fixed number. The probability measure is the uniform distribution on this space.

Theorem 17. *The weight $h(n)$ of the w-NAF of the integer n is asymptotically normally distributed with mean $\frac{\log_2 N}{w+1} + \mathcal{O}(1)$ and variance $\frac{2}{(w+1)^3}\log_2 N + \mathcal{O}(1)$, that is*

$$\mathbb{P}\left(\frac{h(n) - \frac{\log_2 N}{w+1}}{\sqrt{\frac{2}{(w+1)^3}\log_2 N}} < x\right) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{y^2}{2}}\,dy + \mathcal{O}\left(\frac{1}{\sqrt[4]{\log N}}\right)$$

for all $x \in \mathbb{R}$.

The remainder of this section consists of the proof of Theorem 17.

We calculate the exponential function $f(n) := e^{ith(n)}$. The computation follows along the proof of Theorem 6 in [10]. We define the following matrices M_ε for each digit $\varepsilon \in \{0, 1\}$: the (k, l)-th entry is equal to e^{ith} if reading an ε in state k means writing an h and going to state l, or it is 0 if there is no edge from state k to l with input label ε. So we obtain the two matrices

$$M_0 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ \vdots & \ddots & \ddots & 0 & \vdots & \vdots \\ 0 & \cdots & 0 & 1 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 \\ 0 & \vdots & \ddots & \vdots & 1 & 0 \\ z & 0 & \cdots & 0 & 0 & 0 \end{pmatrix}, \qquad M_1 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ \vdots & \ddots & \ddots & 0 & \vdots & \vdots \\ 0 & \cdots & 0 & 1 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 \\ z & \vdots & \ddots & \vdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 \end{pmatrix}.$$

Thereby the labels of the states in the transducer in Figure 3 correspond to the enumeration of the rows and columns of the matrices and $z = e^{it}$.

Now we can rewrite the function $f(n)$ into

$$f(n) = \vec{v}^T \prod_{l=0}^{L} M_{n_l} \cdot M_0^w \vec{v},$$

where $\vec{v}^T = (0, \ldots, 0, 1, 0)$ is the w-th unit vector and $n = \sum_{l=0}^{L} 2^l n_l$ is the binary expansion of n. The product describes all possible paths from any state to any other state, using edges with input labels corresponding to the input n. The exponent of the entries of the matrix product is the sum of output labels on this paths. Since we are interested in paths starting and ending in state w, we multiply by $\vec{v}^T$ from the left and $\vec{v}$ from the right. The factor M_0^w is due to the fact that we want to stop at state w, but maybe the input n is not of the correct length to stop there. So we just append w zeros at the end of the input, since we need at most w steps reading 0 to get to the state w.

We define $M(n) := \prod_{l=0}^{L} M_{n_l}$. The function $M(n)$ is 2-multiplicative (see, for example [3]), that is

$$M\left(\sum_{l=0}^{L} 2^l n_l\right) = \prod_{l=0}^{L} M(n_l).$$

Furthermore we define the following summatory functions

$$E(N) := \sum_{n<N} e^{ith(n)},$$
$$F(N) := \sum_{n<N} M(n).$$

Next we want to find some recursion relation for $F(N)$. By the 2-multiplicativity we

have

$$\begin{aligned} F(2N) &= \sum_{\varepsilon=0}^{1} \sum_{2n+\varepsilon<2N} M(2n+\varepsilon) \\ &= \sum_{\varepsilon=0}^{1} M_\varepsilon F(N) \end{aligned}$$

and for $N \geq 1$ we have

$$\begin{aligned} F(2N+1) &= \sum_{\varepsilon=0}^{1} \sum_{2n+\varepsilon<2N+1} M(2n+\varepsilon) \\ &= \sum_{\varepsilon=0}^{1} M_\varepsilon \sum_{n<N+\frac{1-\varepsilon}{2}} M(n) \\ &= \sum_{\varepsilon=0}^{1} M_\varepsilon F(N) + M_0 M(N). \end{aligned}$$

With

$$A := M_0 + M_1 = \begin{pmatrix} 0 & 2 & 0 & \cdots & & 0 & 0 \\ 0 & 0 & 2 & \ddots & & \vdots & \vdots \\ \vdots & & \ddots & \ddots & & & \\ 0 & \cdots & & 0 & 2 & 0 & 0 \\ 0 & \cdots & & & & 1 & 1 \\ z & 0 & \cdots & & & 1 & 0 \\ z & 0 & \cdots & & & 0 & 1 \end{pmatrix}$$

we have

$$\begin{aligned} F(2N) &= AF(N), \\ F(2N+1) &= AF(N) + M_0 M(N) \end{aligned}$$

and by iterating we obtain

$$F\left(\sum_{l=0}^{L} 2^l \varepsilon_l\right) = \sum_{l=0}^{L} \varepsilon_l A^l M_0 \prod_{j=l+1}^{L} M_{\varepsilon_j}.$$

Now we want to split this formula into two parts. One part is coming from the dominating eigenvalue. The other part, coming from the other eigenvalues, will be the remainder term. The characteristic polynomial of M_0 and M_1 is $(-1)^{w+1}x^w(x-1)$ and therefore independent of z, which is obtained by Laplace expansion. The eigenvalues of the matrices M_0 and M_1 are therefore 0 and 1. Thus the interesting matrix is A.

w	2	3	4	5	6	7	8	9	10
$\beta(0)$	1	1.41421	1.64512	1.77055	1.84348	1.8886	1.91797	1.9379	1.95189

Table 5: Values of $\beta(0)$, modulus of the second largest eigenvalue of A at $t = 0$ for $w = 2, \ldots, 10$

The underlying graph of the transducer in Figure 3 is strongly connected and the matrix A is positive at $t = 0$. So we can use the theorem of Perron-Frobenius (see, for example, [13]). Since we can reach every node with exactly w steps starting at node w, and we can reach the node w starting from any node with exactly w steps, the adjacency matrix A is primitive at $t = 0$ (since $A^{2w} > 0$). Therefore we know that the dominating eigenvalue of the adjacency matrix A is unique at $t = 0$. Since eigenvalues are continuous, we have $\beta(t) < |\mu(t)|$ around $t = 0$ where $\mu(t)$ is the dominating eigenvalue and $\beta(t)$ is the modulus of the second largest eigenvalue. In other words, $\mu(t)$ is the dominating eigenvalue.

With Laplace expansion we get the characteristic polynomial of A

$$(x-1)(x^w - x^{w-1} - 2^{w-1}e^{it}).$$

At $t = 0$ we have a solution $x = 2$. We have $\mu(0) = 2$, because for $|x| > 2$ we have

$$|x^{w-1} + 2^{w-1}| = |x^{w-1}| \cdot \left|1 + \left(\frac{2}{x}\right)^{w-1}\right| < |x^{w-1}| \cdot 2 < |x^w|$$

by the triangle inequality. Thus there cannot be a greater eigenvalue.

The value of $\beta(0)$ depends on w. In Table 5 the values for $\beta(0)$ for some small w can be found. The Taylor expansion of $\mu(t)$ around $t = 0$ is

$$\mu(t) = 2 + \frac{2i}{w+1}t - \frac{w+3}{(w+1)^3}t^2 + \mathcal{O}(t^3).$$

Let $T^{-1}AT = J$ be a Jordan decomposition of A with the eigenvalue $\mu(t)$ in the top left entry: $J_{1,1} = \mu(t)$. To split the formulas, we define $\Lambda := T\mathrm{diag}(\mu^{-1}, 0, \ldots, 0)T^{-1}$ and $R = T\left(J - \mathrm{diag}(\mu, 0, \ldots, 0)\right)T^{-1}$. Then $A^l = \mu^L \Lambda^{L-l} + R^l$ holds for $l \leq L$. The largest eigenvalue of the matrix R has modulus β. We also define

$$\Lambda(x_0, x_1, \ldots) := \sum_{l=0}^{\infty} x_l \Lambda^l M_0 \prod_{p=0}^{l-1} M_{x_p},$$

$$R(\varepsilon_L, \ldots, \varepsilon_0) := \sum_{l=0}^{L} \varepsilon_l R^l M_0 \prod_{p=l+1}^{L} M_{\varepsilon_p},$$

where $\coprod_{l=a}^{b} m_l = m_b \cdot m_{b-1} \cdot \ldots \cdot m_a$. The function Λ is continuous on the infinite product space $\{0,1\}^{\mathbb{N}_0}$. Altogether we have

$$F\left(\sum_{l=0}^{L} 2^l \varepsilon_l\right) = \mu(t)^L \Lambda(\varepsilon_L, \ldots, \varepsilon_0, 0^\omega) + R(\varepsilon_L, \ldots, \varepsilon_0).$$

We can define R to be a function on $\mathbb{N}$. The function Λ is defined on the infinite product space $\{0,1\}^{\mathbb{N}}$, but we can define Λ on the interval $[0,1)$ by

$$\Lambda\left(\sum_{l\geq 1} \varepsilon_l 2^{-l}\right) = \Lambda(\varepsilon_1, \varepsilon_2, \ldots),$$

where we prefer representations ending on 0^ω to representations ending on 1^ω if there is a choice. Then we get

$$E(N) = \vec{v}^T F(N) M_0^w \vec{v} = \mu^{\log_2 N} \Psi(\log_2 N, t) + \vec{v}^T R(N) M_0^w \vec{v}, \tag{10}$$

with $\Psi(\log_2 N, t) = \mu^{-\{\log_2 N\}} \vec{v}^T \Lambda(2^{\{\log_2 N\}}) M_0^w \vec{v}$.

Since the eigenvalues of M_ε are 0 and 1 and the largest eigenvalue of R has modulus $\beta(t)$, we know the error term

$$\left|\vec{v}^T R(N) M_0^w \vec{v}\right| = \mathcal{O}(N^{\log_2 \beta(t)})$$

and the following expression for $E(N)$

$$E(N) = N^{1+i/((w+1)\log 2)t - 1/\left((w+1)^3 \log 2\right)t^2 + \mathcal{O}(t^3)} \Psi(\log_2 N, t) + \mathcal{O}(N^{\log_2 \beta(t)}). \tag{11}$$

The function $\Psi(x,t)$ is periodic in x with period 1 and is well defined for all $x \in \mathbb{R}^+$. Therefore $\Psi(x,t) = \mathcal{O}(1)$. To prove continuity in x we first note that continuity is obvious for $x \in \mathbb{R}$ with $x = \log_2 y$ where y is not a dyadic rational. To prove it for $x = \log_2 y$ with $y = \sum_{l=1}^{L} \varepsilon_l 2^{-l}$ a dyadic rational, we observe that the two one-sided limits exist. Next we prove that they are the same. Therefore we look at the two sequences $N_k = y2^{L+k+1} + 2^k$ and $\tilde{N}_k = y2^{L+k+1} + 2^k - 1$. Then

$$\lim_{k\to\infty} 2^{\{\log_2 N_k\}} = (.\varepsilon_1 \varepsilon_2 \ldots \varepsilon_L 10^\omega) \text{ and } \lim_{k\to\infty} 2^{\{\log_2 \tilde{N}_k\}} = (.\varepsilon_1 \varepsilon_2 \ldots \varepsilon_L 01^\omega).$$

If we insert these two sequences in (10) we get

$$\mathcal{O}(1) = E(N_k) - E(\tilde{N}_k) = N_k \Psi(\log_2 N_k, t) - \tilde{N}_k \Psi(\log_2 \tilde{N}_k, t) + \mathcal{O}(N_k^{\log_2 \beta(t)}),$$

and hence $\lim_{k\to\infty} \Psi(\{\log_2 N_k\}) = \lim_{k\to\infty} \Psi(\{\log_2 \tilde{N}_k\})$. Therefore $\Psi(x,t)$ is continuous in x.

In t, $\Psi(x,t)$ is also continuous, because the eigenvalues of a matrix are continuous. Furthermore the function $\Psi(x,t)$ is arbitrarily often differentiable in t, because it is dominated by a geometric series.

Furthermore the error term $\vec{v}^T R(N) M_0^w \vec{v}$ is also differentiable with respect to t because it is dominated by a geometric series.

If we insert $t = 0$ in (11), we get the value of $\Psi(\log_2 N, 0) = 1 + \mathcal{O}(N^{\log_2 \beta(0)-1})$.

