LETTER

Two Lower Bounds for Shortest Double-Base Number System

Parinya CHALERMSOOK[†], ***Nonmember***, **Hiroshi IMAI**[††], ***and*** **Vorapong SUPPAKITPAISARN**[†††,††††a)], ***Members***

SUMMARY In this letter, we derive two lower bounds for the number of terms in a double-base number system (DBNS), when the digit set is $\{1\}$. For a positive integer n, we show that the number of terms obtained from the greedy algorithm proposed by Dimitrov, Imbert, and Mishra [1] is $\Theta\left(\frac{\log n}{\log\log n}\right)$. Also, we show that the number of terms in the shortest double-base chain is $\Theta(\log n)$.

key words: *analysis of algorithms, number representation, elliptic curve cryptography, double-base number system, double-base chain*

1. Introduction and Notations

A *double-base number system* (DBNS) is a redundant number representation, which is used for reducing the computation time of elliptic curve cryptography (ECC) [2]. The representation in the number system can be defined as follows.

Definition 1 (Double-Base Number Representation)**.** *Let* $\mathbf{c} = \langle c_i\rangle_{i=1}^m, \mathbf{x} = \langle x_i\rangle_{i=1}^m, \mathbf{y} = \langle y_i\rangle_{i=1}^m$ *be vectors of natural numbers of length m, and let* $R = \langle \mathbf{c}, \mathbf{x}, \mathbf{y}\rangle$*. If*

$$n = \sum_{i=1}^{m} c_i 2^{x_i} 3^{y_i},$$

then R is a ***double-base number representation (DBR)*** *of n.*

A positive integer that can be written in the form $2^{x_i}3^{y_i}$ for $x_i, y_i \in \mathbb{N}$ is known as as 3-*smooth number*, and thus we can consider a DBR to be a linear combination of 3-smooth numbers. We usually assume that c_i is a member of a small finite set called the *digit set* D_S. When $D_S = \{1\}$, the double-base number representations of 7 include $R_1 = \langle\langle 1,1,1\rangle, \langle 2,1,0\rangle, \langle 0,0,0\rangle\rangle$, $R_2 = \langle\langle 1,1\rangle, \langle 1,0\rangle, \langle 1,0\rangle\rangle$, and $R_3 = \langle\langle 1,1\rangle, \langle 2,0\rangle, \langle 0,1\rangle\rangle$.

We denote the number of terms in R, $|R| = |\langle \mathbf{c}, \mathbf{x}, \mathbf{y}\rangle|$, as the length of vector $\mathbf{c}$. Recalling R_1, R_2, R_3 defined in the previous paragraph, we get $|R_1| = 3$, and $|R_2| = |R_3| = 2$. It is shown in [2] that we can perform a faster ECC if we can find a DBR R with a smaller $|R|$. When $D_S = \{1\}$, Dimitrov, Imbert, and Mishra showed that for any $n \in \mathbb{Z}_+$, the DBR R of n obtained from Algorithm 1 has $|R|$ in $O\left(\frac{\log n}{\log\log n}\right)$ [1]. The result implies that a DBNS can asymptotically improve the binary representation, where the number of terms is $O(\log n)$ for any D_S. Also, the result leads us to a sublinear-time algorithm for the ECC.

```
Input: A positive integer $n$
Output: A DBR $R$ of $n$
$n' \leftarrow n$
$m \leftarrow 1$
while $n' \neq 0$ do
    Let $2^x 3^y$ be the largest 3-smooth number less than or equal to $n'$.
    $c_m \leftarrow 1, x_m \leftarrow x, y_m \leftarrow y$
    $m \leftarrow m + 1$
    $n' \leftarrow n' - 2^x 3^y$
end
$R = \langle\langle c_i\rangle_{i=1}^m, \langle x_i\rangle_{i=1}^m, \langle y_i\rangle_{i=1}^m\rangle$
```

Algorithm 1: A greedy algorithm [1] for calculating a short DBR of a positive integer n

Elliptic curve cryptography using a DBNS is very fast. However, the number of elliptic points that must be stored in memory at any given time can be as large as $O\left(\frac{\log n}{\log\log n}\right)$. We cannot afford that in computation environments with limited resources. To solve this problem, the *double-base chain* (DBC) was proposed in [3]. This representation is defined as follows.

