Journal of Theoretical and Applied Information Technology
15th March 2020. Vol.98. No 05

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

COMPUTATIONAL ANALYSIS OF DNA SEQUENCES BASED UPON AN INNOVATIVE MATHEMATICAL HYBRIDIZATION MECHANISM OF PROBABILISTIC CELLULAR AUTOMATA AND PARTICLE SWARM OPTIMIZATION

WESAM M. ELSAYED[1], MOHAMMED ELMOGY[2], and B.S. EL-DESOUKY[3]

[1]Mathematics Dept., Faculty of Science, Mansoura University, Mansoura, 35516, Egypt.
[2]Information Technology Dept., Faculty of Computers and Information, Mansoura University, Mansoura, 35516, Egypt
[3]Mathematics Dept., Mathematical Statistics & Combinatorics, Faculty of Science, Mansoura University, Mansoura, 35516, Egypt
E-mail: [1]W_mahmoud93@yahoo.com, [2]melmogy@mans.edu.eg, and [3]b_desouky@mans.edu.eg

ABSTRACT

The deoxyribonucleic acid (DNA) sequence reconstruction problem is a very complex issue of computational biology. In this paper, we introduce a modified procedure for the reconstruction process based on probabilistic cellular automata (PCA) integrated with a particle swarm optimization (PSO) algorithm. PSO is utilized to detect the optimal and adequate transition rules of cellular automata (CA) for the reconstruction process. This integration makes our algorithm more efficient. The evolution of organisms occurs due to mutations of DNA sequences. As a result, we attempt to model the evolutions of DNA sequences using our proposed system. In Particular, we determine the impact of neighboring DNA base pairs on the mutation process. We used CA rules for analysis and prediction of the DNA sequence. Our innovative model leans on the hypothesis that mutations are probabilistic events. As a result, their evolution can be simulated as a PCA model, and this enables us to discover the effects of some neighborhood base-pairs on a DNA segment evolution. The main target of this paper is to analyze various DNA sequences and try to predict the changes that occur in DNA sequences during evolution (mutations). We use a similarity score as our measure of fitness to detect symmetry relations, which in turn makes our method appropriate for comparison of numerous extremely long sequences. Phylogenetic trees are exhibited in order to view our investigated samples. Unlike using Markov chains, our proposed technique does not reveal biases in mutation rates that depend on the neighboring bases, which indicates the effect of neighbors on mutations. Incorporating probabilistic components in our proposed technique helps to produce a tool capable of foretelling the likelihood of specific mutations.

Keywords: *DNA Sequence Reconstruction, Computational Biology, Mutation Rates, Probabilistic Cellular Automata (PCA), Particle Swarm Optimization (PSO).*

1. INTRODUCTION

Each living cell of every human being contains deoxyribonucleic acid (DNA). Each molecule of human DNA holds the text of three billion characters that formulate the whole set of commands for outgrowth and conservation of the body. A considerable scheme of molecular biology is to dictate the sequence of characters that compose this text. Therefore, the task of identifying a DNA sequence has seen significant technological progress over the last decades.

The work in this paper is an extension of what we had done previously [1], where we used our suggested probabilistic rules to study DNA evolution. Because cellular automata (CA) looked to be a promising tool in modeling DNA, we use these probabilistic rules again to reconstruct DNA sequences. Our work aims to detect the rules for neighborhoods, considering the effect of mutations that occurred throughout the evolution of the sequences. Besides, this study attempts to get rid of the uncertainties in the intermediate sequences. This is done by introducing stochastic elements into our

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

modified algorithm, which leads us to a more qualified simulation.

Work on the reconstruction problem began in the late 1960s. Software for reconstructing sequences emerged in the late 1970s. Software for reconstructing sequences came into sight in the late 1970s. Gingeras et al. [2] determined the status of the overlapping between fragments and how it is oriented. An error was accommodated by estimating a heuristic alignment for each overlap. The overall target was to compare all fragments and compute an alignment for every pair. The authors picked out an overlap, and the two fragments were incorporated and substituted with a continuous sequence called a contig. This procedure was repeated until one contig remained, or no overlaps could be detected.