Differentiating $E(N)$ with respect to t and inserting $t = 0$ produces

$$\sum_{n<N} h(n) = \frac{1}{w+1} N \log_2 N + N\Psi_1(\log_2 N) + \mathcal{O}(N^{\log_2 \beta(0)} \log N),$$

with $\Psi_1(x) = -i\frac{\partial}{\partial t}\Psi(x,t)|_{t=0}$. Differentiating once more produces

$$\begin{aligned}\sum_{n<N} h^2(n) = {} & \frac{1}{(w+1)^2} N \log_2^2 N + \frac{2}{(w+1)^3} N \log_2 N \\ & + \frac{2}{w+1} N \log_2 N \Psi_1(\log_2 N) + N\Psi_2(\log_2 N) + \mathcal{O}(N^{\log_2 \beta(0)} \log^2 N),\end{aligned}$$

with $\Psi_2(x) = -\frac{\partial^2}{\partial t^2}\Psi(x,t)|_{t=0}$. Both functions $\Psi_1(x)$ and $\Psi_2(x)$ are continuous with respect to x and periodic with period 1 due to the same arguments as above. Therefore they are bounded.

From these two equations we can compute the expected value and the variance of the random variable $X = h(n)$. The expected value is

$$\frac{1}{N}\sum_{n<N} h(n) = \frac{1}{w+1}\log_2 N + \Psi_1(\log_2 N) + \mathcal{O}(N^{\log_2 \beta(0)-1} \log N),$$

and the variance is

$$\begin{aligned}\frac{1}{N}\sum_{n<N} h^2(n) - \left(\frac{1}{N}\sum_{n<N} h(n)\right)^2 = {} & \frac{2}{(w+1)^3}\log_2 N - \Psi_1(\log_2 N)^2 \\ & + \Psi_2(\log_2 N) + \mathcal{O}\left(N^{\log_2 \beta(0)-1} \log^2 N\right).\end{aligned}$$

Now we can calculate the characteristic function $\hat{g}_N(t)$ of the standardized random variable

$$Z = \frac{X - \frac{\log_2 N}{w+1}}{\sqrt{\frac{2}{(w+1)^3}\log_2 N}}.$$

With this we can compare the asymptotic distribution of Z with the normal distribution. For this we use Vaaler's version [20] of the Berry-Esseen inequality, Theorem 11.

First we have to compute the characteristic function of Z for $t = o(\sqrt{\log N})$. We do so by multiplying (10) with $\frac{1}{N}\exp\left(-it\frac{\frac{\log_2 N}{w+1}}{\sqrt{\frac{2}{(w+1)^3}\log_2 N}}\right)$ and inserting $\frac{t}{\sqrt{\frac{2}{(w+1)^3}\log_2 N}}$

instead of t:

$$\begin{aligned}\hat{g}_N(t) &= \frac{1}{N}\sum_{n<N}\exp\left(it\frac{h(n)-\frac{\log_2 N}{w+1}}{\sqrt{\frac{2}{(w+1)^3}\log_2 N}}\right)\\ &= e^{-\frac{t^2}{2}}\left(1+\mathcal{O}\left(\frac{t^3}{\log^{3/2} N}\right)\right)\Psi\left(\log_2 N, \frac{t}{\sqrt{\frac{2}{(w+1)^3}\log_2 N}}\right)\\ &\quad + \tilde{R}\left(N, \frac{t}{\sqrt{\frac{2}{(w+1)^3}\log_2 N}}\right)\exp\left(-it\sqrt{\frac{(w+1)\log_2 N}{2}}\right),\end{aligned}$$

where $\tilde{R}(N,t) = \frac{1}{N}\vec{v}^T R(N) M_0^w \vec{v}$. Since $\hat{g}_N(t)$ is a characteristic function, we have

$$1 = \hat{g}_N(0) = \psi_0 + r_0 \tag{12}$$

with $\psi_0 := \Psi(\log_2 N, 0)$ and $r_0 := \tilde{R}(N,0)$. We know that $r_0 = \mathcal{O}(N^{-\varepsilon})$ for some $\varepsilon > 0$, since $\log_2 \beta(0) < 1$. Furthermore we know that $\tilde{R}(N,t) = \mathcal{O}(N^{-\varepsilon})$, since there is a dominating eigenvalue around $t = 0$.

We also know that the characteristic function of the normal distribution with mean 0 and variance 1 is $\hat{f}(t) := \exp(-\frac{t^2}{2})$. Now we look at the difference between these two characteristic functions

$$\begin{aligned}\left|\hat{g}_N(t) - \hat{f}(t)\right| &= \left|e^{-\frac{t^2}{2}}\left(1+\mathcal{O}\left(\frac{t^3}{\log^{3/2} N}\right)\right)\left(\psi_0+\mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right)\right)\right.\\ &\quad \left. + \tilde{R}\left(N, \frac{t}{\sqrt{\frac{2}{(w+1)^3}\log_2 N}}\right)\exp\left(-it\sqrt{\frac{(w+1)\log_2 N}{2}}\right) - (\psi_0+r_0)e^{-\frac{t^2}{2}}\right|\\ &\leq e^{-\frac{t^2}{2}}\mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right) + \left|\left(r_0+\mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right)\right)\left(1+\mathcal{O}\left(t\sqrt{\log N}\right)\right) - r_0\right|\\ &\leq \mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right) + \mathcal{O}\left(\frac{t\sqrt{\log N}}{N^{\varepsilon}}\right) + \mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right) = \mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right),\end{aligned} \tag{13}$$

where we used (12), $e^{-\frac{t^2}{2}} \leq 1$ and $\tilde{R}(N,t) = r_0 + \mathcal{O}(t) = \mathcal{O}(N^{-\varepsilon})$ for some $\varepsilon > 0$ around $t = 0$. Let

$$g_N(x) := \mathbb{P}\left(\frac{h(n)-\frac{\log_2 N}{w+1}}{\sqrt{\frac{2}{(w+1)^3}\log_2 N}} < x\right)$$

be the distribution function of the variable Z and $f(x) := \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{y^2}{2}}dy$ be the normal distribution with mean 0 and variance 1. Since the density function of the normal

distribution is bounded from above by 1, the Berry-Esseen inequality, Theorem 11, implies

$$|g_N(x) - f(x)| \leq \int_{-T}^{T} \hat{J}(T^{-1}t)\frac{1}{2\pi t}\left|\hat{g}_N(t) - e^{-\frac{t^2}{2}}\right| dt + \frac{1}{2T}\left(1 + \int_{-T}^{T} \hat{K}(T^{-1}t)\left(\hat{g}_N(t) - e^{-\frac{t^2}{2}}\right) dt\right).$$

Now we estimate the integral on the right hand side with $\hat{J}(t) \leq 1$, $e^{-\frac{t^2}{2}} \leq 1$ and (13):

$$\begin{aligned}|g_N(x) - f(x)| &\leq \int_{-T}^{T} \frac{1}{t}\mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right) dt + \frac{1}{2T}\left(1 + \int_{-T}^{T}\left(1 - \left|T^{-1}t\right|\right)\mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right) dt\right) \\ &\leq \mathcal{O}\left(\frac{T}{\sqrt{\log N}}\right) + \frac{1}{2T}.\end{aligned}$$

If we choose $T = \sqrt[4]{\log N}$, we get

$$\mathbb{P}\left(\frac{h(n) - \frac{\log_2 N}{w+1}}{\sqrt{\frac{2}{(w+1)^3}\log_2 N}} < x\right) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{y^2}{2}} dy + \mathcal{O}\left(\frac{1}{\sqrt[4]{\log N}}\right).$$

4 Asymmetric digit sets in dimension one

In this section we investigate minimal weight representations with asymmetric digit sets. We will give the asymptotic distribution of the minimal Hamming weight. In [11], Heuberger and Muir present an algorithm for constructing d-dimensional joint representations in base 2 with minimal weight. First we will restrict ourselves to $d = 1$. Later in Section 5 we will describe how to generalize the result to arbitrary d.

For definitions see Section 2.7. Algorithm 3 in [11] computes a minimal weight joint representation for arbitrary d. In the case of $d = 1$, some simplifications can be done. First of all, even digits, except 0 will not be used, therefore u is assumed to be odd. As we only want to compute the weight of the representation, we sometimes can abbreviate the while loop. This leads to Algorithm 8.

Algorithm 8 Algorithm to compute the minimal weight of a representation over the digit set $D_{l,u}$ in dimension one

Input: Integers n, $l \leq 0$, $u > 0$, u odd, $n \geq 0$ if $l = 0$
Output: Minimal weight of a representation of n with digits in $D_{l,u}$

1: $h = 0$
2: **while** $n \neq 0$ **do**
3: **if** $n \equiv 0 \mod 2$ **then**
4: $a = 0$
5: $h = h + 0$
6: $m = \frac{n}{2}$
7: **else**
8: $a = l + ((n - l) \bmod 2^{w-1})$
9: $h = h + 1$
10: $m = \frac{n-a}{2^{w-1}}$
11: **if** $m \equiv 1 \mod 2$ **and** $(n - l) \bmod 2^{w-1} \leq u - l - 2^{w-1}$ **then**
12: $a = a + 2^{w-1}$
13: $m = m - 1$
14: **end if**
15: **end if**
16: $n = m$
17: **end while**
18: **return** h

Now we want to construct a transducer, doing the same calculation as Algorithm 8. It will look similar to the transducer in Figure 3. We start at some state 0. Then there is some vertical block of states $(0, 0)_i$, $(1, 0)_i$, $(0, 1)_i$ and $(1, 1)_i$ for $i = 1, \ldots, w-1$. After this block we either go back to state 0, or to a similar state 1, or again to the block of states (see Figure 5). We call the states 0 and 1 the beginning states. Their labels signify the carry which is to be processed. The state 0 is also the final state.

The block of states corresponds to the **if** statement in line 11 in Algorithm 8. In this

line we have to check the inequality $(n-l) \bmod 2^{w-1} \leq u-l-2^{w-1}$. A first step to this aim is to compare $n+\tilde{l} \leq \tilde{u}$ with $\tilde{l} := -l$ and $\tilde{u} := u-l-2^{w-1}$. Therefore we use Automaton 4 in Figure 4.

This automaton takes three binary expansions of the integers a, b and c as input. It answers the question whether $a+b \leq c$ is true. The states are (s,t) with $s, t \in \{0,1\}$. The label s signifies the carry of the addition $a+b$ which still has to be processed. The label t corresponds to the truth value of the expression $(a+b) \bmod 2^i > c \bmod 2^i$ where i is the number of read digits up to now. So the automaton accepts the input if it stops in state $(0,0)$ where there is no carry anymore and $a+b>c$ is false. The initial state is $(0,0)$.

Therefore there is a path from $(0,0)$ to (s,t) in Automaton 4 with input label

$$\begin{pmatrix} \alpha_{i-1} \dots \alpha_0 \\ \beta_{i-1} \dots \beta_0 \\ \gamma_{i-1} \dots \gamma_0 \end{pmatrix}$$

if and only if

$$s = \left\lfloor \frac{(\alpha_{i-1} \dots \alpha_0) + (\beta_{i-1} \dots \beta_0)}{2^i} \right\rfloor$$

and $t = [((\alpha_{i-1} \dots \alpha_0) + (\beta_{i-1} \dots \beta_0)) \bmod 2^i > (\gamma_{i-1} \dots \gamma_0)]$. Here we use Iverson's notation, that is $[\textit{expression}]$ is 1 if *expression* is true and 0 else. From this the rules for the transitions follow. There is a transition $(s,t) \xrightarrow{(\alpha,\beta,\gamma)^T} (s',t')$ if and only if $s' = \left\lfloor \frac{\alpha+\beta+s}{2} \right\rfloor$ and $t' = [(\alpha+\beta+s) \bmod 2 > \gamma - t]$.

Next we examine the binary expansions of $\tilde{u}$ and $\tilde{l}$. Since we have assumed that $l > -2^{w-1}$, we know that the length of the binary expansion of $\tilde{l}$ is at most $w-1$. Furthermore $-1 \leq \tilde{u} < 2^{w-1}$. In the case $\tilde{u} = -1$ the set $\mathbf{nonunique}(D_{l,u})$ is empty and we have no choices for the digits, therefore we will neglect this case. Then the length of the binary expansion of $\tilde{u}$ is at most $w-1$. Let $\tilde{l} = (l_{w-2} \dots l_0)$ and $\tilde{u} = (u_{w-2} \dots u_0)$ be the binary expansions.

