Unbalanced digit sets and the closest choice strategy for minimal weight integer representations

Clemens Heuberger*
Institut für Mathematik B
Technische Universität Graz, Graz, Austria
http://www.opt.math.tugraz.at/~cheub/

James A. Muir†
Department of Mathematics and Computing Science
Saint Mary's University, Halifax, Canada
http://cs.smu.ca/~jamuir/

25 March 2008

Abstract

An online algorithm is presented that produces an optimal radix-2 representation of an input integer n using digits from the set $D_{\ell,u} = \{a \in \mathbb{Z} : \ell \leq a \leq u\}$, where $\ell \leq 0$ and $u \geq 1$. The algorithm works by scanning the digits of the binary representation of n from left-to-right (i.e., from most-significant to least-significant). The output representation is optimal in the sense that, of all radix-2 representations of n with digits from $D_{\ell,u}$, it has as few nonzero digits as possible (i.e., it has *minimal weight*). Such representations are useful in the efficient implementation of elliptic curve cryptography. The strategy the algorithm utilizes is to choose an integer of the form $d2^i$, where $d \in D_{\ell,u}$, that is closest to n with respect to a particular distance function. It is possible to choose values of ℓ and u so that the set $D_{\ell,u}$ is unbalanced in the sense that it contains more negative digits than positive digits, or more positive digits than negative digits. Our distance function takes the possible unbalanced nature of $D_{\ell,u}$ into account.

1 Introduction

In grade school, students are taught a *radix*-10 (or base-10) number system wherein positive integers are represented using *strings* of digits from the set $D = \{0, 1, 2, \dots, 9\}$. For example, the integer thirty-one thousand four hundred fifteen is represented as "31415". This manner of representing numbers can be generalized, as in the following definition.

Definition 1.1. Let $D \subset \mathbb{Z}$ be a finite set with $0 \in D$ and let $r \geq 2$ be an integer. A *radix-r representation of an integer n with digit set D* is a finite string $a_{s-1} \dots a_1 a_0$ with each $a_i \in D$ such that

$$(a_{s-1} \dots a_1 a_0)_r := \sum_{i=0}^{s-1} a_i r^i = n.$$

The sum operation defines a function from the set D^* of all finite length digit-strings to $\mathbb{Z}$. ◊

*C. Heuberger is supported by the Austrian Science Foundation FWF, project S9606, that is part of the Austrian National Research Network "Analytic Combinatorics and Probabilistic Number Theory."

†This work was initiated during J. Muir's visit to TU Graz which was funded by the FWF-project S9606.

Example 1.2. Consider the set $D = \{0, \pm 1, \pm 2, \pm 3\}$ and the following three strings from D^*:

$$111101010110111,\ 10000\bar{1}0\bar{1}0\bar{1}00\bar{1}00\bar{1},\ 100000\bar{3}00300\bar{1}00\bar{1}.$$

Note that, for typographic reasons, we denote the digits $-1, -2, -3$ by $\bar{1}, \bar{2}, \bar{3}$. Each string is a radix-2 representation of 31415. $\diamondsuit$

Our interest is mainly in radix-2 representations which use a digit set D containing $0, 1$ and other integers. To convert a radix-2 representation $a_{s-1} \ldots a_1 a_0$ into a number, we simply need to evaluate the sum $(a_{s-1} \ldots a_1 a_0)_2$. One way to do this is based on Horner's rule for evaluating polynomials, as in Algorithm 1. Notice there that the number of times the addition operation on line 5 is carried out is equal to one less than the number of nonzero digits in $a_{s-1} \ldots a_1 a_0$, assuming that $a_{s-1} \neq 0$.

Algorithm 1 Horner's Rule

Input: a radix-2 representation $a_{s-1} \ldots a_1 a_0$.
Output: the integer $n = (a_{s-1} \ldots a_1 a_0)_2$.

1: $n \leftarrow a_{s-1}$
2: **for** $i = s - 2$ **downto** 0 **do**
3: $\quad n \leftarrow 2n$
4: $\quad$ **if** $a_i \neq 0$ **then**
5: $\qquad n \leftarrow n + a_i$
6: **return** n

If Algorithm 1 is modified slightly, it can be used to compute

$$nP = \underbrace{P + P + \cdots + P}_{n}$$

where P is an element of a group and "+" denotes the group operation. This computation is commonly required in elliptic curve cryptography, which utilizes the abelian group formed by points on an elliptic curve defined over a finite field. There are well known formulae for doubling a point (i.e., computing $2P$) and adding two unequal points. Thus, if we have a radix-2 representation $a_{s-1} \ldots a_1 a_0$ of n, then we can use it to compute nP, as in Algorithm 2.

Algorithm 2 Scalar Multiplication via Horner's Rule

Input: a radix-2 representation $a_{s-1} \ldots a_1 a_0$, an elliptic curve point P.
Output: the point $nP = (a_{s-1} \ldots a_1 a_0)_2 P$.

1: $Q \leftarrow a_{s-1} P$
2: **for** $i = s - 2$ **downto** 0 **do**
3: $\quad Q \leftarrow 2Q$
4: $\quad$ **if** $a_i \neq 0$ **then**
5: $\qquad Q \leftarrow Q + a_i P$
6: **return** Q

Note that some of the computations required in Algorithm 2 can be done in advance if we happen to know which digit set D the radix-2 representation is built from. If D is known, then for each $d \in D$ the point dP can be precomputed and stored; this permits $Q + a_i P$ (line 5) to be evaluated at a cost of one table look-up (to retrieve $a_i P$) and one elliptic curve addition. Precomputation is advantageous if nP must be evaluated for several different values of n. If we do not count the cost of precomputation, the number of elliptic curve additions required to compute nP is equal to one less than the number of nonzero digits in $a_{s-1} \ldots a_1 a_0$, assuming that $a_{s-1} \neq 0$. Since elliptic curve additions are computationally expensive, it is desirable to do only as few of them as necessary.

Example 1.3. Consider again the digit set $D = \{0, \pm 1, \pm 2, \pm 3\}$. Here are three strings from D^*:

$$101001011110100100100 1, \; 1010011000 0\bar{1}\bar{1}001001001, \; 300\bar{3}00\bar{2}0000\bar{3}000200201.$$

Each of these is a radix-2 representation of 2718281, and each can be used in Algorithm 2 to compute $2718281P$. The first string contains 11 nonzero digits, the second contains 9, and the third contains 7. Thus, the number of times the elliptic curve addition operation on line 5 is performed is 10, 8 and 6, respectively. We will see later that the second and third representations can be computed from the binary representation of 2718281. Thus, if 2718281 is initially encoded in binary, before Algorithm 2 is executed it may be beneficial to construct one of these alternate representations. $\Diamond$

The preceding example suggests the following minimization problem: given a set of digits D and an integer n, of all strings $\alpha \in D^*$ such that $n = (\alpha)_2$, find one that has as few nonzero digits as possible. This problem is only interesting when n has a number of different radix-2 representations in D^*. Note that if $\{0, \pm 1\} \subset D$ then it is easily seen that each nonzero integer has an infinite number of radix-2 representations in D^*.

Several authors have presented right-to-left constructions for minimal weight representations which work for particular families of digit sets; a brief survey is given by Muir and Heuberger [5]. The earliest of these goes back to Reitwiesner [14] and uses the digit set $\{0, \pm 1\}$. By "right-to-left" we mean that the the digits of α are determined in turn from least- to most-significant. Phillips and Burgess [13] generalize all previously known right-to-left constructions by presenting a construction which produces minimal weight[1] representations using the digit set $D_{\ell,u}$, which is defined as follows:

$$D_{\ell,u} := \{a \in \mathbb{Z} : \ell \leq a \leq u\}, \text{ where } \ell \leq 0, u \geq 1.$$

This digit set is unusual when compared to the digit sets of other constructions since it may contain more positive digits than negative, or more negative digits than positive.

Suppose that we fix numbers ℓ and u, and precompute dP for each $d \in D_{\ell,u}$. To compute nP using Algorithm 2 where n is encoded in binary, we can first compute a minimal weight radix-2 representation of n with digits in $D_{\ell,u}$ using the right-to-left method of Phillips and Burgess [13]. However, this approach presents a slight annoyance to implementors: Algorithm 2 processes the digits of $\alpha = a_{s-1} \dots a_1 a_0$ from *left to right*. This means that all the digits of α must be computed and stored before the computations in Algorithm 2 can proceed. This problem of opposing directions has been remarked by both Müller [11, pp. 224–225] and Solinas [16, p. 200]. If the digits of α could instead be determined left-to-right, then, as each digit is computed, one iteration of the "for" loop could proceed. In this way, it is not necessary to store the digits of α — they are just determined on the fly as needed.

Our Contributions. We present an algorithm which, for any $\ell \leq 0$ and $u \geq 1$, produces a minimal weight radix-2 representation of a positive integer n using the digit set $D_{\ell,u}$. The algorithm works by scanning the digits of the binary representation of n from left to right. The algorithm is *online* in the sense that it is able to compute a digit of its output after scanning only a finite number of the most-significant bits of n.

The main strategy our algorithm employs is to determine an element from the set

$$W_1 := \{d2^i : i \in \mathbb{Z}, i \geq 0, d \in D_{\ell,u} \text{ and } d \neq 0\} \tag{1}$$

that is closest to n. The function we use to quantify closeness differs from the standard metric $d(a,b) = |a - b|$. Our distance function incorporates the parameters ℓ and u and takes the possible unbalanced nature of $D_{\ell,u}$ into account. Interestingly, we find that for certain values of ℓ and u there

[1] In fact, Phillips and Burgess consider the construction of minimal weight representations using an arbitrary radix $r \geq 2$. However, with the exception of $r = 2$, their minimality proof imposes a number of restrictions on the parameters ℓ and u [13, p. 671].

are inputs n where it is not possible to determine a value $c \in W_1$ closest to n without reading *all* the bits of n. Nevertheless, we find that we can obtain an online algorithm by relaxing our choice of $c \in W_1$. We show that to build a minimal weight representation of n, it suffices to choose $c \in W_1$ which is "almost" closest to n.

