International Journal of Engineering Trends and Technology
ISSN: 2231 – 5381 /doi:10.14445/22315381/IJETT-V69I2P205
Volume 69 Issue 2, 32-38, February 2021

Generation of Addition Chain using Bacteria Foraging Optimization Algorithm

Dr.K.Mani[1], A. Mullai[2]

[1]*Associate Professor in Computer Science, Nehru Memorial College(Autonomous), Puthanampatti, Affiliated to Bharathidasan University, Tiruchirappalli, Tamil Nadu, India.*

[2]*Associate Professor in Computer Science, Seethalakshmi Ramaswami College (Autonomous), Affiliated to Bharathidasan University, Tiruchirappalli, Tamil Nadu, India.*

[1]nitishmanik@gmail.com, [2]mullai_a@yahoo.com

Abstract

In many number-theoretic cryptographic algorithms, encryption and decryption are of the form $x^n \bmod p$ where x, n and p are integers. When exponentiation operation is involved in many cryptosystems, it takes more time than any normal arithmetic operations. The computation time can be reduced by using repeated multiplications rather than using exponential operation. This can also be further reduced by using addition chain. Modular exponentiation with addition chain is used to determine the correct sequence of multiplications. There exist several algorithms in the literature to generate the optimal addition chain for the given integer. A novel bacteria foraging optimization algorithm based addition chain has been proposed and it is verified with the existing state of art of addition chain algorithms like genetic algorithm, evolutionary programming etc., in this paper.

Keywords *- Addition Chain, RSA, ECC, PSO, SSO, BFOA, Optimization.*

I. INTRODUCTION

An Addition Chain (AC) can be thought of as a sequence of integers in which the first number is always 1 and the last number is always n, where n is an integer for which ACs are to be generated. For finite fields, operations such as square roots or inversions, exponentiations can be performed efficiently by utilizing an optimal AC, the smallest such AC sequence to reach n. In particular, fast exponentiation and inversion are paramount to the performance of scalar point multiplication $k[P]$ where k is a scalar and P is a point in elliptic curve (EC) in elliptic curve cryptography (ECC) [1] [2], pairings in pairing-based cryptosystems, and computing isogenies in the quantum-resistant isogeny-based cryptosystems [23]. To get the next number, there are two steps normally used in AC. They are addition and doubling steps, i.e., to get the next number (intermediate number) in AC, any two previous numbers are added together in addition step, whereas in the doubling step, the current number is multiplied by two. To generate the AC for given n, two types of algorithms are normally used viz., deterministic and stochastic or bio-inspired.

In deterministic algorithms, since everything is deterministic and the optimal AC may not be obtained at all times. The binary method, factor method, window method, sliding window method, Fibonacci method, Lucas method, continuous fraction method, etc., are examples of the deterministic algorithm. Evolutionary algorithms or bio-inspired are inspired by the idea of either natural evolution or social behavior of insects or birds. The optimal ACs produced by evolutionary algorithms are not obtained by a single run. Many more runs are needed to obtain optimal AC, which will eventually take more times. Some examples of evolutionary algorithms are Genetic Algorithm(GA), Artificial Immune System(AIS), Ant Colony Optimization(ACO), Particle Swarm Optimization (PSO), Simplified Swarm Optimization(SSO), etc. Generating optimal AC for the given integer is an NP-hard problem because too many optimal ACs are generated. For example, different possible optimal ACs for the number 21 with length i.e., l(21) = 6 are:

1-2-3-4-7-14-21	1-2-3-6-9-15-21	1-2-4-5-10-20-21	1-2-3-6-9-12-21
1-2-3-5-7 -14-21	1-2-3-6-9-18-21	1-2-4-8-9-12-21	1-2-4-8-12-13-21
1-2-3-5-8-13-21	1-2-3-6-12-15-21	1-2-4-8-9-13-21	1-2-4-8-12-20-21
1-2-3-5 -8-16-21	1-2-3-6-12-18-21	1-2-4-8-9-17-21	1-2-4-8-16-17-21
1-2-3-5-10-11-21	1-2-4-6-7-14-21	1-2-4-8-10-11-21	1-2-3-6-7-14-21
1-2-3-5-10-20-21	1-2-4-8-16-20-21	1-2-4-8-10-20-21	

This is because 7 can be obtained by adding (7 = 3 + 4, 7 = 2 + 5, 7 = 1 + 6), 8 can be obtained by adding (8 = 4 + 4, 8 = 3 + 5) etc.