Now we can verify $n + \tilde{l} \bmod 2^{w-1} \leq (u_{w-2} \dots u_0)$ by checking the label t of the state (s,t) after reading $w-1$ digits from the binary expansion of $(n, \tilde{l}, \tilde{u})^T$ in Automaton 4. If $t=0$, then the inequality is true, otherwise it is false. Since the length of $\tilde{u}$ is less than or equal $w-1$ there are no digits of $\tilde{u}$ left. Only a possible carry of the addition $n+\tilde{l}$ is left. This carry is the label s of the current state (s,t). Therefore in fact we have checked $n+\tilde{l} \bmod 2^{w-1} \leq \tilde{u}$. To ensure that we read exactly $w-1$ digits, the transducer in Figure 5 has $w-1$ copies of the four states of Automaton 4. The transitions start in a state of the i-th copy and go to an appropriate state of the $(i+1)$-th copy while reading the i-th digit of the expansion.

In the **if** statement in line 11 in Algorithm 8 we have to check the other condition $m \equiv 1 \mod 2$ too. Let $(s,t)_{w-1}$ be the current state at the end of the block of states. We know that $m = \frac{(n+\tilde{l}) - (n+\tilde{l}) \bmod 2^{w-1}}{2^{w-1}}$, therefore the least significant digit of m is simply the next digit of the addition $n+\tilde{l}$. Since there are no digits of the expansion of $\tilde{l}$ left, we only have to look at the next digit of n and consider the carry s. Thus we have $m \equiv s + \varepsilon \mod 2$.

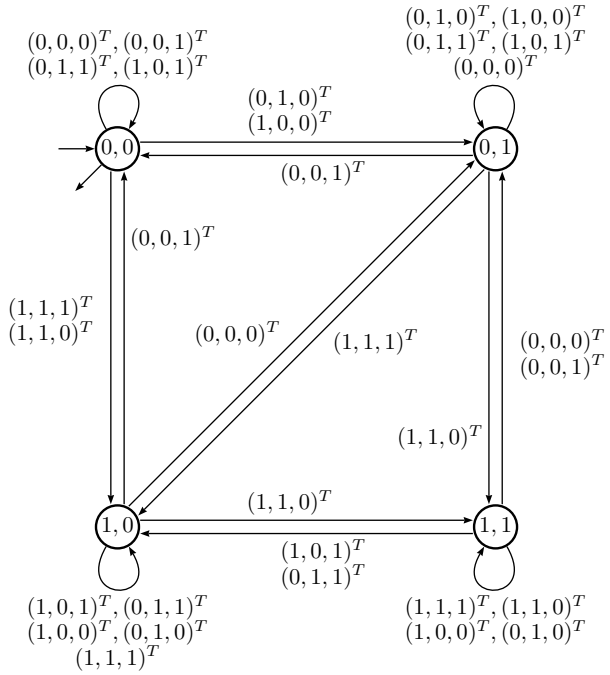

Figure 4: Automaton 4 to compare three integers a, b and c, if $a + b \leq c$

If the inequality of the **if** statement is satisfied, that is if $t = 0$, then whatever digit ε we read next, the transducer starts from a beginning state again. If m is even, then the next written digit is a zero anyway. If m is odd, we can change the digit in the representation (because it is non-unique) and m becomes even too. We only have to remember the carry. If $s = 0$ or $s = 1$ and we read $\varepsilon = 0$ then there will be no carry propagation and we start at state 0. If $s = 1$ and we read $\varepsilon = 1$, then there is a carry propagation and we start at state 1.

If the inequality is not satisfied, that is if $t = 1$ and $m \equiv s + \varepsilon \mod 2$ is not even, then we have to start with the $w - 1$ transitions of Automaton 4 immediately. If m is even however, then the transducer starts from a beginning state again. In both cases we have to consider the carry propagation as well.

At state $s \in \{0, 1\}$ we stay in state s as long as we read s. If we read $1 - s$ we start with the $w - 1$ transitions of Automaton 4.

To summarize we have the following transitions in the transducer in Figure 5 for s, s', t, t', $\varepsilon \in \{0, 1\}$ and $i \in \{1, \ldots, w - 2\}$:

- $s \xrightarrow{\varepsilon|0} s$ if $s = \varepsilon$.
- $s \xrightarrow{\varepsilon|1} (s', t')_1$ if $s = \varepsilon$ and $(s, 0) \xrightarrow{(\varepsilon, l_0, u_0)^T} (s', t')$ is a transition in Automaton 4.
- $(s, t)_i \xrightarrow{\varepsilon|0} (s', t')_{i+1}$ if $(s, t) \xrightarrow{(\varepsilon, l_i, u_i)^T} (s', t')$ is a transition in Automaton 4.
- $(s, t)_{w-1} \xrightarrow{\varepsilon|0} s'$ if $t = 0$ or $\varepsilon + s \equiv 0 \mod 2$, and $s' = \lfloor \frac{\varepsilon + s}{2} \rfloor$.

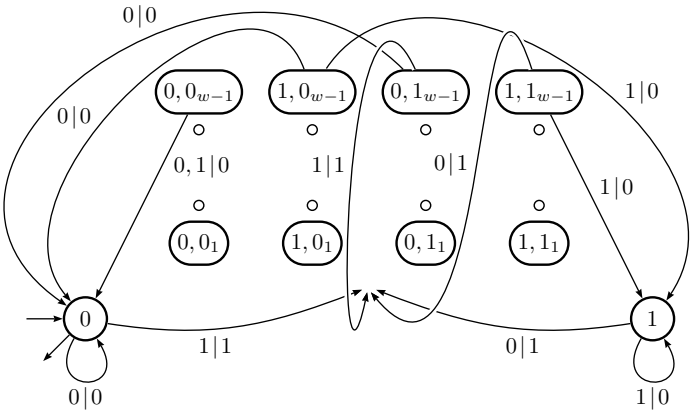

Figure 5: Transducer to compute the weight of a minimal weight representation with digits in $D_{l,u}$

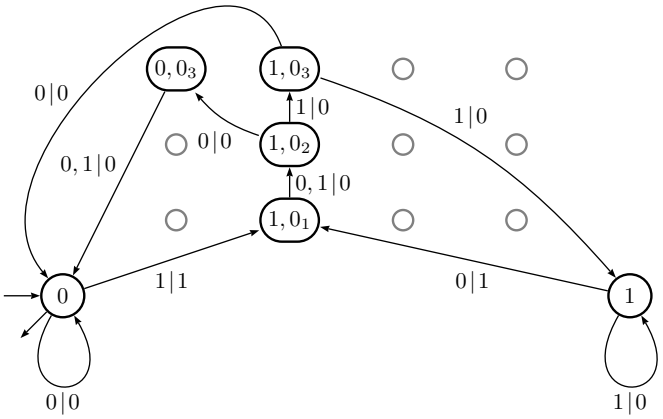

Figure 6: Transducer to compute the weight of a minimal weight representation with digits in $D_{-3,11}$

- $(s,t)_{w-1} \xrightarrow{\varepsilon|1} (s',t')_1$ if $t = 1$, $\varepsilon + s \equiv 1 \mod 2$ and $(s,0) \xrightarrow{(\varepsilon,l_0,u_0)^T} (s',t')$ is a transition in Automaton 4.

We note that there is only one accessible state in the first row, because the transitions $0 \xrightarrow{1|1} (s,t)_1$ and $1 \xrightarrow{0|1} (s,t)_1$ have both the same target state. This target state depends on l and u.

Example 4.1. *For $l = -3$ and $u = 11$ we have $w = 4$, $\tilde{l} = (011)_2$ and $\tilde{u} = u - l - 2^{w-1} = (110)_2$. The transducer can be seen in Figure 6, where all non-accessible states are gray.*

The next theorem states the properties of the distribution of the weight.

Theorem 18. *The weight $h(n)$ of the colexicographically minimal and minimal weight representation over the digit set $D_{l,u}$ of the integer n is asymptotically normally distributed*

with mean

$$\frac{1}{w+\frac{c_n}{2^{w-1}}}\log_2 N+\mathcal{O}(1)$$

and variance

$$\frac{(2-\frac{c_n}{2^{w-1}})(1+\frac{c_n}{2^{w-1}})}{(w+\frac{c_n}{2^{w-1}})^3}\log_2 N+\mathcal{O}(1),$$

that is

$$\mathbb{P}\left(\frac{h(n)-\frac{1}{w+\frac{c_n}{2^{w-1}}}\log_2 N}{\sqrt{\frac{(2-\frac{c_n}{2^{w-1}})(1+\frac{c_n}{2^{w-1}})}{(w+\frac{c_n}{2^{w-1}})^3}\log_2 N}}<x\right)=\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x}e^{-\frac{y^2}{2}}dy+\mathcal{O}\left(\frac{1}{\sqrt[4]{\log N}}\right)$$

for all $x\in\mathbb{R}$ *with*

$$c_n=|\{a\in\textbf{nonunique}(D_{l,u})\mid a\equiv 1 \mod 2\}|.$$

The remainder of this section consists of the proof of Theorem 18. The proof is similar to the proof of Theorem 17. We define matrices M_ε for $\varepsilon\in\{0,1\}$ for the transducer in Figure 5 as above. The order of the states is $(0,0)_1$, $(1,0)_1$, $(0,\bar{1})_1$, $(1,\bar{1})_1,\ldots,(0,0)_{w-1}$, $(1,0)_{w-1}$, $(0,\bar{1})_{w-1}$, $(1,\bar{1})_{w-1}$, 0, 1.

$$M_0=\begin{pmatrix}\mathbf{0} & M_{(0,l_1,u_1)} & \mathbf{0} & \mathbf{0} & 0 & 0\\ \vdots & \ddots & \ddots & \mathbf{0} & \vdots & \vdots\\ \mathbf{0} & \cdots & \mathbf{0} & M_{(0,l_{w-2},u_{w-2})} & 0 & 0\\ \vec{0} & \vec{0} & \cdots & \vec{0} & 1 & 0\\ \vec{0} & \vdots & \ddots & \vdots & 1 & 0\\ \vec{0} & & & & 1 & 0\\ z\vec{a} & & & & 0 & 0\\ \vec{0} & & & & 1 & 0\\ z\vec{a} & \vec{0} & \cdots & \vec{0} & 0 & 0\end{pmatrix},$$

$$M_1=\begin{pmatrix}\mathbf{0} & M_{(1,l_1,u_1)} & \mathbf{0} & \mathbf{0} & 0 & 0\\ \vdots & \ddots & \ddots & \mathbf{0} & \vdots & \vdots\\ \mathbf{0} & \cdots & \mathbf{0} & M_{(1,l_{w-2},u_{w-2})} & 0 & 0\\ \vec{0} & \vec{0} & \cdots & \vec{0} & 1 & 0\\ \vec{0} & \vdots & \ddots & \vdots & 0 & 1\\ z\vec{a} & & & & 0 & 0\\ \vec{0} & & & & 0 & 1\\ z\vec{a} & & & & 0 & 0\\ \vec{0} & \vec{0} & \cdots & \vec{0} & 0 & 1\end{pmatrix}.$$

Here $\mathbf{0}$ is a 4×4 zero matrix, $\vec{0}=(0,0,0,0)$, $\vec{a}$ is the first row vector of the matrix $M_{(1,l_0,u_0)}$ and $z=e^{it}$. The matrices $M_{(\gamma,\delta,\varepsilon)}$, with γ, δ, $\varepsilon\in\{0,1\}$, are the matrices of Automaton 4:

$$M_{(0,0,0)}=\begin{pmatrix}1&0&0&0\\0&0&1&0\\0&0&1&0\\0&0&1&0\end{pmatrix},\ M_{(0,0,1)}=\begin{pmatrix}1&0&0&0\\1&0&0&0\\1&0&0&0\\0&0&1&0\end{pmatrix},$$

$$M_{(0,1,0)}=M_{(1,0,0)}=\begin{pmatrix}0&0&1&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix},$$

$$M_{(1,1,0)}=\begin{pmatrix}0&1&0&0\\0&0&0&1\\0&0&0&1\\0&0&0&1\end{pmatrix},\ M_{(1,1,1)}=\begin{pmatrix}0&1&0&0\\0&1&0&0\\0&1&0&0\\0&0&0&1\end{pmatrix},$$

$$M_{(0,1,1)}=M_{(1,0,1)}=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&1&0&0\end{pmatrix}.$$