Related Work. In the cryptographic literature, the first left-to-right algorithm for minimal weight radix-2 representations using the digits $\{0, \pm 1\}$ was proposed by Joye and Yen [6]. Several authors later proposed left-to-right constructions using the digits $\{0, \pm 1, \pm 3, \ldots, \pm(2^{w-1} - 1)\}$ [1] [12] [10]. Following these, Möller [8] gave a left-to-right construction using the digits $\{0, \pm 1, \pm 3, \ldots, \pm m\}$ where m is any odd positive integer; the same construction can be found in work by Khabbazian, Gulliver and Bhargava [7]. Grabner, Heuberger, Prodinger and Thuswaldner [3] and Heuberger, Katti, Prodinger and Ruan [4] also propose left-to-right algorithms using the so-called alternating greedy expansion. They give constructions for minimal weight representations, both in the case of the digits $\{0, \pm 1, \pm 3, \ldots, \pm(2^{w-1} - 1)\}$ as well as in the case of joint representations of several integers with digits $\{0, \pm 1\}$.

Of the left-to-right constructions mentioned above, only the method proposed by Muir and Stinson [10] explicitly uses the strategy of computing integers of the form $d2^i$ that are closest to n, with respect to the standard Euclidean distance. This construction is most similar to the one in the current work except that here we must use a different distance function and our digit set is more general.

In the special case where only the digits $\{0, \pm 1\}$ are allowed, the strategy of choosing 2^i closest to n to construct a radix-2 representation is a very natural one. It is essentially just a greedy strategy, and it is not surprising to find this construction proposed elsewhere in the computer science literature. For example, Ganesan and Manku [2] present such representations in their study of optimal routing in a circular network. Also, in an unpublished manuscript, Shallit [15, p. 3] presents an algorithm based on this construction and claims that it outputs minimal weight representations.

Outline. We begin with some preliminary definitions, notations and results in §2. Then, in §3, we explain the basic strategy underlying our algorithm along with our main results (i.e., the algorithm itself and the results we use to prove its correctness and optimality). Proofs of these main results follow in §4. We end by giving an online implementation of our algorithm in §5.

2 Preliminaries

Here we present some preliminary definitions and notations. When we speak about a *digit set*, we mean a finite set of integers which contains 0.

Definition 2.1. Let D be a digit set and let $\alpha = a_{s-1} \ldots a_1 a_0$ be a string of digits from D (i.e., $\alpha \in D^*$). The *weight* of α is the number of nonzero digits it contains; it is denoted by $\mathsf{wt}(\alpha)$. $\Diamond$

Definition 2.2. Let D be a digit set and let $n \in \mathbb{Z}$. If n has some representation $\alpha \in D^*$, then the *minimal weight* of n with respect to D, denoted by $\mathsf{wt}^*(n)$, is the number

$$\mathsf{wt}^*(n) := \min\{\mathsf{wt}(\alpha) : \alpha \in D^* \text{ and } (\alpha)_2 = n\}.$$

In the case where n has no representation in D^*, then $\mathsf{wt}^*(n)$ is undefined. $\Diamond$

We say that $\alpha \in D^*$ is a *minimal weight representation* if $\mathsf{wt}(\alpha) = \mathsf{wt}^*(n)$ where $n = (\alpha)_2$.

Let $\ell \leq 0$ and $u \geq 1$ be integers. We consider the left-to-right construction of minimal weight representations using the digit set

$$D = D_{\ell,u} := \{a \in \mathbb{Z} : \ell \leq a \leq u\}.$$

This family of digit sets has been studied previously by Phillips and Burgess [13] and Heuberger and Muir [5]. Both works contain algorithms which construct minimal weight representations from right to left. Thus, for any $n \in \mathbb{Z}$ and digit set $D_{\ell,u}$, we already have a way of computing $\mathsf{wt}^*(n)$.[2]

Example 2.3. Consider again the three representations of 31415 listed in Example 1.2. These three representations were constructed using the right to left algorithm from [5], but with different values of ℓ and u. Thus, each representation is in fact a minimal weight representation. When $\ell = 0, u = 1$, we get the digit set $D_{0,1} = \{0, 1\}$; of course, there is only one representation of 31415 using these digits, and it contains exactly 11 nonzero digits. When $\ell = -1, u = 1$, we get the digit set $D_{-1,1} = \{0, \pm 1\}$. From the output of the algorithm, we see that any minimal weight representation of 31415 with digits from $D_{-1,1}$ contains exactly 6 nonzero digits.[3] When $\ell = -3, u = 3$, we get the digit set $D_{-3,3} = \{0, \pm 1, \pm 2, \pm 3\}$. As before, from the output of the algorithm, we see that any minimal weight representation of 31415 with digits from $D_{-3,3}$ contains exactly 5 nonzero digits. $\Diamond$

Because of the bounds on ℓ and u, it is always true that $\{0, 1\} \subseteq D_{\ell,u}$. Thus, every nonnegative integer n has a representation with digits in $D_{\ell,u}$. This also implies that $\mathsf{wt}^*(n)$ is always defined for $n \geq 0$. A negative integer has a representation with digits in $D_{\ell,u}$ if and only if $\ell \leq -1$. Thus, in the case where $\ell = 0$, $\mathsf{wt}^*(n)$ is defined only for $n \geq 0$.

2.1 Subadditivity of wt^*

As a first result on minimal weight representations, we prove that wt^* is a subadditive function. Apart from being an interesting fact on its own, it will be a valuable tool in several proofs because it enables us not to worry about carries when manipulating representations.

Proposition 2.4. *Let m and n be integers. Then*

$$\mathsf{wt}^*(m + n) \leq \mathsf{wt}^*(m) + \mathsf{wt}^*(n).$$

Proof. It is sufficient to prove

$$\mathsf{wt}^*(m + n) \leq \mathsf{wt}^*(m) + 1 \text{ for all } m, n \in \mathbb{Z} \text{ with } \mathsf{wt}^*(n) = 1, \tag{2}$$

since the result for arbitrary integers n follows by repeated application of (2).

We prove (2) by induction on $\mathsf{wt}^*(m)$. For $\mathsf{wt}^*(m) = 0$, there is nothing to show.

Take a minimal weight representation $a_r \ldots a_0$ of m. We write $n = d \cdot 2^j$ for some $d \in D_{\ell,u}$ and some nonnegative integer j. By increasing j if necessary, we can assume that d is odd. Next, we only have to consider the case that $j = 0$, because otherwise, we can write $m = m_1 2^j + m_0$ with $m_1 = (a_r \ldots a_j)_2$ and $m_0 = (a_{j-1} \ldots a_0)_2$ and we can consider the addition of m_1 and d instead.

If $a_0 = 0$, then $a_r \ldots a_1 d$ is a representation of $m + n$ of weight $\mathsf{wt}^*(m) + 1$ and we are done.

If a_0 is even and nonzero, we use (2) on $(m - a_0)/2$ and $a_0/2$ to see that $\mathsf{wt}^*(m/2) \leq \mathsf{wt}^*(m) - 1 + 1 = \mathsf{wt}^*(m)$. This lower bound on $\mathsf{wt}^*(m)$ implies that there is a minimal weight representation of m which arises by appending a 0 to a minimal weight representation of $m/2$. We may assume that our representation $a_r \ldots a_0$ has this property, i.e., $a_0 = 0$ or a_0 is odd.

Finally, we consider the case of an odd a_0. In this case, $(a_0 + d)/2 \in D_{\ell,u}$. We use (2) on $(m - a_0)/2$ and $(a_0 + d)/2$ to see that $\mathsf{wt}^*((m + d)/2) \leq \mathsf{wt}^*(m) - 1 + 1 = \mathsf{wt}^*(m)$. Again, we get a representation of $m + d$ of weight at most $\mathsf{wt}^*(m)$ by appending a zero to a representation of $(m + d)/2$. $\square$

[2] Note that both [13] and [5] provide statistical analyses of $\mathsf{wt}^*(n)$; cf. [13, Equation (13)] and [5, Theorem 6.7]. Of course, these results also apply to the weight of the representations proposed in this work.

[3] Readers may recognize the given representation as one of Reitwiesner's so-called *nonadjacent forms* [14].

2.2 The parity of ℓ and u

The digit set $D_{\ell,u-1}$ contains one less positive digit than the digit set $D_{\ell,u}$. If we decrease the cardinality of our digit set in this way, then for a given integer n, $\mathsf{wt}^*(n)$ will either increase or stay the same. However, in the case where u is even, we can be more precise: if u is even, then changing u to $u-1$ will never increase $\mathsf{wt}^*(n)$. An analogous result holds for the parameter ℓ: if ℓ is even and nonzero, then changing ℓ to $\ell+1$ will never increase $\mathsf{wt}^*(n)$. These two facts can be deduced from [5, Lemma 4.6], but, for completeness, we establish them here.

Proposition 2.5. *Let $u \geq 1$ and $\ell \leq 0$ be integers and set*

$$u' := \begin{cases} u, & \text{if } u \text{ is odd,} \\ u-1, & \text{if } u \text{ is even,} \end{cases} \qquad \ell' := \begin{cases} \ell, & \text{if } \ell \text{ is odd or } \ell = 0, \\ \ell+1, & \text{if } \ell \text{ is even and nonzero.} \end{cases}$$

Suppose $n \in \mathbb{Z}$ has a minimal weight representation with digits in $D_{\ell,u}$. Then n also has a minimal weight representation with digits in $D_{\ell',u'}$, and these two representations have equal weight.

Proof. Note that either ℓ' and u' are both odd, or $\ell' = 0$ and u' is odd. In both cases we have $\ell \leq \ell'$ and $u' \leq u$. Our strategy will be to take a minimal weight representation of n with digits in $D_{\ell,u}$ and modify it, without changing the number of nonzero digits, to obtain a representation of n with digits in $D_{\ell',u'}$. The modification essentially involves pushing any even nonzero digits left until they become odd. This will show that $w' \leq w$ where w' is the minimal weight of n with respect to $D_{\ell',u'}$ and w is the minimal weight of n with respect to $D_{\ell,u}$. Of course, $w \leq w'$ since $D_{\ell',u'} \subseteq D_{\ell,u}$. Hence, we get $w = w'$ which gives us the desired result.

Let $b_j \dots b_1 b_0$ be a minimal weight representation of n with digits in $D_{\ell,u}$. If all nonzero digits of this representation are odd, then each digit is also in $D_{\ell',u'}$, and we are done. So suppose that $b_j \dots b_1 b_0$ contains an even nonzero digit. More precisely, assume that b_k is an even nonzero digit such that all nonzero digits to its left are odd.