Definition 2 (Double-Base Chain)**.** *Let* $R = \langle\langle c_i\rangle_{i=1}^m, \langle x_i\rangle_{i=1}^m, \langle y_i\rangle_{i=1}^m\rangle$ *be a DBR of n. If* $x_1 \geq x_2 \geq \cdots \geq x_m$ *and* $y_1 \geq y_2 \geq \cdots \geq y_m$*, then R is a* ***double-base chain*** *of n.*

Recall R_1, R_2, R_3 previously defined in this section. By Definition 2, we know that R_1 and R_2 are DBCs of 7, while R_3 is not.

When using a DBC, the number of points that must be stored at a given time is reduced to $O(1)$. However, it has not been proved that a DBC can asymptotically improve the binary representation, as is true for a DBNS. Although

Manuscript received August 25, 2014.
Manuscript revised February 11, 2015.

[†] The author is with the Department of Algorithms and Complexity, Max-Planck-Institut für Informatik, Saarbücken, 66123 Germany.

[††] The author is with the Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, 113-0033 Japan.

[†††] The author is with Global Research Center for Big Data Mathematics, National Institute of Informatics, Tokyo, 101-8430 Japan.

[††††] The author is also with JST, ERATO Kawarabayashi Large Graph Project, Tokyo, 101-8430 Japan.

a) E-mail: vorapong@nii.ac.jp

DOI: 10.1587/transfun.E98.A.1310

finding the value of R with smallest $|R|$ is believed to be a hard problem for a DBNS, Suppakitpaisarn, Edahiro, and Imai [4] proposed a dynamic programming algorithm that finds the shortest DBC in polynomial time.

1.1 Our Contributions

In this letter, we show the following two negative results for DBNSs and DBCs.

1. There are infinitely many positive integers n for which Algorithm 1 gives a DBR R with $|R| \in \Omega\left(\frac{\log n}{\log\log n}\right)$. We can conclude from this that the bound given in [1] is tight. The only way to reduce the asymptotic complexity of ECC via a DBNS is to improve Algorithm 1. We will discuss this further in Sect. 2.
2. When $D_S = \{1\}$, there are infinitely many positive integers n for which the shortest DBC R of n has $|R| \in \Omega(\log n)$. From this, we know that, in the worst case, DBC cannot asymptotically improve the binary representation. We will discuss this further in Sect. 3.

2. A Lower Bound for a Greedy DBNS

We will use the following theorem to show that there are infinitely many n for which the DBR R obtained using Algorithm 1 has $|R|$ in $\Omega\left(\frac{\log n}{\log\log n}\right)$.

Theorem 1 ([5])**.** *Let a and b be 3-smooth numbers such that $3 < a < b$. Then,*

$$b - a > \frac{a}{(\log a)^C},$$

for some constant C.

Although Tijdeman [5] claims that the value of the constant C is efficiently computable, the explicit value was not given in that paper. If the theorem is satisfied when the constant $C < 10$, we know that the theorem is also satisfied when $C = 10$. Because of that, we can assume that the constant C in Theorem 1 is no smaller than 10. We will need this assumption to prove some of the following statements.

Beside the above theorem, we will also use the following two lemmas.

Lemma 1. *Let $f : \mathbb{R} \to \mathbb{R}$ be a function such that $f(x) = \frac{x}{(\log x)^C}$. Then, $f(x)$ is non-decreasing for all $x \geq e^C$.*

Proof. The function f is non-decreasing because its derivative,

$$f'(x) = \frac{(\log x)^C - C(\log x)^{C-1}}{(\log x)^{2C}},$$

is non-negative for all $x \geq e^C$. □

Lemma 2. *For any constant $C \geq 10$, if $x \geq e^C$, then*

$$(\log x)^{2C}/2 > (\log x + 2C \log\log x - \log 2)^C.$$

Proof. It is obvious that $\log x < x^{0.25}$ for $x \geq e^{10}$. Thus,

$$\begin{aligned}
&(\log x)^{2C}/2 - (\log x + 2C\log\log x - \log 2)^C \\
&\quad > (\log x)^{2C}/2 - (\log x + 2C\log\log x)^C \\
&\quad > (\log x)^{2C}/2 - (\log x + 0.5C\log x)^C \\
&\quad \geq (\log x)^{2C}/2 - (0.6C\log x)^C \\
&\quad \geq C^C\left(C^C/2 - (0.6C)^C\right) > 0.
\end{aligned}$$

□

From these results, we have the following theorem, which is the target of this section.