Example 4.2. *The matrix $A=M_0+M_1$, with the non-accessible states, in Example 4.1 is*

$$A=\begin{pmatrix}
0&0&0&0&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&2&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&2&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&1&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&2&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1\\
0&z&0&0&0&0&0&0&0&0&0&0&1&0\\
0&z&0&0&0&0&0&0&0&0&0&0&0&1\\
0&z&0&0&0&0&0&0&0&0&0&0&1&0\\
0&z&0&0&0&0&0&0&0&0&0&0&0&1
\end{pmatrix}.$$

Its characteristic polynomial is $(x-1)x^9(x^4-x^3-8z)$ and its eigenvalues at $t=0$ are 2, $0.239\pm 1.628i$, -1.478, 1 and 0 with algebraic multiplicity 9. The Taylor expansion of the largest eigenvalue around $t=0$ is

$$\mu(t)=2+\frac{2i}{5}t-\frac{7}{125}t^2+\mathcal{O}(t^3).$$

Next we define the functions $f(n)$, $F(N)$, $E(N)$ and $M(N)$ as above. Then we have

$$f(n) = \vec{v}^T \prod_{l=0}^{L} M_{n_l} \cdot M_0^{2w} \vec{v}.$$

The factor M_0^{2w} is due to the fact that for some u and l we have to go through the block twice, to finish the computation at state 0. We have the same recursion formula for $F(N)$

$$\begin{aligned} F(2N) &= AF(N), \\ F(2N+1) &= AF(N) + M_0 M(N), \end{aligned}$$

and the following explicit formula for $F(N)$

$$F\left(\sum_{l=0}^{L} 2^l \varepsilon_l\right) = \sum_{l=0}^{L} \varepsilon_l A^l M_0 \prod_{j=l+1}^{L} M_{\varepsilon_j}.$$

Now we want to investigate the eigenvalues of the matrices M_ε for $\varepsilon \in \{0,1\}$ and $A = M_0 + M_1$. First we use the theorem of Geršgorin (see [8]), to get an upper bound for the eigenvalues. For the matrices M_ε the union of the Geršgorin discs is the closed disc $\overline{B(0,1)}$, with center 0 and radius 1. Therefore all eigenvalues have modulus less than or equal 1. For the matrix A the union of the Geršgorin discs is $\overline{B(0,2)}$. Hence all eigenvalues have modulus less than or equal 2. Since $\vec{x} = (1,\ldots,1)^T$ is a solution to the equation $(A - 2I)\vec{x} = 0$ at $t = 0$, the matrix A has an eigenvalue 2 at $t = 0$.

If we want to apply the theorem of Perron-Frobenius (see, for example, [13]), we have to get rid of the non-accessible states. A state is called non-accessible if there is no path from the initial state 0 to this state. Suppose we have already done this, then we just look at the accessible states with the corresponding matrix $\tilde{A}$. The underlying graph of these states is strongly connected and we can get from any node to any other node with exactly $4w$ steps. Hence $\tilde{A}^{4w} > 0$ at $t = 0$ and $\tilde{A}$ is a primitive matrix at $t = 0$. By the theorem of Perron-Frobenius and the continuity of eigenvalues the dominating eigenvalue of $\tilde{A}$ is unique around $t = 0$.

What happens with the other states? We just rewrite the matrix A with a permutation of the rows and columns. First there should be the accessible states with the same order as before, and then there should be the non-accessible states with the same order as before as well. Then the permutation of the matrix A looks as follows

$$\begin{pmatrix} \tilde{A} & \mathbf{0} \\ * & D \end{pmatrix},$$

where $*$ is any matrix and D is a strictly upper triangular matrix if state 1 is accessible or D is an upper triangular matrix with zeros on the diagonal except for one entry one if state 1 is not accessible. The zero matrix follows from the fact that a non-accessible state cannot be a direct successor of an accessible state, otherwise it would be accessible.

The matrix D is an upper triangular matrix because inside the block of states there are only transitions from one level to the next, there is only a loop at state 1. Furthermore transitions from the last row either go to a beginning state, then the entry is above the diagonal of D, or go to the only accessible state in the first row, then the entry is in $*$. Thus D is an upper triangular matrix. Therefore the characteristic polynomial of A is the same as the characteristic polynomial of $\tilde{A}$ except for a factor $(-x)^k$ or $(-x)^{k-1}(1-x)$, where k is the number of non-accessible states. Hence the dominating eigenvalue of A is unique around $t = 0$ and the modulus of the second largest eigenvalue $\beta(t) < |\mu(t)|$. Therefore the dominating eigenvalue at $t = 0$ is $\mu(0) = 2$.

With the same calculations and definitions as above we have

$$E(N) = \mu^{\log_2 N}\Psi(\log_2 N, t) + \vec{v}^T R(N) M_0^{2w}\vec{v},$$

with $\Psi(\log_2 Nt) = \mu^{-\{\log_2 N\}}\vec{v}^T\Lambda(2^{\{\log_2 N\}})M_0^{2w}\vec{v}$ and $\mu(t) = 2+\mathcal{O}(t)$ around $t = 0$. Because the spectral radius of M_ε is 1 and the largest eigenvalue of R has modulus $\beta(t)$, we have

$$\left|\vec{v}^T R(N) M_0^{2w}\vec{v}\right| = \mathcal{O}\left(N^{\log_2 \beta(t)}\right).$$

So we get

$$E(N) = N^{1+a_1 t+a_2 t^2+\mathcal{O}(t^3)}\Psi(\log_2 N, t) + \mathcal{O}\left(N^{\log_2 \beta(t)}\right)$$

for some a_1, $a_2 \in \mathbb{C}$. By differentiating $E(N)$ with respect to t and inserting $t = 0$, we get the expected value of the random variable $X = h(n)$

$$\frac{1}{N}\sum_{n<N} h(n) = b_1 \log_2 N + \Psi_1(\log_2 N) + \mathcal{O}(N^{\log_2 \beta(0)-1}\log N) \tag{14}$$

with $b_1 = -ia_1 \log 2$. Differentiating another time gives

$$\begin{aligned}\frac{1}{N}\sum_{n<N} h^2(n) = {} & b_2 \log_2 N + b_1^2 \log_2^2 N + 2b_1 \log_2 N \Psi_1(\log_2 N) + \Psi_2(\log_2 N) \\ & + \mathcal{O}(N^{\log_2 \beta(0)-1}\log^2 N)\end{aligned}$$

with $b_2 = -2a_2 \log 2$. Hence, the variance is

$$\begin{aligned}\frac{1}{N}\sum_{n<N} h^2(n) - \left(\frac{1}{N}\sum_{n<N} h(n)\right)^2 = {} & b_2 \log_2 N - \Psi_1^2(\log_2 N) + \Psi_2(\log_2 N) \\ & + \mathcal{O}(N^{\log_2 \beta(0)-1}\log^2 N).\end{aligned} \tag{15}$$

To determine the values b_1 and b_2, we compute the expected value and the variance in the case of $N = 2^k$ like Heuberger and Prodinger in [12]. Therefore we use the probability generating function of the random variable $X_k = h(n)$ with n an integer in $[0, 2^k)$. We define the probability generating function of the random variable X_k

$$G_k(Y) := \frac{1}{2^k}\sum_{n<2^k} Y^{h(n)}.$$

Then we have the following recursion for $G_k(Y)$ for $k \geq w$.

$$\begin{aligned}
G_k(Y) &= \frac{1}{2^k} \sum_{\substack{n \equiv 0 \bmod 2 \\ n < 2^k}} Y^{h(\frac{n}{2})} + \frac{1}{2^k} \sum_{\substack{a \in \mathbf{unique}(D_{l,u}) \\ a \equiv 1 \bmod 2}} \sum_{\substack{n \equiv a \bmod 2^{w-1} \\ n < 2^k}} Y^{1+h(\frac{n-a}{2^{w-1}})} \\
&\quad + \frac{1}{2^k} \sum_{\substack{a \in \mathbf{nonunique}(D_{l,u}) \\ a \equiv 1 \bmod 2}} \sum_{\substack{n \equiv a \bmod 2^{w} \\ n < 2^k}} Y^{1+h(\frac{n-a}{2^{w}})} \\
&= \frac{1}{2^k} \sum_{m < 2^{k-1}} Y^{h(m)} + \frac{1}{2^k} \sum_{\substack{a \in \mathbf{unique}(D_{l,u}) \\ a \equiv 1 \bmod 2}} \left(Y \sum_{m < 2^{k-w+1}} Y^{h(m)} + H_a(Y) \right) \\
&\quad + \frac{1}{2^k} \sum_{\substack{a \in \mathbf{nonunique}(D_{l,u}) \\ a \equiv 1 \bmod 2}} \left(Y \sum_{m < 2^{k-w}} Y^{h(m)} + H_a(Y) \right) \\
&= \frac{1}{2} G_{k-1}(Y) + \frac{c_u}{2^{w-1}} Y G_{k-w+1}(Y) + \frac{c_n}{2^{k-w}} Y G_{k-w}(Y) + \frac{1}{2^k} \tilde{H}(Y),
\end{aligned}$$

where

$$\begin{aligned}
H_a(Y) &:= \begin{cases} Y \displaystyle\sum_{-\frac{a}{2^{w-b}} \leq m < 0} \left(Y^{h(m)} - Y^{h(2^{k-w+b}+m)} \right) & \text{if } a > 0, \\ Y \displaystyle\sum_{0 \leq m < -\frac{a}{2^{w-b}}} \left(-Y^{h(m)} + Y^{h(2^{k-w+b}+m)} \right) & \text{if } a < 0, \end{cases} \\
b &:= [a \in \mathbf{unique}(D_{l,u})], \\
c_u &:= |\{a \in \mathbf{unique}(D_{l,u}) \mid a \equiv 1 \mod 2\}|, \\
c_n &:= |\{a \in \mathbf{nonunique}(D_{l,u}) \mid a \equiv 1 \mod 2\}|, \\
\tilde{H}(Y) &:= \sum_{\substack{a \in D_{l,u} \\ a \equiv 1 \bmod 2}} H_a(Y).
\end{aligned}$$

At first look, $H_a(Y)$ depends on k. But since the range of summation is quite small we can prove that it is in fact independent of k. For $a \in \mathbf{nonunique}(D_{l,u})$ we have $|a| < 2^w$ since $-l$, $u < 2^w$. Therefore we have $\left|\frac{a}{2^w}\right| < 1$. For $a \in \mathbf{unique}(D_{l,u})$ we have $u - 2^{w-1} < a < l + 2^{w-1}$ and therefore $\left|\frac{a}{2^{w-1}}\right| < 1$. Hence we know that $H_a(Y) = 0$ if $a > 0$ or $H_a(Y) = Y(Y-1)$ if $a < 0$ and thus independent of k.

If we define $G(Y, Z) := \sum_{k=0}^{\infty} Z^k G_k(Y)$ then we get

$$G(Y,Z) = \frac{1}{2} Z G(Y,Z) + \frac{c_u}{2^{w-1}} Y Z^{w-1} G(Y,Z) + \frac{c_n}{2^w} Y Z^w G(Y,Z) + \frac{1}{1 - \frac{Z}{2}} \tilde{H}(Y) + H(Y,Z),$$

where

$$H(Y,Z) = -\frac{1}{2} Z \sum_{k=0}^{w-2} G_k(Y) Z^k - \frac{c_u}{2^{w-1}} Y Z^{w-1} G_0(Y) + \sum_{k=0}^{w-1} G_k(Y) Z^k$$

is a polynomial contributed by $G_k(Y)$ for $k < w$ where the recursion for $G_k(Y)$ is not true. Thus we have

$$G(Y,Z) = \frac{\tilde{H}(Y) + H(Y,Z)(1-\frac{Z}{2})}{(1-\frac{Z}{2})(1-\frac{1}{2}Z - \frac{c_u}{2^{w-1}}YZ^{w-1} - \frac{c_n}{2^w}YZ^w)}.$$

Now we are interested in the coefficient of Z^k in $\frac{\partial}{\partial Y}G(Y,Z)|_{Y=1} = G_Y(1,Z)$ which is the expected value of the random variable X_k. To do so we follow the computation of the mean and the variance in [11]. Therefore we first investigate the occurring functions at $Y = 1$. We have $G_k(1) = 1$ for all $k \geq 0$ and thus $G(1,Z) = \frac{1}{1-Z}$. Furthermore we have $H_a(1) = 0$ and therefore $\tilde{H}(1) = 0$.