Modify the representation $b_j \dots b_k \dots b_1 b_0$ as follows. Write $b_k = 2^s b$ where $s \geq 1$ and $b \in D_{\ell,u}$ is odd. Let $d = b_{k+s} + b$. Now, replace b_k with 0 and b_{k+s} with d. This modification clearly results in a representation of n, although we do not yet know if $d \in D_{\ell,u}$. Note that d must be nonzero since otherwise the new representation of n would have too few nonzero digits (recall we started with a minimal weight representation of n). If it was true that $b_{k+s} = 0$, then we could say that all digits of the new representation are in $D_{\ell,u}$ (since d would then be equal to b), and that there is now one less nonzero even digit. This is exactly what we want since it implies that we can eliminate all nonzero even digits in this manner.

That b_{k+s} equals 0 is, in fact, necessarily true. Suppose that $b_{k+s} \neq 0$. Since b_{k+s} was to the left of b_k, it must be odd. Thus, d is even and nonzero, and $d/2 \in D_{\ell,u}$. By subadditivity (Proposition 2.4) we see that $\mathsf{wt}^*((b_j \dots b_{k+s+1})_2 + d/2) \leq \mathsf{wt}^*((b_j \dots b_{k+s+1})_2) + 1$, which results in a representation of n of weight $\leq \mathsf{wt}^*(n) - 1$, since we decreased the Hamming weight twice (in positions k and $k+s$) and increased it at most once. This is a contradiction. So, it must be that $b_{k+s} = 0$, and the result follows. ☐

Example 2.6. Let n be any integer. We now know that the digits of $D_{-4,6}$ will not admit a minimal weight representation of n that has fewer nonzero digits than a minimal weight representation of n with digits from $D_{-3,5}$. In fact, the proof of Proposition 2.5 shows that the eleven digits of $D_{-4,6}$ are no better than the six digits of $\{-3, -1, 0, 1, 3, 5\}$. Similarly, the digits of $D_{0,8}$ do not allow any minimal weight representations of n with fewer nonzero digits than a minimal weight representation of n with digits from $D_{0,7}$. ◊

As a result of Proposition 2.5, in the remainder of the paper we consider only two cases for the parameters ℓ and u: 1) $\ell = 0$ and u is odd, 2) both ℓ and u are odd. A fair question to ask now is: why bother to use any even nonzero digits from $D_{\ell,u}$ at all? The answer is convenience, as we will see in the coming sections.

3 Strategy and main results

Fix a digit set $D_{\ell,u}$ so that either $\ell = 0$ and u is odd, or both ℓ and u are odd. The set of all integers c with $\mathsf{wt}^*(c) = 1$ is denoted by

$$W_1 := \{c \in \mathbb{Z} : \mathsf{wt}^*(c) = 1\}.$$

Observe that this is the same set as given in (1).

Given an integer n, if we read the digits of a minimal weight representation of n from left to right, then each nonzero digit we read corresponds to some $c_i \in W_1$. If $\mathsf{wt}^*(n) = t$, then this correspondence gives us t elements of W_1, call them $c_1, c_2, \dots c_t$. These numbers can be interpreted as successive approximations to n:

$$\begin{aligned}
&c_1\\
&c_1 + c_2\\
&\quad\vdots\\
&c_1 + c_2 + \cdots + c_t = n.
\end{aligned}$$

When building a minimal weight representation of n from scratch, we do not know which values from W_1 to choose for $c_1, \dots, c_t$. We develop an algorithm which chooses c_i so that it is a close approximation to $n - (c_1 + c_2 + \cdots + c_{i-1})$.

There are two elements in W_1 that are closer to n than any others. We define the *left* and the *right neighbour* of n as

$$\begin{aligned}
N^-(n) &:= \max\{c \in W_1 : c \le n\},\\
N^+(n) &:= \min\{c \in W_1 : n \le c\}.
\end{aligned}$$

Of course, when $n \in W_1$ we have $N^-(n) = N^+(n)$. It can be shown that n always has a minimal weight representation with most significant term equal to $N^-(n)$ or $N^+(n)$; this is essentially the content of the following result, which will be proved in Section 4.2:

Proposition 3.1. *Let n be a nonzero integer. Then*

$$\mathsf{wt}^*(n) = \mathsf{wt}^*\left(n - N^-(n)\right) + 1 \quad \textit{or} \quad \mathsf{wt}^*(n) = \mathsf{wt}^*\left(n - N^+(n)\right) + 1. \tag{3}$$

If $\ell = 0$, then we have

$$\mathsf{wt}^*(n) = \mathsf{wt}^*\left(n - N^-(n)\right) + 1 \tag{4}$$

for all positive integers n.

Our algorithm will choose $c \in \{N^-(n), N^+(n)\}$ so that $\mathsf{wt}^*(n) = \mathsf{wt}^*(n - c) + 1$, replace n with $n - c$, and then repeat these two steps until n equals zero. The sequence $\mathsf{wt}^*(n)$ formed by the variable n decreases by one in each step, thus this algorithm will terminate ($n = 0$ is the only integer with $\mathsf{wt}^*(n) = 0$); moreover, if the input integer n has $\mathsf{wt}^*(n) = t$, then the algorithm will terminate after exactly t steps. The problem we must now consider is how to decide between $N^-(n)$ and $N^+(n)$.

Example 3.2. Consider the digit set $D_{-3,5}$. It is easy to verify that, for all $n \in \{65, 66, \dots, 79\}$, $N^-(n) = 4 \cdot 2^4 = 64$ and $N^+(n) = 5 \cdot 2^4 = 80$. In the following table, we compute $\mathsf{wt}^*(n), \mathsf{wt}^*(n -$

$N^-(n)), \mathsf{wt}^*(n - N^+(n))$ for each n in this range.

n	$\mathsf{wt}^*(n)$	$\mathsf{wt}^*(n - N^-(n))$	$\mathsf{wt}^*(n - N^+(n))$
65	2	**1**	2
66	2	**1**	2
67	2	**1**	2
68	2	**1**	**1**
69	2	**1**	2
70	2	**1**	2
71	3	**2**	**2**
72	2	**1**	**1**
73	3	**2**	**2**
74	2	**1**	**1**
75	3	**2**	**2**
76	2	**1**	**1**
77	2	2	**1**
78	2	2	**1**
79	2	2	**1**

There are seven rows in the table where both $c = N^-(n)$ and $c = N^+(n)$ satisfy $\mathsf{wt}^*(n) = \mathsf{wt}^*(n - c) + 1$; each of these rows contains two numbers in boldface. In the other eight rows, just one value of $c \in \{N^-(n), N^+(n)\}$ satisfies $\mathsf{wt}^*(n) = \mathsf{wt}^*(n - c) + 1$. Since we always want to choose c with $\mathsf{wt}^*(n) = \mathsf{wt}^*(n - c) + 1$, it is apparent that whether we choose $N^-(n)$ or $N^+(n)$ does matter. $\Diamond$

When deciding whether we should approximate n by its left or its right neighbour, we calculate the "distance" from $N^-(n)$ and $N^+(n)$ to n in a way that takes the possibly unbalanced nature of $D_{\ell,u}$ into account. For $m \in \mathbb{Z}$, we define *the norm of* m, denoted $\|m\|$, as

$$\|m\| := \begin{cases} 0, & \text{if } m = 0, \\ m/u, & \text{if } m > 0, \\ m/\ell, & \text{if } m < 0 \text{ and } \ell < 0, \\ \infty, & \text{if } m < 0 \text{ and } \ell = 0. \end{cases}$$

We note that this is not a vector norm in the usual sense since the sign of m matters. Using this norm, we calculate the distance from n to each of $N^-(n)$ and $N^+(n)$ as

$$\left\|n - N^-(n)\right\| \quad \text{and} \quad \left\|n - N^+(n)\right\|.$$

Observe that since $N^-(n) \leq n \leq N^+(n)$, we have

$$\left\|n - N^-(n)\right\| = \frac{n - N^-(n)}{u} = \frac{|n - N^-(n)|}{|u|},$$

and provided $\ell \neq 0$,

$$\left\|n - N^+(n)\right\| = \frac{n - N^+(n)}{\ell} = \frac{|n - N^+(n)|}{|\ell|}.$$

The idea behind this particular choice of norm is the following. If n is approximated by $N^-(n)$, then the difference $n - N^-(n)$ is positive and may be in the range $0 \leq n - N^-(n) \leq (uu\ldots u)_2$ for an appropriate number of digits u. On the other hand, if n is approximated by $N^+(n)$, then the difference $n - N^+(n)$ is negative and may be in the range $(\ell\ell\ldots\ell)_2 \leq n - N^+(n) \leq 0$. To balance these different ranges, it seems to be appropriate to divide the approximation error by u if it is positive and by ℓ if it is negative.

Example 3.3. Consider the digit set $D_{-1,5}$. If $n = 29$, then it is easily seen that $N^-(n) = 24$ and $N^+(n) = 32$. Since $29 \notin W_1$, we have $\mathsf{wt}^*(29) \geq 2$. However, $29 = (3005)_2$, thus we see that $\mathsf{wt}^*(29) = 2$. With respect to Euclidean distance, we would say that n is closer to $N^+(n)$ than $N^-(n)$ (distance 3 compared to distance 5). However, for our purposes, 32 is not a good approximation to 29 as $\mathsf{wt}^*(29-32) = \mathsf{wt}^*(-3) = 2 = \mathsf{wt}^*(29)$; i.e., taking $c = 32$ does not satisfy $\mathsf{wt}^*(n-c) = \mathsf{wt}^*(n) - 1$. Using the norm defined above we have

$$\|29 - 24\| = \frac{5}{5} = 1 \quad \text{and} \quad \|29 - 32\| = \frac{-3}{-1} = 3.$$

According to this notion of distance, 24 is the better approximation to 29. Indeed, $\mathsf{wt}^*(29-24) = \mathsf{wt}^*(5) = 1 = \mathsf{wt}^*(29) - 1$. ♢

For any nonzero integer n, the set $\mathsf{closest}(n) \subseteq W_1$ is defined to be

$$\mathsf{closest}(n) = \{c \in W_1 : \|n - c\| \leq \|n - c'\| \text{ for all } c' \in W_1\}.$$

It is clear that $\mathsf{closest}(n) \subseteq \{N^-(n), N^+(n)\}$. Depending on the values of ℓ and u, the set $\mathsf{closest}(n)$ might contain both neighbours of n rather than just one (i.e., sometimes there is more than one element of W_1 that is closest to n). We will see (as a consequence of a more general result) that for any nonzero integer n, $c \in \mathsf{closest}(n)$ implies $\mathsf{wt}^*(n) = \mathsf{wt}^*(n-c) + 1$. Thus, in each step of our algorithm, we might try to compute $c \in \mathsf{closest}(n)$. However, this approach has an interesting deficiency.