Theorem 2. *There exists a sequence $\langle n_t\rangle_{t=1}^{\infty}$ such that the DBR R_t of n_t obtained from Algorithm 1 has $|R_t| \geq t$, and $t \in \Omega\left(\frac{\log n_t}{\log\log n_t}\right)$.*

Proof. Recall the constant C defined in Theorem 1. Let n_1 be a positive integer larger than $2 \cdot e^C$. Let a_t denote the smallest 3-smooth number such that $\frac{a_t}{(\log a_t)^C} > n_t$. We assign $n_{t+1} = n_t + a_t$.

We will now use induction to show that the DBR R_t of n_t obtained from Algorithm 1 has $|R_t| \geq t$. It is obvious that $|R_1| \geq 1$ and we will show that a_t is the largest 3-smooth number less than or equal to n_{t+1}. This is true because, by Theorem 1, all 3-smooth numbers s such that $s > a_t$ satisfy $s > a_t + \frac{a_t}{(\log a_t)^C} \geq n_{t+1}$. Therefore, we have $|R_{t+1}| = |R_t| + 1$, and $|R_t| \geq t$.

In the remaining part of this proof, we will show that $t \in \Omega\left(\frac{\log n_t}{\log\log n_t}\right)$. Recall the function f defined in Lemma 1, and let $x_t = \frac{n_t(\log n_t)^{2C}}{2}$. It is clear that $x_t \geq e^C$ for all t. Thus, we know that

$$f(x_t) = \frac{n_t(\log n_t)^{2C}/2}{(\log n_t + 2C\log\log n_t - \log 2)^C}.$$

We can assume that $C \geq 10$ from the discussion following Theorem 1. By the assumption, we know from Lemma 2 that $f(x_t) > n_t$. Let b_t be the smallest 3-smooth number that is greater than or equal to x_t. By Lemma 1, we know that $f(b_t) \geq f(x_t)$. Also, we know that $x_t \leq b_t \leq 2x_t$, as there must be a 3-smooth number between x_t and $2x_t$ for $x_t > 1$.

Recall from our sequence construction that a_t is the smallest 3-smooth number such that $f(a_t) > n_t$. We know that $a_t \leq b_t$, since f is non-decreasing and b_t is a 3-smooth number such that $f(b_t) > n_t$. Hence $a_t \leq b_t \leq 2x = n_t(\log n_t)^{2C}$. We thus have

$$n_{t+1} = n_t + a_t \leq n_t + n_t(\log n_t)^{2C} \leq 2n_t(\log n_t)^{2C}.$$

and

$$n_t \leq n_1(2\log n_t)^{2Ct} \in 2^{O(t\log\log n_t)}.$$

By taking the logarithm of each side, we get $\log n_t \in O(t\log\log n_t)$ and $t \in \Omega\left(\frac{\log n_t}{\log\log n_t}\right)$. □

3. A Lower Bound for the DBC

We let R_n denote the DBC of n that has the smallest $|R_n|$.

By assuming that $D_S = \{1\}$, we will show that there are infinitely many positive integers n such that $|R_n| \in \Omega(\log n)$. To achieve this, we will refer to the following results.

Theorem 3 ([4])**.** *If $n = 0$, $|R_n| = 0$. When $n > 0$, the value of $|R_n|$ can be calculated by the following recurrence equation.*

$$|R_n| = \begin{cases} \min(|R_{\frac{n}{2}}|, |R_{\frac{n}{3}}|), & \text{if } n \equiv 0 \pmod 6 \\ \min(|R_{\frac{n-1}{2}}|, |R_{\frac{n-1}{3}}|) + 1, & \text{if } n \equiv 1 \pmod 6 \\ |R_{\frac{n}{2}}|, & \text{if } n \equiv 2 \pmod 6 \\ \min(|R_{\frac{n-1}{2}}| + 1, |R_{\frac{n}{3}}|), & \text{if } n \equiv 3 \pmod 6 \\ \min(|R_{\frac{n}{2}}|, |R_{\frac{n-1}{3}}| + 1), & \text{if } n \equiv 4 \pmod 6 \\ |R_{\frac{n-1}{2}}| + 1, & \text{if } n \equiv 5 \pmod 6 \end{cases}$$

By using the previous theorem, we can obtain the following theorem, which is the target theorem of this section.

