# Challenges in data representation for efficient execution of encryption operation

**Mohamad Afendee Mohamed[1], Yahaya Garba Shawai[1,2], Mohammed Amin Almaiah[3,4], Mohd Noor Derahman[5], Abdalwali Lutfi[6,7], Khairul Azmi Abu Bakar[8]**

[1]Department of Computer Science, Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Besut, Malaysia
[2]National Open University of Nigeria, Victoria Island, Lagos, Nigeria
[3]Faculty of Information Technology, Aqaba University of Technology, Aqaba, Jordan
[4]Applied Science Research Center, Applied Science Private University, Amman, Jordan
[5]Department of Communication Technology and Network, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Selangor, Malaysia
[6]Department of Accounting, College of Business, King Faisal University, Al-Ahsa, Saudi Arabia
[7]MEU Research Unit, Middle East University Amman, Amman, Jordan
[8]Center for Cyber Security, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Bangi, Malaysia

**Article Info**

***Article history:***

Received Dec 7, 2022
Revised Sep 5, 2023
Accepted Oct 19, 2023

***Keywords:***

Efficient execution
Encryption operation
Minimal representation
Modular exponentiation
Non-zero digits

## ABSTRACT

Big number operation has always been a bottleneck to computer system as it imposes high demand on computing power. With a limited power available, operations such as exponentiation and multiplication involving large integer belonging to encryption process requires grave scrutiny. One way to address this issue is by replacing an original complex computation into a sequence of small computations that in the end produces the same results. This paper takes an evolutionary approach to survey numerous articles that have contributed to the advancement of integer representation. Numerous representations were proposed, those that come into play concentrated on reducing non-zero digits and limiting non-zero spacing other than allowing subtraction operation. A comparison was made to distinguish the properties of each method from the others. This detailed outlook can be a guide for identifying the correct representation to be chosen for implementation within specific application.

***Corresponding Author:***

Mohamad Afendee Mohamed
Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin
Besut, Malaysia
Email: mafendee@unisza.edu.my

## 1. INTRODUCTION

Computer was designed to work with only numbers. Characters and symbols must all be converted to numbers in the form of zeros and ones. These are the only form that are understood by computer. Any type of operations from whatever applications deals only with this form of numbers. Speaking of using computer in our daily life, one of the most important measurements that comes to our head would be efficiency. Efficiency in itself is not a real parameter as it has no specific unit for it. But it can be broken into specific parameters such as time taken for executing some tasks, and resources needed to accomplish certain tasks. Time itself can be further subcategorized into speed and delay [1]. Speed and delay are just like an antonym to each other. One measures how fast can you make it to a destination and the other how late will you be there. Resources are much bigger a measurement, it can be divided into space or memory resources, capacity or CPU resources. Memory is a place where computer store data required for processing while CPU is the strength of computing processor to execute given instructions [2].

Integer representation is indeed important in various aspects of computing and mathematics. Integers are whole numbers that can be positive, negative, or zero, and they are commonly used to represent quantities, perform calculations, and store data in computer systems. Integers are used to represent and store numerical data in computer memory. They are typically stored using a fixed amount of memory, allowing for efficient storage and manipulation of large sets of integers. Integers are essential for performing arithmetic operations like addition, subtraction, multiplication, and division [3]. These operations are fundamental in computer programs and mathematical computations. In low-level programming and computer architecture, integers are represented as sequences of bits. This representation allows for bitwise operations like shifting, masking, and logical operations [4], which are used in various applications such as data encoding, encryption, and optimization algorithms. Integers are often the subject of study in algorithm design and analysis. Many algorithms and data structures are designed specifically to efficiently handle integer inputs or produce integer outputs. Overall, integer representation is fundamental in computer science and mathematics, serving as a cornerstone for data representation, numerical computations, and algorithmic problem-solving [4]. Integer representation has been exploited before for many different applications of computing such as encryption [5] and compression [6]. It has been successfully implemented and beginning to offer some fruitful results. For encryption, speed would be the main goal while for compression, a shorter transmission time or a smaller disk space is desirable for data transmission and data storage respectively.

Data security has been a hot topic since a few decades ago and it gets even hotter these days. The emergence of industrial revolution 4.0 relies largely on data security for all of its functionalities and accomplishments [7]–[9]. Its success depends on the safe and secure communication between connected devices. Data security plays the most important role that determines not only the success but also trust and confidence between communicators. Data security in its true meaning ensure the properties of data in terms of confidentiality and integrity are preserved, and those who handle the data require authentication to access and dismiss any future repudiation. Encryption has been a single most critical function in offering protection to data transmission and storage. Encryption tries to achieve the highest possible security to date while retaining the efficiency of its execution. Balancing the twos has been a challenge in itself and the trade-off is pegged to which areas it has been used such as the military, commercial and individual. Security is achieved through the size of the secret information (key) that is used for the encryption, other than the choices of algorithms which imposes different level of complexities to the computer processor [10].

Data representation can be manipulated to reduce the cost of execution and thus improve the speed. Good representation can reduce the number of operations and thus the instruction. Encryption deals with large number for its key, therefore by being able to represent this number in a different representation offer a chance to reduce its complexity. In cryptography, this large number operation is simply converted into a chunk of simple but iterative execution of cheaper operations involving smaller number.

Obviously, any method that was designed to take the recoded minimal representation to iteratively compute the modular exponentiation or modular multiplication will produce an identical result but with different speed. Based on Figure 1, we can categorize integer representation by the availability of its sign. Signed representation allows substitution or division operation into the sequence. Meanwhile, radix is another approach into converting integer base 10 into selected basis. In this survey article, we try to comprehensively review various integer representations that have been used for the purpose of speeding up encryption operations. Our focus is to relate the properties of minimal representation in terms of its length and non-zero density.