Let $f(Y,Z)$ and $g(Y,Z)$ be polynomials so that

$$G(Y,Z) = \frac{f(Y,Z)}{(1-\frac{Z}{2})g(Y,Z)}$$

follows. Then we have $(1-\frac{Z}{2})g(1,Z) = (1-Z)f(1,Z) = (1-Z)(1-\frac{Z}{2})H(1,Z)$ and consequently $g(1,Z) = (1-Z)H(1,Z)$ and $f(1,Z) = (1-\frac{Z}{2})H(1,Z)$. Hence

$$0 = g(1,1) = \frac{1}{2} - \frac{c_u}{2^{w-1}} - \frac{c_n}{2^w}. \tag{16}$$

Hence we have

$$\begin{aligned} G_Y(1,Z) &= \frac{f_Y(1,Z)}{(1-\frac{Z}{2})g(1,Z)} - \frac{f(1,Z)g_Y(1,Z)(1-\frac{Z}{2})}{(1-\frac{Z}{2})^2 g^2(1,Z)} \\ &= \frac{f_Y(1,Z)}{(1-\frac{Z}{2})(1-Z)H(1,Z)} - \frac{H(1,Z)(1-\frac{Z}{2})^2 g_Y(1,Z)}{(1-\frac{Z}{2})^2(1-Z)^2H^2(1,Z)} \\ &= \frac{f_Y(1,Z)}{(1-\frac{Z}{2})(1-Z)H(1,Z)} - \frac{g_Y(1,Z)}{(1-Z)^2H(1,Z)}. \end{aligned}$$

The function $G(1,Z)$, and therefore $G_Y(1,Z)$, has a pole at $Z = 1$. For $|Z| \leq 1$ and $Z \neq 1$ it is analytic, because by the triangle inequality and (16) we have

$$\left|\frac{1}{2}Z + \frac{c_u}{2^{w-1}}Z^{w-1} + \frac{c_n}{2^w}Z^w\right| \leq 1.$$

Equality only holds if $\frac{c_u}{2^{w-1}}Z^{w-1}$ and $\frac{c_n}{2^w}Z^w$ are nonnegative real multiples of $\frac{1}{2}Z$, because of the triangle inequality, and $|Z| = 1$. Thus $g(1,Z) = 0$ for $|Z| \leq 1$ only if $Z = 1$. As a result the main term of the coefficient of Z^k is contributed by the pole $Z = 1$. We therefore compute the Laurent series at $Z = 1$. So we get

$$\begin{aligned} G_Y(1,Z) = \frac{1}{(1-Z)^2}\left(-\frac{g_Y(1,1)}{H(1,1)}\right) + \frac{1}{1-Z}\left(\frac{2f_Y(1,1)}{H(1,1)} + \frac{g_{YZ}(1,1)}{H(1,1)} - \frac{g_Y(1,1)H_Z(1,1)}{H^2(1,1)}\right) \\ + \text{ power series in } (Z-1). \end{aligned}$$

Hence the expected value is

$$\begin{aligned}\mathbb{E}X_k &= [Z^k]G_Y(1,Z)\\ &= (k+1)\left(-\frac{g_Y(1,1)}{H(1,1)}\right)+\left(\frac{2f_Y(1,1)}{H(1,1)}+\frac{g_{YZ}(1,1)}{H(1,1)}-\frac{g_Y(1,1)H_Z(1,1)}{H^2(1,1)}\right)+\mathcal{O}(\varepsilon^k)\\ &= k\left(-\frac{g_Y(1,1)}{H(1,1)}\right)+\mathcal{O}(\varepsilon^k)\end{aligned} \tag{17}$$

for an $0<\varepsilon<1$, since $G_Y(1,Z)$ has no other poles in $|Z|\le 1$.

We have $H(1,1)=\frac{w}{2}+\frac{c_n}{2^w}$ and $g_Y(1,1)=-\frac{1}{2}$ by the definitions of $H(Y,Z)$ and $g(Y,Z)$. Thus the expected value is

$$\mathbb{E}(X_k)=\frac{1}{w+\frac{c_n}{2^{w-1}}}k+\mathcal{O}(1).$$

To compute the variance $\mathbb{V}X_k$ we use the formula $\mathbb{V}X_k=\mathbb{E}(X_k^2)-(\mathbb{E}X_k)^2$. We know that

$$\mathbb{E}(X_k^2)=[Z^k]\left(G_{YY}(1,Z)+G_Y(1,Z)\right).$$

By differentiating $G(Y,Z)=\frac{f(Y,Z)}{(1-\frac{Z}{2})g(Y,Z)}$ twice with respect to Y we obtain

$$G_{YY}(1,Z)=\frac{f_{YY}(1,Z)}{(1-\frac{Z}{2})(1-Z)H(1,Z)}-\frac{2f_Y(1,Z)g_Y(1,Z)}{(1-\frac{Z}{2})(1-Z)^2H^2(1,Z)}+\frac{2g_Y(1,Z)}{(1-Z)^3H^2(1,Z)},$$

where we used $g_{YY}(Y,Z)=0$. Now we compute the Laurent series, which is

$$\begin{aligned}G_{YY}(1,Z)=&\frac{1}{(1-Z)^3}\left(\frac{2g_Y^2(1,1)}{H^2(1,1)}\right)\\ &+\frac{1}{(1-Z)^2}\left(-\frac{4g_Y(1,1)g_{YZ}(1,1)}{H^2(1,1)}+\frac{4g_Y^2(1,1)H_Z(1,1)}{H^3(1,1)}-\frac{4f_Y(1,1)g_Y(1,1)}{H^2(1,1)}\right)\\ &+\frac{c}{1-Z}+\text{ power series in }(Z-1).\end{aligned}$$

Therefore we have

$$\begin{aligned}\mathbb{E}(X_k^2)&=[Z^k]\left(G_{YY}(1,Z)+G_Y(1,Z)\right)\\ &=\frac{1}{2}(k+2)(k+1)\frac{2g_Y^2(1,1)}{H^2(1,1)}+(k+1)\left(-\frac{4g_Y(1,1)g_{YZ}(1,1)}{H^2(1,1)}\right.\\ &\quad\left.+\frac{4g_Y^2(1,1)H_Z(1,1)}{H^3(1,1)}-\frac{4f_Y(1,1)g_Y(1,1)}{H^2(1,1)}-\frac{g_Y(1,1)}{H(1,1)}\right)+\mathcal{O}(1).\end{aligned} \tag{18}$$

If we subtract the square of (17) from (18) we obtain

$$\begin{aligned}\mathbb{V}X_k&=\mathbb{E}(X_k^2)-(\mathbb{E}X_k)^2\\ &=k\left(\frac{g_Y^2(1,1)}{H^2(1,1)}-\frac{2g_Y(1,1)g_{YZ}(1,1)}{H^2(1,1)}+\frac{2g_Y^2(1,1)H_Z(1,1)}{H^3(1,1)}-\frac{g_Y(1,1)}{H(1,1)}\right)+\mathcal{O}(1).\end{aligned}$$

By the definitions of $H(Y,Z)$ and $g(Y,Z)$ we have

$$g_{YZ}(1,1) = \frac{1-w}{2} - \frac{c_n}{2^w},$$
$$H_Z(1,1) = \frac{1}{4}(w-1)(w-2+\frac{c_n}{2^{w-2}}).$$

Thus we have the variance

$$\mathbb{V}X_k = \frac{(2-\frac{c_n}{2^{w-1}})(1+\frac{c_n}{2^{w-1}})}{(w+\frac{c_n}{2^{w-1}})^3}k + \mathcal{O}(1).$$

Since the constants b_1 and b_2 in (14) and (15) are independent of N, we get

$$b_1 = \frac{1}{w+\frac{c_n}{2^{w-1}}} \text{ and } b_2 = \frac{(2-\frac{c_n}{2^{w-1}})(1+\frac{c_n}{2^{w-1}})}{(w+\frac{c_n}{2^{w-1}})^3}.$$

We note that these are exactly the same values as in Theorem 16 for $d = 1$ since $\lambda = \frac{c_n}{2^{w-1}} + 1$.

Now we want to estimate the distribution function of the standardized random variable

$$Z = \frac{X - b_1 \log_2 N}{\sqrt{b_2 \log_2 N}}$$

as in Section 3 for arbitrary N. Then the characteristic function $\hat{g}_N(t)$ of the variable Z for $t = o(\sqrt{\log N})$ is

$$\begin{aligned}\hat{g}_N(t) &= \frac{1}{N}\sum_{n<N} e^{it\frac{h(n)-b_1\log_2 N}{\sqrt{b_2\log_2 N}}} \\ &= e^{\frac{-t^2}{2}}\left(1+\mathcal{O}\left(\frac{t^3}{\log^{3/2} N}\right)\right)\psi\left(\log_2 N, \frac{t}{\sqrt{b_2\log_2 N}}\right) \\ &\quad + \tilde{R}\left(N, \frac{t}{\sqrt{b_2\log_2 N}}\right)e^{-it\frac{b_1}{\sqrt{b_2}}\sqrt{\log_2 N}}\end{aligned}$$

and the difference from the characteristic function $\hat{f}(t) = e^{-\frac{t^2}{2}}$ of the normal distribution with mean 0 and variance 1 is

$$|\hat{g}_N(t) - \hat{f}(t)| = \mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right).$$

Therefore the Berry-Esseen inequality, Theorem 11, implies

$$\mathbb{P}\left(\frac{X - b_1\log_2 N}{\sqrt{b_2\log_2 N}} < x\right) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{y^2}{2}}dy + \mathcal{O}\left(\frac{1}{\sqrt[4]{\log N}}\right).$$

5 Asymmetric digit sets in higher dimensions

The algorithm presented by Heuberger and Muir in [11] also works for higher dimension d. Now we want to generalize the result of the previous chapter to an arbitrary dimension d.

Algorithm 9 for general dimension d is slightly different from Algorithm 8 for dimension one. The most important part is that the **if** statement in line 11 of Algorithm 8 becomes much more complicated. The difference to Algorithm 7 is that we abbreviate the while loop. All variables are the same as in the previous chapter. For simplicity we write $n + a$, for a vector n and an integer a, and mean that we add a to every coordinate of the vector n.

Algorithm 9 Algorithm to compute the minimal weight of a representation over the digit set $D_{l,u}$ in higher dimension

Input: A vector of integers n, integers $l \leq 0$, $u > 0$, $n \geq 0$ if $l = 0$
Output: Minimal weight representation of n with digits in $D_{l,u}$

1: **while** $n \neq 0$ **do**
2: **if** $n \equiv 0 \mod 2$ **then**
3: $a = 0$, $h = h + 0$
4: $m = \frac{n}{2}$
5: **else**
6: $a = l + ((n - l) \bmod 2^{w-1})$
7: $h = h + 1$
8: $m = \frac{n-a}{2^{w-1}}$
9: $I_{\text{unique}} = \{j \in \{1, 2, \ldots, d\} | a_j \in \textbf{unique}(D_{l,u})\}$
10: $I_{\text{nonunique}} = \{j \in \{1, 2, \ldots, d\} | a_j \in \textbf{nonunique}(D_{l,u})\}$
11: **if** $m_j \equiv 0 \mod 2$ for all $j \in I_{\text{unique}}$ **then**
12: **for** $j \in I_{\text{nonunique}}$ such that m_j is odd **do**
13: $a_j = a_j + 2^{w-1}$
14: $m_j = m_j - 1$
15: **end for**
16: **else**
17: **for** $j \in I_{\text{nonunique}}$ such that $m_j \equiv u + 1 \mod 2^{w-1}$ **do**
18: $a_j = a_j + 2^{w-1}$
19: $m_j = m_j - 1$
20: **end for**
21: **end if**
22: **end if**
23: $n = m$
24: **end while**

Now we want to construct a transducer calculating the weight of a colexicographically minimal joint integer representation. For convenience we start with a slightly wrong transducer, like the algorithm on page 306 of [11]. Therefore we skip the **else** statement in

line 16 of Algorithm 9. The resulting provisional transducer is similar to the transducer in Figure 5.

