Discrete Mathematics and Theoretical Computer Science DMTCS vol. **11:1**, 2009, 153–172

Diophantine Approximation, Ostrowski Numeration and the Double-Base Number System

Valérie Berthé[1] and Laurent Imbert[1 2]

[1] *CNRS, LIRMM, Université Montpellier 2, 161 rue Ada, 34390 Montpellier, France*
[2] *CNRS, PIMS, Centre for Information Security and Cryptography, University of Calgary, 2500 University Drive NW, Calgary, AB, T2N 1N4, Canada*

received July 16, 2008, *revised January 15, 2009*, *accepted March 6, 2009*.

A partition of $x > 0$ of the form $x = \sum_i 2^{a_i} 3^{b_i}$ with distinct parts is called a double-base expansion of x. Such a representation can be obtained using a greedy approach, assuming one can efficiently compute the largest $\{2, 3\}$-integer, i.e., a number of the form $2^a 3^b$, less than or equal to x. In order to solve this problem, we propose an algorithm based on continued fractions in the vein of the Ostrowski number system, we prove its correctness and we analyse its complexity. In a second part, we present some experimental results on the length of double-base expansions when only a few iterations of our algorithm are performed.

Keywords: Double-base number system, Ostrowski numeration, diophantine approximation

1 Introduction

In this paper, we consider representations of integers in two coprime bases. More exactly, we analyze decompositions of positive integers in the form

$$x = \sum_{i=1}^{\ell} 2^{a_i} 3^{b_i}, \tag{1}$$

with $a_i, b_i \geq 0$ and $(a_i, b_i) \neq (a_j, b_j)$ if $i \neq j$.

Apart from its practical interest in real life applications such as digital signal processing [13] and cryptography [10, 14], this so-called double-base number system (DBNS) has many intriguing properties and leads to interesting Diophantine approximation problems. We focus on bases 2 and 3 but most of the results and algorithms presented in this paper remain valid for arbitrary bases.

Clearly, such a decomposition always exists, the binary expansion is a special case of (1). In fact, this number system is even extremely redundant. For a given integer $x > 0$, the number $f(x)$ of DBNS representations is given by (see [11] for more details):

$$f(x) = \begin{cases} 1 & \text{if } x = 1, \\ f(x-1) + f(x/3) & \text{if } x \equiv 0 \pmod 3, \\ f(x-1) & \text{otherwise.} \end{cases}$$

It is not difficult to see that the elements of this sequence go by triples and correspond to the number of partitions of $3x$ into powers of 3, also called ternary partitions (see [3] and the sequence #A005704 from the on-line encyclopedia of integer sequences), whose first terms are:

$$1, 2, 3, 5, 7, 9, 12, 15, 18, 23, 28, 33, 40, 47, 54, 63, 72, 81, 93, 105, 117, \ldots$$

For a given $x > 0$, finding an expansion of minimal length ℓ (i.e., it is not possible to write x with $\ell - 1$ or fewer terms) is believed to be a difficult problem. Those representations of minimal length are extremely sparse. For example, among its 783 representations, the integer 127 can be written as a sum of exactly three terms in 6 different ways:

$$\begin{aligned}
127 &= 2^2 3^3 + 2^1 3^2 + 2^0 3^0 = 108 + 18 + 1 \\
127 &= 2^2 3^3 + 2^4 3^0 + 2^0 3^1 = 108 + 16 + 3 \\
127 &= 2^5 3^1 + 2^0 3^3 + 2^2 3^0 = 96 + 27 + 4 \\
127 &= 2^3 3^2 + 2^1 3^3 + 2^0 3^0 = 72 + 54 + 1 \\
127 &= 2^6 3^0 + 2^1 3^3 + 2^0 3^2 = 64 + 54 + 9 \\
127 &= 2^6 3^0 + 2^2 3^2 + 2^0 3^3 = 64 + 36 + 27
\end{aligned}$$

This observation suggests that 127 cannot be expressed with fewer than three terms. In fact, the smallest integer requiring three summands is 23 (as pointed out in [27] by Tijdeman); the smallest integer requiring four summands is 431. Similarly, the next smallest numbers requiring five, six and seven summands are 18431, 3448733, and 1441896119 (a 21-bit integer), respectively.

For large numbers, such as those used in cryptographic applications, finding a representation of minimal length, in a reasonable amount of time, seems very hard. Fortunately, one can use a greedy approach (see Algorithm 1) to find a fairly sparse representation very quickly, based on the determination of the best default approximation of x (i.e., the largest integer $\leq x$) of the form $z = 2^a 3^b$.

Algorithm 1 Greedy decomposition

Input : An integer $x > 0$
Output : The sequence of exponents $(a_i, b_i)_{i>0}$ s.t. $x = \sum_{i=1}^{\ell} 2^{a_i} 3^{b_i}$ as in (1)

```
while x ≠ 0 do
    Find the best default approximation of x of the form z = 2^a 3^b
    Print (a, b)
    x := x − z
```

Although it sometimes fails in finding a minimal representation (the smallest example is 41: the minimal representation is $32 + 9$, whereas Algorithm 1 returns $41 = 36 + 4 + 1$), it is very easy to implement.

More importantly, it guarantees an expansion of sublinear length ℓ. Indeed, an important theorem from Dimitrov [12] states that ℓ is in $O(\log x / \log\log x)$. The proof is based on a result by Tijdeman [26], which says that there exists an absolute constant C such that there is always a number of the form $2^a 3^b$ in the interval $\left[x - x/(\log x)^C, x\right]$. See [12] for a complete proof.

In this paper we investigate the problem of finding the best default approximation of x of the form $z = 2^a 3^b$. This operation is clearly of crucial importance for Algorithm 1. We present an algorithm as well as inhomogeneous approximation results in the vein of a number system from Ostrowski [20], which uses the series of convergents of the continued fraction expansion of an irrational number.

We introduce the problem in Diophantine terms in Section 2 and recall basic facts on Ostrowski's numeration in Section 3. In Section 4, we describe our algorithm and prove its correctness. Moreover, we show that for all x of bounded binary length, our algorithm terminates in $O(\log\log x)$ iterations. The natural extension to signed digits and some implementation considerations are discussed in Section 5. We then conclude by presenting some numerical experiments in Section 6. The present paper is an extended version of [7].

2 Problem and notation

Let $x \geq 2$ be a given positive integer. We want to find two integers $a, b \geq 0$ such that $2^a 3^b \leq x$ and among the solutions to this problem, $2^a 3^b$ is the largest possible value. In other words

$$2^a 3^b = \max\left\{2^c 3^d \leq x : (c,d) \in \mathbb{N}^2\right\}. \tag{2}$$

Equivalently, one can solve the following Diophantine inequality in non-negative integers

$$a \log 2 + b \log 3 \leq \log x, \tag{3}$$

with the constraint that no other integers $c, d \geq 0$ give a better left approximation to $\log x$; meaning that if (a,b) is a maximal solution of (3), then for all integers c, d with $(c,d) \neq (a,b)$, we have

$$c\alpha + d < a\alpha + b \leq \log_3 x, \quad \text{where } \alpha = \log_3 2.$$

In the following, $\{t\}$ denotes the fractional part of t. Graphically, the solutions to $c\alpha + d \leq \log_3 x$ are the points (c,d) with non-negative integer coordinates located under the line of equation

$$v = -\alpha u + \log_3 x.$$

These are the integer points in the gray area of Figure 1. Among those points, the best left approximation of x we are looking for is the point (a,b), which has the smallest vertical distance to the line, that we denote by $\delta(u) = \{-\alpha u + \log_3 x\}$, for $u \geq 0$.

