Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

A novel cryptosystem based on DNA cryptography, hyperchaotic systems and a randomly generated Moore machine for cyber physical systems

Pramod Pavithran [a], Sheena Mathew [a], Suyel Namasudra [b,c], Gautam Srivastava [d,e,*]

[a] Division of Computer Engineering, School of Engineering, Cochin University of Science and Technology, Kochi, India
[b] Department of Computer Science and Engineering, National Institute of Technology Patna, Bihar, India
[c] Universidad Internacional de La Rioja, Logroño, Spain
[d] Dept. of Math and Computer Science, Brandon University, Canada
[e] Research Centre for Interneural Computing, China Medical University, Taichung, Taiwan

ARTICLE INFO

Keywords:
Moore machine
Hyperchaotic system
DNA sequence
Encryption
Decryption

ABSTRACT

Nowadays, a large amount of data is transmitted over the Internet and in Cyber–Physical Systems (CPS). A significant portion of this data is highly confidential. Bank account details, credit card details, One Time Passwords (OTP), financial data and other sensitive information are some examples of highly confidential data. A secure encryption and decryption technique is needed to ensure the secure transmission of data. These techniques use a secret key and guarantee that confidential data is only accessible to authorized individuals. This paper proposes a novel encryption process based on Deoxyribonucleic Acid (DNA) cryptography, a hyperchaotic system and a Moore machine. The hyperchaotic system generates four pseudo-random number sequences used in DNA-based operations. The Moore machine performs substitutions in the DNA sequence that makes the system more secure. The proposed technique can protect a system from various attacks, namely man-in-the-middle attacks, ciphertext-only attacks, known-plaintext attacks, brute force attacks and differential cryptanalysis attacks. The proposed scheme gives an average avalanche effect of 54.75%, which guarantees a high level of robustness. Moreover, experimental results show that the proposed scheme is more secure and efficient than the existing schemes.

1. Introduction

Data is considered one of the most valuable resources in this digital age, especially in cyber–physical systems. The CPS consists of a network of hardware and software components that performs well-defined functions. In these systems, a lot of confidential data is stored and transmitted between different components of the network, namely computational devices, communication devices and control systems. Hence, data must be stored and shared securely, so that authorized users can access the data. Data encryption is the most preferred security method in which data is encoded in a format that is unreadable to unauthorized users. In data encryption, a secret key encodes the data, which is known as the encryption key. Here, when a sender has to send any sensitive data to a receiver, it generates an encryption key and encodes the data using that key. The sender uses a secure communication medium to transmit the encoded data to the receiver. Here, the receiver retrieves the secret key, then, performs the decryption process to decode the encrypted data and recovers the original data. A cryptosystem is the collection of encryption, decryption, key generation and key retrieval processes. Thus, highly efficient and secure cryptosystems are required to transmit massive amounts of confidential data among different entities of a computer network.

A DNA-based cryptosystem uses DNA cryptography to perform encryption and decryption processes [1]. Nowadays, DNA cryptography and hyperchaotic systems are commonly used in the design of cryptosystems. In conventional cryptography, data is coded as binary bits, whereas in DNA-based cryptography, data is coded as DNA bases, namely Adenine (A), Guanine (G), Thymine (T) and Cytosine (C) [2]. Here, the DNA bases are coded using 2 bit binary numbers. The data to transmit is converted into a binary string and a DNA base is used to replace each 2 bit of the binary string. As a result, the binary string is converted into a sequence of DNA bases known as DNA sequence. For example, the data (plaintext) "Hello" is replaced by a DNA sequence "ATCGGATCGATACATCGAAC". Thus, the plaintext is converted into a long DNA sequence. Then, the encryption process performs many operations on the DNA sequence to generate the encrypted data (ciphertext).

Hyperchaotic systems generate sequences of pseudo-random numbers, that are normalized and converted to binary strings. Then, the

* Corresponding author at: Dept. of Math and Computer Science, Brandon University, Canada.
E-mail addresses: pramod_p@cusat.ac.in (P. Pavithran), sheenamathew@cusat.ac.in (S. Mathew), suyelnamasudra@gmail.com (S. Namasudra), srivastavag@brandonu.ca (G. Srivastava).

https://doi.org/10.1016/j.comcom.2022.02.008
Received 13 October 2021; Received in revised form 1 February 2022; Accepted 13 February 2022
Available online 18 February 2022

binary strings are converted into DNA sequences and many operations are performed between the DNA sequences and the DNA sequence generated from the plaintext. Randomness is a very important criterion to implement a secure cryptosystem as it is difficult to predict the plaintext when randomness increases [3,4]. Since hyperchaotic systems generate random numbers, the generated DNA sequences are also random. Thus, cryptosystems based on DNA cryptography and hyperchaotic systems are used for the secure transmission of confidential data.

Many novel schemes are proposed to improve the performance of cryptosystems [5]. Thangavel and Varalakshmi [6] have proposed a novel scheme in which many Exclusive OR (EXOR) operations and 16*16 matrix operations are involved to improve data security. Here, the key generation algorithm uses a large prime number. However, the algorithms of this scheme are computationally intensive. Kaundal and Verma [7] have proposed a scheme based on Feistel-inspired structure and an OTP is implemented in this scheme. The main disadvantage of this scheme is that the DNA sequence used for encryption is too short, thus, it faces security issues. Their scheme experience high encryption and decryption times. Majumdar et al. [8] have proposed a scheme that makes extensive use of two-dimensional matrix operations and a complex S-Box design algorithm. The algorithm uses two-dimensional matrix computations that result in time complexity of $O(n^2)$. There are many complex operations in this scheme. Furthermore, it is noted that the times taken for data encryption, data decryption, key generation and key retrieval are too high in all the above-mentioned schemes. In addition, these schemes are not secured against man-in-the-middle attacks, brute force attacks, known-plaintext attacks, ciphertext-only attacks and differential cryptanalysis attacks.

To overcome the limitations of the existing schemes, a novel cryptosystem is proposed in this paper to secure users' confidential data or files. The proposed scheme is based on a hyperchaotic system with four dimensions, DNA cryptography and Moore machine. Here, by using DNA cryptography, DNA coding rules convert the plaintext into a DNA sequence. The four-dimensional hyperchaotic system generates four pseudo-random number sequences that are converted into DNA sequences. Then, the DNA sequences are used to perform many operations with the DNA based plaintext to generate a random DNA sequence. The Moore machine, i.e. a finite state machine is used to perform substitution operations in the DNA sequence. In the proposed scheme, the transition rules of the Moore machine replace a DNA base present in the DNA sequence with a random DNA base. Finally, a random DNA sequence is generated and represents the ciphertext. Here, a key pair generator provides a set of private and public keys to the sender and receiver. The sender uses these keys to encrypt the key values, ciphertext and system parameters, and sends them to the receiver. The main contributions of the paper are summarized as below:

1. A novel encryption technique with a keyspace of 2^{195} is proposed in this paper by using a hyperchaotic system, DNA computing and Moore machine.
2. In this paper, a key-pair generator is utilized for ensuring data security that generates key pairs (public and private keys) for the receiver and sender. The key-pairs facilitate secure transmission of key values, ciphertext and system parameters from the sender to the receiver.
3. Experimental results prove that the proposed scheme is more secure and efficient than the existing schemes.

The remaining part of the paper consists of several sections. Section 2 discusses the related works. The background of the proposed scheme is given in Section 3. Section 4 discusses the proposed scheme in detail and Section 5 analyses the security of the proposed system. This is followed by the performance analysis in Section 6. The conclusion of the proposed scheme and future works are discussed in Section 7.

2. Related works

In the last few years, many cryptographic algorithms based on DNA cryptography and hyperchaotic systems have been developed. In the scheme proposed by Amin et al. [9], DNA computing is used in the encryption algorithm. The plaintext is converted into a DNA sequence using DNA coding rules. In this scheme, search algorithms are used to find multiple locations of DNA quadruplets (four DNA bases) within the DNA sequence. Then, all the values of obtained location are added together to form the ciphertext. Here, an exhaustive search is applied and the algorithm has high time complexity. Yunpeng et al. [10] have proposed a symmetric block cipher that uses blocks of DNA bases. Here, the encryption key is generated using a random DNA sequence. Logistic mapping functions are used in this scheme to generate the key. However, in this scheme, the mathematical operations are computationally complex. Magaria et al. [11] have proposed a deep learning algorithm to ensure the security of Interne of Things (IoT) application. Their scheme gives the concept of industrial IoT and its applications. They have also surveyed many deep learning models, such as deep reinforcement learning, convolutional neural networks mand many more, for industrial IoT in smart cities. The scheme proposed by Atito et al. [12] uses a cryptosystem that implements a data hiding strategy using DNA nucleotides. In the first phase of this scheme, the characters of the plaintext are replaced with a DNA sequence. After this, the DNA sequences are converted into an amino acid-based sequence. In the second phase, a random DNA sequence is generated and is used to hide the amino-acid-based sequence. This cryptosystem is vulnerable to the chosen-plaintext attacks. In [13], Guimaraes et al. have proposed an intelligent model for improving network security. The algorithms used in this model is based on optimum-path forest clustering, and their meta-heuristic optimization technique improves the OPF classifier, which supports to detect anomaly in any wireless sensor network. However, this model takes much time in its execution, as well as faces security issues.