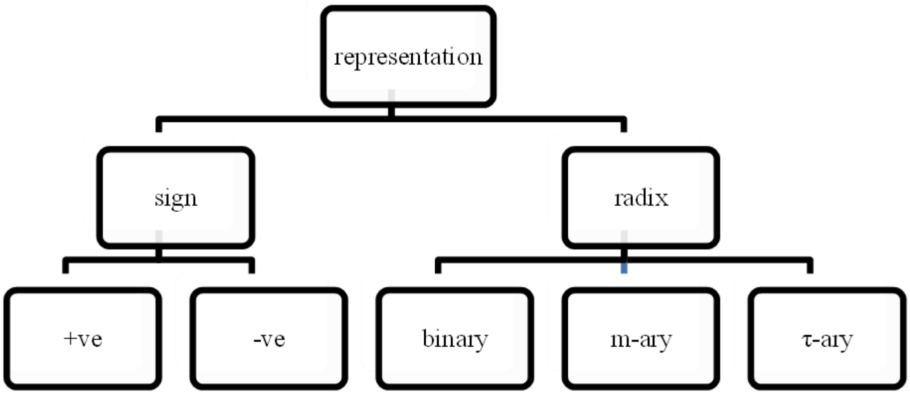

Figure 1. Taxonomy of integer representation

## 2. INTEGER REPRESENTATION

In complex modular arithmetic, the scalar $n$ (multiplicand/exponent) is assumed to be a positive integer. Some form of representations for $n$ were introduced such that the technique for computing result is efficient. In achieving this objective, several factors were identified namely, the density of non-zeros, the cardinality of coefficient set and the length of zero runs in the representation.

Definition 1: every integer $n \in \mathbb{Z}^{+}$can be expressed by a summation of its coefficient multiplied by its radix as:

$$n = \sum_{i=0}^{r-1} c_i m^i = c_{r-1}m^{r-1} + c_{r-2}m^{r-2} + \cdots + c_1 m + c_0$$

where $c_i \in \{0,1,\ldots,m-1\}$ are the coefficients, $\lambda(n) = \lfloor log_m n \rfloor + 1$ is the length of the representation and $v(n)$ gives the number of non-zero digits. The *m* is called the radix for *m*-ary representation.

### 2.1. Binary representation

Binary number is the most suitable representation for computer systems as it able to work directly with no extra efforts are required for further conversion because computer system understands 0's and 1's perfectly. Other than the original, to increase efficiency, many representations emerged as a result of weaknesses found within their predecessors. An unsigned binary representation is the most common form of binary number used to represent an integer *n*.

Definition 2: unsigned binary. Let an integer $n \in \mathbb{Z}^{+}$, a classical binary representation for *n* is given by:

$$n = \sum_{i=0}^{r-1} b_i 2^i = b_{r-1}2^{r-1} + b_{r-2}2^{r-2} + \cdots + b_1 2 + b_0$$

where $b_i \in \{0,1\}, \lambda(n) = \lfloor log_2 n \rfloor + 1$, and the average non-zero density is given by $v_A(n) = \frac{r}{2}$. This representation is unique, there is one-to-one correspondence between $n$ and its associated binary representation. However, the appearance of non-zeros seems to be redundant in this representation. As it will be discussed in the next section, efficiency in computing $Q$ can be further improved by reducing the number of 1's. Out of unsigned binary representation, Booth [11] introduced the idea of signed binary representation.

Definition 3: signed binary. Given an integer $n \in \mathbb{Z}^{+}$, a signed binary representation for *n* is given by:

$$n = \sum_{i=0}^{r-1} b'_i 2^i = b'_{r-1}2^{r-1} + b'_{r-2}2^{r-2} + \cdots + b'_1 2 + b'_0$$

where $b'_i \in \{0, \pm 1\}, \lambda(n) = \lfloor log_2 n \rfloor + 1$ and $v_A(n) = \frac{r}{2}$. This representation is also known as modified radix-*m* form in the theory of arithmetic codes. It manages to reduce the non-zero density. However, the representation is not unique since for each *n*, there exist more than one signed binary representations. For this reason, a non-adjacent form (NAF) was introduced by Reitwiesner [12], which consists of minimal hamming weight, and most importantly every integer has a unique form of this sort [12]–[14]. Hamming weight for this type of representation is a measure of its non-zero density.

Definition 4: non-adjacent form. Every integer $n \in \mathbb{Z}^{+}$can be represented by non-adjacent form as:

$$n = \sum_{i=0}^{r} b'_i 2^i = b'_r 2^r + b'_{r-1}2^{r-1} + \cdots + b'_1 2 + b'_0$$

where $b'_i = \begin{cases} +1,0 & if\ i \geq n \\ \pm 1,0 & if\ i < n \end{cases}$ and $\lfloor log_2 n \rfloor \leq \lambda(n) - 1 \leq \lfloor log_2 n \rfloor + 1$ and $v_A(n) = \frac{r}{3}$. The conception is to disallow two consecutive non-zero bits. Comparing to that of unsigned binary, an average non-zero density is reduced by one sixth of the original length. Researchers [12], [15] specified an iterative method to generate NAF from an unsigned binary as well as signed binary representations. That of [12] is known as right-to-left recoding algorithm and is compact and efficient. Researchers [16], [17] showed that the algorithm from Reitwiesner is based on the formula $3n - n$. Alternatively, Chang and Tsao-Wu [17] produced an iterative method called weight minimization algorithm (WMA) to generate NAF, including theorems for the sparseness and uniqueness of the output. Another notably development was due to [18] with recoding from right to left is based on dividing $n$ repeatedly by 2 and assigning $b'_i$ to be 0 if $n$ is even, 1 if $(n-1)/2$ is even, or -1 if $(n+1)/2$ is even.