A naive approach consists in computing $\delta(u)$ for every integer $0 \leq u \leq \log_2 x$ and to keep the point (u,v) which gives the smallest value. Since $\log_3 x < \log_2 x$, a little more efficient solution is to exchange the coordinate axis, i.e., to consider the line $v' = u'/\alpha - \log_2 x$, and to keep the closest point among all integers $0 \leq u' \leq \log_3 x$ (we perform here the change of coordinates system $u' = v$, $v' = -u$). Let us note that in some cases, it may be interesting to compute the best right approximation, i.e., the smallest integer of the form $2^a 3^b$ greater than x. Observe that it can be easily obtained by symmetry, by looking for the point (a', b') located under and with the smallest vertical distance to the line of equation $v' = -\alpha u' + \alpha\lceil\log_2 x\rceil + \lceil\log_3 x\rceil - \log_3(x)$ (the dashed line in Figure 1), and to apply the change of coordinates system $u = \lceil\log_2 x\rceil - u'$, $v = \lceil\log_3 x\rceil - v'$.

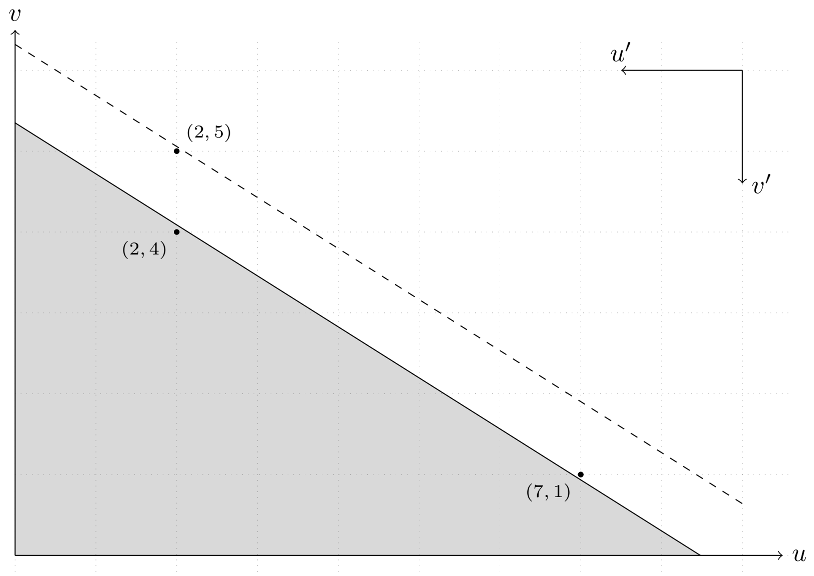


Fig. 1: Graphical interpretation of the problem

Example 1 *In Figure 1, we consider an example with* $x = 358$*. We have* $\log_2 x \approx 8.48$*,* $\log_3 x \approx 5.35$*. The best left approximation is the point of coordinates* $(2, 4)$*. Similarly the closest point below the dashed line is the point* $(2, 5)$*, which by symmetry gives the best right approximation:* $(9 - 2, 6 - 5) = (7, 1)$*. We easily verify that* $2\alpha + 4 \approx 5.26 < \log_3 x < 7\alpha + 1 \approx 5.41$*.*

The complexity of all those graphical approaches is $O(\log x)$. In the next sections, we give an algorithm which solves the problem in $O(\log \log x)$.

In the following, we define $\beta = \{\log_3 x\}$. We assume $0 < \beta < 1$. Indeed, if $\beta = 0$, then we take $a = 0$ and $b = \log_3 x \in \mathbb{N}$. Clearly, if (a, b) is a maximal solution of (3), then we have

$$\lfloor \log_3 x \rfloor \leq a\alpha + b \leq \beta + \lfloor \log_3 x \rfloor,$$

with $0 \leq a \leq \lfloor \log_2 x \rfloor$ and $0 \leq b \leq \lfloor \log_3 x \rfloor$. Indeed, the pair $(c, d) = (0, \lfloor \log_3 x \rfloor)$ is a solution which provides $c\alpha + d = \lfloor \log_3 x \rfloor$.

Let us briefly describe the strategy followed in the present paper. We set $k = a$ and $l = \lfloor \log_3 x \rfloor - b$. We have $0 \leq k\alpha - l \leq \beta$.

We thus are looking for $(k, l) \in \mathbb{N}^2$ such that $0 \leq k\alpha - l \leq \beta$ and

$$k\alpha - l = \max_{(r,s) \in \mathbb{N}^2} \{r\alpha - s : 0 \leq r\alpha - s \leq \beta,\ 0 \leq r \leq \lfloor \log_2 x \rfloor,\ 0 \leq s \leq \lfloor \log_3 x \rfloor\}. \tag{4}$$

We shall define two increasing sequences $(k_n)_{n \geq 0}, (l_n)_{n \geq 0}$ such that, for some $j > 0$, $k_j = k$, $l_j = l$ satisfy (4). We then set $a = k$ and $b = \lfloor \log_3 x \rfloor - l$ to get the solution of (2).

We will consider this classical inhomogeneous approximation problem in Section 4, where the sequence of inhomogeneous best approximations of β by points of the form $N\alpha$ will be explicitly given. For that purpose, we first introduce Ostrowski's number system.

3 Ostrowski's number system

In this section we introduce a number system due to Ostrowski [20]. Let us first recall some basic facts about continued fractions (see [17] for more details).

A simple continued fraction is an expression of the form

$$\alpha = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cdots}}},$$

where $a_0 = \lfloor \alpha \rfloor$ and $a_1, a_2, \ldots$ are integers ≥ 1. The sequence $(a_n)_{n \in \mathbb{N}}$ of partial quotients can either be finite or infinite.

Every rational number α can be expressed as a finite simple continued fraction, denoted by $\alpha = [a_0, a_1, \ldots, a_n]$ in the more convenient compact notation. Similarly, every irrational number α can be expressed uniquely as an infinite simple continued fraction. We write $\alpha = [a_0, a_1, a_2, \ldots]$. For example, the continued fraction decomposition of $\log 2 / \log 3 = 0.630929753571\ldots$ is

$$\log_3 2 = [0, 1, 1, 1, 2, 2, 3, 1, 5, 2, 23, 2, \ldots].$$

The rational number obtained by restricting the continued fraction of α to its first $n+1$ partial quotients

$$\frac{p_n}{q_n} = [a_0, a_1, a_2, \ldots, a_n]$$

is called the nth convergent of α. The integers p_n, q_n are easily computable; we have $p_{-1} = 1$, $q_{-1} = 0$, $p_0 = a_0$, $q_0 = 1$, and

$$\begin{aligned} p_{n+1} &= a_{n+1}\, p_n + p_{n-1} \\ q_{n+1} &= a_{n+1}\, q_n + q_{n-1}. \end{aligned}$$

It is well known that the sequences $(p_n)_{n \geq 0}, (q_n)_{n \geq 0}$ satisfy $\lim_{n \to \infty} p_n / q_n = \alpha$ and $p_{n+1}q_n - p_n q_{n+1} = (-1)^n$. Moreover, simple continued fractions provide the sequence of best rational approximations of an irrational number.