Our ultimate goal is to devise an *online* algorithm that creates a minimal weight representation by processing the digits of the binary representation of n from left to right. This means that if we want to compute $c \in \mathsf{closest}(n)$, the only information we have to work from is a fixed number of most significant digits of the binary representation of n. To be clear, when we refer to a binary representation of an integer, we mean a radix-2 representation with digits from $\{0, 1\}$. As the following example shows, it is not always possible to determine $c \in \mathsf{closest}(n)$ in this manner.

Example 3.4. Consider the digit set $D_{-1,5}$. For any integer $i \geq 0$, it is easily seen that no integer strictly between $3 \cdot 2^i$ and $4 \cdot 2^i$ is in W_1, cf. Lemma 4.5. Thus, for any n with $3 \cdot 2^i < n < 4 \cdot 2^i$, we have $N^-(n) = 3 \cdot 2^i$ and $N^+(n) = 4 \cdot 2^i$. Suppose n is an integer in this interval. Suppose further that there exists a function f which, upon input i and some fixed number k of the most significant digits of the binary representation of n, correctly computes $c \in \mathsf{closest}(n) \subseteq \{3 \cdot 2^i, 4 \cdot 2^i\}$. We will construct two integers which demonstrate that f cannot exist.

By computing $\left\|x - 3 \cdot 2^i\right\|$ and $\left\|x - 4 \cdot 2^i\right\|$, it can be verified that $x = 3 \cdot 2^i + 5/6 \cdot 2^i$ is equidistant to $3 \cdot 2^i$ and $4 \cdot 2^i$. Thus, all n with $3 \cdot 2^i < n < x$ have $\mathsf{closest}(n) = \{3 \cdot 2^i\}$, and all n with $x < n < 4 \cdot 2^i$ have $\mathsf{closest}(n) = \{4 \cdot 2^i\}$. Observe that

$$x = 3 \cdot 2^i + 5/6 \cdot 2^i = (11)_2 \cdot 2^i + (0.11010101\ldots)_2 \cdot 2^i = (11.11010101\ldots)_2 \cdot 2^i$$

Choose i so that it is greater than k and consider the two integers

$$n^- = (\overbrace{11.110101\ldots}^{k}00)_2 \cdot 2^i \quad \text{and} \quad n^+ = (\overbrace{11.110101\ldots}^{k}11)_2 \cdot 2^i;$$

i.e., the k most significant digits of the binary representations of n^- and n^+ are the same. Observe that $3 \cdot 2^i < n^- < x < n^+ < 4 \cdot 2^i$, so $\mathsf{closest}(n^-) = \{3 \cdot 2^i\}$ and $\mathsf{closest}(n^+) = \{4 \cdot 2^i\}$. However, when f is applied to each of these integers, the return values will be equal since they are generated by equal inputs (i.e., i and the same k digits). So, the output of f is not correct for one of n^- or n^+, contrary to the definition of f. Therefore, f cannot exist. ♢

Note that when $\ell = 0$, deciding between $N^-(n)$ and $N^+(n)$ is easy. For every positive integer n, we have $\mathsf{closest}(n) = \{N^-(n)\}$, thus there is no decision to be made.

Fortunately, in the case where $\ell \neq 0$, we can determine $c \in W_1$ with $\mathsf{wt}^*(n) = \mathsf{wt}^*(n-c)+1$ by taking c to be "almost closest" to n. We fix a positive number δ such that

$$\delta < \min\left\{\frac{1}{|\ell|}, \frac{1}{|u|}\right\}, \tag{5}$$

and then define

$$\mathsf{closest}_\delta(n) = \{c \in W_1 : \|n-c\| \le \|n-c'\|\,(1+\delta) \text{ for all } c' \in W_1\} \cap \{N^-(n), N^+(n)\}. \tag{6}$$

An element of $\mathsf{closest}_\delta(n)$ is "almost" a closest element to n — its relative error is at most δ (i.e., if $c \in \mathsf{closest}_\delta(n)$ and $c^* \in \mathsf{closest}(n)$, then $\frac{\|n-c\|-\|n-c^*\|}{\|n-c^*\|} \le \delta$). Observe that by definition, for any nonzero integer n, $\mathsf{closest}_\delta(n) \subseteq \{N^-(n), N^+(n)\}$.[4]

We will see in Section 5 that it is possible to compute $c \in \mathsf{closest}_\delta(n)$ by examining only a fixed number (dependent on the value of δ) of the most significant digits of the binary representation of n. This is how we will decide between $N^-(n)$ and $N^+(n)$.

We phrase our main results with respect to Algorithm 3.

Algorithm 3 Compute $t = \mathsf{wt}^*(n)$.

Input: $n \in \mathbb{Z}$ (if $\ell = 0$, then we require $n \ge 0$)
Output: A nonnegative integer t and a list $c_1, c_2, \ldots c_t$ with $c_i \in W_1$ and $\sum_i c_i = n$.

```
1: t ← 0
2: while n ≠ 0 do
3:     t ← t + 1
4:     Choose c_t ∈ closest_δ(n)
5:     n ← n − c_t
6: return t, c_1, c_2, ... c_t
```

Note that Algorithm 3 is nondeterministic; i.e., for an input n, there can be more than one output. This is due to the fact that at line 4, there may be more than one choice for $c_t \in \mathsf{closest}_\delta(n)$.

Theorem 1. *For any valid input $n \in \mathbb{Z}$, Algorithm 3 terminates, and for the resulting output $t, c_1, c_2, \ldots, c_t$, we have $t = \mathsf{wt}^*(n)$ and $\sum_{i=1}^{t} c_i = n$.*

Of course, for a given $n \in \mathbb{Z}$, rather than a sum $c_1 + c_2 + \cdots + c_t = n$ with $\mathsf{wt}^*(n) = t$, what we really want is a string $\alpha \in D_{\ell,u}$ with $(\alpha)_2 = n$ and $\mathsf{wt}(\alpha) = \mathsf{wt}^*(n)$. It is possible to convert $c_1 + c_2 + \cdots + c_t = n$ into a minimal weight representation of n by assigning digits to each c_i. Note that elements of W_1 can have several representations $d \cdot 2^j$ with $d \in D_{\ell,u}$ and $j \ge 0$, and when we assign digits to each term of $c_1 + c_2 + \cdots + c_t$ we need to ensure that the resulting sequence of exponents is strictly decreasing. However, it turns out that every possible assignment of digits has this property.

Theorem 2. *Let $t, c_1, c_2, \ldots, c_t$ be the output of Algorithm 3 for any valid input n. Then every assignment of digits from $D_{\ell,u}$ to $c_1, c_2, \ldots, c_t$ yields a minimal weight representation of n.*

Our online algorithm, presented in Section 5, is essentially an implementation of Algorithm 3; it builds a minimal weight representation by encoding the list $c_1, c_2, \ldots, c_t$ as a string of digits from $D_{\ell,u}$.

[4] To ensure that $\mathsf{closest}_\delta(n) \subseteq \{N^-(n), N^+(n)\}$, we use an intersection operation in (6). However, when $\delta < 1/2$, it is easily shown that the intersection operation is unnecessary; i.e., whenever $\delta < 1/2$, then the set $\{c \in W_1 : \|n-c\| \le \|n-c'\|\,(1+\delta) \text{ for all } c' \in W_1\}$ is a subset of $\{N^-(n), N^+(n)\}$.

4 Proofs

Here we provide proofs for Proposition 3.1, Theorem 1, and Theorem 2. However, we first need to establish some facts about the elements of the set W_1.

4.1 The set W_1

In general, an element $c \in W_1$ can be written in different ways, e.g., we have $d \cdot 2^j = (2d) \cdot 2^{j-1}$ if $0 < d \leq u/2$ or $\ell/2 \leq d < 0$. There are at least two natural strategies to enforce a unique representation: one can require that d is odd or one can require that d is in a certain range (e.g., $u/2 < d \leq u$ or $\ell \leq d < \ell/2$). In our proofs, we will frequently adopt the second strategy; the main results, however, are independent of this choice by Theorem 2.

We define the following digit sets:

$$L := \{d \in D_{\ell,u} : \ell \leq d < \ell/2\} \quad \text{and} \quad U := \{d \in D_{\ell,u} : u/2 < d \leq u\}.$$

When $\ell = 0$, the set L is empty; otherwise, both ℓ and u are odd, and we have

$$\max L = (\ell - 1)/2 \quad \text{and} \quad \min U = (u + 1)/2,$$

and thus

$$\ell = \min L = 1 + 2 \max L \quad \text{and} \quad u = \max U = -1 + 2 \min U. \tag{7}$$

The following simple lemma will turn out to be useful.

Lemma 4.1. *For $d \in L \cup U$, we have $(d - 1) \in W_1 \cup \{0\}$ and $(d + 1) \in W_1 \cup \{0\}$.*

Proof. Assume that $d \in U$. For $d < u$, we clearly have $(d + 1) \in U \subseteq W_1$. For $d = u$, we have $d + 1 = 2 \min U$ by (7), which is an element of W_1. As for $d - 1$, since $0 < d \leq u$, we obviously have $0 \leq d - 1 \leq u - 1$ and therefore $d - 1 \in D_{\ell,u} \subseteq W_1 \cup \{0\}$.