There is an initial state and some similar states. For every vector $s \in \{0,1\}^d$ there is such a state. We call them beginning states. The vector s signifies the carry at each coordinate. Furthermore there is a block of states. The states inside the block have the labels $(s,t)_i$ where s, $t \in \{0,1\}^d$, and i is the row in the block. The coordinates of s and t have the same meaning as in the previous section, that is s is the carry of the addition $n + \tilde{l}$ and t signifies whether the digit is in **nonunique**$(D_{l,u})$ or not.

If $s \in \{0,1\}^d$ is a beginning state, then there is a loop with label $s|0$ at this state. Because if we read $\varepsilon = s$, then we have $\varepsilon + s \equiv 0 \mod 2$ and we just restart the transducer. Thereby carry propagation occurs. If we read $\varepsilon \neq s$, then we start with the $w-1$ transitions of Automaton 4 in Figure 4 in each coordinate. These $w-1$ transitions can be processed independently for every coordinate. Therefore we need 4^d states in each row and $w-1$ rows to process exactly $w-1$ transitions of Automaton 4.

At the end of the block of states we either go back to a beginning state or start again with the block of states. Let $(s,t)_{w-1}$ be the current state in the last row and ε the next input digit. As in the previous chapter we have $m \equiv \varepsilon + s \mod 2$. If for every coordinate j, $t_j = 1$ implies m_j is even, then we have to process the **if** branch in line 11. We write this condition as $t \cdot (s + \varepsilon \bmod 2) = 0$. In this case the next written digit will be zero and we start the transducer in a beginning state s' again. We only have to consider carry propagation. Thus we have $s' = \lfloor \frac{s+\varepsilon}{2} \rfloor$.

If $t \cdot (s + \varepsilon \bmod 2) = 0$ does not hold, then we would have to process the **else** branch in line 16. But since we skip this part for a while we simply have to restart the transducer with the input m in the case $t \cdot (s + \varepsilon \bmod 2) > 0$. We know that m is the original next input plus the carry s. In this case $s \neq \varepsilon$, otherwise $t \cdot (s + \varepsilon \bmod 2) > 0$ would be false. Therefore there is a transition $s \xrightarrow{\varepsilon|1} (s',t')_1$ in this transducer. This ensures that, when restarting the transducer with input m, we immediately go on to the state (s',t'). Hence we have a transition $(s,t)_{w-1} \xrightarrow{\varepsilon|1} (s',t')$ in the provisional transducer.

Altogether, for s, s', t, $t' \in \{0,1\}^d$, $i \in \{1, \dots w-2\}$, $j \in \{1 \dots d\}$ and $\varepsilon \in \{0,1\}^d$, we have the following transitions in this provisional transducer:

- $s \xrightarrow{\varepsilon|0} s$ if $\varepsilon = s$
- $s \xrightarrow{\varepsilon|1} (s',t')_1$ if $\varepsilon \neq s$ and $\forall j : (s_j, 0) \xrightarrow{(\varepsilon_j, l_0, u_0)^T} (s'_j, t'_j)$ is a transition in Automaton 4
- $(s,t)_i \xrightarrow{\varepsilon|0} (s',t')_{i+1}$ if $\forall j : (s_j, t_j) \xrightarrow{(\varepsilon_j, l_i, u_i)^T} (s'_j, t'_j)$ is a transition in Automaton 4
- $(s,t)_{w-1} \xrightarrow{\varepsilon|0} s'$ if $t \cdot (s + \varepsilon \bmod 2) = 0$ and $s' = \lfloor \frac{s+\varepsilon}{2} \rfloor$
- $(s,t)_{w-1} \xrightarrow{\varepsilon|1} (s',t')_1$ if $t \cdot (s + \varepsilon \bmod 2) > 0$ and $s \xrightarrow{\varepsilon|1} (s',t')_1$ is a transition in this transducer

This transducer does the same as Algorithm 9 without the **else** branch in line 16. Now we must consider the **else** statement.

Let $(s,t)_{w-1}$ be the current state in the last row and ε be the next input digit. To process the **else** branch, $t \cdot (s + \varepsilon \bmod 2) > 0$ must hold in the state $(s,t)_{w-1}$. Otherwise we would process the **if** branch. First let us examine one coordinate j. If $t_j = 1$ nothing is done in the **else** branch, because the digit at this coordinate is unique. If $t_j = 0$, we have to decide if $m_j \equiv u+1 \mod 2^{w-1}$. Here $m_j \bmod 2^{w-1}$ corresponds to the next $w-1$ input digits plus the carry s_j from the current state $(s,t)_{w-1}$. So we just have to compare the input letters plus the carry with the binary expansion of $u+1 \bmod 2^{w-1}$ or, equivalently, we compare $m_j - l \bmod 2^{w-1}$ with $\tilde{v} = u - l + 1 \bmod 2^{w-1}$. If they are not the same at some point, then we just go on like we did in the provisional transducer.

If they are the same, we have to process the **else** branch. There we would have taken $m_j - 1$ as the next input of the algorithm instead of m_j. Therefore we have to decide where we would be in the provisional transducer, when starting in $(s,t)_{w-1}$ and the input is the original input minus one. This case only happens if originally the next nonzero digit is unique, but changing the current digit ensures that the next nonzero digit is non-unique. Nevertheless the next digit will not be zero, since this is the case when the **if** branch is processed. Therefore we would start in $(s,t)_{w-1}$ with original input minus one and immediately go to the block of states again. Otherwise the next digit would be zero. Thus after $w-1$ transitions we are again in a state $(s',t')_{w-1}$ in the last row. Since the next digit is non-unique, we have $t'_j = 0$.

To determine the value of s'_j we have to decide whether there is a carry at position $w-1$ in the addition of $m_j - 1$ and $\tilde{l}$. We have $m_j - 1 \bmod 2^{w-1} = u + k2^{w-1}$ for $k \in \mathbb{Z}$. Since $0 \le u \le 2^w - 1$, we have $k \in \{0, -1\}$. Then the carry is

$$\begin{aligned} s'_j &= \left\lfloor \frac{(m_j - 1) \bmod 2^{w-1} + \tilde{l} \bmod 2^{w-1}}{2^{w-1}} \right\rfloor \\ &= \left\lfloor \frac{u + \tilde{l} + k2^{w-1}}{2^{w-1}} \right\rfloor \\ &= 1 + k, \end{aligned}$$

because $2^{w-1} \le u + \tilde{l} < 2^w$. Therefore we have

$$s'_j = \begin{cases} 0 & \text{if } u \ge 2^{w-1}, \\ 1 & \text{if } u < 2^{w-1}. \end{cases}$$

As a result the state $(s',t')_{w-1}$ where we would be in the provisional transducer has

$$\left(\left[u < 2^{w-1}\right], 0\right)$$

in the j-th coordinate.

To remember that we can change the j-th coordinate at the end of the block we have to use a second identical block $\{j\}$. Let $\emptyset$ be the first block, which already exists in the

provisional transducer. Let $(s,t)_i^C$ be a state in block C. At the end of block $\emptyset$ we go to block $\{j\}$ if $t \cdot (s + \varepsilon \bmod 2) > 0$ and $t_j = 0$ and else to a beginning state or to the block $\emptyset$. If we find out that $m_j \not\equiv u+1 \mod 2^{w-1}$ in block $\{j\}$, then we go back to the appropriate state in block $\emptyset$. At the end of block $\{j\}$ in the state $(s,t)_{w-1}^{\{j\}}$ we go to the same states as we would go from the state with $([u < 2^{w-1}], 0)_{w-1}^{\emptyset}$ in the j-th coordinate.

Up to now we only considered one coordinate. Now we combine this approach for all coordinates. Since we have to remember for each coordinate whether we are allowed to change it or not, we need one block for every subset of coordinates. Let block $C \subseteq \{1, \ldots, d\}$ be the block where we can change the coordinates in C. The states in block C are denoted by $(s,t,)_i^C$. The block $\emptyset$ is the block which already exists in the provisional transducer. The block $\{1, \ldots, d\}$ is not accessible, since we need at least one unique coordinate and only non-unique coordinates can be changed.

If we are in block $C \neq \emptyset$ and we find out that not every coordinate $j \in C$ satisfies $m_j \equiv u + 1 \mod 2^{w-1}$, we go to the appropriate state in block $C' = C \setminus \{j \in \{1, \ldots, d\} \mid m_j \not\equiv u + 1 \mod 2^{w-1}\}$. At the end of block C in state $(s,t)_{w-1}^C$ we can change the coordinates in C and all other coordinates remain the same. Therefore we go to the same states as we would go from $(\tilde{s}, \tilde{t})_{w-1}^{\emptyset}$ where $\tilde{s}_j = [u < 2^{w-1}]$, $\tilde{t}_j = 0$ for $j \in C$ and all other coordinates stay the same, that is $\tilde{s}_j = s_j$ and $\tilde{t}_j = t_j$ for $j \notin C$.

Let $(v_{w-2} \ldots v_0)$ be the binary expansion of $\tilde{v}$. Further let s, s', t, $t' \in \{0,1\}^d$, C, $C' \subsetneq \{1, \ldots, d\}$, $i \in \{1, \ldots, w-2\}$ and $\varepsilon \in \{0,1\}^d$. Then altogether there are the following transitions in the final transducer:

- $s \xrightarrow{\varepsilon|0} s$ if $s = \varepsilon$
- $s \xrightarrow{\varepsilon|1} (s',t')_1^{\emptyset}$ if $s \xrightarrow{\varepsilon|1} (s',t')_1$ is a transition in the provisional transducer
- $(s,t)_i^C \xrightarrow{\varepsilon|0} (s',t')_{i+1}^{C'}$ if $(s,t)_i \xrightarrow{\varepsilon|0} (s',t')_{i+1}$ is a transition in the provisional transducer and $C' = C \setminus \{j : s_j + \varepsilon_j + l_i \bmod 2 \neq v_i\}$
- $(s,t)_{w-1}^{\emptyset} \xrightarrow{\varepsilon|0} s'$ if $t \cdot (s + \varepsilon \bmod 2) = 0$ and $s' = \left\lfloor \frac{s+\varepsilon}{2} \right\rfloor$
- $(s,t)_{w-1}^{\emptyset} \xrightarrow{\varepsilon|1} (s',t')_1^{C'}$ if $t \cdot (s + \varepsilon \bmod 2) > 0$, $(s,t)_{w-1} \xrightarrow{\varepsilon|1} (s',t')_1$ is a transition in the provisional transducer and $C' = \{j : s_j + \varepsilon_j + l_0 \bmod 2 = v_0 \text{ and } t_j = 0\}$
- $(s,t)_{w-1}^C \xrightarrow{\varepsilon|1} (s',t')_1^{C'}$ if $C \neq \emptyset$ and $(\tilde{s}, \tilde{t})_{w-1}^{\emptyset} \xrightarrow{\varepsilon|1} (s',t')_1^{C'}$ is a transition in this transducer with $\tilde{s}_j = [u < 2^{w-1}]$, $\tilde{t}_j = 0$ for $j \in C$ and $\tilde{s}_j = s_j$, $\tilde{t}_j = t_j$ for $j \notin C$

Next we go on like in the previous sections. We define $f(m_1, \ldots, m_d) := e^{ith(m_1,\ldots,m_d)}$ and the matrices $M_{\varepsilon_1,\ldots,\varepsilon_d}$ for $\varepsilon_i \in \{0,1\}$. The (k,l)-th entry of the matrix $M_{\varepsilon_1,\ldots,\varepsilon_d}$ is e^{ith} if there is a transition from state k to l with input label $(\varepsilon_1, \ldots, \varepsilon_d)^T$ and output label h. The entry is 0 if there is no transition from state k to l with this input label. In this section we only use the accessible states to define this matrix. From the ordering of the states we demand that the states of the blocks are sorted by their row i. All blocks of any first row

come first. At the end are the beginning states. The initial and final state $(0,\dots,0)^T$ is the last one of all beginning states. Then we have

$$f(m_1,\dots,m_d) = \vec{v}^T \prod_{l=0}^{L} M_{m_{1,l},\dots,m_{d,l}} M_{0,\dots,0}^{4w} \vec{v}$$

for $\vec{v}^T = (0,\dots,0,1)$ and $m_i = \sum_{l=0}^{L} m_{i,l} 2^l$.