Ostrowski's number system [20] (see [2] for an introduction) is associated with the numeration scale $(q_n)_{n \geq 0}$ of denominators of the convergents of the continued fraction expansion of an irrational number $0 < \alpha < 1$. The following proposition holds.

Proposition 1 *Let $0 < \alpha < 1$ be an irrational number. Every integer N can be written uniquely in the form*

$$N = \sum_{k=1}^{m} b_k q_{k-1},$$

where $0 \leq b_1 \leq a_1 - 1$; $0 \leq b_k \leq a_k$, for $k \geq 2$ and $b_k = 0$ if $b_{k+1} = a_{k+1}$.

For example, for $\alpha = \frac{1+\sqrt{5}}{2} = [1, 1, 1, 1, \ldots]$, the sequence $(q_n)_{n\geq 0}$ corresponds to the Fibonacci numbers [28]. The unicity condition $b_k = 0$ if $b_{k+1} = a_{k+1}$ means that we do not have two consecutive ones in the corresponding Zeckendorf representation [29].

Ostrowski's representation of integers can be extended to real numbers (see e.g. the survey [5]). The base is given by the sequence $(\theta_n)_{n\geq 0}$, where $\theta_n = q_n\alpha - p_n$. Note that the sign of θ_n is equal to $(-1)^n$.

Proposition 2 *Let $0 < \alpha < 1$ be an irrational number. Every real number $-\alpha \leq \beta < 1 - \alpha$ can be written uniquely in the form*

$$\beta = \sum_{k=1}^{+\infty} b_k \theta_{k-1},$$

where $0 \leq b_1 \leq a_1 - 1$; $0 \leq b_k \leq a_k$, for $k \geq 2$; $b_k = 0$ if $b_{k+1} = a_{k+1}$, and $b_k \neq a_k$ for infinitely many odd integers.

Proposition 2 can be used to approximate β modulo 1 by numbers of the form $N\alpha$, with $N \in \mathbb{N}$. Indeed, one verifies that the integers

$$N_n = \sum_{k=1}^{n} b_k q_{k-1}$$

provides a series of arbitrarily good approximations to β on both sides, i.e., the fractional part of $N_n\alpha$ can either be greater or smaller than β.

If we are only interested in the best left approximations to β, which is our case, we can express β in base $(|\theta_n|)_{n\geq 0}$ (see Section 4 below for the exact definition of best left approximations). The following proposition holds.

Proposition 3 *Let $0 < \alpha < 1$ be an irrational number. Every real number $0 \leq \beta < 1$ can be written uniquely in the form*

$$\beta = \sum_{k=1}^{+\infty} d_k |\theta_{k-1}|,$$

where $0 \leq d_k \leq a_k$ for $k \geq 1$; $d_{k+1} = 0$ if $d_k = a_k$ and $d_k \neq a_k$ for infinitely many even and odd integers.

In this case, the sequence of best left approximations is more difficult to define due to the alternating signs of θ_n. Indeed, the corresponding numeration system on integers is defined with respect to the numeration scale $((-1)^n q_n)_{n\geq 0}$. Nevertheless, none of these expansions provides the sequence of best left approximations (see Remark 3 below for more details). In the next sections, we present an algorithm inspired by [22] and [6], we prove its correctness and analyse its complexity.

4 The sequence of inhomogeneous best approximations of β

From now on we assume α irrational, $0 < \alpha < 1$ and $0 < \beta \leq 1$. Inhomogeneous left approximations of β are numbers of the form $k\alpha + l \leq \beta$, with k, l integers. Clearly, there exist infinitely many such approximations. We want to define two increasing sequences of integers $(k_n)_{n\geq 0}$ and $(l_n)_{n\geq 0}$, such that

$$0 < k_n\alpha - l_n < k_{n+1}\alpha - l_{n+1} < \beta,$$

and, furthermore, $\forall k < k_{n+1}$, $k \neq k_n$, and $\forall l \in \mathbb{Z}$ such that $0 \leq k\alpha - l \leq \beta$, we have

$$0 < k\alpha - l < k_n\alpha - l_n < \beta.$$

For simplicity, we define $f_n = |\theta_n|$ for all n. We have $f_{-1} = 1$, $f_0 = \alpha$, $f_1 = 1 - a_1\alpha$, and for $n \in \mathbb{N}$

$$f_{n-1} = a_{n+1}f_n + f_{n+1}. \tag{5}$$

One has $f_n > 0$ for all n, since α is irrational. Hence, the sequence $(f_{n-1} + f_n)_{n\geq 0}$ is strictly decreasing and tends to zero. Since $0 < \beta \leq 1$, there exists a unique integer $n \geq 0$ such that

$$f_n + f_{n+1} < \beta \leq f_{n-1} + f_n. \tag{6}$$

Before we give the algorithm that defines the series of inhomogeneous best left approximations of β, we need to prove the following two lemmas.

Lemma 1 *Let $0 < \beta \leq 1$ and $(f_n)_{n\geq 0}$ be defined as above. There exists a unique integer $n \geq 0$, a unique integer $c \geq 1$, and a unique $e \in \mathbb{R}$ such that*

$$\beta = cf_n + f_{n+1} + e, \tag{7}$$

with $0 < e \leq f_n$, and $1 \leq c \leq a_1 - 1$ if $n = 0$, $1 \leq c \leq a_{n+1}$ if $n \geq 1$.

Proof: If $n \in \mathbb{N}$ satisfies (7), then $f_n + f_{n+1} < \beta \leq f_{n-1} + f_n$, and n is uniquely determined since $(f_n + f_{n+1})_{n\geq -1}$ is strictly decreasing. Now, let $n \in \mathbb{N}$ be the unique integer that satisfies $f_n + f_{n+1} < \beta \leq f_{n-1} + f_n$. If $n \geq 1$, then $f_n + f_{n+1} < \beta \leq a_{n+1}f_n + f_{n+1} + f_n$ by (5). If $n = 0$, then $f_0 + f_1 < \beta \leq 1 = f_{-1} = (a_1 - 1)f_0 + f_1 + f_0$. (Note that $a_1 \geq 2$ in this case.) □

Lemma 2 *Let α irrational, $0 < \alpha < 1$. Let $0 < \beta \leq 1$ and $(f_n)_{n\geq 0}$ be defined as above, and let n, c, e be the unique numbers satisfying (7). We define $k, l \in \mathbb{N}$ by setting*

$$(k, l) = \begin{cases} (q_n, p_n) & \text{if } n \text{ is even,} \\ (-cq_n + q_{n+1}, -cp_n + p_{n+1}) & \text{if } n \text{ is odd.} \end{cases}$$

Then we have $0 < k\alpha - l < \beta$.

Proof: Let us prove that $0 < \beta - (k\alpha - l) < \beta$. Assume first that n is even. In this case, $q_n\alpha - p_n = f_n$. We have $\beta - (k\alpha - l) = \beta - f_n$, and thus $0 < \beta - (k\alpha - l) < \beta$. Now, if n is odd, $q_n\alpha - p_n = -f_n$ and thus $\beta - (k\alpha - l) = \beta + c(q_n\alpha - p_n) - (q_{n+1}\alpha - p_{n+1}) = \beta - cf_n - f_{n+1} = e$, and thus, $0 < \beta - (k\alpha - l) \leq f_n < \beta$. □

Algorithm 2 below computes the infinite sequence $(k_n\alpha - l_n)_{n\geq 1}$ of inhomogeneous best left approximations to β.