However, the direction of scanning the bits can be a constraint in some occasions. As a result, Joye and Yen [19] brought up an idea to produce binary representation with NAF properties of having minimal hamming weight but contrary to earlier works, the technique scans the bits of input from left to right. In certain cases, this may lead to some improvements in software as well as hardware performance especially in a case where a memory resource is very limited. This representation is an elegant left-to-right algorithm of comparable performance to Reitwiesner right-to-left. It has minimal hamming weight but not always sparse, in other words, it allows two consecutive non-zeros [20]. Extended this idea of left-to-right algorithm and came out with a new canonical and unique representation for signed binary namely mutual opposite form (MOF). Unlike NAF, MOF does not always have minimal hamming weight and it allows consecutive non-zeros but with opposite sign and the most non-zero significant bit and the least non-zero significant bit are 1 and $\bar{1}$ respectively.

Definition 5: mutual opposite form. Every unsigned binary for any $n \in \mathbb{Z}^+$can be converted into MOF.

$$n = \sum_{i=0}^{r} b_i' 2^i = b_r' 2^r + b_{r-1}' 2^{r-1} + \cdots + b_1' 2 + b_0'$$

where $b_i' \in \{0, \pm 1\}, b_r' = b_{r-1}, b_0' = -b_0$ and $b_i' = b_{i-1} - b_i$ for $i = 1,2,\ldots,r-1$. The conversion is achieved using the formula $2n \ominus n$, where $\ominus$ is a bitwise subtraction. Properties of MOF is quite similar to that of NAF except for it is not being sparse. The left-to-right MOF recoding algorithm generates MOF representation by taking an unsigned binary representation. The advantage of MOF is its ability to recode from left to right although in some cases it will produce longer addition chain than NAF. Another version of NAF is due to [21]. The idea is to increase the zero runs in a NAF representation without changing its weight. All recoding algorithms discussed are based on iteration and can be quite a constraint [22] came out with a new recoding technique based on common multiplicand [23] which could speed up the recoding from binary to NAF.

Definition 6: complementary recoding. Given an unsigned binary representation for $n \in \mathbb{Z}^+$, its complementary recoding representation is given by:

$$n = \sum_{i=0}^{r-1} b_i 2^i = (100 \ldots 0)_{(r+1)bits} - \bar{n} - 1$$

where $\bar{n} = \bar{n}_{i-1}\bar{n}_{i-2} \ldots \bar{n}_0$, if $b_i = 1$ then $\bar{n}_i = 0$, if $b_i = 0$ then $\bar{n}_i = 1$, for $i = 0,1,2,\ldots,n-1$. This representation is longer than its input binary representation by 1 bit. Due to a much simpler recoding algorithm which requires no iteration, the average running time for recoding an integer *n* using this algorithm is significantly shorter than that of NAF or MOF. However, this method does not always produce minimal weight representation and it is totally different from NAF or MOF. As such, the number of operations will not be as minimal as NAF or MOF. Therefore, for general case, NAF is still considered as the best recoding technique to date.

Integer representation discussed thus far is known as signed binary representation. In case when extra memory resource is available, precomputation is allowed, signed representation using larger digit set should be taken into account. This representation enlarges the coefficient set by reading *w* bits NAF input at one time. There are two different ways to construct this type of representation, namely by applying sliding window technique on signed binary representation [21], [24], and *w*-NAF which is computed directly from binary strings using a generalization of NAF recoding technique [25]–[27].

Definition 7: *w-NAF*. Every integer $n \in \mathbb{Z}^+$, can be represented by *w-NAF* as:

$$n = \sum_{i=0}^{r} c_i 2^i = c_r 2^r + c_{r-1} 2^{r-1} + \cdots + c_1 2 + c_0$$

where $c_i \in \{|2k + 1| < 2^w - 1 : k \in \mathbb{Z}^+\}, \lfloor log_2 n \rfloor \leq \lambda(n) - 1 \leq \lfloor log_2 n \rfloor + 1$ and $v_A(n) = \frac{r}{1+w}$. The *w*-NAF is just a generalization of NAF ($w = 2$). It simply inherits properties like uniqueness, no non-zero adjacent with the least non-zero density among other representation having the same coefficient set. The *w*-NAF(*n*) can be computed similar to NAF(*n*). A reduction modulo $2^w$ is done to ensure that *w* consecutive digits contain at most one non-zero digit [26] recodes unsigned binary representation into *w*-NAF, operating from right to left. Unfortunately, this type of *w*-NAF representation can only be generated from right to left due to the *carry–over* bit. Independently, by [28]–[30] introduced a left-to-right version equivalent to *w*-NAF which was able to perform scalar multiplication on-the-fly. The one due to Avanzi is called *w*-LtoR recoding.

Irrespective to operating direction, both algorithm produces minimal weight representation. However, the memory consumption level is still inefficient for a limited storage device. The search for a *memory−less* left-to-right algorithm only came to an end after [29] introduced a left-to-right memory-less algorithm called $w$-NAF*, a generalization of $w$-NAF. Interestingly enough, this algorithm bears all properties that belongs to $w$-NAF. Another left-to-right recoding technique called width $w$ window MOF, shortened as $w$-MOF was due to [20] can also be used to decrease the non-zero density of MOF.

Definition 8: *w-MOF*. Every integer $n \in \mathbb{Z}^{+}$can be represented by *w-MOF* as:

$$n = \sum_{i=0}^{r} c_i 2^i = c_r 2^r + c_{r-1} 2^{r-1} + \cdots + c_1 2 + c_0$$

where $c_i \in \{|2k+1| < 2^w - 1 : k \in \mathbb{Z}^{+}\}, \lfloor log_2 n \rfloor \leq \lambda(n) - 1 \leq \lfloor log_2 n \rfloor + 1$ and $v_A(n) = \frac{r}{1+w}$.