Mandge and Choudhary [14] have proposed a DNA-based encryption technique that uses several matrix manipulations to implement a secure key generation algorithm. This scheme uses many EXOR operations and the time taken to generate the key and retrieve the key is high. Parah et al. [15] have suggested a high payload electronic healthcare record embedding framework for improving security and for verifying the received contents. This framework is mainly based on pixel repetition method, left data mapping, checksum computation and RC4 encryption. In this scheme, a logo is used to embed it into a cover image to detect at the early stage. The novel scheme proposed by Wang and Zhang [16] is based on DNA computing. Here, DNA coding and the RSA algorithm are used to encrypt the message. This technique has many complex operations, and hence, the encryption time and decryption time are not satisfactory.

Aich et al. [17] have introduced a cryptographic model using symmetric keys that is implemented in two phases. At first, an OTP is generated and converted to a binary string. Then, DNA bases are used to replace the binary string. In the second phase, the OTP is used to represent the plaintext as a DNA sequence and various DNA operations are performed to generate the ciphertext. In this scheme, when the length of the plaintext is increased, the computation becomes too complex. A novel scheme has been proposed by Pirbhulal et al. [18] that supports resource allocation by adopting the energy drain, security and cost factors. The experimental results show that this scheme outperforms other existing schemes when considering delay, security, mobility management, and important factors with optimum resource allocation. Kar et al. [19] have proposed a DNA-based round encryption method that consists of two rounds. In the first round, cipher block chaining is used and a binary string is generated from a sequence of random numbers obtained from a random number generator. A collection of DNA sequences is maintained in their scheme. In the second round, a binary string is generated from a DNA sequence that is randomly selected from the collection of DNA sequences. The binary strings generated in both rounds are added and converted into a DNA sequence that represents the ciphertext. This scheme has a high space complexity. In the scheme proposed by Jain et al. [20], a cryptosystem accepts the plaintext and converts it to its binary representation. Then, the resultant binary string is converted into a DNA sequence using a DNA coding rule that is defined in a DNA sequence table. The limitation of this scheme is that the DNA sequence table is always static. Tanaka et al. [21] have utilized DNA computing to design a public key system. Here, a one-way function and simple substitution operations are performed to generate the ciphertext. However, the scheme faces security issues against brute force attack.

3. Background of the proposed scheme

This section gives a brief introduction to hyperchaotic systems, DNA cryptography and Moore machines.

3.1. Hyperchaotic systems

Hyperchaotic systems are non-linear dynamical systems that exhibit random behavior, when subjected to some initial conditions, i.e. the values assigned to the system parameters as discussed below [22]. Small variations in the initial conditions lead to random variations in the output [23]. A four-dimensional hyperchaotic system generates four pseudo-random number sequences [24]. These random numbers are used to enhance the cryptosystem's diffusion property. The diffusion property states that a small change in the plaintext generates a significant change in the ciphertext. Hyperchaotic systems have system parameters and initial conditions that help the cryptosystem to have a large keyspace. In the proposed scheme, the fourth-order Rossler hyperchaotic system generates the random numbers using Eq. (1).

$$\left.\begin{array}{c} y_1 = a\left(y_2 - y_1\right) \\ y_2 = -\left(y_1 * y_3\right) + \left(d * y_1\right) + \left(c * y_2 * y_4\right) \\ y_3 = \left(y_1 * y_2\right) - \left(b * y_3\right) \\ y_4 = y_1 + k \end{array}\right\} \tag{1}$$

Here, the chaotic system has four state variables y_1, y_2, y_3 and y_4, and five system parameters a, b, c, d and k. The system enters a state with maximum randomness, when the values of the system parameters a, b, c and d are 36, 3, 28 and −16, respectively. The parameter k can have a value from [−0.7, 0.7] and the commonly assigned value of k is 0.5. Thus, four sequences of pseudo-random numbers are generated, which are used in various cryptographic operations by recursively substituting the values in Eq. (1).

3.2. DNA cryptography

In DNA cryptography, data is encoded as DNA bases, i.e. G, A, T and C. Let the DNA bases (G, A, T and C) are assigned as 11, 00, 01 and 10, respectively. Then, at first, the binary value of the data is obtained, and each 2 bit in the binary value is replaced with the respective DNA base. Assume 'A' is the data that is changed into DNA bases. The binary value 01000001 that represents 'A' is now converted into the DNA sequence 'TAAT' using DNA codes or bases. Thus, each character in the plaintext is replaced with four DNA bases, and finally, the plaintext is converted to a long DNA sequence. In the proposed scheme, dynamic DNA codes and DNA sequence operations are used and they are discussed in the following subsections.

Table 1
Rules for dynamic DNA coding.

	Rules							
Binary	R0	R1	R2	R3	R4	R5	R6	R7
00	T	A	A	C	T	G	G	C
01	A	T	C	A	G	C	T	G
10	G	C	T	G	A	T	C	A
11	C	G	G	T	C	A	A	T

Table 2
Rules for DNA EXOR.

ADD	G	A	C	T
G	A	G	T	C
A	G	A	C	T
C	T	C	A	G
T	C	T	G	A

Table 3
Rules for DNA addition.

SUB	G	A	C	T
G	C	G	T	A
A	G	A	C	T
C	T	C	A	G
T	A	T	G	C

3.2.1. Dynamic DNA code

To convert binary strings into DNA sequences, dynamic DNA code is used. Consider the DNA bases G, A, C and T that have values 11, 00, 10 and 01, respectively. In this case, the DNA complementary pairs are (A, G) and (T, C). Thus, the four DNA bases that satisfy the base complementary criterion can be assigned to eight different 2 bit values. As result, Table 1 shows 8 possible rules of DNA coding. The first 8 bits of the plaintext are used to select the coding rule from Table 1. The modulus operation is performed between the decimal value of the 8 bits and the decimal value 8. Assume that the first 8 bits of the plaintext are 00001101, which is the decimal value of 13. Then, modulus operation is performed between 13 and 8 to yield the value 5 associated with the rule R5 of Table 1. Thus, rule R5 is applied to convert the 8 bit binary value to the DNA sequence 'GGAC'. This process is repeatedly executed for the remaining bits of the plaintext by converting each set of 8 bit by using the next rule of Table 1 in a cyclic order. So, one of the eight coding rules is randomly selected to convert each character of the plaintext to a DNA sequence consisting of four DNA bases. Therefore, the dynamic DNA code increases the randomness of the encryption process.

3.2.2. DNA sequence operation

As the DNA bases are considered as 2 bit binary representation, the operations are carried out in binary before replacing by their equivalent DNA base. The proposed scheme performs three DNA sequence operations: DNA Exclusive OR (DNA EXOR), DNA subtraction and DNA addition. The rules used to perform these operations are given in Tables 2–4. The following is an example of the sequence operations.

$$\underbrace{\begin{array}{c} DNA\ EXOR \\ TACG \oplus ATCG \end{array}}_{TTAA} \qquad \underbrace{\begin{array}{c} DNA\ addition \\ TACG + ATCG \end{array}}_{TTAC}$$

$$\underbrace{\begin{array}{c} DNA\ subtraction \\ ATCG - TACG \end{array}}_{GTAA}$$

3.3. Moore machine

A finite state machine with six parameters, i.e. namely a set of states (Q) set of input alphabets (Σ), set of output alphabet (O), input

Table 4
Rules for DNA subtraction.

EXOR	G	A	C	T
G	A	G	T	C
A	T	A	C	G
C	G	C	A	T
T	C	T	G	A

transition function (δ), output transition function (f) tt the initial state (q_0), is known as Moore machine [25]. The input transition function, $\delta : Qx\sum \to Q$, states that for each state in Q and each input in Σ, the machine makes a transition to a state in Q. In the output transition function, $f : Q \to O$, each state outputs a value from the set O. The transition functions or rules of the Moore machine are used to convert an input string into a new string. In the proposed scheme, the input and output of the Moore machine are DNA sequences. The Moore machine starts reading the first DNA base from the input, then, moves to a new state and outputs the DNA base associated with that state. The process is repeated until the Moore machine reads all of the DNA bases from the input string. As a result, the Moore machine transforms a set of DNA bases into a new DNA sequence. The design of the Moore machine is described in the following subsections.