Bacteria foraging is one of the optimization and evolutionary algorithms. Kevin M. Passino proposed it in 2000, and it has been widely accepted as a new nature-inspired optimization algorithm [15]. It is inspired by the social foraging behavior of Escherichia Coli, i.e., a bacteria present in the human intestine and has drawn many researcher's attention. The underlying biology behind foraging is locomotion. During the foraging of the real bacteria, locomotion can be performed by a set of the tensile flagella and optimization process is achieved by foraging behaviour of bacteria in bacterium seeks to maximize the energy obtained per unit time spent during foraging. Suppose the flagella are rotated in the clockwise direction by the bacterium. In that case, the flagellum pulls on the cells, which results in independent movement of flagella, and the bacterium tumbles with lesser numbers of tumbling. Swimming at a very fast rate of the bacterium is performed with the flagella moving in the counter-clockwise direction.

The foraging strategy of E.coli is achieved by four processes viz., chemotaxis, swarming, reproduction and dispersal. Chemotaxis is a process which simulates the movement of E.coli cell through swimming and tumbling via flagella. Movement of E.coli bacterium can be performed in two ways viz., (i) swim for some time in the same direction or tumble (ii) alternate between the swim and tumble for the entire lifetime. In the swarming process, a group of E.coli cells arranged themselves in a traveling ring by moving up the nutrient gradient when placed amidst a semisolid matrix with a single nutrient chemo-effecter. The healthy bacteria asexually split into two bacteria, which are then placed in the same location while the least healthy bacteria eventually die in the reproduction process. In the elimination and dispersal process, gradual or sudden changes in the local environment, i.e., the significant local rise of temperature or due to unavoidable events, all the bacteria in a region are killed, or a group is dispersed into the new location.

In BFOA, generally, the bacteria can be moved for a long distance in a friendly environment. When sufficient food they had, their length also increased and will break in the middle to form a replica of themselves in the presence of a suitable environment. In swarm intelligence concept, this chemotactic progress may be eliminated and also a group of bacteria can move on to some areas or introduce some others related to the occurrences environmental changes like the event of elimination- dispersal done in the real bacterial population (where all the bacteria in a region are killed or a group will be dispersed into a new part of the environment).

II. RELATED WORK

In [3], Hugo Volger presented several results on $l(n)$. In particular, they determined $l(n)$ for all n satisfying $l(n)\leq 3$ and proved $\lfloor log n \rfloor + 2 \leq l(n)$ for all n satisfying $s(n) \geq 3$, where $s(n)$ is the extended sum of digits of n. In [4], Y.H. Tsai and Y.H. Chin found some mathematical properties of the shortest-length AC for certain integers whose binary patterns meet some special forms; and the correctness of these properties was proved. In [5], Bergeron et al. proposed generating the shortest AC based on the continued fraction. They gave a general upper bound for the complexity of continued fraction methods as a chosen strategy function. Thus, the total number of operations required for the generation of an AC for all integers up to n was shown to be $(n\ log^2 n\gamma n)$, where γn is the complexity of computing the set of choices corresponding to the strategy and proved an analogy of the Scholz-Brauer conjecture.

In [6], F. Bergeron et al. generated a method of fast addition chains with a small length of positive integer n, using continued fraction up to 1000 obtained with optimal length, (with 29 exceptions optimal length plus one). A new algorithm of optimal ACs described in [8] and also faster than the best-known methods. It is applicable for single values and slower than the best-known methods. This does not require any pre-computed values and is considered suitable for finding optimal ACs for point values.

Bounds on sums of ACs and properties of optimal ACs are discussed in [9].The study results that the final step in an optimal AC of an even number always have doubling, and also the sum of an optimal AC for an odd number $n$is asymptotically nearly $5n^2$. In [10], Noboru Kunihiro and Hirosuke Yamamoto developed two systematic methods viz., run-length encoding (RLE) and hybrid for generating short AC. They proved that the hybrid method was far better than RLE with a reduced 8% of the AC length.

In[11], Nareli Cruz-Cortéset al. explored the usage of a GA approach for the problem of finding optimal (shortest) ACs for optimal field exponentiation computations. The GA heuristic presented in this work was capable of finding almost all the optimal ACs for any given fixed exponent $e$with $e<4096$. They found that our GA strategy's percentage error was within 0.4% of the optimal for all cases considered. In other words, for any given fixed exponent e with $e<4096$, they found that strategy was able to find the requested shortest AC in at least 99.6% of the cases. In [12], N. Cruz- Cortés et al. proposed an artificial immune system(AIS) to generate an optimal AC. In that paper, they dealt with the optimal computation of finite field exponentiation, which is a well-studied problem with many important applications in error-correcting codes and cryptography.