The proof for $d \in L$ is analogous. □

Example 4.2. The sets L and U provide us with a convenient way to enumerate the elements of W_1 which are not in $D_{\ell,u}$. Consider $D_{-3,5}$. Then $L = \{-3, -2\}$ and $U = \{3, 4, 5\}$. It is easily shown that any $c \in W_1$ with $c \notin D_{-3,5}$ appears in some row of the array below.

$$\begin{array}{rrcrrr}
-6 & -4 & \qquad & 6 & 8 & 10 \\
-12 & -8 & & 12 & 16 & 20 \\
-24 & -16 & & 24 & 32 & 40 \\
\vdots & \vdots & & \vdots & \vdots & \vdots \\
-3 \cdot 2^j & -2 \cdot 2^j & & 3 \cdot 2^j & 4 \cdot 2^j & 5 \cdot 2^j
\end{array}$$

Observe that, for the exponent j, we always have $j \geq 1$. ◊

Lemma 4.3. *Let n be an integer with $\mathsf{wt}^*(n) > 1$ and $c \in \{N^-(n), N^+(n)\}$. Then there is some $d \in L \cup U$ and integer $j \geq 1$ such that $c = d \cdot 2^j$.*

Proof. Since $n \notin W_1$ and $n \neq 0$, we have $n < \ell - 1$ or $u + 1 < n$, and this implies that $N^-(n), N^+(n)$ are not in $D_{\ell,u}$. We write $c = d \cdot 2^j$ for some $d \in D_{\ell,u}$ and integer j which is chosen as small as possible. This implies that $d \in L \cup U$, since otherwise, replacing d by $2d$ and j by $j - 1$ would be possible. Since $c > u$ or $c < \ell$, we conclude that $j \geq 1$. □

Next, we are interested in the successor and predecessor functions on W_1:

Definition 4.4. Let $c \in W_1$. We define

$$\mathsf{succ}(c) = \min\{c' \in W_1 : c < c'\},$$
$$\mathsf{pred}(c) = \max\{c' \in W_1 : c' < c\}.$$

The successor and the predecessor functions can be computed explicitly:

Lemma 4.5. *Let $c = d \cdot 2^j \in W_1$ with $j \geq 1$ and $d \in L \cup U$. Then we have*

$$\mathsf{succ}(d \cdot 2^j) = \begin{cases} (d+1)2^j, & \text{if } d \neq \max L, \\ (2d+1)2^{j-1}, & \text{if } d = \max L, \end{cases}$$

$$\mathsf{pred}(d \cdot 2^j) = \begin{cases} (d-1)2^j, & \text{if } d \neq \min U, \\ (2d-1)2^{j-1}, & \text{if } d = \min U. \end{cases}$$

Proof. We only prove the lemma under the assumption that $d \in U$, the other case being analogous.

We first show that $\mathsf{succ}(d \cdot 2^j) = (d+1)2^j$. By Lemma 4.1, we know that $d+1 \in W_1$ and therefore $(d+1)2^j \in W_1$. Assume that there is an element $d' \cdot 2^{j'} \in W_1$, where $d' \in D_{\ell,u}$, with

$$d \cdot 2^j < d' 2^{j'} < (d+1)2^j.$$

There are no multiples of 2^j strictly between $d \cdot 2^j$ and $(d+1)2^j$, thus it must be that $j' < j$. Dividing by $2^{j'}$ yields

$$d \cdot 2^{j-j'} < d' < (d+1)2^{j-j'}.$$

However, $u + 1 \leq d \cdot 2^{j-j'} < d'$, and this contradicts the fact that $d' \in D_{\ell,u}$.

The predecessor function can be computed from the knowledge of the successor function: If $d > \min U$, then we just proved that $\mathsf{succ}((d-1)2^j) = d \cdot 2^j$, which implies that $\mathsf{pred}(d \cdot 2^j) = (d-1)2^j$, as required. If $d = \min U$, then we have $(2d-1)2^{j-1} = u \cdot 2^{j-1}$ by (7), and $\mathsf{succ}(u \cdot 2^{j-1}) = (u+1)2^{j-1} = \min U \cdot 2^j = d \cdot 2^j$ and we are done once again. □

From Lemma 4.5, we see, for example, that if an integer n has $N^-(n) = d \cdot 2^j$ with $d \in U$, then $N^+(n) = (d+1)2^j$. Similarly, if n has $N^+(n) = d \cdot 2^j$ with $d \in L$, then $N^-(n) = (d-1)2^j$. Another consequence of the lemma is that if $\mathsf{wt}(n) \geq 2$, then $N^+(n) - N^-(n) = 2^j$ for some $j \geq 1$.

4.2 Proposition 3.1

Proof of Proposition 3.1. From the subadditivity of wt^* (Proposition 2.4), it is clear that $\mathsf{wt}^*(n) \leq \mathsf{wt}^*(n - N^-(n)) + 1$ and $\mathsf{wt}^*(n) \leq \mathsf{wt}^*(n - N^+(n)) + 1$, so it only remains to show that the other direction holds for at least one of these inequalities.

Let $b_r \ldots b_1 b_0$ be a minimal weight representation of n with $b_r \neq 0$, and define the integer

$$k^* := \max\{k \in \mathbb{Z} : N^-(n) \leq (b_r \ldots b_k)_2 \cdot 2^k \leq N^+(n)\}.$$

Note that k^* is well defined since when $k = 0$ we have $(b_r \ldots b_k)_2 \cdot 2^k = n$ and $N^-(n) \leq n \leq N^+(n)$. Observe that $r \geq k^* \geq 0$.

If $r = k^*$, then $b_r 2^r \in \{N^-(n), N^+(n)\}$, and we get the desired inequality by observing that $\mathsf{wt}^*(n - b_r 2^r) \leq \mathsf{wt}^*(n) - 1$.

Assume that $r > k^*$. By the maximality of k^*, it must be that

$$(b_r \ldots b_{k^*+1} 0)_2 \cdot 2^{k^*} < N^-(n) \quad \text{or} \quad N^+(n) < (b_r \ldots b_{k^*+1} 0)_2 \cdot 2^{k^*}. \tag{8}$$

However, $N^-(n) \leq (b_r \ldots b_{k^*})_2 \cdot 2^{k^*} \leq N^+(n)$, and so we must have $b_{k^*} \neq 0$. Therefore, $b_r \ldots b_{k^*}$ contains at least two nonzero digits, and hence $\mathsf{wt}^*(n) > 1$.

Suppose n is positive. By Lemma 4.3, we can write $N^-(n) = d \cdot 2^j$ for some $d \in U$ and $j \geq 1$. And by Lemma 4.5, we have $N^+(n) = (d+1) \cdot 2^j$. Now,

$$d \cdot 2^j \leq (b_r \dots b_{k^*})_2 \cdot 2^{k^*} \leq (d+1) \cdot 2^j.$$

Since there are no multiples of 2^j strictly between $d \cdot 2^j$ and $(d+1) \cdot 2^j$, we see that either $k^* < j$ or $(b_r \dots b_{k^*})_2 \cdot 2^{k^*}$ equals $d \cdot 2^j$ or $(d+1) \cdot 2^j$. However, the latter possibility gives $n = d \cdot 2^j + (b_{k^*-1} \dots b_1 b_0)_2$ or $n = (d+1)2^j + (b_{k^*-1} \dots b_1 b_0)_2$, and each of these sums yields a representation of n with too few nonzero digits by the subadditivity of wt^* (Proposition 2.4). Therefore, $k^* < j$.

By (8), $(b_r \dots b_{k^*+1} 0)_2 \cdot 2^{k^*}$ is either less than $d \cdot 2^j$ or greater than $(d+1) \cdot 2^j$. In the first possibility, we have

$$\begin{aligned} & (b_r \dots b_{k^*+1} 0)_2 \cdot 2^{k^*} < d \cdot 2^j < (b_r \dots b_{k^*+1} b_{k^*})_2 \cdot 2^{k^*} \\ \Longrightarrow\ & (b_r \dots b_{k^*+1} 0)_2 < d \cdot 2^{j-k^*} < (b_r \dots b_{k^*+1} 0)_2 + b_{k^*}. \end{aligned}$$

From this last inequality, we see that there must exist some $a \in D_{\ell,u}$ with $0 < a < b_{k^*}$ such that

$$\begin{aligned} & d \cdot 2^{j-k^*} + a = (b_r \dots b_{k^*+1} b_{k^*})_2 \\ \Longrightarrow\ & d \cdot 2^j + a \cdot 2^{k^*} = (b_r \dots b_{k^*+1} b_{k^*})_2 \cdot 2^{k^*} \\ \Longrightarrow\ & d \cdot 2^j + a \cdot 2^{k^*} + (b_{k^*-1} \dots b_0)_2 = (b_r \dots b_0)_2 \\ \Longrightarrow\ & d \cdot 2^j + (a b_{k^*-1} \dots b_0)_2 = n. \end{aligned}$$

Thus, we have a representation $(a b_{k*-1} \dots b_0)_2$ of $n - N^-(n)$ of weight $\mathsf{wt}^*(n) - 1$, which implies that $\mathsf{wt}^*(n - N^-(n)) \leq \mathsf{wt}^*(n) - 1$.

In the second possibility (i.e., $(d+1) \cdot 2^j < (b_r \dots b_{k^*+1} 0)_2 \cdot 2^{k^*}$), using similar reasoning we obtain a sum

$$(d+1) \cdot 2^j + (a b_{k^*-1} \dots b_0)_2 = n$$

where $a \in D_{\ell,u}$ and $b_{k^*} < a < 0$. This gives us $\mathsf{wt}^*(n - N^+(n)) \leq \mathsf{wt}^*(n) - 1$.

Note that when $\ell = 0$, all the digits of a minimal weight representation $b_r \dots b_1 b_0$ of n are nonnegative. Thus, for all k with $r \geq k \geq 0$, we have $(b_r \dots b_k)_2 \cdot 2^k \leq n < N^+(n)$. This implies that for the parameter k^* we have

$$(b_r \dots b_{k^*+1} 0)_2 \cdot 2^{k^*} < N^-(n),$$

i.e., we are always in the first case and obtain (4).

The case where n is negative is handled in the same manner. This proves the result. ☐

4.3 The set $\mathsf{closest}_\delta(n)$

Before we can proceed with the remaining proofs, we require a result on the set $\mathsf{closest}_\delta(n)$. This result tells us which integers n, between $N^-(n)$ and $N^+(n)$, have $N^-(n) \in \mathsf{closest}_\delta(n)$ and which have $N^+(n) \in \mathsf{closest}_\delta(n)$.