Theorem 4. *There exists a sequence $\langle n_t \rangle_{t=1}^{\infty}$ such that $|R_{n_t}| \geq t$ and $t \in \Omega(\log n_t)$.*

Proof. Assign $n_t = 6 \cdot 2^t - 1$ for all $t \geq 1$. Clearly, $n_t \in O(2^t)$ and $t \in \Omega(\log n_t)$.

In the remaining part of this proof, we will use induction to show that $|R_{n_t}| \geq t$. The statement is true when $t = 1$, since $R_5 = \langle\langle 1, 1\rangle, \langle 2, 0\rangle, \langle 0, 0\rangle\rangle$ and $|R_5| = 2 > 1$. Since $n_t = 6 \cdot 2^t - 1$, it is obvious that $n_t \equiv 5 \pmod 6$. By Theorem 3, we know that

$$\begin{aligned} |R_{n_t}| &= |R_{\frac{n_t - 1}{2}}| + 1 \\ &= |R_{\frac{6 \cdot 2^t - 2}{2}}| + 1 \\ &= |R_{6 \cdot 2^{t-1} - 1}| + 1 = |R_{n_{t-1}}| + 1. \end{aligned}$$

Since $|R_{n_{t-1}}| \geq t - 1$ by induction, we get $|R_{n_t}| \geq t$. □

4. Conclusion and Future Works

In this study, we found two tight lower bounds for a DBNS using greedy algorithm, and DBC using digit set $D_S = \{1\}$. We are currently working on algorithms that can find a DBR R with smaller value for $|R|$. It is widely believed that the problem is NP-hard, and there is currently no algorithm that has a bounded approximation ratio in the literature.

We are also aiming to extend our result in Sect. 3 to the case in which $D_S \neq \{1\}$. For those cases, there are several fast, suboptimal algorithms that have been proposed in literature (e.g., [6]). Even though the experimental results show that those algorithms can output DBRs that are very close to optimal, the theoretical results show that the number of terms in those DBRs are in $\Omega(\log n)$. Because of this, we strongly believe that we can obtain a similar result for a general D_S, and we are seeking for a proof of this.

We expect that our result in Sect. 2 might also be used for a triple-base number system (TBNS) where three bases (2, 3, and 5) are allowed. It was shown in [7] that there are infinitely many positive integers n for which the shortest triple-base representation R has $|R| \in \Omega\left(\frac{\log n}{\log\log n \log\log\log n}\right)$. By following the same argument in our proof, one might obtain the same lower bound, $|R| \in \Omega\left(\frac{\log n}{\log\log n}\right)$, for a TBNS, when the representation is found by a greedy algorithm.

Acknowledgments

The authors would like to thank Ms. Chitchanok Chuengsatiansup and an anonymous reviewer for valuable comments during this course of research.

References

[1] V. Dimitrov, L. Imbert, and P. Mishra, "The double-base number system and its application to elliptic curve cryptography," Mathematics of Computation, vol.77, no.262, pp.1075–1104, 2008.

[2] N. Méloni and M.A. Hasan, "Elliptic curve scalar multiplication combining Yao's algorithm and double bases," Proc. CHES'09, pp.304–316, 2009.

[3] V. Dimitrov, L. Imbert, and P.K. Mishra, "Efficient and secure elliptic curve point multiplication using double-base chains," Proc. ASIACRYPT'05, pp.59–78, 2005.

[4] V. Suppakitpaisarn, H. Imai, and E. Masato, "Fastest multi-scalar multiplication based on optimal double-base chains," Proc. 2012 World Congress on Information Security (WorldCIS), pp.93–98, IEEE, 2012.

[5] R. Tijdeman, "On integers with many small prime factors," Compositio Mathematica, vol.26, no.3, pp.319–330, 1973.

[6] D.J. Bernstein, P. Birkner, T. Lange, and C. Peters, "Optimizing double-base elliptic-curve single-scalar multiplication," Proc. INDOCRYPT'07, pp.167–182, 2007.

[7] W. Yu, K. Wang, B. Li, and S. Tian, "On the expansion length of triple-base number systems," Proc. AFRICACRYPT'13, pp.424–432, 2013.