We further define the following summatory functions

$$E(N) = \sum_{m_1,\dots,m_d<N} f(m_1,\dots,m_d)$$
$$F(N) = \sum_{m_1,\dots,m_d<N} M(m_1,\dots,m_d),$$

with

$$M(m_1,\dots,m_d) = \prod_{l=0}^{L} M_{m_{1,l},\dots,m_{d,l}}.$$

To write down a recursion formula for $F(N)$, we need the following matrices

$$B_{C,D} := \sum_{\substack{\varepsilon_i=0,1 \\ i\notin C\cup D}} \sum_{\substack{\varepsilon_i=0 \\ i\in C}} \sum_{\substack{\varepsilon_i=1 \\ i\in D}} M_{\varepsilon_1,\dots,\varepsilon_d}$$

for C, $D \subseteq \{1,\dots,d\}$. The first index C of $B_{C,D}$ is the set of coordinates where the digit is zero. The second index D is the set of coordinates where the digit is one. All other coordinates in $(C\cup D)^c$ can be any digit. These matrices all have eigenvectors $(1,\dots,1)^T$ at $t=0$, because every summand has one entry with modulus 1 in every row. Therefore the eigenvalue to this eigenvector of $B_{C,D}$ is $2^{d-|C|-|D|}$, the number of summands. This is the largest eigenvalue of this matrix, since all Geršgorin discs are contained in $\overline{B(0,2^{d-|C|-|D|})}$. As a special matrix we define $A = B_{\emptyset,\emptyset}$.

Furthermore we define the following functions $G_C(N)$ for every set $C \subseteq \{1,\dots,d\}$

$$G_C(N) := \sum_{\substack{m_i<N \\ i\notin C}} \sum_{\substack{m_i=N \\ i\in C}} M(m_1,\dots,m_d).$$

Then we have $F(N) = G_\emptyset(N)$ and the functions satisfy the following recursion formulas

$$\begin{aligned}
G_C(2N) &= \sum_{\substack{\varepsilon_i=0,1\\ i\notin C}} \sum_{\substack{\varepsilon_i=0\\ i\in C}} \sum_{\substack{2m_i+\varepsilon_i<2N\\ i\notin C}} \sum_{\substack{2m_i+\varepsilon_i=2N\\ i\in C}} M(2m_1+\varepsilon_1,\dots,2m_d+\varepsilon_d) \\
&= B_{C,\emptyset} G_C(N), \\
G_C(2N+1) &= \sum_{\substack{\varepsilon_i=0,1\\ i\notin C}} \sum_{\substack{\varepsilon_i=1\\ i\in C}} \sum_{\substack{2m_i+\varepsilon_i<2N+1\\ i\notin C}} \sum_{\substack{2m_i+\varepsilon_i=2N+1\\ i\in C}} M(2m_1+\varepsilon_1,\dots,2m_d+\varepsilon_d) \\
&= \sum_{D\subseteq C^c} \sum_{\substack{\varepsilon_i=0,1\\ i\notin C\cup D}} \sum_{\substack{\varepsilon_i=0\\ i\in D}} \sum_{\substack{\varepsilon_i=1\\ i\in C}} M_{\varepsilon_1,\dots,\varepsilon_d} \sum_{\substack{m_i<N\\ i\notin C\cup D}} \sum_{\substack{m_i=N\\ i\in C\cup D}} M(m_1,\dots,m_d) \\
&= \sum_{D\subseteq C^c} B_{D,C} G_{C\cup D}(N).
\end{aligned}$$

From this recursion we can determine $G_C(N)$ inductively, because all needed functions $G_{C'}(N)$ have $C' \supsetneqq C$. Therefore we have the following recursion formula for $F(N) = G_\emptyset(N)$

$$F(2N+\varepsilon) = AF(N) + \varepsilon H(N)$$

for $N \geq 1$, $\varepsilon \in \{0,1\}$ and

$$H(N) = \sum_{\emptyset\neq D\subseteq\{1,\dots,d\}} B_{D,\emptyset} G_D(N).$$

To solve this recursion we use the next lemma.

Lemma 13. *Let $G(N) : \mathbb{N} \to \mathbb{C}^{n\times n}$ be a function which satisfies the recurrence relation*

$$G(2N+\varepsilon) = A_\varepsilon G(N) + \varepsilon H(N)$$

for $N \geq 1$. Here we have $\varepsilon \in \{0,1\}$, A_ε two matrices in $\mathbb{C}^{n\times n}$ and $H(N) : \mathbb{N} \to \mathbb{C}^{n\times n}$ a known function. We additionally assume that $H(0) = G(1)$. Then

$$G\left(\sum_{l=0}^{L} \varepsilon_l 2^l\right) = \sum_{l=0}^{L} \varepsilon_l \left(\prod_{j=0}^{l-1} A_{\varepsilon_j}\right) H\left(\sum_{j=l+1}^{L} \varepsilon_j 2^{j-l-1}\right).$$

Proof. We prove this lemma by induction on L. If $L = 0$, then $N = 1$ and we have

$$G(1) = 1\left(\prod_{j=0}^{-1} A_{\varepsilon_j}\right) H(0) = H(0)$$

which is true by assumption. If $L \geq 1$, then we have

$$\begin{aligned} G\left(2\sum_{l=1}^{L}\varepsilon_l 2^{l-1}+\varepsilon_0\right) &= A_{\varepsilon_0}G\left(\sum_{l=1}^{L}\varepsilon_l 2^{l-1}\right)+\varepsilon_0 H\left(\sum_{l=1}^{L}\varepsilon_l 2^{l-1}\right) \\ &= A_{\varepsilon_0}\sum_{l=1}^{L}\varepsilon_l\left(\prod_{j=1}^{l-1}A_{\varepsilon_j}\right)H\left(\sum_{j=l+1}^{L}\varepsilon_j 2^{j-l-1}\right)+\varepsilon_0 H\left(\sum_{l=1}^{L}\varepsilon_l 2^{l-1}\right) \\ &= \sum_{l=0}^{L}\varepsilon_l\left(\prod_{j=0}^{l-1}A_{\varepsilon_j}\right)H\left(\sum_{j=l+1}^{L}\varepsilon_j 2^{j-l-1}\right) \end{aligned}$$

by the recurrence relation and the induction hypothesis. □

If we define $H(0) = G_\emptyset(1)$, we can use this lemma and get

$$F\left(\sum_{l=0}^{L}\varepsilon_l 2^l\right) = \sum_{l=0}^{L}\varepsilon_l A^l H\left(\sum_{j=l+1}^{L}\varepsilon_j 2^{j-l-1}\right). \tag{19}$$

Thereby $H(N)$ is known because it is a sum of functions $G_C(N)$, which are recursively known by Lemma 13.

From the definition of $G_C(N)$ we can derive the growth rates of the functions $G_C(N)$ and $H(N)$. Because the eigenvalues of $M_{\varepsilon_1,\ldots,\varepsilon_d}$ are less than or equal to 1 and there are $N^{d-|C|}$ summands in the definition, we have $G_C(N) = \mathcal{O}(N^{d-|C|})$ and $H(N) = \mathcal{O}(N^{d-1})$.

Next we further investigate the eigenvalues of the matrix A. For $t = 0$ the matrix A is the adjacency matrix of the underlying graph of the accessible parts of the transducer described above. It has eigenvalue 2^d with eigenvector $(1,\ldots,1)^T$ at $t = 0$, since in every row of A the sum of the entries is 2^d. The Geršgorin discs are all contained in $\overline{B(0,2^d)}$ because the sum of the modulus of the entries of a row is 2^d at $t = 0$. Therefore all eigenvalues have moduli less than or equal 2^d which is 2^d an eigenvalue. The underlying graph of the transducer is strongly connected, therefore we can use the theorem of Perron-Frobenius [13] at $t = 0$. Furthermore, we can reach every node starting from any node with exactly $4w$ steps, hence A is primitive at $t = 0$. Therefore the dominating eigenvalue $\mu(t)$ is unique and the modulus of the second largest eigenvalue $\beta(t)$ satisfies $\beta(t) < |\mu(t)|$ around $t = 0$.

Furthermore all other matrices in (19) have eigenvalues less than 2^{d-1}.

Now we want to split up (19) into two parts, one for the dominating eigenvalue and one for the remaining eigenvalues. Therefore let $J = T^{-1}AT$ be a Jordan decomposition of A. We define $\Lambda := T\mathrm{diag}(\mu(t)^{-1},0,\ldots,0)T^{-1}$ and $R := T(J - \mathrm{diag}(\mu(t),0,\ldots,0))T^{-1}$. Then $A^l = \mu^L\Lambda^{L-l} + R^l$ holds for $l \leq L$. Further we define

$$\Lambda(x_0,x_1,\ldots) = \sum_{l=0}^{\infty}x_l\Lambda^l H\left(\sum_{j=0}^{l-1}x_j 2^{l-1-j}\right),$$

$$R(\varepsilon_L,\ldots,\varepsilon_0) = \sum_{l=}^{L}\varepsilon_l R^l H\left(\sum_{j=l+1}^{L}\varepsilon_j 2^{j-l-1}\right).$$

The function Λ is well defined on the infinite product space $\{0,1\}^{\mathbb{N}}$ because it is dominated by a geometric series because $H(N) = \mathcal{O}(N^{d-1})$. We define Λ to be a function on $[0,1)$, by choosing the representation of $x \in [0,1)$ which ends on 0^ω if there is a choice. We can define the function R to be a function on $\mathbb{N}$.

Then we have

$$F\left(\sum_{l=0}^{L} \varepsilon_l 2^l\right) = \mu^L \Lambda(\varepsilon_L, \varepsilon_{L-1}, \ldots, \varepsilon_0, 0^\omega) + R(\varepsilon_L, \ldots, \varepsilon_0)$$

and

$$E(N) = \mu(t)^{\log_2 N} \Psi(\log_2 N, t) + \vec{v}^T R(N) M_{0,\ldots,0}^{4w} \vec{v},$$

with $\Psi(x,t) = \mu(t)^{-\{x\}} \vec{v}^T \Lambda(2^{\{x\}}) M_{0,\ldots,0}^{4w} \vec{v}$.