In [13], Raveen R. Gounder et al. discussed a new strategy for doubling-free (SPA-resistant) short addition-subtraction chain(GRASC) for an arbitrary integer by using a precise golden ratio. In this, 12% to 28% reduction was obtained in the average chain length compared to other doubling-free AC methods. In [14], Alejandro Le´on-Javier et al. discussed the PSO algorithm to find short ACs with different exponents.

In[16], Mohamed M. Abd. Eldayamet al. proposed an algorithm for shorter AC based on the window method with small width using 2's complement. They proved that the proposed algorithm was more efficient than the last result with a 20% minimum. In[17], S Domínguez-Isidro and E Mezura-Montes et al. proposed an algorithm using evolutionary programming to find the minimal length AC and the results obtained were more promising than the other nature-inspired metaheuristic approaches but with a lower number of evaluations per run. The proposed EP algorithm comprised the solution encoding with suitable fitness function and initial population, a mutation operator, and the survivor selection mechanism, and EP does not use other operators such as crossover nor additional mechanisms like parent selection in GAs.

In [18], a note an addition chain was presented. Niel Michael Clift [19] proved the perfect matches in the Scholz–Brauer conjecture l(2n −1) = l(n) + n − 1 for new values. The minimal sequence of minimal multiplications required for performing modular exponentiation using Brauer Chains' concept by GA discussed in [20].

In [21], K. Mani proposed division based AC to generate the optimal ACs for the small exponents, exactly matched with ACs generated by the latest methods. But, for some large exponents, there was a very small increase in chain length (at most three).

In [24], a survey of the AC problem for optimizing the AC was made and effectively applied to implement a public-key cryptosystem. Mani K and Viswambari M [25] implemented a new method for the generation of the AC using graph G(V,E) where in the G's vertices refer to the numbers in the AC and edges refer to the move from one to another number in the AC. They have proposed two methods viz., Graph-Based All Possible AC (GBAPAC) generated all possible optimum ACs for the given integer n and Graph-Based Minimal AC (GBMAC), which generated the minimum number of optimum ACs by considering mutually exclusive edges starting from every number and also proved with the conjectures like Scholz-Brauer.

In [26], P. Anuradha Kameswari and B. Ravitheja derived a Lucas AC for any integer n to obtain Lucas sequence $V_n(a, 1)$ and also proved that the computation of $V_n(a, 1)$ using this Lucas AC is based on $V_{x+y}(a, 1)$ for $x, y, x - y$ in the Lucas AC. In [27], Stjepan Picek et al. derived that the GA approach with an novel encoding using crossover and mutation operators to minimize the length of the ACs with respect to a given exponent. Aaron Hutchinson and Koray Karabina implemented algorithms[28], for multidimensional differential ACs and applied these chains to ECC. This algorithm has the unique key features using n dimension. With key efficiency cum security features like uniformity, parallelized, and differential addition formulas were adopted by allowing speed using precomputation cost and storage requirements.

Dustin Moody and Amadou Tall [29], derived minimal chains with low Hamming weight using addition-subtraction chains with Lucas addition-subtraction in using $\ell-(n)$ the minimal length n, and proved that $|\ell-(2n) - \ell-(n)| \leq 1$ for all integers n of $Hammingweight \leq 4$ to have arrived a conclusion that minimal addition-subtraction chains for low Hamming weight integers, with the consideration of odd integers. In [30], Hazem M. Bahig and Yasser Kotb implemented a new parallel algorithm to obtain minimal AC for n. The experimental studies on multicore systems revealed that this algorithm's run time worked faster than the sequential one and obtained the maximum speed up of 2.5 times than the best known sequential algorithm.

In [31], A. Mullai and K. Mani proposed Particle Swarm Optimization (PSO) and Simplified Swarm Optimization (SSO) with ACs in RSA and ECC with two emulators, android and window. The processing time, power consumption was taken for encryption, decryption process, and security of the above was analyzed and also proved that the SSOAC optimization with RSA to reduce operational power and SSOAC optimization with ECC for more security. Narendra Mohan [32], discussed in Wireless Sensor Networks (WSNs), to enhance the network lifetime and minimize the energy consumption in sink nodes contains additional resources like long-range antenna, powerful batteries, large memory. This should be achieved using Enhanced Emperor Penguin Optimization (EEPO) algorithm.