Peltola et al. [3] described the first program to control this process. Their proposed technique was implemented in three stages. The premier stage calculated overlaps among pairs of fragments. They showed these fragments as edges in a directed graph having a vertex for each fragment. Secondly, their procedure elected overlaps from the graph. Finally, the third phase integrated these overlaps into a synchronous alignment of the fragments from which a sequence was extracted.

Because our model is probabilistic, we form a combination of particle swarm optimization (PSO) and probabilistic cellular automata (PCA) to extract CA rules properly. Many studies used this combination successfully. Fengxia and Gang [4] used CA combined with PSO for simulation. Their experiments showed that their algorithm had a strong global search capability, and could effectively improve the capability for premature convergence. Their method made improvements but did not completely solve the premature convergence problem.

Djemame and Batouche [5] presented a method for detecting contours, from the application of a CA rule. They proposed an approach for resolving the problem of optimizing the search space and extracting the subset of rules likely to achieve the desired task by using metaheuristic PSO.

Dewangan et al. [6] proposed a solution for the DNA sequence assembly issue by applying PSO with naïve Crossover and shortest position value (SPV) rule. For that purpose, a model regarding PSO was used in order to have naïve crossover in addition to the SPV rule.

As indicated above, there are some limitations to the related work to date. However, these algorithms contained the premier explanation of the sequence reconstruction problem with the error. They did not solve or approximate this problem. Nearly most of the existing research has considered the sequence reconstruction problem from the perspective of computational learning theory [7-11]. Also, these methods deal with the DNA reconstruction process taking into consideration that mutations are deterministic events. However, it is commonly admissible that mutations occur randomly. For instance, a survey using Markov chains manifested biases in mutation rates that rely on the neighboring bases detecting a neighbor-dependent impact on mutations [12,26-28].

To overcome the limitations mentioned above, we combine PSO and PCA to introduce a technique for the DNA reconstruction problem. We propose a new PCA-PSO algorithm, which integrates the cellular space, cells, and neighbors of CA with the state of the particles and a PSO configuration. The PSO algorithm is applied to discover the optimal and convenient transition rules of CA for the reconstruction process. Our study seems to be more rational and convenient in the process of mutagenesis. Dealing with the model as a stochastic technique enables us to foresee the likelihood of specific mutations. So, it will be easy to predict probable mutations of viruses and other pathogens.

The rest of this paper is summarized as follows. In Section 2, we discuss the design of DNA mutations as a PCA model. Section 3 is a brief introduction to the PSO algorithm. In Section 4, we explain in detail how we merged PCA and the PSO algorithm. Also, we show our innovative algorithm's evolution rules. All the stages of our algorithm are clarified in a flow diagram chart in Section 4. Section 5 discusses the experimental simulation and results. The conclusion and the future expectations of this work are stated in Section 6.

2. DNA MODELING REGARDING PCA

In this section, we tackle DNA sequence modeling as a stochastic procedure. We took this assumption from observations that clarified how mutations happen at random. For example, Arndt et al. [12] deduced a neighbor-dependent impact in the mutagenesis process.

CAs are discrete-state systems comprising a countable lattice of analogous cells that communicate with their neighbors [13-15]. These networks can take any number of dimensions, beginning from a one-dimensional sequence of cells. The state of a CA is entirely detected by the values of the variables for each cell. Here, we use a one-dimensional CA for DNA modeling, since a DNA sequence can be expressed as a line of adjacent bases [16,17, and 31].

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Its mathematical formulation can be defined as a 4-uplet:

$$Z = (Q_d, S, N, f) \quad (1)$$

where Z is a CA system, Q is the cell space, d denotes the dimension of this space. S describes the states of all cells. N denotes a set of all cells within the neighborhood. Finally, f is the evolution rule where it determines how the cell's state can change.

In probabilistic CA, local rules may include a probabilistic element within them, instead of dictating the state of an updated cell. The rule provides the probability that an updated cell will be in each of some certain states. PCA is constructed by introducing probabilistic elements into deterministic local CA rules. The CA grid is composed of similar cells, which are,

$$\ldots, i-3, i-2, i-1, i, i+1, i+2, i+3, \ldots$$

The states of these cells are

$S_{i-3}, S_{i-2}, S_{i-1}, S_i, S_{i+1}, S_{i+2}$ and S_{i+3} .