Lemma 4.6. *Let n be a nonzero integer. Then*

$$N^-(n) \in \mathsf{closest}_\delta(n) \iff n \leq N^-(n) + \frac{|u|(1+\delta)}{|\ell| + |u|(1+\delta)}(N^+(n) - N^-(n)), \tag{9}$$

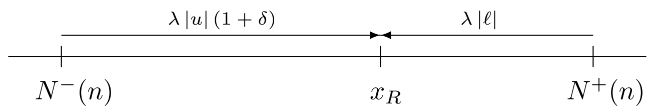
$$N^+(n) \in \mathsf{closest}_\delta(n) \iff n \geq N^+(n) - \frac{|\ell|(1+\delta)}{|u| + |\ell|(1+\delta)}(N^+(n) - N^-(n)). \tag{10}$$

Proof. If $n \in W_1$, then $n = N^-(n) = N^+(n)$, and each equivalence reduces to $n \in \{n\} \iff n \leq n$, which is clearly true. Thus, we may assume $n \notin W_1$.

By the definition of $\mathsf{closest}_\delta(n)$ in (6), $N^-(n) \in \mathsf{closest}_\delta(n)$ is equivalent to

$$\begin{aligned} &\left\|n - N^-(n)\right\| \leq \left\|n - N^+(n)\right\| (1+\delta) \\ \iff &\frac{|n - N^-(n)|}{|n - N^+(n)|} \leq \frac{|u|\,(1+\delta)}{|\ell|}. \end{aligned}$$

Geometrically, this is equivalent to $n \leq x_R$, where x_R is the point subdividing the interval $[N^-(n), N^+(n)]$ with the ratio $|u|\,(1+\delta) : |\ell|$, as illustrated in the following diagram:



Calculating x_R explicitly as $N^-(n)$ plus a positive constant λ times $|u|\,(1+\delta)$ yields

$$x_R = N^-(n) + \frac{|u|\,(1+\delta)}{|\ell| + |u|\,(1+\delta)} \left(N^+(n) - N^-(n)\right),$$

which gives us (9).

Similarly, we have $N^+(n) \in \mathsf{closest}_\delta(n)$ if and only if

$$\frac{|n - N^+(n)|}{|n - N^-(n)|} \leq \frac{|\ell|\,(1+\delta)}{|u|},$$

which is equivalent to $x_L \leq n$, where x_L is the point subdividing the interval $[N^-(n), N^+(n)]$ with the ratio $|u| : |\ell|\,(1+\delta)$. Calculating x_L explicitly as $N^+(n)$ minus a positive constant times $|\ell|\,(1+\delta)$ yields

$$x_L = N^+(n) - \frac{|\ell|\,(1+\delta)}{|u| + |\ell|\,(1+\delta)} \left(N^+(n) - N^-(n)\right),$$

which gives us (10). $\square$

For the numbers x_L and x_R defined in the previous proof, we justify our choice of notation as follows. By definition, we have $x_L < x < x_R$, where x is the point subdividing the interval with the ratio $|u| : |\ell|$. So, x_L is always to the left of x, and x_R is always to the right. Note that $N^-(n) \in \mathsf{closest}(n)$ if and only if $n \leq x$ and $N^+(n) \in \mathsf{closest}(n)$ if and only if $x \leq n$.

Example 4.7. Consider the digit set $D_{-3,5}$. For all n with $4 \cdot 2^8 \leq n \leq 5 \cdot 2^8$, we have $N^-(n) = 4 \cdot 2^8$ and $N^+(n) = 5 \cdot 2^8$. We set $\delta = 1/8$ and use Lemma 4.6 to describe $\mathsf{closest}_\delta(n)$ for n in this range. Note that $\delta = 1/8$ is a valid choice for δ as $1/8 < \min\{\frac{1}{|-3|}, \frac{1}{|5|}\}$.

According to the lemma, for an integer n with $1024 = 4 \cdot 2^8 \leq n \leq 5 \cdot 2^8 = 1280$, we have

$$\begin{aligned} 4 \cdot 2^8 \in \mathsf{closest}_\delta(n) &\iff n \leq \left\lfloor 4 \cdot 2^8 + \frac{5(1+1/8)}{3 + 5(1+1/8)} 2^8 \right\rfloor = 1190, \\ 5 \cdot 2^8 \in \mathsf{closest}_\delta(n) &\iff n \geq \left\lceil 5 \cdot 2^8 - \frac{3(1+1/8)}{5 + 3(1+1/8)} 2^8 \right\rceil = 1177. \end{aligned}$$

From this, we conclude that

$$\begin{aligned} 1024 \leq n \leq 1176 &\implies \mathsf{closest}_\delta(n) = \{1024\}, \\ 1177 \leq n \leq 1190 &\implies \mathsf{closest}_\delta(n) = \{1024, 1280\}, \\ 1191 \leq n \leq 1280 &\implies \mathsf{closest}_\delta(n) = \{1280\}. \end{aligned}$$

By using a larger value of δ, the number of integers with $\mathsf{closest}_\delta(n) = \{1024, 1280\}$ can be increased. By using a smaller value of δ, the number of integers with $\mathsf{closest}_\delta(n) = \{1024, 1280\}$ can be decreased. $\Diamond$

4.4 Theorem 1

With the following lemma, the proof of Theorem 1 follows easily.

Lemma 4.8. *Let n be a nonzero integer and $c \in \mathsf{closest}_\delta(n)$. Then $\mathsf{wt}^*(n) = \mathsf{wt}^*(n-c)+1$.*

Proof. For $\ell = 0$, we have $\mathsf{closest}_\delta(n) = \{N^-(n)\}$. Thus the result follows from Proposition 3.1. So we restrict ourselves to the case $\ell < 0$.

By subadditivity (Proposition 2.4), we have $\mathsf{wt}^*(n) \le \mathsf{wt}^*(n-c) + \mathsf{wt}^*(c) = \mathsf{wt}^*(n-c)+1$. So, we only have to prove the other direction. We prove this by induction on $\mathsf{wt}^*(n)$. When $n \in W_1$, $\mathsf{closest}_\delta(n) = \{n\}$ and the result is clearly true. Thus we may assume that $\mathsf{wt}^*(n) > 1$.

We consider the case that $c = N^+(n)$ (the other case $c = N^-(n)$ follows from analogous arguments or simply by considering $-n$ and the digit set $D_{-u,-\ell}$). By Proposition 3.1, we have $\mathsf{wt}^*(n) = \mathsf{wt}^*(n - N^+(n)) + 1$ or $\mathsf{wt}^*(n) = \mathsf{wt}^*(n - N^-(n)) + 1$. In the former case, we are done. Thus we consider the latter case. By Lemma 4.5, we have

$$c - N^-(n) = N^+(n) - N^-(n) = 2^j$$

for an appropriate nonnegative integer j.

We set $m = n - N^-(n)$. By assumption, we have $\mathsf{wt}^*(m) = \mathsf{wt}^*(n) - 1$. Since $c = N^+(n) \in \mathsf{closest}_\delta(n)$, we have

$$0 < c - n \le \frac{|\ell|\,(1+\delta)}{|u| + |\ell|\,(1+\delta)} 2^j \tag{11}$$

by Lemma 4.6.

Let

$$c_1 := \begin{cases} N^+(m), & \text{if } N^+(m) \in \mathsf{closest}_\delta(m), \\ N^-(m), & \text{if } N^+(m) \notin \mathsf{closest}_\delta(m), \end{cases}$$

which ensures that $c_1 \in \mathsf{closest}_\delta(m)$. If $c_1 < m$, we have $N^+(m) \notin \mathsf{closest}_\delta(m)$, which is equivalent to

$$m < N^-(m) + \frac{|u|}{|u| + |\ell|\,(1+\delta)} \left(N^+(m) - N^-(m)\right)$$

by Lemma 4.6. In every case, we have

$$m < c_1 + \frac{|u|}{|u| + |\ell|\,(1+\delta)} \left(N^+(m) - N^-(m)\right). \tag{12}$$

By the induction hypothesis, we have $\mathsf{wt}^*(m - c_1) = \mathsf{wt}^*(m) - 1$. We write $c_1 = d2^k$ for a $d \in D_{\ell,u}$ and a $k \ge 0$. We obtain the estimate $N^+(m) - N^-(m) \le 2^k$. Since $0 \le m = n - N^-(n) < 2^j$, we also have $2^k \le d \cdot 2^k \le N^+(m) \le 2^j$, which implies $k \le j$.

If $d \ge 2^{j-k} + \ell$, then $(d - 2^{j-k}) \in D_{\ell,u}$. Using the subadditivity of wt^* (Proposition 2.4), we obtain

$$\begin{aligned} \mathsf{wt}^*(n-c) &= \mathsf{wt}^*(m - c_1 + c_1 - 2^j) = \mathsf{wt}^*((m-c_1) + (d - 2^{j-k})2^k) \\ &\le \mathsf{wt}^*(m - c_1) + 1 = \mathsf{wt}^*(m) = \mathsf{wt}^*(n) - 1 \end{aligned}$$

and we are done.

For the remainder of the proof, we can therefore assume that

$$d \le 2^{j-k} + \ell - 1. \tag{13}$$

From (11), we get

$$m \ge \frac{|u|}{|u| + |\ell|\,(1+\delta)} \cdot 2^j. \tag{14}$$

Combining (12) and (14) yields

$$|u|\,(2^{j-k}-1) < d(|u| + |\ell|\,(1+\delta)). \tag{15}$$

Using the trivial estimate $d \le u = |u|$ yields

$$2^{j-k} - 1 < |u| + |\ell|\,(1+\delta).$$

Since $|\ell|\,\delta < 1$ by (5), this can be sharpened to

$$2^{j-k} - 1 \le |u| + |\ell|\,. \tag{16}$$

However, equality in (16) would imply that 2^{j-k} equals an odd number ≥ 3, which is clearly a contradiction. Thus, we can further sharpen (16) to

$$2^{j-k} - |\ell| - 1 \le |u| - 1. \tag{17}$$

After converting absolute value signs, inserting (13) in (15) yields

$$u(2^{j-k}-1) \le (2^{j-k} + \ell - 1)(u - \ell(1+\delta)),$$

which is equivalent to

$$0 \le -u + (1+\delta)(2^{j-k} + \ell - 1).$$

Inserting (17) and $\delta < 1/u$ into this last inequality yields

$$0 \le -u + \left(1 + \frac{1}{u}\right)(u-1) = -\frac{1}{u},$$

a contradiction. ☐

Proof of Theorem 1. Any integer n is a valid input to Algorithm 3 when $\ell \ne 0$, but when $\ell = 0$ we require $n \ge 0$.