3.3.1. Randomly designed moore machine

The main work of this section is to assign values to six parameters of the Moore machine. Since the input and output of the Moore machine are DNA sequences, the parameters Σ and O are sets of DNA bases. Now, because there are four DNA bases and each state produces a different DNA base, the machine has four states: 0, 1, 2 and 3. The outputs A, T, C and G are assigned to the states 0, 1, 2 and 3, respectively. Here, the Moore machine has an initial state, i.e. state 0. Thus, the values assigned to the parameters are summarized as follows:

$$\Sigma = \{G, A, C, T\}$$
$$Q = \{0, 1, 2, 3\}$$
$$O = \{G, A, C, T\}$$
$$q_0 = 0$$
$$f(0) = A,\ f(1) = T,\ f(2) = C,\ f(3) = G$$

Now, the input transition function, i.e. $\delta : Qx\sum \to Q$, is randomly generated and the results are stored in the transition table.

3.3.2. Generation of the transition table

The transition table stores the input and output transition functions. The state obtained from each input transition function is stored in the transition table, i.e. Table 5. At first, the modulus operation between a random number and decimal 4 is performed. This generates a value from 0 to 3 and stores it in the cell corresponding to the state '0' and input 'A'. The remaining cells, one for each state and input, are assigned unique values ranging from 0 to 3. The last column of Table 5 represents the output of each state, i.e. the output transition function. The new state for each transition of the Moore machine is given in Table 5.

3.3.3. Transition diagram

As shown in Fig. 1, the Moore machine's transition diagram is represented by circles connected by directed arrows. The circles represent the Moore machine's states, and the arrows represent the Moore machine's state transitions. Inside the circle, the notation '0/A' (state/output) represents the state '0' and its corresponding output 'A'. Consider a DNA sequence, i.e. ATCG, is given to the Moore machine as an input. Now, the initial state accepts the first input 'A' and makes a transition to state '2' and outputs 'C'. In state '2', the input 'T' results in a transition to state '0' and outputs 'A'. After reading the remaining two inputs, the DNA sequence is generated as CAAT. In this manner, the Moore machine accepts a DNA sequence as an input and outputs a new DNA sequence.

Table 5
Transition table of the Moore machine.

	Transition function for input $\delta : Qx\sum \to Q$				$f : Q \to O$
State	Input (G)	Input (A)	Input (C)	Input (T)	Output
0	1	2	0	3	A
1	0	1	3	2	T
2	2	3	1	0	C
3	3	0	2	1	G

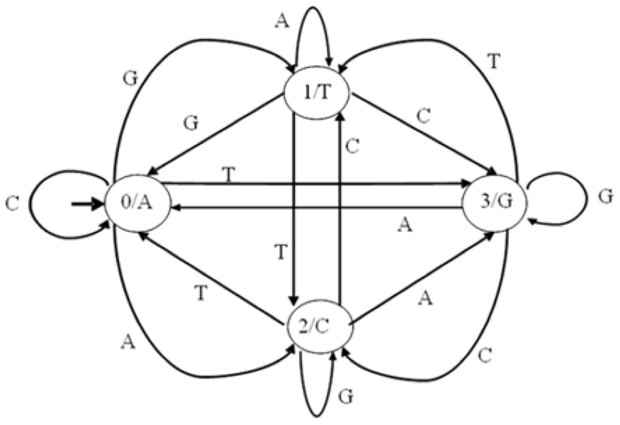


Fig. 1. Design of the Moore machine.

4. Proposed scheme

In the proposed scheme, DNA cryptography uses dynamic DNA codes to enhance randomness in the encryption algorithm. At first, the sender converts the plaintext to a random DNA sequence using DNA code rules, and then, the hyperchaotic system generates four sequences of pseudo-random numbers. The random numbers are converted into DNA sequences and many DNA sequence operations are performed on the DNA-based plaintext to generate a random DNA sequence. Then, a Moore machine is implemented with random characteristics on the DNA bases before the encryption process. Moore machine converts the DNA sequence to a new DNA sequence. Hence, the Moore machine generates a random DNA sequence to transmit each plaintext. Then, many operations are performed on the DNA sequence to generate the ciphertext. Thus, the randomness introduced in the proposed scheme establishes no relationship between the plaintext and ciphertext. All the computations of the proposed scheme consist of simple mathematical operations performed on one-dimensional inputs, i.e. plaintexts, binary strings or DNA sequences. Hence, the time complexity of the algorithms is less in the proposed scheme. Thus, the main motivation of using DNA computing, hyperchaotic system and Moore machine in the proposed scheme is to minimize the time complexity and to increase the randomness of the ciphertext. The system model of the proposed system is shown in Fig. 2, and it consists of the following three entities:

1. **Key-Pair Generator (KPG):** This entity generates key pairs for the data owner and data user, and provides them based on the requests from the data owner and data user.
2. **Data Owner (DO):** The DO accepts data requests from the data user. The DO encrypts the data using the pseudo-random numbers generated by the hyperchaotic system. The DO encrypts the ciphertext and system's parameters using the DO's private key and DU's public key and sends them to the DU.
3. **Data User (DU):** The DU sends a data access request to the DO, and the DO provides the ciphertext to the DU, if the user is authenticated. The DU retrieves the ciphertext and parameters using the DU's private key and DO's public key. To obtain the

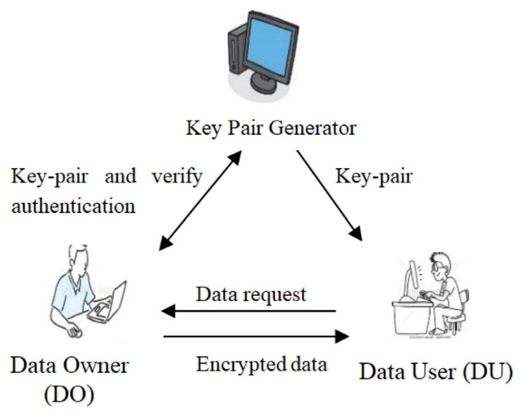


Fig. 2. Proposed scheme system model.

plaintext, the retrieved ciphertext is again decrypted using the key values.

The proposed system is divided into several phases: (i) system setup, (ii) Moore machine implementation, (iii) random sequence generation, (iv) DNA coded Moore machine generation, (v) key generation, (vi) data encryption, (vi) key retrieval, and (vii) data decryption.

4.1. System setup

At first, all the DOs' and DUs' are registered with the KPG. When the DU requires data, it sends a request to the DO, and the DO verifies the user's authenticity from the KPG. If the user is authenticated, the KPG generates key-pair and distributes them to the DO and DU. The key-pair is used for the secure exchange of information between the DO and DU. Secure Socket Layer (SSL) is established between the DO and DU to distribute the key-pairs. SSL is used to ensure that the key pairs are secured from unauthorized users. After this, the DO starts implementing the Moore machine.

4.2. Implementation of the moore machine

To encrypt each plaintext, a novel Moore machine is implemented. The machine is implemented as discussed in Section 3.3. Then, the hyperchaotic system generates four random number sequences.

4.3. Random sequence generation

The four-dimensional hyperchaotic system generates four random number sequences using its state variables y_1, y_2, y_3 and y_4. The sequences are generated by executing the following steps.

Step 1. Input the plaintext and assign random values for state variables y_1, y_2, y_3 and y_4.

Step 2. In the second step, system parameters a, b, c and d are assigned with values 36, 3, 28 and −16, respectively.

Step 3. Convert each character of plaintext to its ASCII value. Let SUM be the sum of ASCII values and*len* be the length of the plaintext.

Step 4. In this step, values of y_1, y_2, y_3 and y_4 are updated using Eq. (2).