III. THEORETICAL BACKGROUND

This section describes some mathematical preliminaries required for AC.

Definition 3.1 (Addition Chain)

An AC [7] for a positive integer n is a sequence, $1 = a_0 \leq a_1 \leq \cdots \leq a_r = n$ such that each member after a_0 is the sum of two earlier (not necessarily distinct) ones. The number $l(n)$ is called the length of the AC. It is noted that if the value of n is relatively small, the exact value of $l(n)$ is known.

Definition 3.2 (Optimal Addition Chain)

An AC is optimal if its length is the smallest among all possible ACs. For example, 1− 2 − 3 − 6 − 12 − 13 is one of the optimal chains for 13, and with l(13) =5.

The construction [20] of each element of an AC is called a step. For an AC, $1 = a_0 \leq a_1 \leq \cdots \leq a_r = n$, the following steps are involved. Doubling step: $a_i = 2a_{i-1}$, $i > 0$. Non-doubling step: $a_i = a_j + a_k$, $i > j > k \geq 0$. The steps of the form $a_i = 2a_j$, $j \leq i - 2$ are defined as non-doubling steps.

Big step: $\lambda(a_i) = \lambda(a_{i-1}) + 1$.

Small step: $\lambda(a_i) = \lambda(a_{i-1})$.

Thus, the length of the AC, $l(n)$ can be split into two components as $l(n) = \lambda(n) + S(n)$. where $S(n)$ is the number of small steps in an optimal AC for n.

IV. BFOA_AC - PROPOSED METHODOLOGY

In the proposed methodology, the concept of BFOA is used to generate the optimum length AC for an integer n, which utilizes the foraging behaviors of bacteria. i.e., chemotaxis, swarming, reproduction, and elimination dispersal [15], are the four principal mechanisms used in BFOA. In this optimization, a virtual bacterium called search agent is one trial solution that moves on the functional surface to find the optimal length AC. The cost or fitness function is computed with a minimum length approach based on the nutrient concentration of the bacterium's immediate environment, searching for numbers in AC. The swarming step is not considered for the generation of AC in this method. The following notations are used in generating the optimal AC in this paper.

j	Index for the chemotactic step
k	Index for the reproduction step
i	Index for the elimination-dispersal event
S	Total number of the bacterium in the population
d	The dimension of the search space. Here, d = 1
S_w	The swarming length
RP_n	Number of reproduction steps
ED_n	Number of elimination-dispersal events
P_{ed}	Elimination-dispersal probability
$C(i)$	The magnitude of the next number in the random direction specified by the tumble

To generate the AC for any integer n, the first number is always 1, and the second number is 2, i.e., AC starts with $a_0 = 1$ and $a_1 = 2$ and last number $a_r = n$. Let $(i, k, l) = \{(j, k, l | i = 1,2, \ldots, S)$ represents each number in the AC in the population S at the j^{th} chemotactic, k^{th} reproduction, and l^{th} elimination-dispersal steps. It is noted that initially, the length of AC is taken as very large for the given integer n. Too many ACs are generated for n, but all ACs generated are not necessarily optimum. Moreover, the generation of optimal AC is an NP-hard problem. The prime steps used in BFOA related to generating the AC are as follows.

A. Search Space

Here, the search space is taken as one dimension(i.e.,d=1), and also the integer numbers are involved in generating AC for any n. Since the difference between intermediate numbers in AC is finite, the search space is also finite.

B. Chemotaxis

The movement of an E.coli cell through swimming and tumbling via flagella is simulated by the chemotaxis process. When a bacterium meets a favourable environment (rich in nutrients and noxious free), it will continue swimming in the same direction. When it meets an unfavorable environment, it will tumble, i.e., change its direction**.** In BFOA[22], E.coli can swim for a period of time in the same direction, or it may tumble and alternate between these two modes of operation for the entire life time. It is the most important step in determining the optimal AC for n. For AC generation, swimming and tumbling represent addition and doubling step, respectively. The goal is to move to let the bacterium search for the next number in the AC with minimal step.