The theorem follows directly from Lemma 4.8 and the fact that the only integer m for which $\mathsf{wt}^*(m) = 0$ is $m = 0$. ☐

Note that by Lemma 4.8 and the fact that $c \in \mathsf{closest}(n)$ implies $c \in \mathsf{closest}_\delta(n)$, we have now established the result

$$c \in \mathsf{closest}(n) \implies \mathsf{wt}^*(n-c) = \mathsf{wt}^*(n) - 1.$$

4.5 Theorem 2

Proof of Theorem 2. The result is clearly true for all inputs n with $\mathsf{wt}^*(n) = 1$. Thus, we assume $\mathsf{wt}^*(n) \ge 2$. Write each c_i as $d_i 2^{j_i}$ where $d_i \in D_{\ell,u}$ and $j_i \ge 0$. We show that $j_1 > j_2 > \cdots > j_t$. Note that since $t-1, c_2, \ldots, c_t$ is an output of Algorithm 3 for the input $n - c_1$, to conclude that $j_1 > j_2 > \cdots > j_t$, we need only prove that $j_1 > j_2$.

We consider the case $c_2 > 0$. The other case is analogous or can even be handled by considering $-n$ and the digit set $D_{-u,-\ell}$. Since $c_2 > 0$ it must be that $c_1 = N^-(n)$. Note that c_2 equals either $N^-(n-c_1)$ or $N^+(n-c_1)$. By Lemma 4.6, we have

$$n - c_1 \le \frac{|u|(1+\delta)}{|\ell| + |u|(1+\delta)} \left(N^+(n) - N^-(n)\right)$$

and, assuming that $c_2 = N^+(n-c_1)$,

$$n - c_1 - c_2 \ge -\frac{|\ell|(1+\delta)}{|u| + |\ell|(1+\delta)} \left(N^+(n-c_1) - N^-(n-c_1)\right).$$

However, when $c_2 = N^-(n-c_1)$, this last inequality is trivially true since its left side is non-negative and the right side is negative. Combining these two estimates yields

$$c_2 \le (1+\delta)\left(\frac{|u|\,(N^+(n)-N^-(n))}{|\ell|+|u|(1+\delta)} + \frac{|\ell|\,(N^+(n-c_1)-N^-(n-c_1))}{|u|+|\ell|(1+\delta)}\right). \tag{18}$$

By Lemma 4.3, we have $c_1 = d_1 2^{j_1'}$ for some $d_1' \in L \cup U$ and $j_1' \ge 1$. By construction and Lemma 4.5, we have

$$N^+(n) - N^-(n) \le 2^{j_1'} \quad \text{and} \quad 1 \le j_1' \le j_1. \tag{19}$$

If $c_2 < \min U$, then we set $j_2' = 0$, otherwise, we may write $c_2 = d_2' 2^{j_2'}$ for $d_2' \in U$ and $j_2' \ge 0$. In both cases, we have

$$N^+(n-c_1) - N^-(n-c_1) \le 2^{j_2'} \quad \text{and} \quad j_2' \le j_2. \tag{20}$$

We assume that $j_2 \ge j_1$ and set $j = \max\{j_1', j_2'\}$. From (19) and (20) it is clear that $j \le j_2$. From (18), (19) and (20), we obtain

$$2^{j_2} \le c_2 = d_2 2^{j_2} \le \left(\frac{|u|(1+\delta)}{|\ell|+|u|(1+\delta)} 2^{j_1'-j} + \frac{|\ell|(1+\delta)}{|u|+|\ell|(1+\delta)} 2^{j_2'-j}\right) 2^{j_2} 2^{j-j_2}. \tag{21}$$

We claim that

$$\frac{1}{2} \cdot \frac{|u|(1+\delta)}{|\ell|+|u|(1+\delta)} + \frac{|\ell|(1+\delta)}{|u|+|\ell|(1+\delta)} < 1, \tag{22}$$

$$\frac{|u|(1+\delta)}{|\ell|+|u|(1+\delta)} + \frac{1}{2} \cdot \frac{|\ell|(1+\delta)}{|u|+|\ell|(1+\delta)} < 1. \tag{23}$$

Indeed, (22) is equivalent to

$$|\ell|\left((1+\delta)^2 - 2\right) < |u|(1+\delta).$$

If $(1+\delta)^2 \le 2$, this is obviously true. Otherwise, by (5), we have $|\ell| = |u| = 1$ and are left with

$$(1+\delta)^2 - 2 < 1 + \delta,$$

which is true for $\delta < 1$. The estimate (23) follows analogously.

If $j_1' < j$ or $j_2' < j$, (21), (22) and (23) yield

$$2^{j_2} < 2^{j_2} 2^{j-j_2},$$

a contradiction. Thus we have $j_1' = j_2' = j \ge 1$. In this case, we use the estimate

$$\frac{|u|(1+\delta)}{|\ell|+|u|(1+\delta)} + \frac{|\ell|(1+\delta)}{|u|+|\ell|(1+\delta)} \le \frac{|u|(1+\delta)}{|u|+|\ell|} + \frac{|\ell|(1+\delta)}{|u|+|\ell|} = (1+\delta),$$

and (21) yields

$$2^{j_2} \le d_2 2^{j_2} \le (1+\delta) 2^{j_2} 2^{j-j_2},$$

which implies $j_2 = j$ and therefore also $j_1 = j$ and $c_2 = 2^{j_2}$. By definition of d_2', we obtain $1 \in U$. This implies that $U = \{1\}$.

Since $2 \le 2^j$ and $c_2 = 2^j \in \mathsf{closest}_\delta(n-c_1)$, we conclude that $n - c_1 > 0$ and therefore $n > c_1$, which implies that $n < \mathsf{succ}(c_1) \le c_1 + 2^j$. Thus $n - c_1 < 2^j = c_2$, which implies that $c_2 = N^+(n-c_1)$ and therefore $N^-(n-c_1) = c_2/2$ and $N^+(n-c_1) - N^-(n-c_1) = 2^{j-1}$. Plugging this in (18) and using (23) yields a contradiction. □

5 Online implementations

As shown in Example 3.4, it is, in general, impossible to decide which of $N^-(n)$ and $N^+(n)$ is closest to n without knowing the full binary representation of n. To circumvent this problem, the sets $\mathsf{closest}_\delta$ have been studied. The purpose of this section is to explicitly demonstrate how this relaxation can be used to determine an element of $\mathsf{closest}_\delta$ by only reading a bounded number of digits of the binary representation. This will result in a refinement of Algorithm 3 to an online algorithm, which could also be implemented by a transducer automaton.

As the cases $\ell = 0$ and $\ell < 0$ differ substantially, we treat them in different subsections. Nevertheless, before forking the discussion, we note how $N^-(n)$ can be read from the digits of the binary representation of n.

Let $b_r \dots b_1 b_0$ be the binary representation of an integer $n \geq 0$. Then for any i with $r \geq i \geq 0$, we have

$$(b_r \dots b_i)_2 \cdot 2^i \leq n < (b_r \dots b_i)_2 \cdot 2^i + 2^i.$$

Suppose that $(b_r \dots b_i)_2 \in U$. Then Lemma 4.5 gives us

$$N^-(n) = (b_r \dots b_i)_2 \cdot 2^i \quad \text{and} \quad N^+(n) = ((b_r \dots b_i)_2 + 1) \cdot 2^i. \tag{24}$$

5.1 $\ell = 0$

The case $\ell = 0$ can be handled quite easily. When $\ell = 0$ we have no need of δ or the set $\mathsf{closest}_\delta(n)$. The set $\mathsf{closest}(n)$ is always equal to $\{N^-(n)\}$, so we simply compute $N^-(n)$ using (24).

Algorithm 4 Compute a minimal weight representation of n for $\ell = 0$.

Input: $b_r \dots b_1 b_0$, the binary representation of an integer n.
Output: $a_r \dots a_1 a_0$, a minimal weight representation of n with each $a_i \in D_{0,u}$.

$U \leftarrow \{a \in \mathbb{Z} : u/2 < a \leq u\}$
$d \leftarrow 0$
for $i = r$ **downto** 0 **do**
  $d \leftarrow 2d + b_i$
  {We have $m := n - \sum_{k=i+1}^{r} a_k 2^k = d2^i + \sum_{k=0}^{i-1} b_k 2^k$}
  if $d \in U$ **then**
    {We have $N^-(m) = d2^i$}
    $a_i \leftarrow d,\ d \leftarrow 0$
  else
    {We have $0 \leq d \leq (u-1)/2$}
    $a_i \leftarrow 0$
if $d \neq 0$ **then**
  $a_0 \leftarrow d$
return $a_r \dots a_1 a_0$

The invariants stated as comments in Algorithm 4 are easily verified by an inductive proof. From these, the correctness of the algorithm follows.

5.2 $\ell < 0$

Fix any $\delta < \min\{\frac{1}{|\ell|}, \frac{1}{|u|}\}$. Let $b_r \dots b_1 b_0$ be the binary representation of an integer $n \geq 0$ and assume that (24) holds. We must now determine $c \in \{N^-(n), N^+(n)\}$ such that $c \in \mathsf{closest}_\delta(n)$ by reading no more than some finite number of digits to the right of b_i (i.e., we must make a correct decision using only a finite *look-ahead*).

By Lemma 4.6, there are numbers x_L, x_R with $N^-(n) < x_L < x_R < N^+(n)$ such that

$$N^-(n) \in \mathsf{closest}_\delta(n) \iff n \le x_R \quad \text{and} \quad N^+(n) \in \mathsf{closest}_\delta(n) \iff n \ge x_L.$$

As $N^+(n) - N^-(n) = 2^i$, we have

$$x_L = N^-(n) + y_L \cdot 2^i \quad \text{and} \quad x_R = N^-(n) + y_R \cdot 2^i$$

where

$$y_L := \frac{|u|}{|u| + |\ell|\,(1+\delta)} \quad \text{and} \quad y_R := \frac{|u|\,(1+\delta)}{|\ell| + |u|\,(1+\delta)}.$$

Since $N^-(n) < x_L < x_R < N^+(n)$, we have $0 < y_L < y_R < 1$. We choose a number Y with $y_L \le Y \le y_R$ and use it to decide membership in $\mathsf{closest}_\delta(n)$, as explained below. We will sometimes write $x_L(\delta)$, $x_R(\delta)$, $y_L(\delta)$, and $y_R(\delta)$ instead of x_L, x_R, y_L, y_R when we need to emphasize the parameter δ involved.