Step 5. Here, Eq. (3) is recursively executed len times that starts with the initial values obtained from Eq. (2). Each iteration's new valuesy_1, y_2, y_3 and y_4, are added to the random sequences X_1, X_2, X_3 and X_4, respectively.

$$\left.\begin{aligned} y_1 &= abs\left(y_1 - \frac{(float)(SUM\%100)}{1000}\right)\\ y_2 &= abs\left(y_2 - \frac{(float)(SUM\%100)}{1000}\right)\\ y_3 &= abs\left(y_3 - \frac{(float)(SUM\%100)}{1000}\right)\\ y_4 &= abs\left(y_4 - \frac{(float)(SUM\%100)}{1000}\right)\end{aligned}\right\} \quad (2)$$

$$\left.\begin{gathered} y_1 = a\left(y_2 - y_1\right)\\ y_2 = -(y_1 * y_3) + (d * y_1) + (c * y_2) - y_4\\ y_3 = (y_1 * y_2) - b3\\ y_4 = y_1 + k\\ y_1 = abs(y_1 - round\left(y_1\right))\\ y_2 = abs(y_2 - round\left(y_2\right))\\ y_3 = abs(y_3 - round\left(y_3\right))\\ y_4 = abs(y_4 - round\left(y_4\right))\end{gathered}\right\} \quad (3)$$

Here, the function abs returns the absolute value of the computed result. The sum of ASCII values of characters in the plaintext is stored in 'SUM'. By using Eq. (2), the sum of ASCII values of characters of the plaintext is stored in the variable 'SUM'. Hence, each plaintext has a unique value for 'SUM'. Thus, the initial values assigned to y_1, y_2, y_3 and y_4 depend on the value of 'SUM'. This helps to add more randomness in generating the final key values.

4.4. DNA coded moore machine (DNAMM) generation

In the proposed scheme, the same transition table is used for encryption and decryption. Therefore, the DO sends the transition table to the DU as a 16 base DNA sequence, i.e. DNAMM. To generate DNAMM, there are three steps:

Step 1. In the first step, 16 digits from columns 2 to 5 of the transition table (Table 5) are copied column-wise to form a 16-digit integer number.

Step 2. Then, each digit of the 16-digit number is replaced by its 2 bit binary equivalent.

Step 3. In this step, rule Rx from Table 1 is chosen randomly to convert the binary string obtained in step 2 into a DNA sequence. Thus, the obtained DNA sequence is DNAMM.

The DNAMM generation process is executed as below.

Step 1. Transition table of the Moore machine (Table 5) has 6 columns. The digits present in columns 2 to 5 are combined to form a 16-digit number.

$$\underbrace{2130}_{Column\ 2}\ \underbrace{3201}_{Column\ 3}\ \underbrace{0312}_{Column\ 4}\ \underbrace{1023}_{Column\ 5}$$

Step 2. Each of the 16 digits is replaced by their corresponding 2 bit binary numbers.

$$\underbrace{10011100}_{Column\ 2}\ \underbrace{11100001}_{Column\ 3}\underbrace{00110110}_{Column\ 4}\ \underbrace{01001011}_{Column\ 5}$$

Step 3. Rule R1 is randomly selected from Table 1 and applied on the binary number in step 2. The following DNA sequence is generated that represents DNAMM.

$$\underbrace{CTGA}_{Column\ 2}\ \underbrace{GCAT}_{Column\ 3}\ \underbrace{AGTC}_{Column\ 4}\ \underbrace{TACG}_{Column\ 5}$$

4.5. Key generation

In Section 4.3, it has been discussed that the DO generates random sequences X_1, X_2, X_3 and X_4. The length of the plaintext len is also computed. Here, the random sequences X_1, X_2, X_3 and X_4 are converted into values from $\{0, 1, 2, \ldots, len-1\}$. Then, X_2, X_3 and X_4 are converted into binary strings $binX_2, binX_3$ and $binX_4$, respectively. Now, the current system time (in millisecond) is used to select the DNA Coding Rule (DCR) using Eq. (4). Then, the DCR is assigned a value from 0 to 7 that corresponds to rules R0 to R7 of Table 1. Thus, DCR and Table 1 are used to convert $binX_2$, $binX_3$ and $binX_4$ into DNA sequences $dnaX_2$, $dnaX_3$ and $dnaX_4$, respectively. Here, X_1, $dnaX_2$, $dnaX_3$ and $dnaX_4$ are the key values used in the encryption algorithm, i.e. Algorithm 1.

$$DCR = system\ time\ \%\ 8 \tag{4}$$

4.6. Data encryption

To encrypt the plaintext, the DO employs the key values generated in Section 4.5. Initially, the ASCII values are scrambled using the index values of X_1. The ASCII values and X_1 are then converted into binary strings $binPT$ and $binX_1$, respectively. Now, an EXOR operation is executed between $binPT$ and $binX_1$. Then, DNA code is used to convert the resultant binary string into a DNA sequence, i.e. dna.

The DNA sequence *dna* is now subjected to three DNA operations. At first, DNA addition (Table 3) is performed between $dnaX_2$ and *dna* to generate dna_1. The second operation is DNA subtraction (Table 4), which is performed between dna_1 and $dnaX_3$ to generate dna_2. At last, DNA EXOR (Table 2) operation is performed between $dnaX_4$ and dna_2 to produce dna_3. The DNA sequence dna_3 is given to the Moore machine as an input, which transforms dna_3 to $dna4$, i.e. a new DNA sequence. Finally, if DCR is less than 3, the DNA bases with even index values in the $dna4$ are complemented, i.e. DNA bases ‘G’ and ‘C’ are replaced with ‘A’ and ‘T’, respectively, and vice versa. If DCR is greater than or equal to 3, the DNA bases with odd index values are complemented. As a result, the final ciphertext is the DNA sequence generated after the complement operation. DNAMM, parameter Rx, DCR, SUM, state variables y_1, y_2, y_3 and y_4 and the ciphertext, are encrypted by using the DU’s public key and DO’s private key. The encrypted parameters are then sent to the DU. The encryption processes are represented in Algorithm 2.

Algorithm 1: Key Generation Algorithm

```
   Input: $X_1$, $X_2$, $X_3$, $X_4$ and $len$
   Output: $X_1$, $dnaX_2$, $dnaX_3$ and $dnaX_4$
1  START
2  CONVERT $X_1$, $X_2$, $X_3$, $X_4$ into values from $\{0, 1, 2, \ldots, len-1\}$
3  CONVERT $X_2$, $X_3$ and $X_4$ into binary values $binX_2$, $binX_3$ and
    $binX_4$.
4  GENERATE DCR from the system time
5  PERFORM dynamic DNA coding on $binX_2$ to generate $dnaX_2$
6  PERFORM dynamic DNA coding on $binX_3$ to generate $dnaX_3$
7  PERFORM dynamic DNA coding on $binX_4$ to generate $dnaX_4$
8  STOP
```

4.7. Key retrieval

Here, at first, the DU receives the state variables y_1, y_2, y_3, y_4 and DCR from the DO. The system parameters a, b, c and d are assigned as 36, 3, 28 and −16, respectively. Then, variables y_1, y_2, y_3 and y_4 are updated using Eq. (2). Now, the state variables are then used to generate random sequences X_1, X_2, X_3 and X_4 as discussed in Section 4.3. In the next step, the random numbers X_1, X_2, X_3 and X_4 are converted into values ranging from 0 to $len-1$. Then, X_2, X_3 and X_4 are converted into binary strings $binX_2, binX_3$ and $binX_4$, respectively. After that, using the DCR and Table 1, $binX_2, bin\ X_3$ and $binX_4$ are converted into DNA sequences $dnaX_2$, $dnaX_3$ and $dnaX_4$, respectively. The steps for key retrieval are given in Algorithm 3.

4.8. Data decryption

Following the retrieval of the key values by the DU, the DNAMM is decoded to generate the Moore machine. At first, the DNA bases with even index values in the ciphertext are complemented, if DCR is less than 3, i.e. DNA bases ‘A’ and ‘T’ are replaced with ‘G’ and ‘C’, respectively, and vice versa. If DCR is greater than or equal to 3, the DNA bases with odd index values are complemented. The new DNA sequence is stored as $dna4$. The Moore machine then accepts $dna4$ and outputs dna_3. Then, an EXOR operation is performed between dna_3 and $dnaX_4$ to generate dna_2. Now, DNA ADD operation is performed between dna_2 and $dnaX_3$ to generate dna_1. After that, DNA SUB operation is performed between dna_1 and $dnaX_2$ to generate dna. Then, dna and X_1 are converted to binary strings $dnabin$ and $binX_1$, respectively. In the next step, EXOR operation is performed between $dnabin$ and $binX_1$ to generate $binPT$. After that, the binary string $binPT$ is converted to ASCII values that are scrambled using the index values of X_1. Finally, the ASCII values are converted into their respective plaintext. The entire decryption process is shown in Algorithm 4.