a) Minimum Intermediate Number in AC

It is noted that the number of intermediate numbers between 2 and n should be minimum and it is obtained by a minimum number of steps as far as possible so that $l(n)$ could be minimized by considering all the directions (previous numbers) from the current bacterium position (present current number) can be chosen for the next step. Initially bacterium i is positioned at number 1, Let $m=0$ i.e., $a_0 = 1$. From 1, then it should move to 2. Now, $m = m + 1$ i.e., $a_1 = 2$, $AC \leftarrow 1 - 2$; $l(AC) = 1$. From 2, it can move to either 3 or 4, Now, $m = m + 1$

$$2a_{m-1} = a_m + a_0,\ i > j > k \geq 0 \quad \ldots(1)$$

$$AC \leftarrow AC || a_m \quad \ldots(2)$$

Now,

$$new\_l(AC) = old\_l(AC) + l \ \ or \ \ l(AC) = m \quad \ldots(3)$$

All the intermediate numbers obtained in this step are added to the minimal set Φ_{min},i.e., $\Phi_{min} = \{a_m\}$. A random intermediate number $\leq a_m$ is chosen from this set, and it indicates the direction of movement (i.e., from which AC starts) of bacterium$i$.

$$\Delta(i) = rand\{\ x \in \Phi_{min}\} \quad \ldots(4)$$

Let, (j, k, l) represents i^{th} bacterium with 1-dimensional vector represented as, 1,2, …, Sat j^{th} chromatic, k^{th} reproductive and l^{th} elimination-dispersal step. Let C(i) be the step size, which is taken as a unity because, from the current number in the AC, only one next number in the AC is generated based on previous numbers. Thus, the movement of the bacterium may be represented in the chemotaxis process as

$$(j+1, K) = \theta^i(j,k) + C(i)\frac{\Delta(i)}{\sqrt{\Delta i^T \Delta(i)}} \quad \ldots(5)$$

Where Δ indicates a vector in the random direction whose elements are [1, x].
The movement of the bacterium is explained with tree diagram as shown in fig. 1.

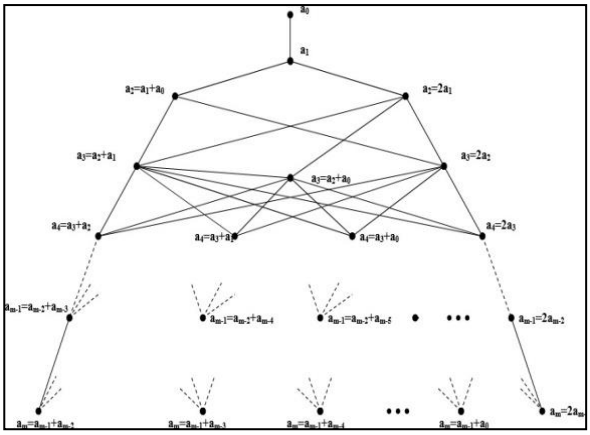

Fig.1: The Movement of Bacterium

C. Reproduction and Dispersal Step

Local search is provided by chemotaxis step, and the speed of convergence is achieved through the reproduction process. The bacteria which yields the maximal length of AC for n is called the least healthy bacteria, and it never produces the optimal length AC, which eventually dies. Each of the healthiest bacteria (yields minimum length AC) is asexually split into two bacteria, placed randomly. The dispersion process happens after a certain number of reproduction process. Depending on the probability, some bacteria were chosen to be killed or move to another position within the environment.