The coefficient set for $w$-MOF is similar to that of $w$-NAF. The recoding process takes an unsigned binary representation into $w$-MOF in a quite similar way to the one from Avanzi. Moreover, it also has the same properties as that of $w$-NAF. Window methods is capable to reduce the non-zero density, but at the expense of precomputation for all elements within coefficient set excluding $P$.

### 2.2. *m*-ary representation

The concept of binary representation can be extended to $m$-ary form. Only two changes are required for this transformation to work, the radix must now be generalized to $m = 2^k$ for which $k > 1$, and the coefficient set is now allowed to have any element less than $m$. Each element from the coefficient set needs to be precomputed prior computing $Q$. All elements contribute to the so-called addition sequence. Bos and Coster [30] discussed in depth on how to generate this sequence using an efficient vectorial addition chain. Similar to binary case, an integer $n$ can be converted to a number of base $m$ through an iterative division operation of $n$ by $m$ until the quotient is zero, taking the remainders as the result. The original study on unsigned $m$-ary representation is due to [31], [32].

Definition 9: *unsigned m-ary*. Every positive integer $n$ can be represented by an unsigned $m$-ary form as:

$$n = \sum_{i=0}^{t-1} d_i m^i = d_{t-1} m^{t-1} + d_{t-2} m^{t-2} + \cdots + d_1 m + d_0$$

where $d_i \in \{0,1,\ldots,m-1\}$, $\lambda(n) = \lfloor log_m n \rfloor + 1$, and $v_A(n) = \frac{m-1}{m} t$. Similar to unsigned binary case, this representation is unique for each integer. Consider $m = x^k$, an unsigned $m$-ary can also be obtained by partitioning the $m = x^1$representation starting from the least significant bit by $k$ size. Each partition should hold a value of less than $m$. An improved representation to unsigned $m$-ary was introduced, namely recoded $m$-ary allows negative coefficients into the set [33], based on the idea of recoded binary representation [11].

Definition 10: *recoded m-ary*. Given a positive integer $n$, its recoded $m$-ary representation is given by:

$$n = \sum_{i=0}^{t-1} d'_i m^i = d'_{t-1} m^{t-1} + d'_{t-2} m^{t-2} + \cdots + d'_1 m + d'_0$$

where $d'_i \in \{0, \pm 1, \ldots, \pm m - 1\}$ and $v_A(n) = \frac{3t+1}{8} + \frac{5t-1}{2^t} \approx \frac{3}{8} t$. This representation is however not unique for each $n$. In order to address this gap, Clark and Liang [34] came out with an idea of general non-adjacent form (GNAF) for any integer $n$.

Definition 11: GNAF. Given a positive integer $n$, its *GNAF* representation can be expressed as:

$$n = \sum_{i=0}^{t} d'_i m^i = d'_t m^t + d'_{t-1} m^{t-1} + \cdots + d'_1 m + d'_0$$

with two conditions, $|d'_i + d'_{i+1}| < m,$ for all $I$ and $|d'_i| < |d'_{i+1}|$ if $d'_i d'_{i+1} < 0,$ where$-m < d'_i < m.$ The average non-zero density denoted as $v_A(t,m)$ of GNAF with radix-$m$ is given by [35] as:

$$v_A(t,m) = \begin{cases} \dfrac{m-1}{m+1} t + \dfrac{2m}{(m+1)^2} - \dfrac{2}{(m+1)^2 m^t - 1} \text{for } t \text{ even} \\ \dfrac{m-1}{m+1} t + \dfrac{2m}{(m+1)^2} - \dfrac{m^2+1}{(m+1)^2 m^t} \text{for } t \text{ odd} \end{cases}$$

where $0 \leq t \leq m^t - 1$ for $t \geq 1$. For large *t*, it can be approximated to $\frac{m-1}{m+1}t$ [36]. The following is the properties of GNAF for every integer *n*. GNAF retains similar properties to that of NAF [17]. Among several versions of GNAF algorithms available, the one due to [35] takes an input of unsigned $m$-ary integer and operates from right to left. Twisting an idea from [16] for NAF, [34] used the formula $(3n + n) - n$ to produce a non-iterative GNAF algorithm for radix 3. Moreover, for an arbitrary radix-*m*, [34] produced an algorithm to accept signed $m$-ary form as an input.

Many research were conducted to study the generation of GNAF from left to right. Earlier results showed that left to right GNAF algorithm is not possible to have. However, Kong *et al.* [37] proved it otherwise. They produced left-to-right GNAF by basing on the idea of left-to-right NAF recoding algorithm earlier. Another version of left-to-right algorithm was due to [38] who came out with another radix-*m* representation namely generalized star form (GSF). This algorithm processes unsigned $m$-ary representation for $n$ from left to right. Moreover, they proved that this representation bears the minimal weight as that of GNAF.

The approach of $m$-ary method is equivalent to partitioning the binary representation into a block of fixed window size and then convert each binary block into an equivalent radix-*m* integer. Although this method seems to reduce the number of terms through shortening the chain, the drawback is the need to precompute all elements of coefficient set.