Algorithm 2: Encryption Algorithm

```
    Input: ASCII values, $X_1$, $dnaX_2$, $dnaX_3$ and $dnaX_4$
    Output: Ciphertext ($dna4$)
1   START
2   SCRAMBLE ASCII values of plaintext using index values of $X_1$
3   REPLACE ASCII values with their binary values to generate
     string $binPT$
4   CONVERT $X_1$ into its equivalent binary string to generate
     $binX_1$
5   EXECUTE EXOR between $binPT$ and $binX_1$
6   CONVERT the binary string into DNA sequence $dna$
7   EXECUTE DNA ADD between $dna$ and $dnaX_2$ to generate $dna_1$
8   EXECUTE DNA SUB between $dna_1$ and $dnaX_3$ to generate $dna_2$
9   EXECUTE DNA EXOR between $dna_2$ and $dnaX_4$ to generate
     $dna_3$
10  INPUT $dna_3$ to Moore machine
11  SAVE output of Moore machine in $dna4$
12  IF DCR $<$ 3
13       COMPLEMENT DNA bases in $dna4$ having an even index
14  IF DCR $\geq$ 3
15       COMPLEMENT DNA bases in $dna4$ having an odd index
16  STOP
```

Algorithm 3: Key Retrieval Algorithm

```
   Input: $y_1$, $y_2$, $y_3$, $y_4$ and DCR
   Output: $X_1$, $dnaX_2$, $dnaX_3$ and $dnaX_4$
1  START
2  GENERATE four random sequences $X_1$, $X_2$, $X_3$ and $X_4$
3  CONVERT $X_1, X_2$, $X_3$ and $X_4$ into values ranging from
    $\{0, 1, 2, \ldots, len-1\}$
4  CONVERT $X_2$, $X_3$ and $X_4$ into binary values $binX_2$, $binX_3$ and
    $binX_4$
5  PERFORM dynamic DNA coding on $binX_2$ to generate $dnaX_2$
6  PERFORM dynamic DNA coding on $binX_3$ to generate $dnaX_3$
7  PERFORM dynamic DNA coding on $binX_4$ to generate $dnaX_4$
8  STOP
```

5. Security analysis

This section presents the security analysis of the proposed scheme.

5.1. Keyspace analysis

The keyspace used in the encryption algorithm determines the security of a cryptosystem. The keyspace is the set of all permutations of characters in the key. The exhaustive attack can be defeated by a cryptosystem with a keyspace greater than 2^{100}. In the proposed scheme, the keyspace is calculated as shown below:

$$\underbrace{\begin{matrix} System\ parameters \\ (y_1, y_2, y_3, y_4) \end{matrix}}_{2^{32} X_2^{32} X_2^{32} X_2^{32}} \qquad \underbrace{\begin{matrix} DNA\ sequence \\ (DNAMM) \end{matrix}}_{2^{32}}$$

$$\underbrace{\begin{matrix} Sum\ of\ plaintext \\ ASCII\ values\ (SUM) \end{matrix}} \quad 2^{32} \quad \underbrace{\begin{matrix} DNA\ coding\ rule \\ (DCR) \end{matrix}}_{2^{3}}$$

Thus, the proposed scheme has a keyspace of 2^{195}, which indicates high security against exhaustive attacks.

Algorithm 4: Decryption Algorithm

```
   Input: Ciphertext, DCR, $X_1$, $dnaX_2$, $dnaX_3$ and $dnaX_4$
   Output: Plaintext
1  START
2  DECODE DNAMM to generate Moore machine
3  IF DCR < 3
4  COMPLEMENT the DNA bases of ciphertext with even index
5  IF DCR ≥ 3
6  COMPLEMENT the DNA bases of ciphertext with odd index
7  SAVE complemented ciphertext as $dna4$
8  INPUT $dna4$ to the Moore machine
9  SAVE output of the Moore machine as $dna_3$
10 EXECUTE DNA EXOR between $dna_3$ and $dnaX_4$ to generate
     $dna_2$
11 EXECUTE DNA ADD between $dna_2$ and $dnaX_3$ to generate $dna_1$
12 EXECUTE DNA SUB between $dna_1$ and $dnaX_2$ to generate $dna$
13 CONVERT $dna$ and $X_1$ into binary strings $dnabin$ and $binX_1$
14 EXECUTE EXOR between $dnabin$ and $binX_1$ to generate $binPT$
15 CONVERT $binPT$ into its equivalent ASCII values
16 SCRAMBLE ASCII values using the index values of $X_1$
17 CONVERT ASCII values into plaintext
18 STOP
```

5.2. Brute-force attacks

Here, the attacker attempts to decrypt the ciphertext using every possible combination of the secret key. It is considered one of the important features of an efficient encryption algorithm. A large keyspace helps a cryptosystem to prevent brute force attacks [26]. To prevent the brute force attack, a keyspace greater than 2^{100} is recommended. As the keyspace of the proposed system is 2^{195}, this attack can be easily resisted.

5.3. Known plaintext attacks

The attacker gains access to both the plaintext and ciphertext in the known-plaintext attack [27]. Here, the attacker attempts to decrypt subsequent messages by analyzing the relationship between the ciphertext and its corresponding plaintext. The Moore machine is designed for supporting randomness in the encryption process of the proposed scheme. After several transitions, the Moore machine converts the DNA sequence generated from the plaintext into a new DNA sequence. Then, several operations convert the new DNA sequence into the ciphertext. As the result, the ciphertexts generated for identical or similar plaintexts are unique. Hence, any relationship between the plaintext and ciphertext cannot be established. Therefore, the proposed scheme can easily resist the known-plaintext attack.

5.4. Ciphertext only attacks

The attacker gains access to many encrypted messages in the ciphertext-only attack [28]. In this attack, the attacker tries to deduce plaintexts as much as possible or to obtain the decryption key. Any ciphertext can be easily decrypted once the decryption key is hacked. During each encryption–decryption process, a new Moore machine is used in the proposed scheme, which is randomly generated. Hence, when two similar ciphertexts in terms of DNA sequences are passed as inputs to the Moore machine, it generates DNA sequences that have no similarity. Hence, there is very little correlation between plaintexts generated from similar ciphertexts. So, the attacker is unable to establish any relationship between ciphertexts and their corresponding plaintexts. Thus, the ciphertext only attack fails in the proposed scheme.

5.5. Man-in-the-middle attack

The man-in-the-middle attack is considered an eavesdropping attack [29]. Here, the attacker intercepts the communication channel between two parties to get confidential data, modify data, corrupt data, or even disrupt the communication between them. Here, the DO encrypts the state variables y_1, y_2, y_3, y_4, DCR and other system parameters using the DU's public key and the DO's private key prior to sending them to the corresponding DU. The DU retrieves the encrypted variables and parameters by using the public key of the DO and the private key of the DU. The proposed scheme can resist this attack because the private keys are known only to the authorized entities.

5.6. Differential cryptanalysis attacks

In a differential cryptanalysis attack, the attacker examines pairs of plaintexts that differ in some context [30]. The corresponding ciphertexts are then examined to verify whether there are any common differences. If some common relationships between the pairs of plaintexts and pairs of ciphertexts are discovered, the attacker can decipher the secret key. To encrypt each plaintext in the proposed scheme, the hyperchaotic system generates four new sequences of random numbers. In the encryption algorithm, these random sequences are used in four different DNA operations. Here, no common differences are found between the pairs of plaintexts and pairs of ciphertexts. Therefore, the proposed model resists this differential cryptographic attack.

5.7. Avalanche effect

The Avalanche Effect (AE) is considered a critical security measure for analyzing a cryptosystem [31]. Here, 1 bit change in the plaintext outcomes major changes in the ciphertext. A cryptosystem has a good avalanche effect, if 50% of the bits of the ciphertext are modified. Thus, a good AE ensures that an attacker is unable to predict the plaintext from a given ciphertext. Eq. (5) is used to calculate the avalanche effect.

$$AE = \frac{Number\ of\ bits\ changed\ in\ ciphertext}{Number\ of\ bits\ in\ ciphertext} \times 100 \tag{5}$$

To analyze the AE, fifty experiments are performed varying the length of the plaintext, and the average values are considered, which are given in Table 6. The average AE of all sets of data is computed as 54.75%, and it shows that the proposed scheme is highly secured from an external attack.

Table 6
Analysis of avalanche effect.

Plaintext length (char)	Change of bits in the ciphertext
10	51%
20	54%
40	59%
80	55%

6. Performance analysis

The section deals with the experimental setup and performance evaluation of the proposed technique.