V. BFOA_AC – AN EXAMPLE

In order to understand the relevance of the work, let, n = 14, i = 1, m = 0, $a_m = a_0 = 1$ and initially bacteria b_1 is positioned at a_0. With the chemotaxis step, it moves to 2. Now, $m = m + 1$, i.e., $a_1 = 2$ and $l(a_1) = 1$. From a_1, b_1 moves to either 3 or 4 because $a_2 = a_1 + a_0 = 2 + 1 = 3$ or $a_2 = 2a_1 = 4$. Now, m = 2. Thus, Φ_{min}={3,4}. Let the intermediate number in AC randomly selected from Φ_{min}, i.e., $\Delta(1)$ = 3, Thus, the movement of b_1 is from 3, i.e., a_2=3 and the corresponding AC up to this stage is $1 - 2 - 3$ and $l(a_2) = 2$. From a_2, b_1 moves to either 4 or 5 or 6 because $a_3 = 2\,a_2 = 6$ or $a_3 = a_2 + a_0 = 3 + 1 = 4$ or $a_3 = a_2 + a_1 = 3 + 2 = 5$.Now, m = 3. Thus, Φ ={4,5,6}. Let 5 is selected randomly from the set Φ_{min}. Thus, $\Delta(1)$ = 5. The movement of b_1 is from 5, i.e., a_3 = 5. Correspondingly, AC up to this stage is $1 - 2 - 3 - 5$ and $l(a_3)$ = 3. From a_3, b_1 moves to either 6 or 7 or 8 or 10 because $a_4 = 2a_3 = 10$ or $a_4 = a_3 + a_0 = 5 + 1 = 6$ or $a_4 = a_3 + a_1 = 5 + 2 = 7$ or $a_4 = a_3 + a_2 = 5 + 3 = 8$. Now, m = 4. Thus, Φ_{min} = {6,7,8,10}. Let 7 is selected randomly from the set Φ_{min}. Thus, $\Delta(1)$ =7. The movement of b_1 is from 7, i.e., a_4 = 7. Correspondingly, AC up to this stage is $1 - 2 - 3 - 5 - 7$ and $l(a_4) = 4$. From a_4, b_1 moves to either 8 or 9 or 10 or 12 or 14 because $a_5 = a_4 + a_0 = 7 + 1 = 8$ or $a_5 = a_4 + a_1 = 7+2 = 9$ or $a_5 = a_4 + a_2 = 7+3 = 10$ or $a_5 = a_4 + a_3 = 7 + 5 = 12$ or $a_5 = 2(a_4) = 2(7) = 14$. Now, m = 5, Thus, Φ_{min} = {8,9,10,12,14}. Let 14 is selected randomly from the set Φ_{min}. Thus, $\Delta(1)$ = 14. The process is terminated because it reaches n = 17. Correspondingly, AC up to this stage is

$$1 - 2 - 3 - 5 - 7 - 14 \text{ and } l(a_5) = 5.$$

Suppose, other numbers from Φ_{min} are selected, even though it reaches 14 in the subsequent stages, l(14) is increased, and the corresponding bacteria will eventually die. Repeat the said process for other numbers, and the other ACs for 14 with l*(14) are given below.

1-2-3-4-7-14	1-2-4-5-7-14	1-2-4-6-8-14
1-2-3-5-7-14	1-2-4-5-9-14	1-2-4-6-10-14
1-2-3-6-7-14	1-2-4-5-10-14	1-2-4-6-12-14
1-2-3-6-8-14	1-2-4-6-7-14	1-2-4-8-10-14
1-2-3-6-12-14		1-2-4-8-12-14

VI. IMPLEMENTATION

The proposed methodology is implemented in VC++ and AC for the numbers up to 1024 are generated. It is shown in table 1. In table 1, l(r) indicates the sum of all optimal addition chains up to r. Table 1 exhibits the total l (up to 1024).

TABLE 1 TOTAL LENGTH OF OPTIMAL ADDITION CHAIN UPTO 1024

r	001-100	101-200	201-300	301-400	401-500	501-600	601-700	701-800	801-900	901-1000	1001-1024
l(r)	663	918	1011	1071	1121	1148	1183	1205	1230	1262	307
Total	4784					6028					307
Grand Total:11119											

Table 2 reveals AC generated for some hard exponents by BFOA where the hard exponent is the one for which AC is not easily found. Table 3 compares the optimal AC up to integers 1024 produced by the existing algorithms and the proposed BFOA.

TABLE 2 AC FOR HARD EXPONENTS BY BFOA

Exponents (E)	Optimal length (E)
2000	1 – 2 – 3 – 6 – 7 – 14 – 15 – 30 – 31 – 62 – 124 – 125 – 250 – 500 – 1000 – 2000.
2048	1 – 2 – 4 – 8 – 16 – 32 – 64 – 128 – 256 – 512 – 1024 - 2048.
4096	1 – 2 – 4 – 8 – 16 – 32 – 64 – 128 – 256 – 512 – 1024 – 2048 - 4096.
65131	1 – 2 – 3 – 5 – 7 – 11 – 19 – 29 – 47 – 71 – 127 – 191 – 379 – 607 – 1087 – 1903 – 3583 – 6271 – 11231 – 18287 – 34303 - 65131.
196591	1 - 2 - 3 - 5 - 7 - 11 - 19 - 29 - 47 - 71 - 127 - 191 - 379 - 607 - 1087 - 1903 - 3583 - 6271 - 11231 - 18287 - 34303 - 65131 - 110591 – 196591.
1176431	- 7 - 11 - 19 - 29 - 47 - 71 - 127 - 191 - 379 - 607 - 1087 - 1903 - 3583 - 6271 - 11231 - 18287 - 34303 - 65131 - 110591 - 196591 - 357887 - 685951 – 1176431.
2211837	1 - 2 - 3 - 6 - 9 - 15 - 30 - 60 - 120 - 126 - 252 - 504 - 1008 - 2016 - 4032 - 8062 - 16128 - 16143 - 32286 - 64572 - 129144 - 258288 - 516576 - 1033152 - 2066304 - 2195448 - 2211591 - 2211717 – 2211837.
4169527	1 - 2 - 3 - 5 - 7 - 11 - 19 - 29 - 47 - 71 - 127 - 191 - 379 - 607 - 1087 - 1903 - 3583 - 6271 - 11231 - 18287 - 34303 - 65131 - 110591 - 196591 - 357887 - 685951 - 1176431 - 2211837 – 4169527.
14143037	1 - 2 - 3 - 5 - 7 - 11 - 19 - 29 - 47 - 71 - 127 - 191 - 379 - 607 - 1087 - 1903 - 3583 - 6271 - 11231 - 18287 - 34303 - 65131 - 110591 - 196591 - 357887 - 685951 - 1176431 - 2211837 - 4169527 - 7624319 – 14143037.