Algorithm 2 Inhomogeneous best left approximations to β

Input : Two real numbers $0 < \alpha < 1$ and $0 < \beta \leq 1$, with α irrational
Output : The infinite sequence $(k_i, l_i)_{i\geq 1}$ of best left approximations to β with $0 < k_i\alpha - l_i < \beta, \forall i$

```
(k_0, l_0) := (0, 0)
while true do
    Compute n_i, c_i, e_i such that β − (k_i α − l_i) = c_i f_{n_i} + f_{n_i+1} + e_i
    if n_i is even then
        (k_{i+1}, l_{i+1}) := (k_i + q_{n_i}, l_i + p_{n_i})
    else
        (k_{i+1}, l_{i+1}) := (k_i − c_i q_{n_i} + q_{n_i+1}, l_i − c_i p_{n_i} + p_{n_i+1})
```

This algorithm is inspired by [22]; similar ones can be found in [23, 24, 25]. Note that $\beta-(k_{i+1}\alpha-l_{i+1})$ is equal to e_i if n_i is odd, and to $(c_i-1)f_{n_i}+f_{n_i+1}+e_i$, if n_i is even. Hence, we may have $n_{i+1}=n_i$. This happens if and only if n_i is even and $c_i > 1$; it therefore happens (c_i-1) times before the sequence (n_i) keeps growing towards $+\infty$. We thus have

$$\beta = \sum_{n_i \text{ even}} c_i f_{n_i} + \sum_{n_i \text{ odd}} (c_i f_{n_i} + f_{n_i+1}). \tag{8}$$

The following proposition proves the correctness of Algorithm 2.

Proposition 4 *Let $0 < \alpha < 1$ and $0 < \beta \leq 1$ be given. We assume α irrational. The increasing sequences of integers $(k_i)_{i\geq 0}$ and $(l_i)_{i\geq 0}$ provided by Algorithm 2 satisfy*

$$0 < k_i\alpha - l_i < k_{i+1}\alpha - l_{i+1} < \beta, \quad \forall i \geq 0 \tag{9}$$

and furthermore, for all i, for all k such that $k_i < k < k_{i+1}$, and for all $l \in \mathbb{Z}$ such that $0 \leq k\alpha - l < \beta$, then

$$0 \leq k\alpha - l < k_i\alpha - l_i < \beta. \tag{10}$$

Proof: We first prove (9). From Lemma 2, we know that, for all i, $0 < k_i\alpha - l_i < \beta$. We consider two cases:

1. If n_i is even, then $\beta > k_{i+1}\alpha - l_{i+1} = k_i\alpha - l_i + q_{n_i}\alpha - p_{n_i} = k_i\alpha - l_i + f_{n_i} > k_i\alpha - l_i > 0$.
2. If n_i is odd, then $\beta > k_{i+1}\alpha - l_{i+1} = k_i\alpha - l_i - c_i(q_{n_i}\alpha - p_{n_i}) + q_{n_i+1}\alpha - p_{n_i+1} = k_i\alpha - l_i + c_i f_{n_i} + f_{n_i+1} > k_i\alpha - l_i > 0$.

Let us now prove (10). Let us assume that $k_i < k < k_{i+1}$, and $0 \leq k\alpha - l < \beta$. By rewriting $\beta-(k\alpha-l)$, we get that $0 \leq \beta-(k\alpha-l) = \beta-(k_i\alpha-l_i)+(k_i\alpha-l_i-k_{i+1}\alpha+l_{i+1})+(k_{i+1}\alpha-l_{i+1}-k\alpha+l) < \beta$. What we prove in the next two cases depending on the parity of n_i, is that $\beta-(k\alpha-l) > \beta-(k_i\alpha-l_i)$.

1. Assume first that n_i is even. We have

$$\beta-(k\alpha-l) = \beta-(k_i\alpha-l_i) - f_{n_i} + (k_{i+1}\alpha - l_{i+1} - k\alpha + l).$$

Thus, what remains to be proved is that $(k_{i+1}\alpha - l_{i+1} - k\alpha + l)$ is greater than f_{n_i}. Since $|k_{i+1} - k| < |k_{i+1} - k_i| = q_{n_i}$, one has $|k_{i+1}\alpha - l_{i+1} - k\alpha + l| > f_{n_i}$ (we use the best approximation property of continued fractions, see e.g. [8]). Next we show that $(k_{i+1}\alpha - l_{i+1} - k\alpha + l)$ cannot be negative by considering two cases. Note first that, from (5) and line 3 of Algorithm 2, we have $|\beta - (k_i\alpha - l_i) - f_{n_i}| \leq f_{n_i-1}$.

(a) If $k_{i+1} - k \neq q_{n_i-1}$, then $|(k_{i+1}\alpha - l_{i+1} - k\alpha + l)| > f_{n_i-1}$, and since $0 \leq k\alpha - l < \beta$, we have $(k_{i+1}\alpha - l_{i+1} - k\alpha + l) > 0$.

(b) If $k_{i+1} - k = q_{n_i-1}$, since $n_i - 1$ is odd, we have $(k_{i+1}\alpha - l_{i+1} - k\alpha + l) = (q_{n_i-1}\alpha - p_{n_i-1}) = -f_{n_i-1} < 0$. And we get $\beta - (k\alpha - l) = \beta - (k_i\alpha - l_i) - f_{n_i} - f_{n_i-1} \leq 0$, which contradicts the hypothesis.

2. If we now assume that n_i is odd, we have

$$\beta - (k\alpha - l) = \beta - (k_i\alpha - l_i) - (c_i f_{n_i} + f_{n_i+1}) + (k_{i+1}\alpha - l_{i+1} - k\alpha + l).$$

Here, what remains to be proved is that $(k_{i+1}\alpha - l_{i+1} - k\alpha + l)$ is greater than $c_i f_{n_i} + f_{n_i+1}$. Since $|k_{i+1} - k| < |k_{i+1} - k_i| = q_{n_i+1} - c_i q_{n_i}$, we have $|k_{i+1}\alpha - l_{i+1} - k\alpha + l| > f_{n_i+1} + c_i f_{n_i}$. We know from (7) and Algorithm 2, that $|\beta - (k_i\alpha - l_i) - c_i f_{n_i} - f_{n_i+1}| \leq f_{n_i}$. Hence $(k_{i+1}\alpha - l_{i+1} - k\alpha + l) > 0$, which implies $(k_{i+1}\alpha - l_{i+1} - k\alpha + l) > c_i f_{n_i} + f_{n_i+1}$.

Thus, in both cases $\beta - (k\alpha - l) > \beta - (k_i\alpha - l_i)$, which concludes the proof. $\square$

As stated in Section 2, we want to find two integers $a, b \geq 0$ such that

$$2^a 3^b = \max\left\{2^c 3^d : (c,d) \in \mathbb{N}^2, \text{ and } 2^c 3^d \leq x\right\}.$$

Let $(a,b) \in \mathbb{N}^2$ be one of the solutions of the equivalent Diophantine inequality

$$a \log 2 + b \log 3 \leq \log x.$$

Let $\alpha = \log_3 2$ (note that α is irrational and $0 < \alpha < 1$), and $\beta = \{\log_3 x\}$. Recall that we assume $0 < \beta < 1$. We have $a\alpha + b \leq \beta + \lfloor \log_3 x \rfloor$ with $0 \leq a \leq \lfloor \log_2 x \rfloor$ and $0 \leq b \leq \lfloor \log_3 x \rfloor$.