The state of the i^{th} cell holds values from a predetermined discrete set:

$$S_i \in \{S_1, S_2, S_3, \ldots, S_n\} \quad (2)$$

Moreover, the system evolves through discrete time steps. The evolution of the probability $P_t(S)$ of state S at time step t is given by:

$$P_{t+1}(S) = \sum_{S'} W(S|S') P_t(S') \quad (3)$$

where $W(S|S')$ is the probability of transition from state S' to state S with the following two properties:

$$W(S|S') \geq 0 \quad (4)$$

$$\sum_{S} W(S|S') = 1 \quad (5)$$

A PCA is simulated by beginning from any preliminary configuration [18]. From this configuration, a sequence of configurations is generated, each one attained from the previous one through a synchronous update of all sites. It is a well-known fact that DNA mutation plays a significant role in DNA sequence evolution [19]. Neighbor-dependent mutations influence the evolution of DNA sequences. We used the mutation rates previously calculated in [1], where we used the calculated frequencies from [12]. So, we got the mutation rates that produced our evolution. Figure 1 presents a simulation of the evolution of a DNA sequence. The simulation begins with an arbitrary DNA sequence comprising 30 bases and generates the sequences for 30 consecutive time steps. Base A is shown in red, C in green, T in blue, and G in yellow.

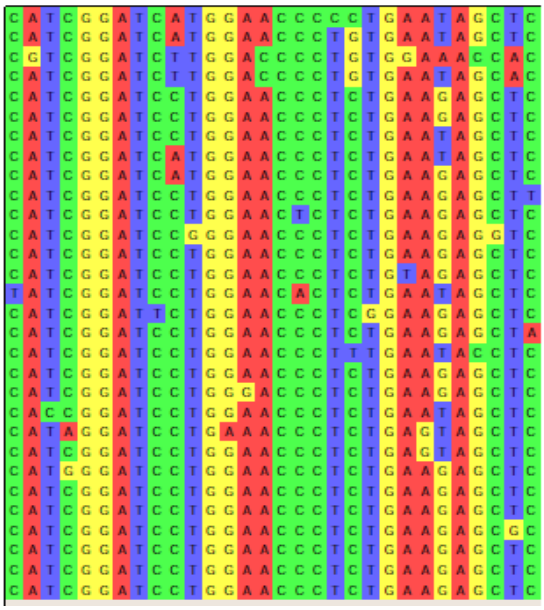


Figure 1: The simulated evolution of an arbitrary DNA sequence: (A: Red; C: Green; T: Blue; G: Yellow).

3. PARTICLE SWARM OPTIMIZATION

The PSO algorithm is an already established population-based stochastic optimization mechanism [4,20, 21, and 29]. The PSO algorithm composes a set of feasible solutions evolving to achieve an adequate solution for a problem. Its main target is to achieve the global optimum of a real-valued function outlined in a search space, named the fitness function.

PSO is initiated with a set of arbitrary particles, and it then looks for an optimum by updating generations. In every iteration, each particle is updated by two subsequent "best" values. The premier solution is the best one it has obtained until now. This value is called P_i (or $P-best$). Another best value that is pursued by the PSO algorithm is the best value, attained so far by any particle. This value is a global best and is symbolized as P_g or $(g-best)$. After detecting these two values, the particle updates its speed according to Eq. (6).

$$V_i^{t+1} = V_i^t + C_1 r_1 (P_i^t - X_i^t) + C_2 r_2 (P_g^t - X_i^t) \quad (6)$$

Moreover, the location is updated using Eq. (7).

$$X_i^{t+1} = X_i^t + V_i^{t+1} \quad (7)$$

where V_i is the speed of each particle, X_i is the current location of each particle, C_1 and C_2 are acceleration constants, r_1 and r_2 are arbitrary numbers in the range [0, 1], P_i is the best position of

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

each particle, and P_g is the best position of the swarm. The original PSO was modified by Shi et al. [20], who introduced an inertia weight ω to balance exploitation and exploration. Then, Eq. (6) becomes:

$$V_i^{t+1} = \omega V_i^t + C_1 r_1 (P_i^t - X_i^t) + C_2 r_2 \left(P_g^t - X_i^t\right) \quad (8)$$

PSO is a simple algorithm easy to execute. The simplicity of PSO implies that the algorithm is inexpensive in terms of memory requirements. These reasons led us to make this hybridization between CA and PSO for a better and fast optimization tool.