Note that the parameter δ serves only to define the endpoints y_L, y_R of a subinterval of $[0, 1]$ from which we select Y. After Y is selected, δ may be discarded as it is not utilized in our implementation. In fact, implementors are free to choose whatever $Y \in [y_L, y_R]$ they wish; however, we will suggest a method that has the advantage that it minimizes the length of the required look-ahead (i.e., no matter what other values of δ or Y might be considered, they cannot result in a shorter length look-ahead).

We set $\delta^* = \min\{\frac{1}{|\ell|}, \frac{1}{|u|}\}$ and

$$y_L^* := y_L(\delta^*) = \frac{|u|}{|u| + |\ell|(1+\delta^*)}, \qquad y_R^* := y_R(\delta^*) = \frac{|u|\,(1+\delta^*)}{|\ell| + |u|\,(1+\delta^*)}.$$

By definition, $0 < y_L^* < y_R^* < 1$. Consider any number Y in the open interval (y_L^*, y_R^*) that has a finite binary representation $Y = (0.h_{-1}h_{-2}\dots h_t)_2$. There must exist some positive $\delta < \delta^*$ with $Y \in [y_L(\delta), y_R(\delta)]$. We now prove the following result based on a look-ahead of $|t|$ digits.

Lemma 5.1. *Let $n = d2^i + (b_{i-1}\dots b_1 b_0)_2$ be an integer partly given by its binary representation and define $b_k = 0$ for all $k < 0$. Suppose that $d2^i = N^-(n)$ and $(d+1)2^i = N^+(n)$. Then there is a δ with $0 < \delta < \delta^*$ such that*

1. *$(b_{i-1}b_{i-2}\dots b_{i+t})_2 < (h_{-1}h_{-2}\dots h_t)_2$ implies $N^-(n) \in \mathsf{closest}_\delta(n)$, and*
2. *$(b_{i-1}b_{i-2}\dots b_{i+t})_2 \ge (h_{-1}h_{-2}\dots h_t)_2$ implies $N^+(n) \in \mathsf{closest}_\delta(n)$.*

Proof. By the definition of Y, there exists some positive $\delta < \delta^*$ with $Y \in [y_L(\delta), y_R(\delta)]$.

Consider the first case. We have

$$\begin{aligned}
&(b_{i-1}b_{i-2}\dots b_{i+t})_2 < (h_{-1}h_{-2}\dots h_t)_2 \\
\implies\ &(0.b_{i-1}b_{i-2}\dots b_{i+t})_2 < (0.h_{-1}h_{-2}\dots h_t)_2 \\
\implies\ &(0.b_{i-1}b_{i-2}\dots b_0)_2 < Y \le y_R(\delta) \\
\implies\ &(b_{i-1}b_{i-2}\dots b_0)_2 < y_R(\delta) \cdot 2^i \\
\implies\ &N^-(n) + (b_{i-1}b_{i-2}\dots b_0)_2 < N^-(n) + y_R(\delta) \cdot 2^i.
\end{aligned}$$

From this last inequality, we conclude that $n < x_R(\delta)$, and therefore $N^-(n) \in \mathsf{closest}_\delta(n)$.

Consider the second case. Using similar reasoning, we see that

$$\begin{aligned}
&(b_{i-1}b_{i-2}\dots b_{i+t})_2 \ge (h_{-1}h_{-2}\dots h_t)_2 \\
\implies\ &N^-(n) + (b_{i-1}b_{i-2}\dots b_0)_2 \ge N^-(n) + y_L(\delta) \cdot 2^i.
\end{aligned}$$

Thus, $n \ge x_L(\delta)$, and therefore $N^+(n) \in \mathsf{closest}_\delta(n)$. $\square$

To determine a Y with the required properties and minimal $|t|$, we can use the binary representations of

$$y_L^* = (0.f_{-1}f_{-2}\ldots)_2 \quad \text{and} \quad y_R^* = (0.g_{-1}g_{-2}\ldots)_2, \tag{25}$$

where in case of non-uniqueness, we choose the representation ending with infinitely many 0 for y_L^* and the representation ending with infinitely many 1 for y_R^*. We choose $t^* < 0$ maximal such that

$$f_k = g_k \text{ for all } k > t^* \text{ and } f_{t^*} = 0,\ g_{t^*} = 1. \tag{26}$$

This is always possible since $y_L^* < y_R^*$. We take $Y = (0.g_{-1}g_{-2}\ldots g_{t^*})_2$. Note that this choice indeed yields $y_L^* < Y < y_R^*$. With this choice of Y, Lemma 5.1 can be translated to Algorithm 5.

Algorithm 5 Compute a minimal weight representation of n for $\ell < 0$.

Input: $b_r \ldots b_1 b_0$, the binary representation of an integer n.
Output: $a_{r+1}\ldots a_1 a_0$, a minimal weight representation of n with each $a_i \in D_{\ell,u}$.

determine t^* and $g_{-1}g_{-2}\ldots g_{t^*}$ according to (26) and (25)
$U \leftarrow \{a \in \mathbb{Z} : u/2 < a \le u\}$
$\widehat{L} \leftarrow \{a \in \mathbb{Z} : \ell - 1 \le a \le (\ell-3)/2\}$
$d \leftarrow 0, a_{r+1} \leftarrow 0, b_{-1} \leftarrow 0, b_{-2} \leftarrow 0, \ldots, b_{t^*} \leftarrow 0$
for $i = r$ **downto** 0 **do**
  $d \leftarrow 2d + b_i$
  {We have $m := n - \sum_{k=i+1}^{r+1} a_k 2^k = d2^i + \sum_{k=0}^{i-1} b_k 2^k$}
  if $d \in \widehat{L} \cup U$ **then**
    {We have $N^-(m) = d2^i$, $N^+(m) = (d+1)2^i$}
    if $(b_{i-1}\ldots b_{i+t^*})_2 < (g_{-1}g_{-2}\ldots g_{t^*})_2$ **then**
      {$d2^i \in \mathsf{closest}_\delta(m)$ for an appropriate $\delta < \delta^*$}
      $a_i \leftarrow d,\ d \leftarrow 0$
    else
      {$(d+1)2^i \in \mathsf{closest}_\delta(m)$ for an appropriate $\delta < \delta^*$}
      $a_i \leftarrow d+1,\ d \leftarrow -1$
    if $a_i \in \{\ell-1, u+1\}$ **then**
      $a_{i+1} \leftarrow a_i/2, a_i \leftarrow 0$
  else
    $a_i \leftarrow 0$
  {We have $(\ell-1)/2 \le d \le (u-1)/2$}
if $d \ne 0$ **then**
  $a_0 \leftarrow d$
return $a_{r+1}\ldots a_1 a_0$

Theorem 3. *Algorithm 5 terminates and is correct. In particular, it computes a minimal weight representation of an integer from its binary representation from left to right with only a finite look-ahead.*

Proof. The invariants stated as comments in the algorithm are easily verified by an inductive proof and using Lemma 5.1. Note that instead of L, the set $\widehat{L} = \{\ell-1, \ldots, (\ell-3)/2\}$ has been chosen so that the successor of $d2^i$ can be written without needing an extra case distinction (i.e., if $d \in \widehat{L}$, then $\mathsf{succ}(d2^i) = (d+1)2^i$ with $d+1 \in L$). The invariants immediately imply the correctness of the algorithm, its termination being immediate. ∎

References

[1] R. Avanzi, *A note on the signed sliding window integer recoding and its left-to-right analogue*, in "Selected Areas in Cryptography 2004". Lecture Notes in Computer Science **3357** (2005), pp. 130–143.

[2] P. Ganesan and G. Singh Manku, *Optimal routing in chord*, in "Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms", 2004, pp. 176–185.

[3] P. J. Grabner, C. Heuberger, H. Prodinger, and J. Thuswaldner, *Analysis of linear combination algorithms in cryptography*, ACM Transactions on Algorithms **1** (2005), pp. 123–142.

[4] C. Heuberger, R. Katti, H. Prodinger, and X. Ruan, *The alternating greedy expansion and applications to left-to-right algorithms in cryptography*, Theoretical Computer Science **341** (2005), pp. 55–72.

[5] C. Heuberger and J. Muir, *Minimal weight and colexicographically minimal integer representations*. Journal of Mathematical Cryptology **1** (2007), pp. 297–328.

[6] M. Joye and S. Yen, *Optimal left-to-right binary signed-digit recoding*. IEEE Transactions on Computers **49** (2000), pp. 740–748.

[7] M. Khabbazian, T. Gulliver and V. Bhargava, *A new minimal average weight representation for left-to-right point multiplication methods*. IEEE Transactions on Computers **54** (2005), pp. 1454–1459.

[8] B. Möller, *Fractional windows revisited: improved signed-digit representations for efficient exponentiation*, in "Information Security and Cryptology – ICISC 2004", Lecture Notes in Computer Science **3506** (2004), pp. 137–153.

[9] J. Muir, *A simple left-to-right algorithm for minimal weight signed radix-r representations*. IEEE Transactions on Information Theory **53** (2007), pp. 1234–1241.

[10] J. Muir and D. Stinson, *New minimal weight representations for left-to-right window methods*, in "Cryptographers Track of the RSA Conference – CT-RSA 2005", Lecture Notes in Computer Science **3376** (2005), pp. 366–383.

[11] V. Müller, *Fast multiplication on elliptic curves over small fields of characteristic two*. Journal of Cryptology **11** (1998), pp. 219–234.

[12] K. Okeya, K. Schmidt-Samoa, C. Spahn and T. Takagi, *Signed binary representations revisited*, in "Advances in Cryptology – CRYPTO 2004". Lecture Notes in Computer Science **3152** (2004) pp. 123–139.

[13] B. Phillips and N. Burgess, *Minimal weight digit set conversions*. IEEE Transactions on Computers **53** (2004), pp. 666–677.

[14] G. Reitwiesner, *Binary arithmetic*, in Advances in Computers, Vol. 1, Academic Press (1960), pp. 231–308.

[15] J. Shallit, *A primer on balanced binary representations*, unpublished manuscript, 1993. Available from `http://www.cs.uwaterloo.ca/~shallit/Papers/bbr.pdf`.

[16] J. Solinas, *Efficient arithmetic on Koblitz curves*. Designs, Codes and Cryptography **19** (2000), pp. 195–249.