Now, we use a slightly modified version of Algorithm 2 (see Remark 4, below) to compute $(k,l) \in \mathbb{N}^2$ such that $0 \leq k\alpha - l \leq \beta$ and

$$k\alpha - l = \max_{(r,s)\in\mathbb{N}^2} \left\{r\alpha - s : 0 \leq r\alpha - s \leq \beta,\ 0 \leq r \leq \lfloor \log_2 x \rfloor,\ 0 \leq s \leq \lfloor \log_3 x \rfloor\right\}.$$

Proposition 5 *Let $x \geq 2$ be a fixed positive integer. Let $\alpha = \log_3 2$; note that α is irrational and $0 < \alpha < 1$. Let $\beta = \{\log_3 x\}$. We assume furthermore that $\beta \neq 0$. Let $(k_n, l_n)_{n\in\mathbb{N}}$ be the sequence of inhomogeneous best left approximations of β. Let v be the uniquely defined integer that satisfies $k_v \leq \lfloor \log_2 x \rfloor < k_{v+1}$. Then*

$$k_v\alpha - l_v = \max_{(r,s)\in\mathbb{N}^2} \left\{r\alpha - s : 0 \leq r\alpha - s \leq \beta,\ 0 \leq r \leq \lfloor \log_2 x \rfloor,\ 0 \leq s \leq \lfloor \log_3 \rfloor\right\}.$$

Proof: From the proof of Proposition 4, we know that Algorithm 2 returns the infinite sequence $(k_i, l_i)_{i\geq 0}$ of best left approximations to $\beta = \{\log_3 x\}$. In order to find (k, l) as above, we only need to perform a finite number of iterations. In fact, this number of iterations is given by the smallest integer v such that $\lfloor \log_2 x \rfloor < k_v$, or equivalently, the smallest integer v such that $\lfloor \log_3 x \rfloor < l_v$. We use this bound in our complexity analysis (see proof of Proposition 6). □

Remark 1 *As pointed out in the proof of Proposition 5, we only need to perform a finite number of iterations of Algorithm 2. Remark that this number of iterations depends on the sequence of partial quotients in the continued fraction expansion of α. For the same reason, we can also use a rational approximation of $\alpha = \log_3 2$, say p_w/q_w, given by the wth convergent. Note that the required precision for the rational approximation of α depends on the binary length of the input x. This is why we state our complexity result for integers x with bounded binary length. We analyse the required precision in Remark 2.*

Proposition 6 *Let $m > 0$. For all x having at most m bits, the finite version of Algorithm 2 (see Remark 4, above) terminates in $O(\log\log x)$ iterations.*

Proof: We use the notation of Algorithm 2. According to (8), we obtain

$$\beta = \sum_{n_i \text{ even}} c_i f_{n_i} + \sum_{n_i \text{ odd}} (c_i f_{n_i} + f_{n_i+1}).$$

For $x \geq 2$, let $v(x)$ be the unique integer such that $k_{v(x)} \leq \lfloor \log_2 x \rfloor < k_{v(x)+1}$. If n_i is even, we compute $k_{i+1} = k_i + q_{n_i}$ and we have seen that this occurs $c_i - 1$ times. If n_i is odd, we compute $k_{i+1} = k_i - c_i q_{n_i} + q_{n_i+1}$. We have

$$k_{v(x)} = \sum_{n_i \text{ even}} c_i q_{n_i} + \sum_{n_i \text{ odd}} \left(-c_i q_{n_i} + q_{n_i+1}\right),$$

and thus

$$k_{v(x)} = \sum_{n_i \text{ even}} c_i q_{n_i} + \sum_{n_i \text{ odd}} \left((a_{n_{i+1}} - c_i) q_{n_i} + q_{n_i-1}\right).$$

Let $w(x) = \max_i\{n_i\}$. Since $a_{n_i+1} - c_i \geq 0$, for all $n_i \geq 1$, and $c_i \geq 1$, for all n_i, we have

$$q_{w(x)-1} \leq \sum_{n_i \text{ even}} q_{n_i} + \sum_{n_i \text{ odd}} q_{n_i-1} \leq k_{v(x)} \leq \lfloor \log_2 x \rfloor.$$

Moreover, we also know that the sequence $(q_i)_{i\geq 0}$ grows at least as fast as the sequence of Fibonacci numbers given by the denominators of the continued fraction expansion of $(1+\sqrt{5})/2 = [1, 1, 1, 1, \ldots]$ (all the partial quotients are equal to 1). Therefore, from Binet's formula (see [28]), we deduce that there exists a constant $C > 0$ such that

$$\forall x,\ w(x) < C \log\log x.$$

Consequently, there exists C' such that for all x having at most m bits, $w(x) < C' \log m$. We now set $A = \max_{i=1\ldots C'\log m}\{a_i\}$. For any m-bit integer x, the number of iterations $s(x)$ satisfies $s(x) \leq Aw(x)$. This concludes the proof. □

Corollary 1 *The greedy algorithm combined with Algorithm 2 is optimal.*

Proof: Since the greedy algorithm produces decompositions with $O(\log x/\log\log x)$ terms according to [12], its overall asymptotic complexity is $O(\log x)$, i.e., it reaches the lower bound for a recoding algorithm (all the bits/digits of x have to be scanned). $\square$

Remark 2 *Let us analyse the required precision for α in terms of the bitlength of x. From the proof of Proposition 6, we have*

$$k_v = \sum_{n_i \text{ even}} c_i q_{n_i} + \sum_{n_i \text{ odd}} (a_{n_i+1} - c_i) q_{n_i} + q_{n_i-1} \leq \lfloor \log_2 x \rfloor$$

with $a_{n_i+1} - c_i \geq 0$ for all $n_i \geq 1$, and $c_i \geq 1$ for all n_i. Therefore, if x is an m-bit integer, we only need the denominators q_i such that $q_i \leq m$ for all $i \geq 1$. Let $w = \max\{i : q_i \leq m\}$. Table 1 gives some useful numerical values for w.

Now, we need to know how many bits of precision μ are given by $\hat{\alpha} = \frac{p_w}{q_w}$, the wth rational approximation of α. It is well known (see [17]) that for all n

$$\left| \alpha - \frac{p_n}{q_n} \right| \leq \frac{1}{q_n q_{n+1}}.$$

Therefore, if $\frac{1}{q_{n+1} q_n} > 2^{-\mu}$, we obtain

$$\mu > \log_2 q_n + \log_2 q_{n+1}. \tag{11}$$

From (11), we can e.g. deduce that

$$1.11022... \times 10^{-16} \approx 2^{-53} < \frac{1}{q_{16} q_{17}} \approx 6.87368... \times 10^{-16},$$

which tells us that a double-precision binary floating-point approximation of $\alpha = \log_3 2$ provides roughly the same precision as the 16th convergent, and that it is "good-enough" for numbers of size up to 17 millions bits!

Size of x (in bits)	w	μ
3 to 7	4	6
8 to 18	5	9
19 to 64	6	12
65 to 83	7	14
84 to 484	8	18
484 to 1053	9	23

Tab. 1: Number w of partial quotients a_i, and convergents p_i/q_i to be computed based on the size of input x. The last column gives the number of bits of precision provided by p_w/q_w, the wth rational approximation of α.