From table 3, it is observed that the total length of optimal AC produced by BFOA with integers up to 1024 is 11119. They are almost the same as the optimal addition chains and their length produced by EP.

TABLE 3 COMPARISON OF AC UPTO INTEGERS 1024(PRODUCED BY EXISTING ALGORITHMS AND THE PROPOSED BFOA)

R	Opt.	AIS	GA	EP	BFOA
[1,512]	4924	4924(+)	4924	4924	4924
[1,1000]	10808	10813(+)	10813	10808	10812
[1.1024]	11115	11120(+)	-	11115	11119

VII. CONCLUSION

BFOA based AC has been thought of and it is implemented successfully. In this paper, ACs produced by some integers are proved both theoretically and experimentally. From the experimental results, up to integers 1024, the proposed BFOA algorithm produces the same optimal length AC which is almost equal to other existing evolutionary algorithms like AIS, GA, and EP. Further, the optimal length of AC for some hard exponents are the same as other existing evolutionary algorithms. This paper also provides an idea about the generation of AC based on BFOA. In future, this concept may be incorporated into public-key algorithms like RSA and ECC to reduce the encryption and decryption time because the said algorithms are used in mobile devices.

REFERENCES

[1] N Koblitz, Elliptic Curve Cryptosystems, Mathematics of Computation, 48(1982) 203-209.

[2] I Blake, G Seroussi and NP Smart, Elliptic Curves in Cryptography, Ser. London Math. Soc. Lecture Note Series, Cambridge Univ. Press,1999.

[3] Hugo Volger, Some Results on Addition/Subtraction Chains, Information Processing Letter, Elsevier, 1985.

[4] Y H TsaiandY H Chin, "A Study of Some Addition Chain Problems", International Journal of Computer Mathematics, 22(02) (1987) 117-134.

[5] R Begeron, J Berstel, S Brlek, and C Duboc, Addition Chains Using Continued Fractions, Journal of Algorithms, Elsevier, 10(1989) 403-412.

[6] Bergeron, JBerstel and S Brlek, Efficient Computation Of Addition Chains", Journalde Théorie des Nombresde Bordeaux, 6(1)(1994) 21-38.

[7] Donald E Knuth, The Art of Computer Programming, Seminumerical Algorithms, 2(3), Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA,1997.

[8] Gordon DM, A Survey of Fast Exponentiation Methods, Journal of Algorithms, 27(1998).

[9] H Zantema, Minimizing Sums of Addition Chains, Journal of Algorithms, Elsevier, 12(2) (1999) 281-307.

[10] Noboru Kunihiro and Hirosuke Yamamoto, New Methods for Generation of Short Addition Chains, IEICE Transactions Fundamental, 83(1)(2000).

[11] Nareli Cruz-Cortés, Francisco Rodriguez-Henriquez, RaúlJuárez-Morales and Carlos A Coello- Coello, Finding Optimal Addition Chains Using a Genetic Algorithm Approach, Springer- Verlag, (2005) 208-215.

[12] N Cruz-Cortes, F Rodriguez-Henriquez, and C A Coello-Coello, An Artificial Immune System Heuristic for Generating Short Addition Chains, IEEE Transactions on Evolutionary Computation, 6(2005) 252–280.

[13] Raveen R Goundar, Ken-ichiShiota, M Toyonaga,New Strategy for Doubling - Free Short Addition-Subtraction Chain, Mathematics,2008.