### 2.3. *ψ*-ary representation

This representation exploits the properties of endomorphism of an elliptic curve $E$ defined over $\mathbb{F}_2$ [39], [40]. It works only with specific family of curves known as anomolous or Koblitz curves defined as $E/\mathbb{F}_2$. The group denoted by $E(\mathbb{F}_{2^m})$ is defined over an extended field, having an order only divisible by an odd large prime. For this, $m$ supposedly be a prime such that $2^m$becomes *nearly* prime. Indeed, the subgroup $E(\mathbb{F}_2)$ has an order of either 2 or 4 and this value divides $\#E(\mathbb{F}_{2^m})$. Consider $\#E(\mathbb{F}_{2^m}) = fu$ where $\#E(\mathbb{F}_2)$=*f*, and $\#E_u(\mathbb{F}_{2^m}) = u$ such that $E_u(\mathbb{F}_{2^m})$ be a main subgroup of $E(\mathbb{F}_{2^m})$. The main subgroup has been found suitable for cryptographic application. Efficient arithmetic is based on decomposing $n$ by Frobenius map *ψ* for which an execution of doubling or its multiples $\psi^i$ for $i \in \mathbb{Z}^+$can be obtained almost for free. The *ψ*-expansion of *n* allows the computation of *nP* with only additions. Endomorphism doublings is just a cyclic shift of the vector representation using normal basis of $\mathbb{F}_{2^m}$ and therefore requires insignificant amount of time. Smart [41] proved that this representation does exist for every *n* and not arbitrarily long with coefficients having absolute values of less than 7. Similar to the binary case, *ψ*-ary representation for *n* can be obtained by repeated division of *n* by *ψ* where the digits $b_i$ are the remainders.

Definition 12: an element $n \in \mathbb{Z}[\psi]$ can be represented by *ψ-ary* as:

$$n = \sum_{i=0}^{s-1} b_i\psi^i = b_{r-1}\psi^{r-1} + b_{r-2}\psi^{r-2} + \cdots + b_1\psi + b_0$$

where $b_i \in \{0,1\}$ and $v_A(n) = \frac{s}{2}$. Since *ψ* is an element of euclidean domain $\mathbb{Z}[(1 + \sqrt{-7})/2]$, any element of the ring is uniquely represented by this representation. In addition, as mentioned by Morain and Olivos [18] for binary representation, where elliptic curve point subtraction costs insignificant resource, NAF like representation for *ψ*-ary called *ψ*NAF can also be obtained.

Definition 13: an element $n \in \mathbb{Z}[\psi]$ can be represented by *ψNAF* as:

$$n = \sum_{i=0}^{s-1} b_i'\psi^i = b_{s-1}'\psi^{s-1} + b_{s-2}'\psi^{s-2} + \cdots + b_1'\psi + b_0'$$

where $b_i' \in \{0, \pm 1\}$ and $v_A(n) = \frac{s}{2}$. In *ψ*NAF, every positive integer *n* has a unique representation, and no two consecutive non-zeros is allowed. To compute $l(n)$, l*et N*(*n*) *be the norm of* $0 \neq n$ *in* $\mathbb{Z}[\psi]$. *If* $s > 30$, *then* $log_2(N(n)) - 0.55 < l(n) < log_2(N(n)) + 3.52$ [26]. Computation of *ψ*NAF(*n*) is similar to that of NAF(*n*) although in the case of *ψ*NAF, *n* is now an algebraic integer and the recoding process is a little more complicated, for various details refer to [26]. However, there is a drawback, using standard *ψ*NAF representation the length of its representation is twice the length of NAF(*n*) [42]. Due to this, *ψ*NAF representation is not necessarily to be more efficient than using binary NAF.

In overcoming this, Meier and Staffelbach [43] showed that for every Koblitz curve with *ψ* defined over $\mathbb{F}_{2^m}$, there is a *ψ*-expansion for *n* of length *m* having *ψ*NAF representation, namely reduced *ψ*NAF.

Observed the fact that $(\psi^m - 1)(P) = \psi^m(P) - P = P - P = \infty$ for any $P \in E(\mathbb{F}_{2^m})$. For if $n \equiv n' mod\ (\psi^m - 1)$ then $nP = n'P$. Hence $l(n') \approx l(\psi^m - 1) \approx m \approx l(NAF(n))$ and $s = m$. The length of $\psi$NAF($n$) is now shortened to that of NAF($n$). Since cryptographic operation takes place in the main subgroup $E_u(\mathbb{F}_{2^m})$, consider $P \in E_u(\mathbb{F}_{2^m})$, then the following expression is true.

$$(\psi\ -\ 1)\frac{\psi^m - 1}{\psi - 1}P = \mathcal{O}$$

for if $(\psi\ -\ 1) \neq O$, then $\frac{\psi^m - 1}{\psi - 1} = \mathcal{O}$. Therefore $n'' \equiv n\ mod\ (\frac{\psi^m - 1}{\psi - 1})$ and obviously $\psi$NAF($n''$)=$\psi$NAF($n$) follows. That the equation is equivalent to each other, reduced $\psi$NAF can be used in place of $\psi$NAF in scalar multiplication. The average non-zero density is also reduced to $\frac{m}{3}$. The right-to-left $\psi NAF$ recoding algorithms [26] responsible to compute this modular reduction can also be found in [44]. It was shown that Frobenius map allows a replacement of many expensive elliptic curve's doublings and additions with fewer elliptic curve additions and some power evaluations in a finite field. By this way, it improves the speed of up to 50% on curve of this sort compares to general methods.

## 3. DISCUSSION

The fact is many ideas have been developed to transform integer into some form that can cheaply be operated by computer. The simplest form would be the binary (2-ary) form which can be directly operated since computer only deals with '0's and '1's. The fact is basic operations allowed are addition and doubling which are known to be the least costly. Nevertheless, m-ary representation can also be beneficial in certain conditions. This unsigned representation is further manipulated by allowing negativity of operation producing a signed representation and thus achieving minimality of non-zero digits and limiting non-zero adjacency. We observed that many ideas and methods have been brought forward for solving an addition chain problem. The methods mainly manipulate the representation of number in terms of basis and what operations are allowed as well as its orientation. Different representation results in different addition chain as well as its length.

Table 1 shows topological properties of integer representations. Given an integer, various techniques have been developed to produced the aforementioned representations. One way to classify the techniques is the selection of the allowed operations. Unsigned representation allows only addition and doubling, whereas signed operation includes a subtraction operation. Another way would be the direction we operate the representation. Some techniques begin with the least significant bit (right-to-left) while others most significant bit (left-to-right). The rest of the columns specifies the intrinsic properties of the representations namely uniqueness, non-zero adjacency and non-zero density. These three properties play the most important role in reducing the number of operations and hence increase the performance of the entire encryption time.