Remark 3 *We have seen that Algorithm 2 provides an expansion of β of the form*

$$\beta = \sum_{n_i \text{ even}} c_i f_{n_i} + \sum_{n_i \text{ odd}} (c_i f_{n_i} + f_{n_i+1}).$$

Let us stress the fact that sequences $(n_i)_{i\geq 0}$ and $(c_i)_{i\geq 0}$ cannot be directly derived from the sequences of digits provided by Proposition 2 and 3 by performing finitely many operations. Note that these latter sequences of digits can be generated in an effective way by maps defined on the unit square as skew products of the Gauss map, such as described in [15] and [16].

A numerical example

We want to find the largest integer of the form $2^a 3^b \leq x = 23832098195$. We have $\lfloor \log_3 x \rfloor = 21$, $\beta = \{\log_3 x\} \approx 0.7495$. The partial quotients and the convergents of the continued fraction expansion of $\alpha = \log_3 2 \simeq 0.63093$ are given in Table 2.

i	a_i	p_i	q_i	$f_i = \lvert q_i\alpha - p_i \rvert$
0	0	0	1	0.63093
1	1	1	1	0.36907
2	1	1	2	0.26186
3	1	2	3	0.10721
4	2	5	8	0.04744
5	2	12	19	0.01234
6	3	41	65	0.01043

Tab. 2: Partial quotients and convergents of $\log_3 2$

Table 3 gives the first inhomogeneous best left approximations to β.

i	$\beta - (k_i\alpha - l_i)$	n_i	c_i	e_i	k_{i+1}	l_{i+1}	$k_{i+1}\alpha - l_{i+1}$
0	0.7495	1	1	0.1186	1	0	0.6309
1	0.1186	4	2	0.0114	9	5	0.6784
2	0.0712	4	1	0.0114	17	10	0.7258
3	0.0237	5	1	0.0009	63	39	0.7486

Tab. 3: Best left approximations of $\beta \simeq 0.7495$ with numbers of the form $k\alpha - l$

The stop condition (see Remark 4) is reached for $i = 3$ because we get a negative ternary exponent $(21 - 39 = -18)$. We thus retain the third approximation (0.7258), which gives $a = 17$ and $b = 21 - 10 = 11$. In order to find the DBNS representation of x, we then apply the same algorithm with the value $x - 2^{17}3^{11} = 613086611$. For completeness, we give the DBNS representation of x provided by the greedy decomposition: $x = 2^{17}3^{11} + 2^7 3^{14} + 2^7 3^8 + 2^2 3^8 + 2^9 3^0 + 2^2 3^1 + 2^0 3^1$.

5 Signed expansions

It is natural to consider signed expansions of the form

$$x = \sum \pm 2^{a_i} 3^{b_i},$$

with $a_i, b_i \geq 0$, $(a_i, b_i) \neq (a_j, b_j)$ if $i \neq j$. The following proposition shows that it is not possible to define a greedy algorithm based on a similar logarithm reduction approach, that is, by working with the sequence of two-sided inhomogeneous best approximations of β. More precisely, Proposition 7 below shows that it may give an erroneous solution infinitely often; this happens only when the logarithm reduction approach returns a solution greater than $\log x$ whereas the best approximation of x is possibly less than x.

Proposition 7 *Let $(K_n)_{n\in\mathbb{N}}$ be the increasing sequence of integers of the form $2^a 3^b$, with $a, b \in \mathbb{N}$. For any positive integer n, we define N_n as the number of positive integers x such that $K_n \leq x < K_{n+1}$ with*

1. $x - K_n \leq K_{n+1} - x$
2. $\log K_{n+1} - \log x < \log x - \log K_n$.

The sequence $(N_n)_{n\in\mathbb{N}}$ tends towards infinity.

Proof: Condition (1) is equivalent to $K_n \leq x \leq \dfrac{K_{n+1} + K_n}{2}$, whereas condition (2) is equivalent to $\dfrac{\log K_n + \log K_{n+1}}{2} < \log x < \log K_{n+1}$, that is,

$$\sqrt{K_n K_{n+1}} < x < K_{n+1}.$$

Hence N_n satifies

$$\Delta_n - 1 \leq N_n \leq \Delta_n + 1$$

with

$$\Delta_n = \frac{K_n + K_{n+1}}{2} - \sqrt{K_n K_{n+1}}.$$

For all n, we set $\delta_n = \dfrac{K_{n+1}}{K_n}$. One has

$$\Delta_n = \left(\frac{\delta_n + 1}{2} - \sqrt{\delta_n}\right) K_n = \frac{\left(\sqrt{\delta_n} - 1\right)^2}{2} K_n = \frac{\left(\delta_n - 1\right)^2}{2} K_n \frac{1}{\left(\sqrt{\delta_n} + 1\right)^2}.$$

According to [26], there exist C, C' with $C > 0$, $C' > 0$ such that for all n

$$\frac{1}{\log {K_n}^{C'}} \leq \delta_n - 1 = \frac{K_{n+1} - K_n}{K_n} \leq \frac{1}{\log {K_n}^{C}}.$$

We thus deduce that $\lim_{n\to+\infty} N_n = +\infty$. □

Nevertheless, we can apply the following strategy in the signed case. Let $x \geq 2$ be a given positive integer. We first find $a, b \geq 0$ such that $2^a 3^b \leq x$ and among the solutions to this problem, $2^a 3^b$ is the largest possible value:

$$2^a 3^b = \max\left\{2^c 3^d \leq x : (c,d) \in \mathbb{N}^2\right\}.$$

We similarly find $a', b' \geq 0$ such that $2^{a'} 3^{b'} \geq x$ and among the solutions to this problem, $2^{a'} 3^{b'}$ is the smallest possible value:

$$2^{a'} 3^{b'} = \min\left\{2^{c'} 3^{d'} \geq x : (c,d) \in \mathbb{N}^2\right\}.$$

For this latter problem, we can either deal with an analogous algorithm providing the sequence of inhomogeneous best right approximations of β, or work with $1-\alpha$ and $1-\beta$, with the notation of the previous sections. It then remains to compare $x - 2^a 3^b$ and $2^{a'} 3^{b'} - x$, and to take the smallest of both values.

6 An approximate greedy algorithm

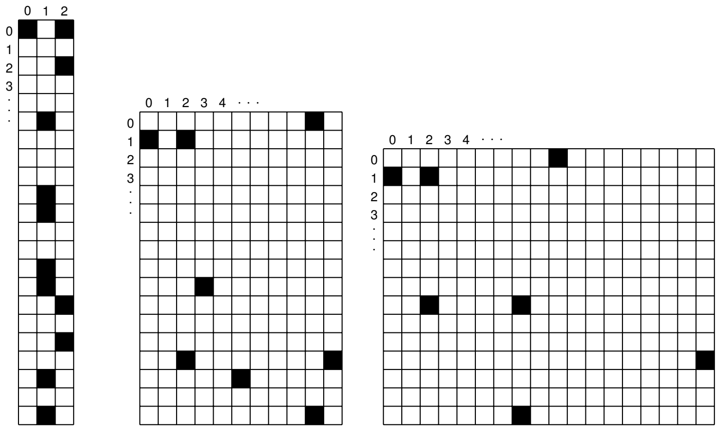
The greedy algorithm is not optimal, in the sense that it does not necessarily produce a DBNS representation of minimal length. However, within this algorithm, we proved that finding the largest number of the form $2^a 3^b$ less than equal to x is optimal in complexity. It seems then natural to investigate the potential advantages of an approximate greedy algorithm, where we only perform a few iterations of Algorithm 2 to define a "good" integer of the form $2^a 3^b$, although not the largest, less than or equal to x.