By the same arguments as in the previous sections $\Psi(x,t)$ is periodic and continuous in x and continuous and two times differentiable in t. Hence, the function $\Psi(x,t)$ and its derivatives are $\mathcal{O}(1)$. Also the error term $\vec{v}^T R(N) M_{0,\ldots,0}^{4w} \vec{v}$ is differentiable, since it is dominated by a geometric series. Furthermore we have

$$|\vec{v}^T R(N) M_{0,\ldots,0}^{4w} \vec{v}| = \mathcal{O}(N^{\log_2 \beta(t)}).$$

So we have

$$E(N) = N^{d + a_1 t + a_2 t^2 + \mathcal{O}(t^3)} \Psi(\log_2 N, t) + \mathcal{O}(N^{\log_2 \beta(t)}).$$

The first and second derivative of $E(N)$ with respect to t at $t = 0$ imply that the expected value of the Hamming weight is

$$\frac{1}{N^d} \sum_{m_i < N} h(m_1, \ldots, m_d) = b_1 \log_2 N + \Psi_1(\log_2 N) + \mathcal{O}(N^{\log_2 \beta(0) - d} \log N)$$

with $b_1 = -i a_1 \log 2$ and $\Psi_1(\log_2 N) = -i \frac{\partial}{\partial t} \Psi(\log_2 N, t)|_{t=0}$, and

$$\frac{1}{N^d} \sum_{m_i < N} h^2(m_1, \ldots, m_d) = b_2 \log_2 N + b_1^2 \log_2^2 N + 2 b_1 \log_2 N \Psi_1(\log_2 N) + \Psi_2(\log_2 N) + \mathcal{O}(N^{\log_2 \beta(0) - d} \log^2 N)$$

with $b_2 = -2 a_2 \log 2$ and $\Psi_2(\log_2 N) = -\frac{\partial^2}{\partial t^2} \Psi(\log_2 N, t)|_{t=0}$. From that we calculate the variance, which is

$$\frac{1}{N^d} \sum_{m_i < N} h^2(m_1, \ldots, m_d) - \left(\frac{1}{N^d} \sum_{m_i < N} h(m_1, \ldots, m_d)\right)^2 = b_2 \log_2 N - \Psi_1^2(\log_2 N) + \Psi_2(\log_2 N) + \mathcal{O}(N^{\log_2 \beta(0) - d} \log^2 N).$$

Example 5.1. *In Figure 7 there is a sketch of the transducer computing the weight of a minimal weight representation over $D_{-2,3}$ in dimension two. The labels of the transitions are omitted in the figure and the transitions going back at the end of a block or inside a block are gray. All non-accessible states also are omitted. We have $w = 3$, $\tilde{u} = (01)$, $\tilde{l} = (10)$ and $\tilde{v} = (10)$.*

For example the state $\binom{01}{11}_2^{\{2\}}$ has transitions to the same states as the state $\binom{01}{10}_2^{\emptyset}$ since $u < 2^{w-1}$. The adjacency matrix A of the underlying graph of this transducer is

$$A = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
3z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
z & 0 & 0 & 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
z & 0 & 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
2z & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
z & 0 & 0 & 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
z & 0 & 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
2z & 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
z & 0 & 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
z & 0 & 0 & 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
z & z & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
2z & 0 & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
2z & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
3z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}.$$

The characteristic polynomial of A is

$$-(x-1)x^7\left(x^2-2z\right)\left(x^3-x^2-xz-2z\right)^2\left(x^5-x^4-7x^3z-20x^2z+6xz^2-24z^2\right).$$

At $t = 0$ the dominating eigenvalue $\mu(0) = 4$ is a root of the fourth factor. Therefore the Taylor expansion of $\mu(t)$ around $t = 0$ is

$$\mu(t) = 4 + \frac{128i}{89}t - \frac{673216}{2114907}t^2 + \mathcal{O}(t^3).$$

Hence the expected value of the weight is

$$0.359551 \log_2 N + \mathcal{O}(1)$$

and the variance is

$$0.0298831 \log_2 N + \mathcal{O}(1).$$

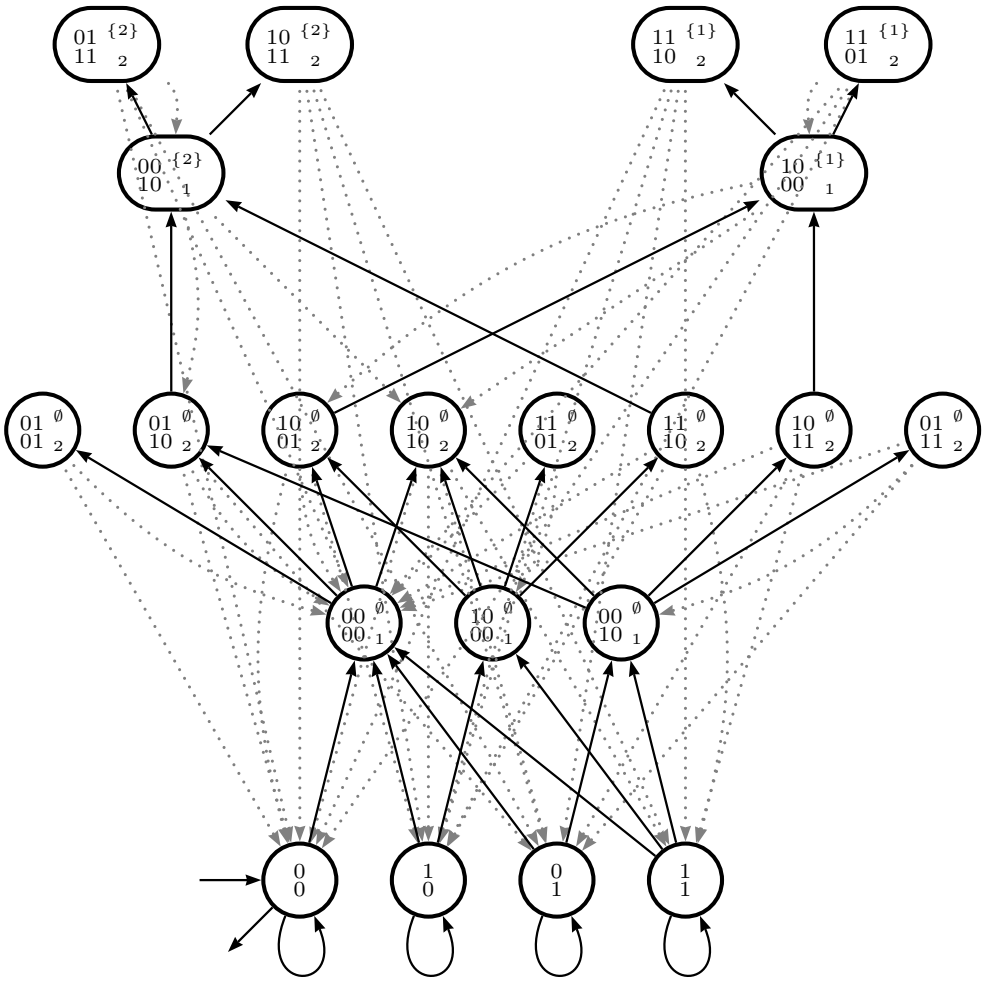

Figure 7: Transducer to compute the Hamming weight of a minimal weight representation in dimension two over the digit set $D_{-2,3}$

Now we can prove the following theorem about the asymptotic expansion. Therefore we again use the probabilistic space $\{(m_1, \dots, m_d)^T \in \mathbb{Z}^d \mid m_i < N\}$ for a fixed integer N with uniform distribution.

Theorem 19. *The Hamming weight $h(m_1, \dots, m_d)$ of the colexicographically minimal and minimal weight representation of an integer vector $(m_1, \dots, m_d)^T$ over the digit set $D_{l,u}$ in dimension d is asymptotically normally distributed. There exist constants b_1, $b_2 \in \mathbb{R}$ depending on u, l and d such that the expected value is $b_1 \log_2 N + \mathcal{O}(1)$ and the variance is $b_2 \log_2 N + \mathcal{O}(1)$. Then we have*

$$\mathbb{P}\left(\frac{h(m_1, \dots, m_d) - b_1 \log_2 N}{\sqrt{b_2 \log_2 N}} < x\right) = \int_{-\infty}^{x} e^{\frac{-y^2}{2}} dy + \mathcal{O}\left(\frac{1}{\sqrt[4]{\log N}}\right)$$

for all $x \in \mathbb{R}$.

Proof. The proof is the same as in the previous sections. We first compute the characteristic function $\hat{g}_N(t)$ of the random variable

$$Z = \frac{h(m_1, \dots, m_d) - b_1 \log_2 N}{\sqrt{b_2 \log_2 N}},$$

which is

$$\begin{aligned}\hat{g}_N(t) &= \frac{1}{N^d} \sum_{n<N} e^{it\frac{h(m_1,\dots,m_d) - b_1 \log_2 N}{\sqrt{b_2 \log_2 N}}} \\ &= e^{\frac{-t^2}{2}} \left(1 + \mathcal{O}\left(\frac{t^3}{\log^{3/2} N}\right)\right) \psi\left(\log_2 N, \frac{t}{\sqrt{b_2 \log_2 N}}\right) \\ &\quad + \tilde{R}\left(N, \frac{t}{\sqrt{b_2 \log_2 N}}\right) e^{-it\frac{b_1}{\sqrt{b_2}}\sqrt{\log_2 N}}.\end{aligned}$$

Next we can estimate the difference from $\hat{g}_N(t)$ to the characteristic function $\hat{f}(t) = e^{-\frac{t^2}{2}}$ of the normal distribution with mean 0 and variance 1, which is

$$|\hat{g}_N(t) - \hat{f}(t)| = \mathcal{O}\left(\frac{t}{\sqrt{\log N}}\right).$$

Therefore the Berry-Esseen inequality, Theorem 11 implies

$$\mathbb{P}\left(\frac{h(m_1, \dots, m_d) - b_1 \log_2 N}{\sqrt{b_2 \log_2 N}} < x\right) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{y^2}{2}} dy + \mathcal{O}\left(\frac{1}{\sqrt[4]{\log N}}\right).$$

□

References

[1] Tom M. Apostol. *Introduction to analytic number theory.* Springer, 1976.

[2] Roberto M. Avanzi. A Note on the Signed Sliding Window Integer Recoding and a Left-to-Right Analogue. In *Selected Areas in Cryptography'04*, pages 130–143, 2004.

[3] Guy Barat and Peter J. Grabner. Distribution of Binomial Coefficients and Digital Functions. *Journal of the London Mathematical Society*, 64(3):523–547, 2001.

[4] Andrew C. Berry. The accuracy of the Gaussian approximation to the sum of independent variates. *Transactions of the American Mathematical Society*, 49:122–136, 1941.

[5] Henri Cohen. Analysis of the Sliding Window Powering Algorithm. *Journal of Cryptology*, 18:63–76, 2005.

[6] Henri Cohen, Atsuko Miyaji, and Takatoshi Ono. Efficient Elliptic Curve Exponentiation Using Mixed Coordinates. In Kazuo Ohta and Dingyi Pei, editors, *Advances in Cryptology - ASIACRYPT´98*, volume 1514 of *Lecture Notes in Computer Science*, pages 51–65. Springer Berlin / Heidelberg, 1998.

[7] Carl-Gustav Esseen. Fourier analysis of distribution functions. A mathematical study of the Laplace-Gaussian law. *Acta Mathematica*, 77:1–125, 1945.

[8] S. Geršgorin. Über die Abgrenzung der Eigenwerte einer Matrix. *Bulletin de l'Académie des Sciences de l'URSS. Classe des sciences mathématiques et naturelles*, 6:749–754, 1931.

[9] Peter Grabner and Jörg Thuswaldner. On The Sum of Digits Function for Number Systems with Negative Bases. *The Ramanujan Journal*, 4:201–220, 2000.

[10] Peter J. Grabner, Clemens Heuberger, and Helmut Prodinger. Distribution results for low-weight binary representations for pairs of integers. *Theoretical Computer Science*, 319:307 – 331, 2004.

[11] Clemens Heuberger and James A. Muir. Minimal Weight and Colexicographically Minimal Integer Representations. *Journal of Mathematical Cryptology*, 1:297–328, 2007.

[12] Clemens Heuberger and Helmut Prodinger. Analysis of Alternative Digit Sets for Nonadjacent Representations. *Monatshefte für Mathematik*, 147:219–248, 2006.

[13] Carl D. Meyer. *Matrix Analysis and Applied Linear Algebra.* Miscellaneous Titles in Applied Mathematics Series. Society for Industrial and Applied Mathematics, 2000.

[14] James A. Muir and Douglas R. Stinson. Minimality and other properties of the width-w nonadjacent form. *Mathematics of Computation*, 75(253):369–384, 2006.

[15] Braden Phillips and Neil Burgess. Minimal Weight Digit Set Conversions. *IEEE Transactions on Computers*, 53:666–677, 2004.

[16] George W. Reitwiesner. Binary Arithmetic. *Advances in Computers*, 1:231–308, 1960.

[17] Jerome A. Solinas. Efficient Arithmetic on Koblitz Curves. *Designs, Codes and Cryptography*, 19:195–249, 2000.

[18] Jerome A. Solinas. Low-Weight Binary Representations for Pairs of Integers. Technical report, University of Waterloo, 2001.

[19] Jörg Thuswaldner. Summatory functions of digital sums occuring in cryptography. *Periodica Mathematica Hungarica*, 38:111–130, 1999.

[20] Jeffrey D. Vaaler. Some extremal functions in Fourier analysis. *Bulletin of the American Mathematical Society*, 12:183–216, 1985.