[14] AlejandroLe´on-Javier,NareliCruz-Cort´es,MarcoAMoreno-Armend´ariz,andSandraOrantes- Jim´enez, Finding Minimal Addition Chains with a Particle Swarm Optimization Algorithm, Advances in Artificial Intelligence, Springer, (2009) 680-691.

[15] Swagatam Das, ArijitBiswas, Sambarta Dasgupta, and Ajith Abraham, "Bacterial Foraging Optimization Algorithm: Theoretical Foundations, Analysis, and Applications", Foundations of Computational Intelligence, Springerlink.com, Springer-Verlag Berlin Heidelberg, 3SCI 203(2009) 23–55.

[16] Mohamed M Abd-Eldayem, EhabT Alnfrawy, and AlyA Fahmya, Addition-Subtraction Chain for 160-bit Integers by using 2's Complex N Cruz-Cortés, F Rodríguez-Henríquez,and C A Coello-Coello, Addition Chain Length Minimization With Evolutionary Programming, Proceedings of Genetic and Evolutionary Computation Conference (GECCO) ACM digital Library, (2011).

[17] S Domínguez-Isidro and E Mezura-Montes, An Evolutionary Programming Algorithm to Find MinimalAdditionChains,ICongresoInternacionaldeIngenieríaElectrónica,Instrumentación y Computación, de Juniodel, Minatitlán Veracruz, México,2011.

[18] Maurice Mignotte, A Note on Addition Chains, International Journal of Algebra, 5(6)(2011).

[19] Neill Michael Clift, Calculating Optimal Addition Chains", Journal of Computing, Springer, 91 (2011) 265–284.

[20] Arturo Rodriguez-Cristerna and Jose Torres-Jimenez, A Genetic Algorithm for the Problem of Minimal Brauer Chains for Large Exponents, Soft Computing Applications in Optimization, Control, and Recognition, Springer, (2013) 27-5.

[21] K. Mani, Generation of Addition Chain using Deterministic Division Based Method, International Journal of Computer Science & Engineering Technology, 4(05) (2013) 553- 560.

[22] Om PrakashVerma, Rashmi Jain, and Vindhya Chhabra, Solution of Travelling Salesman Problem Using Bacteria Foraging Optimization Algorithm, International Journal of Swarm Intelligence, Inderscience publisher, 1(2) (2014).

[23] BrianKoziel, Reza Azarderakhsh, David Jaoand Mehran Mozaari-Kermani, On Fast Calculation of Addition Chains for Isogeny - Based Cryptography, Inscrypt 2016, IACR Cryptology, 2016.

[24] Adamu Muhammad Noma, Abdullah Muhammed, Mohamad Afendee Mohamed, and Zuriati Ahmad Zulkarnain. A Review on Heuristics for Addition Chain Problem: Towards Efficient Public-Key Cryptosystem, Journal of Computer Science, 13(2017) 275-289.

[25] K Mani, M Viswambari, A New Method of Generating Optimal Addition Chain Based on Graph, International Journal of Mathematical Sciences and Computing, 2(2017) 37-54.

[26] P Anuradha Kameswari and B Ravitheja, Addition Chain For Lucas Sequences With Fast Computation Method, International Journal of Applied Engineering Research, 13(11) (2018) 9413–9419.

[27] Stjepan Picek, Carlos A CoelloCoello, Domagoj Jakobovic and Nele Mentens, Finding Short And Implementation - Friendly Addition Chains With Evolutionary Algorithms, Journal of Heuristics, 24 (2018) 457-481.

[28] Aaron Hutchinson and Koray Karabina, Constructing Multidimensional Differential Addition Chains and Their Applications, Springer, Journal of Cryptographic Engineering, 9(2019) 1- 19.

[29] Dustin Moody and Amadou Tall, On Addition-Subtraction Chains of Numbers With Low Hamming Weight", Number Theory Mathematics, 25(2019) 155-168.

[30] Hazem M. Bahig· and Yasser Kotb, An Efficient Multicore Algorithm for Minimal Length Addition Chains, Computers, MDBI, 8(2019).

[31] A Mullai and K Mani, Enhancing The Security In RSA and Elliptic Curve Cryptography Based on Addition Chain Using Simplified Swarm Optimization and Particle Swarm Optimization For Mobile Devices, International Journal of Information Technology, Springer, (2020).

[32] Narendra Mohan Lifetime Enhancement of Sensor Nodes Based On Optimized Sink Node Placement Approach, International Journal of Engineering Trends and Technology 68.10(2020):10-23.