The greedy decomposition described in Algorithm 1 can easily be adapted to compute a DBNS expansion of x, where only d iterations of Algorithm 2 are performed at each step to define the term $z \leq x$. In the following, we shall refer to d as the depth.

In order to visualize DBNS expansions, we use a two-dimensional array where the columns represent the powers of 2 while the rows represent the powers of 3. A black square at position (i, j) indicates that the term $2^i 3^j$ is part of the DBNS decomposition. (The upper-left corner corresponds to the term $2^0 3^0 = 1$.) Figure 2 shows the DBNS representations obtained for $x = 23832098195$ when the greedy decomposition algorithm is applied at depth 1,2, and 3. (Note that depth 3 is equivalent to the complete greedy decomposition.)

As expected, the length of the DBNS decomposition decreases as the depth increases; we get 12 terms at depth 1; 8 terms at depth 2; and 7 terms at depth 3. We note also that the largest binary exponent (the number of columns) is smaller for small depths. These results were to be expected. However, the following questions seem difficult to answer in the general case. By using a greedy decomposition algorithm at a given depth d, how many summands are required to represent x compared to the solution provided by the complete greedy approach? What decrease (resp. increase) can we expect on the largest binary (resp. ternary) exponents? In order to answer those questions, we have implemented the algorithm at different depths, for a thousand randomly chosen numbers of sizes ranging from 128 to 512 bits. The difference in length between the full-greedy approach and the decomposition at fixed depths is shown in Figures 3 to 5. Our experiments are summarized in Table 4.

We first note that, for all our tests (up to 512-bit integers), $d = 5$ is equivalent to the complete greedy algorithm; which means that we perform 6 iterations of Algorithm 2 to get an approximation which corresponds to a negative ternary exponent (the stop condition, cf. Remark 4). With the greedy algorithm at fixed depth d, it is possible that the stop condition occurs before we reach the desired depth, or would occur at the $(d+1)$th iteration. Only in those two cases, we get the optimal solution; i.e, the largest



(a) depth 1 (b) depth 2 (c) depth 3

Fig. 2: DBNS decompositions of $x = 23832098195$ at depth 1,2,3

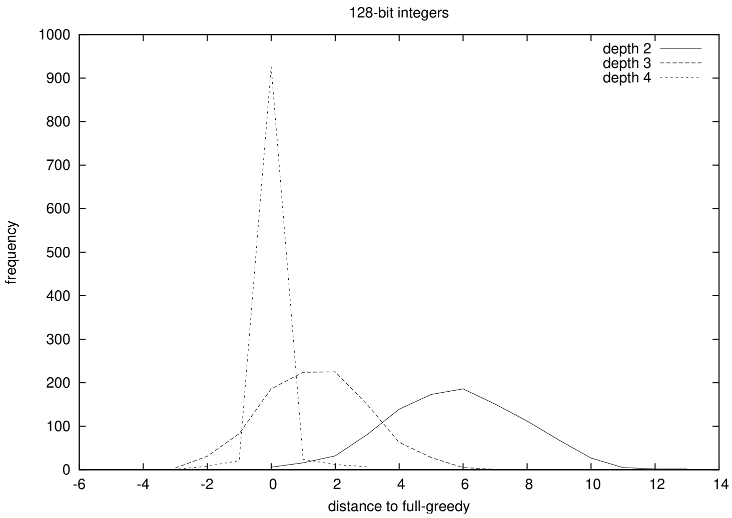


Fig. 3: Distribution of the length difference of DBNS expansions obtained with the full-greedy and the approximate algorithm at depths 2,3,4 for 128-bit numbers

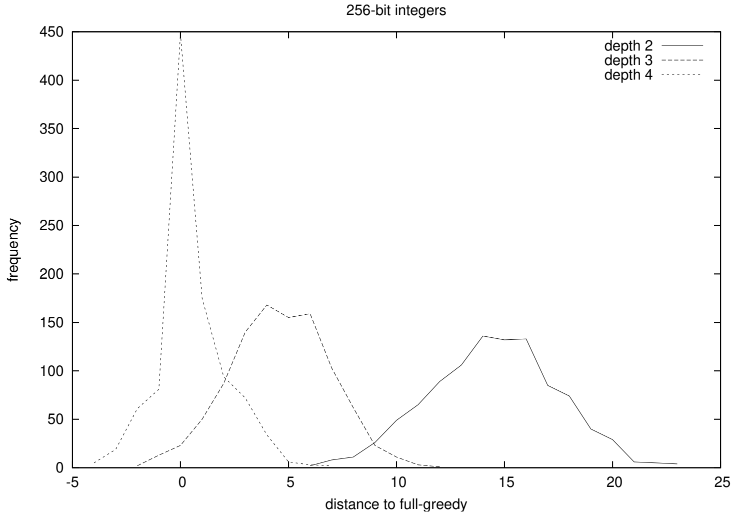


Fig. 4: Distribution of the length difference of DBNS expansions obtained with the full-greedy and the approximate algorithm at depths 2,3,4 for 256-bit numbers

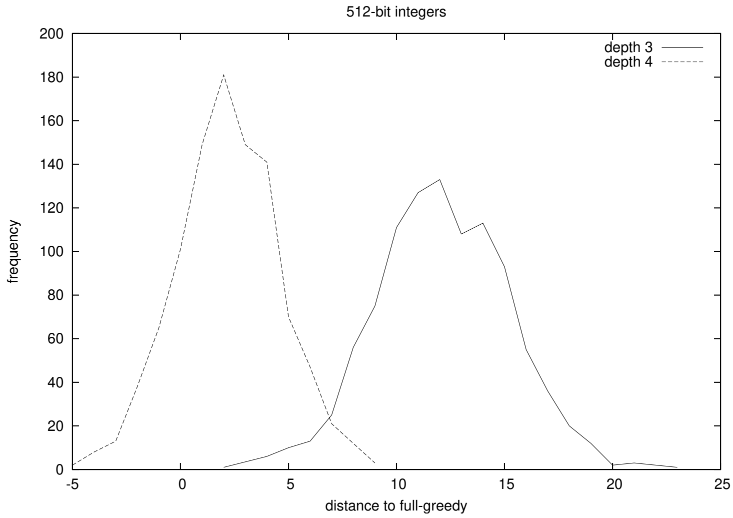


Fig. 5: Distribution of the length difference of DBNS expansions obtained with the full-greedy and the approximate algorithm at depths 3,4 for 512-bit numbers

Size of x (in bits)	Approx # digits with F-greedy	Avg. dist. to greedy at depth …				
		$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$
64	12	9.08	2.35	0.47	0	-
128	20	19.76	5.8	1.4	0.02	0
256	35	-	14.5	4.6	0.5	0
512	62	-	-	12.14	2.2	0

Tab. 4: Average distance to the full-greedy solution at various depths. Note that depth 5 is equivalent to the full-greedy algorithm in all our experiments

$2^a3^b \leq x$. It is also interesting to note that at medium depth ($d = 3$ or 4), the length of the representations can be very close to the full greedy ones, and in some cases even shorter.