6.1. Experimental setup

The proposed system is implemented on an Intel Core i5 desktop with Ubuntu 20.04 operating system and 4 GB RAM and. All algorithms are implemented using the Java programming language on the Eclipse IDE. The 20 Newsgroup dataset is used for evaluation purposes.[1]

6.2. Results and discussion

The experimental results of the proposed scheme are compared to the three state-of-the-art techniques proposed by Thangavel and Varalakshmi [6], Kaundal and Verma [7], and Majumdar et al. [8]. For evaluating the performance of the proposed scheme, the times for encryption and decryption are calculated varying the data length. For each set of data, 50 experiments are performed and the average values are computed. The results are compared with the existing schemes [6–8] and are tabulated in Table 7. The results shown in Figs. 3 and 4 imply that the proposed scheme takes less times for encryption and decryption, when compared to the existing schemes. The existing schemes use complex operations based on large prime numbers and matrices, whereas the proposed scheme uses simple operations, such as multiplication, addition and substitution. Here, in the encryption and decryption algorithms, all computations are performed on one-dimensional vectors.

The inputs to the encryption algorithm consist of one-dimensional numerical values and DNA sequences. Simple DNA-based operations, such as DNA ADD, DNA SUB and DNA EXOR, are performed on the DNA sequences and their time complexity is $\mathcal{O}(n)$. In the Moore machine, simple substitutions of DNA bases are performed, and hence, the time complexity of the Moore machine implementation is also $\mathcal{O}(n)$. Thus, the overall time complexity of the encryption algorithm is $\mathcal{O}(n)$. The decryption algorithm also performs the similar computation in the reverse order, so it has the same time complexity. Thus, the time complexity of the encryption and decryption algorithms is $\mathcal{O}(n)$.

The throughput of the encryption process depends on the encryption time and size of the plaintext. The throughput of all the existing schemes is computed, which is given in Table 8. As shown in Fig. 5, the encryption process of the proposed scheme has higher throughput than the encryption processes of the existing schemes. As the time taken for encryption in the existing schemes is more than the time taken in the proposed scheme, the throughput of the existing schemes is less than that of the proposed scheme. Eq. (6) is used to calculate the throughput of the encryption process.

$$Throughput = (size\ of\ \ plaintext)/(encryption\ time) \tag{6}$$

The times taken for key generation and key retrieval are computed for the proposed scheme, as well as other existing schemes [6–8]. For each dataset, 50 experiments are performed, and the results are shown

[1] http://qwone.com/~jason/20Newsgroups

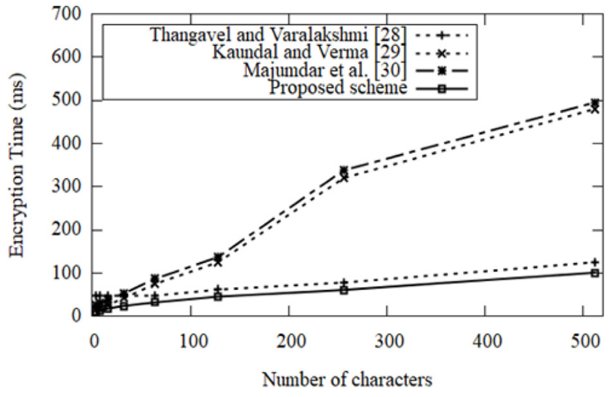


Fig. 3. Encryption time (ms).

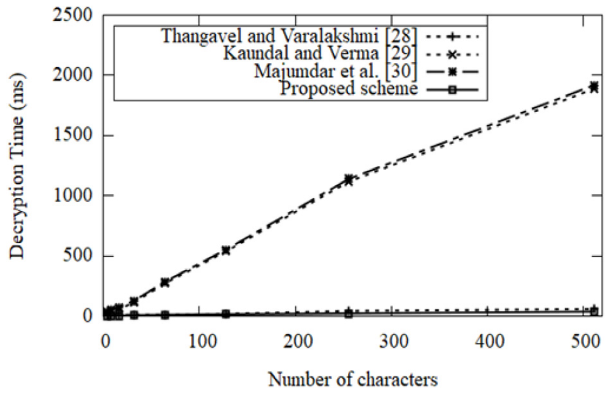


Fig. 4. Decryption time (ms).

in Table 9. The key generation process of Thangavel and Varalakshmi's [6] scheme is dependent on a large prime number, computation of its primitive roots and many modular inverse operations. In the approach of Kaundal and Verma [7], the plaintext is first converted into a binary string. Then, each bit of the binary string is replaced by five DNA bases, and thus, the key size is very huge. Hence, the key generation time is high. In this scheme, a random key sequence that is based on OTP is generated, which utilizes a pseudo-random generator and a 32-byte DNA sequence as the seed value. The algorithms used in the scheme proposed by Majumdar et al. [8] use many modulus operations and matrix multiplication operations, and hence, the time complexity is high.

The proposed scheme generates key values using four random numbers, and the process consists of simple multiplication and addition operations. Then, the system time is used to calculate the DCR and it is used to generate the DNA sequence. The same procedure is followed for the key retrieval process with the same DCR used during key generation. As DCR is not calculated in the key retrieval algorithm, the key retrieval time is slightly less than the key generation time. In the proposed scheme, the time taken for key generation and key retrieval is less compared to the existing schemes [8–10]. The experimental results are graphically shown in Figs. 6 and 7.

In the next experiment, different ciphertexts are generated for identical or similar plaintexts. As random sequences are generated based on the system time, two similar plaintexts are encrypted using two different random number sequences. The correlation coefficient between the two ciphertexts indicates the strength of their linear relationship. If the coefficient value is 0, there is no relationship between the two

Table 7
Encryption and decryption times.

	Ref. [6]		Ref. [7]		Ref. [8]		Proposed scheme	
Plaintext length (char)	Encryption time (ms)	Decryption time (ms)	Encryption time (ms)	Decryption time (ms)	Encryption time (ms)	Decryption time (ms)	Encryption time (ms)	Decryption time (ms)
4	47	15	16	28	26	41	10.21	5.24
8	47	15	18	44	30	56	12.56	6.37
16	47	15	26	62	38	75	18.10	8.10
32	47	15	42	118	53	132	23.94	10.31
64	47	15	75	275	87	289	32.14	13.21
128	62	21	125	542	137	556	45.71	17.62
256	78	46	320	1120	338	1142	61.24	25.11
512	125	63	480	1890	495	1920	101.21	39.82

Table 8
Throughput comparisons.

Plaintext length (char)	Ref. [6] Throughput (bits/ms)	Ref. [7] Throughput (bits/ms)	Ref. [8] Throughput (bits/ms)	Proposed scheme Throughput (bits/ms)
4	0.09	0.25	0.15	0.39
8	0.17	0.44	0.27	0.64
16	0.34	0.62	0.42	0.88
32	0.68	0.76	0.60	1.34
64	1.36	0.85	0.74	1.99
128	2.06	1.02	0.93	2.80
256	3.28	0.80	0.76	4.18
512	4.10	0.25	1.03	5.06

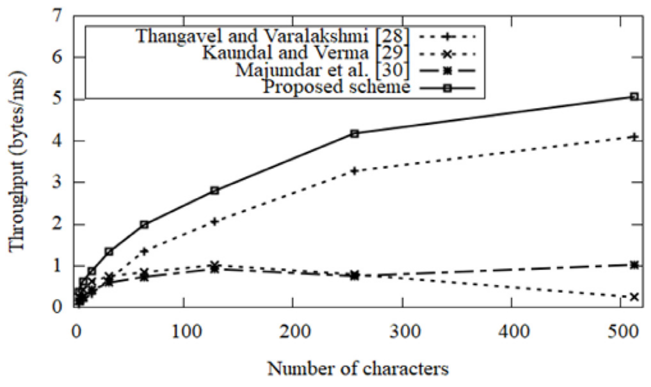


Fig. 5. Throughput (bytes/ms).

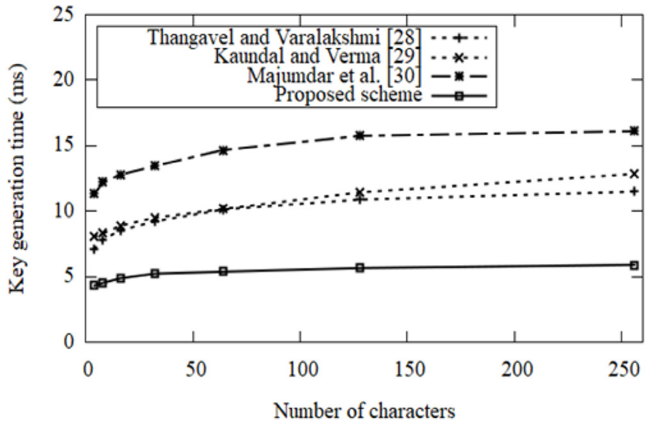


Fig. 6. Key generation time (ms).