Table 1. Properties of integer representations

| Representation | Sign | R->L | L->R | Uniqueness | No non-zero adjacency | Minimal non-zero density |
|---|---|---|---|---|---|---|
| Unsigned binary | unsigned | ✓ | | ✓ | | |
| Signed binary | signed | | ✓ | | | |
| NAF | signed | ✓ | ✓ | ✓ | ✓ | ✓ |
| MOF | signed | | ✓ | ✓ | | |
| CR | signed | ✓ | | ✓ | | |
| $\psi$NAF | signed | ✓ | | | | |
| wNAF | signed | ✓ | ✓ | ✓ | ✓ | ✓ |
| wMOF | signed | | ✓ | ✓ | | ✓ |
| Unsigned m-ary | unsigned | ✓ | | ✓ | | |
| Recoded m-ary | signed | | ✓ | | | |
| GNAF | signed | ✓ | ✓ | ✓ | ✓ | ✓ |
| GSF | signed | | ✓ | ✓ | ✓ | ✓ |

Needless to say, the aforementioned techniques chosen for discussion were based on the efficiency of its execution. They have been practically proved to save the power, storage and time. There are also other representations that are conceptually exceptional, based on some mathematical theorem [45]–[50]. Although some of these concepts were shown to be advantageous, their practicality has never been applied and thus its efficiency is doubtful.

## 4. CONCLUSION

Integer representation plays an important role in ensuring cheap computational operation in computer system and thus the efficiency of the entire application. It happens to be more important when dealing with big numbers in application such that of cryptography. Approaches such as controlling the number of operations executed, the type of operations used, the environment within which the operation takes place are known to improve the computational speed. Cryptography deals with complex computation of modular arithmetic involving multiplication and exponentiation operation within certain algebraic structure. This complexity is normally reduced to repetitive tasks of much simpler operations such as addition and doubling which directly correspond to zero and one in the equivalent binary representation. In some cases, subtraction can be added together in signed representation. That makes the solutions to be categorized into two generic groups namely unsigned and signed representation. By adding an extra operation in signed representation, the entire time factor can be shortened. However, this limit to the selection of specific environment and parameters. In general, one would enjoy experiencing the evolution of these subject.

## ACKNOWLEDGEMENTS

This work was supported in partial by the Ministry of Higher Education, Malaysia via Fundamental Research Grant Scheme project FRGS/1/2018/ICT03/UNISZA/02/2, and the Annual Funding track by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia (Project No. Grant No. 4532).

## REFERENCES

[1] L. Wang, “Analysis of factors affecting computer data processing speed,” *Journal of Physics: Conference Series*, vol. 1648, no. 2, p. 022136, Oct. 2020, doi: 10.1088/1742-6596/1648/2/022136.

[2] K. Asifuzzaman, N. R. Miniskar, A. R. Young, F. Liu, and J. S. Vetter, “A survey on processing-in-memory techniques: advances and challenges,” *Memories - Materials, Devices, Circuits and Systems*, vol. 4, p. 100022, Jul. 2023, doi: 10.1016/j.memori.2022.100022.

[3] H. Cetin, “Explaining the concept and operations of integer in primary school mathematics teaching: opposite model sample,” *Universal Journal of Educational Research*, vol. 7, no. 2, pp. 365–370, Feb. 2019, doi: 10.13189/ujer.2019.070208.

[4] V. A. Krasnobayev and S. A. Koshman, “Method for implementing the arithmetic operation of addition in residue number system based on the use of the principle of circular shift,” *Cybernetics and Systems Analysis*, vol. 55, no. 4, pp. 692–698, Jul. 2019, doi: 10.1007/s10559-019-00179-8.

[5] J. Freixas and S. Kurz, “On minimum integer representations of weighted games,” *Mathematical Social Sciences*, vol. 67, pp. 9–22, Jan. 2014, doi: 10.1016/j.mathsocsci.2013.10.005.

[6] M. A. Mohamed, “A survey on elliptic curve cryptography,” *Applied Mathematical Sciences*, vol. 8, no. 153–156, pp. 7665–7691, 2014, doi: 10.12988/ams.2014.49752.

[7] T. Choudhary, V. Mishra, A. Goswami, and J. Sarangapani, “A comprehensive survey on model compression and acceleration,” *Artificial Intelligence Review*, vol. 53, no. 7, pp. 5113–5155, Oct. 2020, doi: 10.1007/s10462-020-09816-7.

[8] E. C. Ateş, E. Bostanci, and M. S. Guzel, “Security evaluation of industry 4.0: understanding industry 4.0 on the basis of crime, big data, internet of thing (IoT) and cyber physical systems,” *Journal of Security Science*, vol. 9, no. 1, pp. 29–50, 2020.

[9] F. Yang and S. Gu, “Industry 4.0, a revolution that requires technology and national strategies,” *Complex & Intelligent Systems*, vol. 7, no. 3, pp. 1311–1325, Jun. 2021, doi: 10.1007/s40747-020-00267-9.

[10] A. Adel, “Future of industry 5.0 in society: human-centric solutions, challenges and prospective research areas,” *Journal of Cloud Computing*, vol. 11, no. 1, p. 40, Sep. 2022, doi: 10.1186/s13677-022-00314-5.

[11] A. D. Booth, “A signed binary multiplication technique,” *The Quarterly Journal of Mechanics and Applied Mathematics*, vol. 4, no. 2, pp. 236–240, 1951, doi: 10.1093/qjmam/4.2.236.

[12] G. W. Reitwiesner, *Binary arithmetic*, vol. 1. Elsevier, 1960.