For some applications, we might not need the DBNS decomposition given by the complete greedy approach, but rather satisfy with an approximated one, with slightly more terms, especially if the time required for the conversion is reduced. Another consequence of using the greedy decomposition at fixed depth is that the binary exponent is likely to be smaller at small depths than with the complete greedy decomposition. It is clear that the ternary exponent increases at the same time, but clearly not as fast (the ratio is $\alpha \simeq 0.63$).

7 Discussions

The question of the generalization to more than two bases, such as $2^a3^b5^c$ is natural. Nevertheless, there is no canonical generalization of Ostrowski numeration to higher dimensions. This is mainly due to the fact that there is no canonical notion of multidimensional continued fractions. To remedy to the lack of a satisfactory tool replacing continued fractions, several approaches are possible. One can consider e.g. lattice reduction algorithms: let us quote the computation of the n-th Hamming number in [21] based on results from [4]. Let us recall that Hamming numbers are numbers of the form $2^a3^b5^c$ for $a, b, c \in \mathbb{N}$, and that they are called after Hamming who asked for an efficient algorithm to generate in order the list of these numbers. The problem was popularized by E. W. Dijkstra in [9].

Algorithm 2 relies on the so-called three-gap theorem (see [22] and also the survey [5]). In the same vein, an algorithm is given in [18] which computes the points in $\mathbb{Z}^2$ located at a distance smaller than a given ε from a segment. This latter algorithm is based on the three-distance theorem, which can be considered as a dual version of the three-gap theorem. As an application, an algorithm which produces worst cases when trying to round the elementary functions in floating-point arithmetic is given in [19].

Instead of working with integer bases, one can also consider numeration in rational base $(2/3)$. The dynamical and arithmetical behaviour of such a numeration system differs drastically. Let us quote [1] which considers connections to number theory and applications to the problem of the distribution of the fractional part of the powers of rational numbers.

Acknowledgements

The authors would like to thank A. Rémondière for many useful discussions concerning Proposition 7 and the anonymous reviewers for their very careful reading.

References

[1] S. Akiyama, C. Frougny, and J. Sakarovitch. Powers of rationals modulo 1 and rational base number systems. *Israël Journal of Mathematics*, 168(1):53–91, 2008.

[2] J.-P. Allouche and J. Shallit. *Automatic Sequences*. Cambridge University Press, 2003.

[3] G. E. Andrews. Congruence properties of the m-ary partition function. *Journal of Number Theory*, 3(1):104–110, 1971.

[4] A. Barvinok and J. E. Pommersheim. An algorithmic theory of lattice points in polyhedra. In *New perspectives in algebraic combinatorics*, volume 38 of *MSRI Publications*, pages 91–147. Cambridge Univ. Press, 1999. Available at `http://www.msri.org/publications/books/Book38/contents.html`.

[5] V. Berthé. Autour du système de numération d'Ostrowski. *Bulletin of the Belgian Mathematical Society*, 8:209–239, 2001.

[6] V. Berthé, N. Chekhova, and S. Ferenczi. Covering numbers: Arithmetics and dynamics for rotations and internal exchanges. *Journal d'Analyse Mathématique*, 79:1–31, 1999.

[7] V. Berthé and L. Imbert. On converting numbers to the double-base number system. In *Advanced Signal Processing Algorithms, Architecture and Implementations XIV*, volume 5559 of *Proceedings of SPIE*, pages 70–78. SPIE, 2004.

[8] J. W. S. Cassels. *An Introduction to Diophantine Approximation*. Cambridge University Press, 1957.

[9] E. W. Dijkstra. Hamming's exercise in SASL. Originally a privately-circulated handwitten note; available at `http://www.cs.utexas.edu/users/EWD/ewd07xx/EWD792.PDF`, June 1981.

[10] V. Dimitrov, L. Imbert, and P. K. Mishra. Efficient and secure elliptic curve point multiplication using double-base chains. In *Advances in Cryptology, ASIACRYPT'05*, volume 3788 of *Lecture Notes in Computer Science*, pages 59–78. Springer, 2005.

[11] V. Dimitrov, L. Imbert, and P. K. Mishra. The double-base number system and its application to elliptic curve cryptography. *Mathematics of Computation*, 77(262):1075–1104, 2008.

[12] V. Dimitrov, G. A. Jullien, and W. C. Miller. An algorithm for modular exponentiation. *Information Processing Letters*, 66(3):155–159, May 1998.

[13] V. Dimitrov, G. A. Jullien, and W. C. Miller. Theory and applications of the double-base number system. *IEEE Transactions on Computers*, 48(10):1098–1106, Oct. 1999.

[14] C. Doche and L. Imbert. Extended double-base number system with applications to elliptic curve cryptography. In *Progress in Cryptology, INDOCRYPT'06*, volume 4329 of *Lecture Notes in Computer Science*, pages 335–348. Springer, 2006.

[15] S. Ito. Some skew product transformations associated with continued fractions and their invariant measures. *Tokyo Journal of Mathematics*, 9:115–133, 1986.

[16] S. Ito and H. Nakada. Approximations of real numbers by the sequence $\{n\alpha\}$ and their metrical theory. *Acta Mathematica Hungarica*, 52(1-2):91–100, Mar. 1988.

[17] A. Y. Khinchin. *Continued Fractions*. Dover Publications, new edition, 1997.

[18] V. Lefèvre. An algorithm that computes a lower bound on the distance between a segment and $\mathbb{Z}^2$. Research report RR1997-18, Laboratoire de lÍnformatique du Parallélisme, Lyon, France, June 1997. Available at `http://www.vinc17.org/research/publi.en.html`.

[19] V. Lefèvre. New results on the distance between a segment and $\mathbb{Z}^2$. application to exact rounding. In *Proceedings of the 17th IEEE Symposium on Computer Arithmetic*, pages 68–75. IEEE Computer Society Press, June 2005.

[20] V. A. Ostrowski. Bemerkungen zur theorie der diophantischen approximationen. *Abh. Math. Semin. Hamburg Univ*, 1:77–98, 1922.

[21] M. Quercia. Calcul du n-éme entier de Hamming, 1999. Available at `http://pauillac.inria.fr/~quercia/`.

[22] N. B. Slater. Gaps and steps for the sequence $n\theta \bmod 1$. *Mathematical Proceedings of the Cambridge Philosophical Society*, 63:1115–1123, 1967.

[23] V. T. Sós. On the theory of diophantine approximations. I. *Acta Mathematica Hungarica*, 8:461–472, 1957.

[24] V. T. Sós. On the theory of diophantine approximations. II. *Acta Mathematica Hungarica*, 9:229–241, 1958.

[25] V. T. Sós. On a problem of Hartman about normal forms. *Colloquium Mathematicum*, 7:155–160, 1960.

[26] R. Tijdeman. On the maximal distance between integers composed of small primes. *Compositio Mathematica*, 28:159–162, 1974.

[27] R. Tijdeman. Some applications of Diophantine approximation. In *Surveys in Number Theory: Papers from the millennial conference on number theory (University of Illinois at Urbana-Champaign, 2000)*, pages 261–284. A K Peters, 2003.

[28] N. N. Vorobiev. *Fibonacci Numbers*. Birkhäuser, 2002.

[29] E. Zeckendorf. Représentation des nombres naturels par une somme de nombres de Fibonacci ou de nombres de Lucas. *Fibonacci Quarterly*, 10:179–182, 1972.