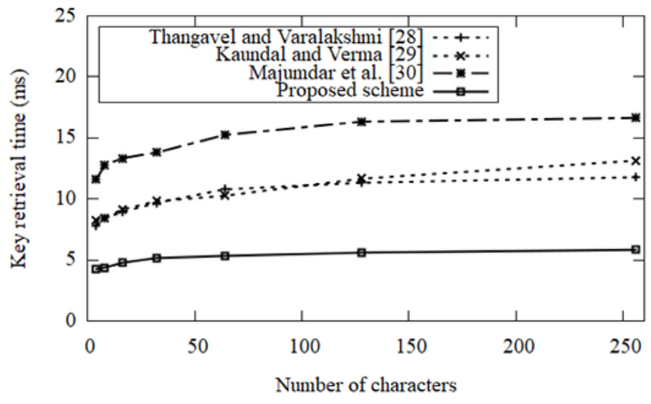


Fig. 7. Key retrieval time (ms).

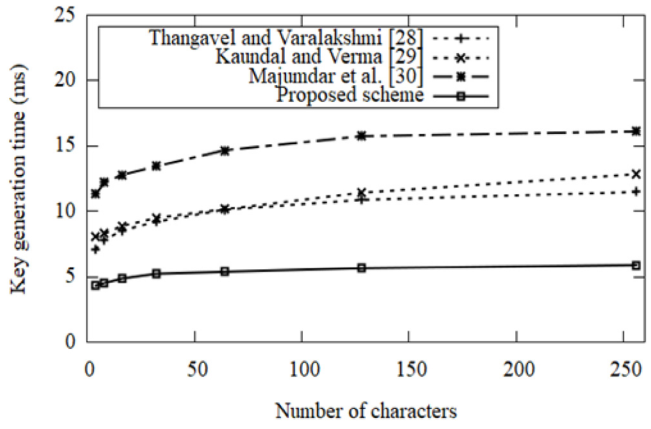


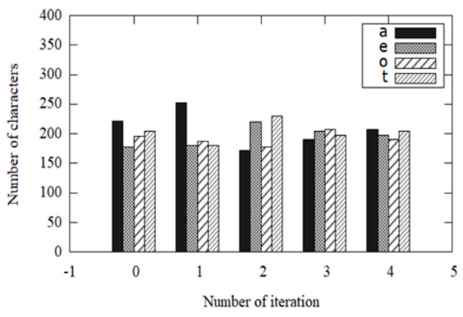
Fig. 8. Correlation coefficient of ciphertext versus plaintext.

ciphertexts. A negative value towards −1 shows a negative relationship, whereas, a positive value towards +1 shows a positive relationship. The correlation coefficient is calculated for the existing schemes and the proposed scheme. Table 10 displays the results for various plaintexts. Fig. 8 shows the correlation coefficients of the proposed scheme, which are less than the coefficients of the existing schemes. Hence, there are many differences between the ciphertexts generated from similar plaintexts in the proposed technique. The proposed scheme is secured because no relationship between plaintext and ciphertext is established.

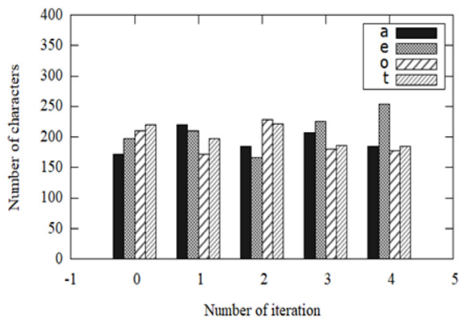
Finally, the ciphertexts generated from four different plaintexts are subjected to frequency analysis. In the experiment, each plaintext contains 200 unique characters from the set $\{a, e, o, t\}$. This set represents

Table 9
Key generation and key retrieval times.

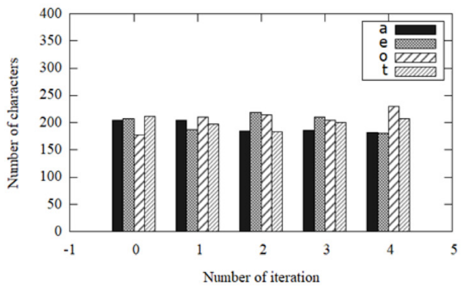
	Ref. [6]		Ref. [7]		Ref. [8]		Proposed scheme	
Plaintext length (char)	Key generation time (ms)	Key retrieval time (ms)	Key generation time (ms)	Key retrieval time (ms)	Key generation time (ms)	Key retrieval time (ms)	Key generation time (ms)	Key retrieval time (ms)
4	7.15	7.87	8.05	8.24	11.35	11.65	4.35	4.28
8	7.85	8.45	8.32	8.45	12.22	12.78	4.55	4.40
16	8.55	8.95	8.90	9.15	12.80	13.35	4.90	4.80
32	9.23	9.69	9.52	9.83	13.45	13.82	5.25	5.18
64	10.15	10.80	10.18	10.28	14.68	15.25	5.40	5.35
128	10.90	11.34	11.45	11.68	15.76	16.35	5.68	5.62
256	11.50	11.80	12.86	13.15	16.12	16.65	5.92	5.85



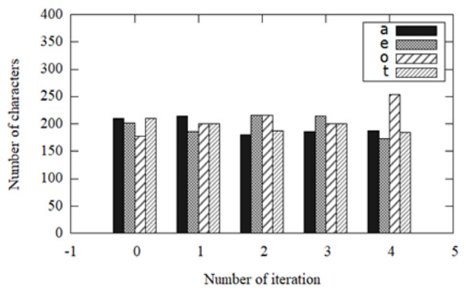
(a) Plaintext1 with 200 A's



(b) Plaintext2 with 200 E's



(c) Plaintext3 with 200 O's



(d) Plaintext4 with 100 T's

Fig. 9. The number of DNA bases in the ciphertext.

Table 10
Correlation coefficient analysis.

Plaintext length (char)	Ref. [6] Correlation coefficient	Ref. [7] Correlation coefficient	Ref. [8] Correlation coefficient	Proposed scheme Correlation coefficient
4	0.412	0.530	0.510	0.312
8	0.350	0.491	0.482	0.256
16	0.275	0.410	0.396	0.194
32	0.195	0.327	0.275	0.118
64	0.108	0.240	0.192	0.075
128	0.090	0.110	0.084	0.042
256	0.015	0.036	0.009	0.001

Table 11
Frequency analysis of ciphertext for Plaintext1.

Input: Plaintext of 200 a's				
	DNA base count			
Iteration	G	A	C	T
1	220	172	210	198
2	198	220	172	210
3	222	184	228	166
4	186	208	180	226
5	184	184	178	254

four most commonly used characters in the English language. Then, all the ciphertexts are analyzed and the number of occurrences of each DNA base is computed. The results are given in Tables 11 to 14. The histograms given in Figs. 9(a) to 9(d) indicate that all four ciphertexts have a nearly uniform distribution of DNA bases. Consider the ciphertext consists of 200 characters generated from any of the four plaintexts is given below.
"ATTCGTAGCGTAGATCGGACTAGTATTATTACTCAATGCATAGTAT-GCCAGATCAGATCTAGGACATAGACTTGGCAAGTCAGAAGGTCACT-ATAGCATGCCAATGAACATAGAGACATTAGACAGATAGACCTTAACA-CAGGTCACACATGGAAACAAACAAATAGACAGATTTAATGGCAGAT-GGCATAGA" (200 char)

$\underbrace{aaaa\ldots aaa}_{Plaintext\ 1}$

$\underbrace{eeee\ldots eee}$ $Plaintext\ 2$ $\underbrace{oooo\ldots ooo}$ $Plaintext\ 3$ $\underbrace{tttt\ldots ttt}_{Plaintext\ 4}$

Table 12
Frequency analysis of ciphertext for Plaintext2.

Input: Plaintext of 200 e's				
	DNA base count			
Iteration	G	A	C	T
1	204	222	196	178
2	180	252	188	180
3	230	172	178	220
4	198	190	208	204
5	204	208	190	198

Table 13
Frequency analysis of ciphertext for Plaintext3.

Input: Plaintext of 200 o's				
	DNA base count			
Iteration	G	A	C	T
1	210	210	178	202
2	200	214	200	186
3	188	180	216	216
4	200	186	200	214
5	184	188	254	174

Table 14
Frequency analysis of ciphertext for Plaintext4.

Input: Plaintext of 200 t's				
	DNA base count			
Iteration	G	A	C	T
1	212	205	178	207
2	198	204	210	188
3	183	185	214	218
4	200	186	204	210
5	208	182	230	180

It is shown that any relationship between the ciphertexts and plaintexts is not established. Thus, the proposed scheme is more secure and efficient.