[13] W. Bosma, “Signed bits and fast exponentiation,” *Journal de Théorie des Nombres de Bordeaux*, vol. 13, no. 1, pp. 27–41, 2001, doi: 10.5802/jtnb.301.

[14] J. Jedwab and C. J. Mitchell, “Minimum weight modified signed-digit representations and fast exponentiation,” *Electronics Letters*, vol. 25, no. 17, p. 1171, 1989, doi: 10.1049/el:19890785.

[15] D. Mandelbaum, “Arithmetic codes with large distance,” *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 237–242, Apr. 1967, doi: 10.1109/TIT.1967.1054015.

[16] J. L. Massey and O. N. García, “Error correcting codes in computer arithmetic,” in *Advances in Information Systems Science*, vol. 4, no. 5, Boston, MA: Springer US, 1971, pp. 273–326.

[17] S.-H. Chang and N. Tsao-Wu, “On the evaluation of minimum distance of binary arithmetic cyclic codes,” *IEEE Transactions on Information Theory*, vol. 15, no. 5, pp. 628–631, Sep. 1969, doi: 10.1109/TIT.1969.1054346.

[18] F. Morain and J. Olivos, “Speeding up the computations on an elliptic curve using addition-subtraction chains,” *RAIRO - Theoretical Informatics and Applications*, vol. 24, no. 6, pp. 531–543, Jan. 1990, doi: 10.1051/ita/1990240605311.

[19] M. Joye and Sung-Ming Yen, “Optimal left-to-right binary signed-digit recoding,” *IEEE Transactions on Computers*, vol. 49, no. 7, pp. 740–748, Jul. 2000, doi: 10.1109/12.863044.

[20] K. Okeya, K. Schmidt-Samoa, C. Spahn, and T. Takagi, “Signed binary representations revisited,” in *Advances in Cryptology — CRYPTO’ 2004*, Springer Berlin Heidelberg, 2004, pp. 123–139.

[21] K. Koyama and Y. Tsuruoka, “Speeding up elliptic cryptosystems by using a signed binary window method,” in *Advances in Cryptology — CRYPTO’ 1992*, Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 345–357.

[22] P. Balasubramaniam and E. Karthikeyan, “Elliptic curve scalar multiplication algorithm using complementary recoding,” *Applied Mathematics and Computation*, vol. 190, no. 1, pp. 51–56, Jul. 2007, doi: 10.1016/j.amc.2007.01.015.

[23] Chin-Chen Chang, Ying-Tse Kuo, and Chu-Hsing Lin, "Fast algorithms for common-multiplicand multiplication and exponentiation by performing complements," in *17th International Conference on Advanced Information Networking and Applications, 2003. AINA 2003*, 2003, pp. 807–811, doi: 10.1109/AINA.2003.1193005.

[24] E. De Win, S. Mister, B. Preneel, and M. Wiener, "On the performance of signature schemes based on elliptic curves," in *Algorithmic Number Theory*, Springer Berlin Heidelberg, 1998, pp. 252–266.

[25] I. Blake, G. Seroussi, and N. Smart, *Elliptic curves in cryptography*. Cambridge University Press, 2004.

[26] J. A. Solinas, "Efficient arithmetic on Koblitz curves," *Designs, Codes, and Cryptography*, vol. 19, no. 2–3, pp. 195–249, 2000, doi: 10.1023/A:1008306223194.

[27] H. Cohen, A. Miyaji, and T. Ono, "Efficient elliptic curve exponentiation," in *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 1997, pp. 282–290.

[28] R. M. Avanzi, "A note on the signed sliding window integer recoding and a left-to-right analogue," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3357, Springer Berlin Heidelberg, 2005, pp. 130–143.

[29] B. King, "WNAF*, an efficient left-to-right signed digit recoding algorithm," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5037 LNCS, Springer Berlin Heidelberg, 2008, pp. 429–445.

[30] J. Bos and M. Coster, "Addition chain heuristics," in *Advances in Cryptology — CRYPTO' 89 Proceedings*, Springer New York, 1989, pp. 400–407.

[31] J. Sauerbrey and A. Dietel, "Resource requirements for the application of addition chains in modulo exponentiation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 658 LNCS, Springer Berlin Heidelberg, 1992, pp. 174–182.

[32] D. E. Knuth, "The art of computer programming," *Vol 2. Seminumeral Algorithms*, Sep. 1981.

[33] Ç. K. Koç, "High-radix and bit recoding techniques for modular exponentiation," *International Journal of Computer Mathematics*, vol. 40, no. 3–4, pp. 139–156, Jan. 1991, doi: 10.1080/00207169108804009.

[34] W. Clark and J. Liang, "On arithmetic weight for a general radix representation of integers," *IEEE Transactions on Information Theory*, vol. 19, no. 6, pp. 823–826, Nov. 1973, doi: 10.1109/TIT.1973.1055100.

[35] M. A. Hassan and H. Wu, "Closed-form expression for the average weight of signed-digit representations," *IEEE Transactions on Computers*, vol. 48, no. 8, pp. 848–851, 1999, doi: 10.1109/12.795126.

[36] S. Arno and F. S. Wheeler, "Signed digit representations of minimal Hamming weight," *IEEE Transactions on Computers*, vol. 42, no. 8, pp. 1007–1010, 1993, doi: 10.1109/12.238495.

[37] D. L. F. Kong, J. Yu, Z. Cai, "Left-to-right generalized non-adjacent form recoding for elliptic curve cryptosystem," in *2006 International Conference on Hybrid Information Technology*, Nov. 2006, vol. 1, pp. 299–303, doi: 10.1109/ICHIT.2006.253503.

[38] M. Joye and S.-M. Yen, "New minimal modified radix-r representation with applications to smart cards," in *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2004, pp. 375–383.

[39] L. Washington, *Elliptic curves: number theory and cryptography*, 2nd Ed. Chapman & Hall, 2008.

[40] R. P. Gallant, R. J. Lambert, and S. A. Vanstone, "Faster point multiplication on elliptic curves with efficient endomorphisms," in *Annual International Cryptology Conference*, Springer Berlin Heidelberg, 2001, pp. 190–200.

[41] N. P. Smart, "Elliptic curve cryptosystems over small fields of odd characteristic," *Journal of Cryptology*, vol. 12, no. 2, pp. 141–151, Mar. 1999, doi: 10.1007/PL00003820.

[42] N. Koblitz, "CM-curves with good cryptographic properties," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 576 LNCS, Springer Berlin Heidelberg, 1991, pp. 279–287.

[43] W. Meier and O. Staffelbach, "Efficient multiplication on certain nonsupersingular elliptic curves," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 740 LNCS, Springer Berlin Heidelberg, 1992, pp. 333–344.

[44] S. V. D. Hankerson and A. Menezes, *Guide to elliptic curve cryptography*. New York: Springer-Verlag, 2004.

[45] M. Z. H. Arslan and A. Altoum, "Integer representations of classical Weyl groups," *arXiv preprint arXiv:2211.00427*, 2022, doi: 10.48550/arXiv.2211.00427.

[46] M. J. C. Lima and V. C. R. Júnior, "Adaptive universal codes for integer representation," *Journal of Communication and Information Systems*, vol. 28, no. 1, pp. 8–13, Apr. 2013, doi: 10.14209/jcis.2013.2.

[47] O.Trifonov and J. Dalton, "Representing positive integers as a sum of a squarefree number and a small prime," *Number Theory (math.NT), arXiv:2301.12585*, 2023.

[48] U. Isnaini, R. Melham, and P. C. Toh, "The number of representations of a positive integer by triangular, square and decagonal numbers," *Bulletin of the Korean Mathematical Society*, vol. 56, no. 5, pp. 1143–1157, 2019, doi: 10.4134/BKMS.b180914.

[49] H. Park, B. Cho, D. Cho, Y. D. Cho, and J. Park, "Representation of integers as sums of Fibonacci and Lucas numbers," *Symmetry*, vol. 12, no. 10, p. 1625, Oct. 2020, doi: 10.3390/sym12101625.

[50] S. P. and N.Thongngam, "Representation of integers of the formx2 + my2 − z2," *Journal of Integer Sequences*, vol. 24, 2021.

## BIOGRAPHIES OF AUTHORS

**Mohamad Afendee Mohamed** received his Ph.D. in Mathematical Cryptography from Universiti Putra Malaysia in 2011. Upon completion, he served the university for three years as a senior lecturer. From 2014, he moved to Universiti Sultan Zainal Abidin and later assumed an associate professor position. His current research interests include both theoretical and application issues in the domain of data security, and mobile and wireless networking. He has authored and co-authored more than 100 articles that have appeared in various journals, book chapters and conference proceedings. He can be contacted at email: mafendee@unisza.edu.my.

**Yahaya Garba Shawai** is a staff of National Open University of Nigeria. He is a Doctoral Researcher at The Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Malaysia. He obtained his Masters at the Field of E-Learning and Mobile Learning, which he published more than 7 research papers in some reputable journal such as theoretical and applied information technology, international journals article. Most of the journals were Scopus based. His research interest for his Ph.D. Program was Cryptography based Chaotic System in conjunction to differential equations. He can be contacted at email: yshawai@noun.edu.ng.

**Mohammed Amin Almaiah** is an Associate Professor in the Department of Cybersecurity and Cloud Computing. He has published over 95 research paper in highly reputed journals such as Engineering and Science Technology, an International Journal, Education and Information Technologies, IEEE Access and others. Most of his publication were indexed under the ISI Web of Science and Scopus. His research interest include cybersecurity and cyber-risk assessment and mobile apps. He can be contacted at email: malmaiah@kfu.edu.sa.

**Mohd Noor Derahman** is a lecturer at Department of Communication Technology and Network, Faculty of Computer Science, and Information Technology, Universiti Putra Malaysia. His research interests include software defined networking and resource optimisation. He can be contacted at email: mnoord@upm.edu.my.

**Abdalwali Lutfi** is an Associate Prof of Accounting or Accounting Information System in King Faisal University (KFU) – KSA. Dr. Lutfi holds an undergraduate degree in Accounting from Irbid University, a master’s degree in accounting from Jadara University, and a Ph.D. in Accounting or Accounting Information System from University Utara Malaysia. His teaching and research interests are in the area of accounting and accounting information system. His current research involves end-user computing, adoption and usage of AIS or ISs, cloud-based technologies, digital transformation, big data analytics, environmental practicesm, and electronic payment systems. He has published many refereed articles in high ranked journals such as Journal of Retailing and Consumer Services, Technology of Society, Int. J. Environ. Res. Public Health, EuroMed Journal of Business, Sustainability, Frontiers in Environmental Science, Global Knowledge, Memory and Communication and Global Business Review. He can be contacted at email: aalkhassawneh@kfu.edu.sa.

**Khairul Azmi Abu Bakar** received a degree in Computer Engineering from Iowa State University, USA and a master's degree in Communication and Computer Engineering from Universiti Kebangsaan Malaysia. He was awarded a Ph.D. Degree in Electrical Engineering from the University of Strathclyde, United Kingdom, for the study on free-riding nodes in an open MANET. He is currently a senior lecturer at the Center for Cyber Security under the Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia. Earlier, he was a researcher at MIMOS Berhad, Malaysia's national applied R&D center in microelectronics and ICT. He has been involved in many R&D projects in micro-controller, smartcards, and security systems under open-source platforms. His primary research interests include network security, internet of things, and computer networks. He is also an IEEE member. He can be contacted at email: khairul.azmi@ukm.edu.my.