7. Conclusions and future work

In this paper, a novel DNA-based cryptosystem is proposed by combining DNA computing, a Moore machine, and a hyperchaotic system. The hyperchaotic system generates four pseudo-random number sequences that are used as key values in DNA-based operations. In the proposed scheme, a Moore machine is generated to randomly perform substitutions of DNA bases. The proposed scheme encrypts the plaintext with a keyspace of 2^{195}. Here, the times for encryption, decryption, key generation and key retrieval are lower, respectively. The proposed scheme is secured against many attacks, namely man-in-the-middle attacks, ciphertext only attacks, known-plaintext attacks, brute force attacks, and differential cryptanalysis attacks. Experimental analysis is performed to evaluate the performance of the proposed scheme. The results obtained for throughput, avalanche effect and frequency analysis of DNA bases in the ciphertext, prove that the proposed scheme outperforms the existing schemes. The proposed scheme uses one-dimensional inputs, i.e. the plaintext, basic arithmetic and logical operations, and hence, the time complexity of the encryption and decryption algorithm is $\mathcal{O}(n)$, respectively. Moreover, the correlation coefficient between ciphertexts generated for the same plaintext is almost zero. Thus, a secure and efficient cryptosystem is developed by using the proposed technique. The proposed system can be implemented in industry, such as finance and healthcare, where sensitive data is protected by several regulations. In the future, the proposed scheme can be extended by proposing a novel authentication technique.

CRediT authorship contribution statement

Pramod Pavithran: Methodology, Experimentation, Data curation, Writing the paper. **Sheena Mathew:** Writing – original draft, Formal analysis, Investigation. **Suyel Namasudra:** Writing – original draft, Visualization, Resources. **Gautam Srivastava:** Draft review, Editing, Supervision, Proofreading.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] S. Basu, M. Karuppiah, M. Nasipuri, A.K. Halder, N. Radhakrishnan, Bio-inspired cryptosystem with DNA cryptography and neural networks, J. Syst. Archit. 94 (2019) 24–31.

[2] S. Namasudra, R. Chakraborty, A. Majumder, N.R. Moparthi, Securing multimedia by using DNA-based encryption in the cloud computing environment, ACM Trans. Multimed. Comput. Commun. Appl. 16 (3s) (2020) 1–19.

[3] S. Ndichu, S. Kim, S. Ozawa, Deobfuscation, unpacking, and decoding of obfuscated malicious javascript for machine learning models detection performance improvement, CAAI Trans. Intell. Technol. 5 (3) (2020) 184–192.

[4] S. Namasudra, P. Roy, P. Vijayakumar, S. Audithan, B. Balusamy, Time efficient secure DNA based access control model for cloud computing environment, Future Gener. Comput. Syst. 73 (2017) 90–105.

[5] R.M. Alguliyev, R.M. Aliguliyev, L.V. Sukhostat, Efficient algorithm for big data clustering on single machine, CAAI Trans. Intell. Technol. 5 (1) (2019) 9–14.

[6] M. Thangavel, P. Varalakshmi, Enhanced DNA and elgamal cryptosystem for secure data storage and retrieval in cloud, Cluster Comput. 21 (2) (2018) 1411–1437.

[7] A.K. Kaundal, A. Verma, Extending feistel structure to DNA cryptography, J. Discrete Math. Sci. Cryptogr. 18 (4) (2015) 349–362.

[8] A. Majumdar, A. Biswas, A. Majumder, S.K. Sood, K.L. Baishnab, A novel DNA-inspired encryption strategy for concealing cloud storage, Front. Comput. Sci. 15 (3) (2021) 1–18.

[9] S.T. Amin, M. Saeb, S. El-Gindi, A DNA-based implementation of YAEA encryption algorithm, in: Computational Intelligence, 2006, pp. 120–125.

[10] Z. Yunpeng, Z. Yu, W. Zhong, R.O. Sinnott, Index-based symmetric DNA encryption algorithm, in: 2011 4th International Congress on Image and Signal Processing, vol. 5, IEEE, 2011, pp. 2290–2294.

[11] N. Magaia, R. Fonseca, K. Muhammad, A.H.F.N. Segundo, A.V.L. Neto, V.H.C. de Albuquerque, Industrial internet-of-things security enhanced with deep learning approaches for smart cities, IEEE Internet Things J. 8 (8) (2020) 6393–6405.

[12] A. Atito, A. Khalifa, S. Rida, DNA-based data encryption and hiding using playfair and insertion techniques, J. Commun. Comput. Eng. 2 (3) (2012) 44.

[13] R.R. Guimaraes, L.A. Passos, R. Holanda Filho, V.H.C. de Albuquerque, J.J. Rodrigues, M.M. Komarov, J.P. Papa, Intelligent network security monitoring based on optimum-path forest clustering, IEEE Netw. 33 (2) (2018) 126–131.

[14] T. Mandge, V. Choudhary, A DNA encryption technique based on matrix manipulation and secure key generation scheme, in: 2013 International Conference on Information Communication and Embedded Systems, ICICES, IEEE, 2013, pp. 47–52.

[15] S.A. Parah, J.A. Kaw, P. Bellavista, N.A. Loan, G. Bhat, K. Muhammad, A. Victor, Efficient security and authentication for edge-based internet of medical things, IEEE Internet Things J. (2020).

[16] X. Wang, Q. Zhang, DNA computing-based cryptography, in: 2009 Fourth International on Conference on Bio-Inspired Computing, IEEE, 2009, pp. 1–3.

[17] A. Aich, A. Sen, S.R. Dash, S. Dehuri, A symmetric key cryptosystem using DNA sequence with OTP key, in: Information Systems Design and Intelligent Applications, Springer, 2015, pp. 207–215.

[18] S. Pirbhulal, W. Wu, K. Muhammad, I. Mehmood, G. Li, V.H.C. de Albuquerque, Mobility enabled security for optimizing IoT based intelligent applications, IEEE Netw. 34 (2) (2020) 72–77.

[19] N. Kar, A. Majumder, A. Saha, S. Deb, M.C. Pal, Data security and cryptography based on DNA sequencing, Int. J. Inf. Technol. Comput. Sci. 10 (3) (2013).

[20] S. Jain, V. Bhatnagar, A novel DNA sequence dictionary method for securing data in DNA using spiral approach and framework of DNA cryptography, in: 2014 International Conference on Advances in Engineering & Technology Research, ICAETR-2014, IEEE, 2014, pp. 1–5.

[21] K. Tanaka, A. Okamoto, I. Saito, Public-key system using DNA as a one-way function for key distribution, Biosystems 81 (1) (2005) 25–29.

[22] X. Xue, Q. Zhang, X. Wei, L. Guo, Q. Wang, An image fusion encryption algorithm based on DNA sequence and multi-chaotic maps, J. Comput. Theor. Nanosci. 7 (2) (2010) 397–403.

[23] Q. Zhang, L. Guo, X. Wei, Image encryption using DNA addition combining with chaotic maps, Math. Comput. Modelling 52 (11–12) (2010) 2028–2035.

[24] X. Wei, L. Guo, Q. Zhang, J. Zhang, S. Lian, A novel color image encryption algorithm based on DNA sequence operation and hyper-chaotic system, J. Syst. Softw. 85 (2) (2012) 290–299.

[25] J.E. Hopcroft, R. Motwani, J.D. Ullman, Introduction to automata theory, languages, and computation, ACM SIGACT News 32 (1) (2001) 60–65.

[26] S. Salamatian, W. Huleihel, A. Beirami, A. Cohen, M. Médard, Why botnets work: Distributed brute-force attacks need no synchronization, IEEE Trans. Inf. Forensics Secur. 14 (9) (2019) 2288–2299.

[27] Y. Yuan, Y. Mo, Security for cyber-physical systems: Secure control against known-plaintext attack, Sci. China Technol. Sci. 63 (9) (2020) 1637–1646.

[28] X. Chang, A. Yan, H. Zhang, Ciphertext-only attack on optical scanning cryptography, Opt. Lasers Eng. 126 (2020) 105901.

[29] K. Li, W. Zhang, C. Yang, N. Yu, Security analysis on one-to-many order preserving encryption-based cloud data search, IEEE Trans. Inf. Forensics Secur. 10 (9) (2015) 1918–1926.

[30] H. Zhao, G. Han, L. Wang, W. Wang, MILP-based differential cryptanalysis on round-reduced midori64, IEEE Access 8 (2020) 95888–95896.

[31] M. Baykara, Z.Z. Gürel, Detection of phishing attacks, in: 2018 6th International Symposium on Digital Forensic and Security, ISDFS, IEEE, 2018, pp. 1–5.