4. The Proposed PCA-PSO Technique

In this paper, we introduce an innovative mechanism for DNA sequence reconstruction based on PSO combined with PCA. In the proposed method, the search space is partitioned into cells by a CA. At any time, in some cells of the CA, a set of particles looks for a local optimum and the best solution found in their neighborhood cells.

In the PSO algorithm, every particle updates its immediate location together with all particle states. In PCA, each cell is closely incorporated with its neighbors and their positions governed by a specific rule. Its immediate state is updated with its neighbor's states. In our proposed hybridized algorithm (PCA-PSO), the state of the cell is evaluated by its state in addition to its neighbor's states. The last speed updates foremost to each cell's state. Then, approaching the optimal state, PCA-PSO updates the current cell's state once more, considering both the immediate state and optimal state. PCA-PSO modeling is as follows:

- Cells: each particle is interpreted as a singular cell.
- Cellular space: this is the group of all cells in the space. (One-dimensional CA is considered).
- State-space: the state of a cell is mentioned as the location of each particle and the state of i^{th} cell is calculated by Eq. 9:

$$S_i(t) = X_i(t) \quad (9)$$

- Neighborhood: All of the cells that may influence variations in the state of the i^{th} cell constitute the neighborhood of this cell. The neighborhood is represented with Eq. 10:

$$N(i,r) = \{S_{i-r}, \dots, S_{i-2}, S_{i-1}, S_i, S_{i+1}, S_{i+2}, , \dots, S_{i+r}\} \quad , r = 0,1,2, \dots, m \quad (10)$$

where r is the size of the neighborhood. Here, we use $r = 1$ then the neighborhood of the i^{th} cell comprises the cell itself and its neighbors directly to the right and left:

$$N(i,1) = \{S_{i-1}, S_i, S_{i+1}\} \quad (11)$$

Update rules: Every cell updates its state with its immediate location, velocity, individual optimal value, itself, and its neighbor's optimal values. The state of the i^{th} cell at a time step $t+1$ is a function of the states of its neighbors at the prior time step:

$$S_i(t+1) = f\left(S_{i-1}(t), S_i(t), S_{i+1}(t)\right) \quad (12)$$

- Discrete time step: an iteration in PSO.

DNA may be modeled as a one-dimensional CA. The four bases (A, C, T, and G) stand for the possible states of a CA cell. The state of the i^{th} cell of this CA takes values from the discrete set that incorporates the four bases:

$$S_i \in \{A, C, T, G\} \quad (13)$$

We symbolize the bases with numbers as follows:

$$\{A \to 0, C \to 1, T \to 2, G \to 3\}$$

Here, we take into account just the rules with a neighborhood size of 1. The transitions of base-pairs through evolution are clarified in Table 1. The right-hand side of each transition can be one of the four base-pairs.

Here, CAs have four states for each cell. Hence, the number of all probable rules is 4^{4^3}. The whole rule space must be investigated to detect the assumed CA rules that control the evolution of DNA sequence. Here, PSO is applied in order to investigate the enormous CA rule space.

Evolution is visualized with the aid of phylogenetic trees representing a group of organisms [22-25, 30, 32]. A phylogenetic tree is a tree demonstrating the evolutionary mutual relations between diverse species or other organisms that are accepted to have a mutual ancestor. Several species, organisms, or genomic sequences are represented on the leaves of the tree. We used the species that are clarified in Figure 2 for our simulation.

Table 1: The left-hand sides of 64 transitions of CA with a neighborhood size of 1.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

State No.	Possible states	State No.	Possible states	State No.	Possible states	State No.	Possible states
0	000	16	001	32	002	48	003
1	100	17	101	33	102	49	103
2	200	18	301	34	302	50	303
3	300	19	201	35	202	51	203
4	010	20	011	36	012	52	013
5	110	21	111	37	112	53	113
6	310	22	311	38	312	54	313
7	210	23	211	39	212	55	213
8	130	24	031	40	032	56	033
9	330	25	131	41	132	57	133
10	230	26	331	42	332	58	233
11	020	27	231	43	232	59	023
12	120	28	021	44	022	60	123
13	320	29	121	45	122	61	323
14	220	30	321	46	322	62	223
15	030	31	221	47	222	63	333

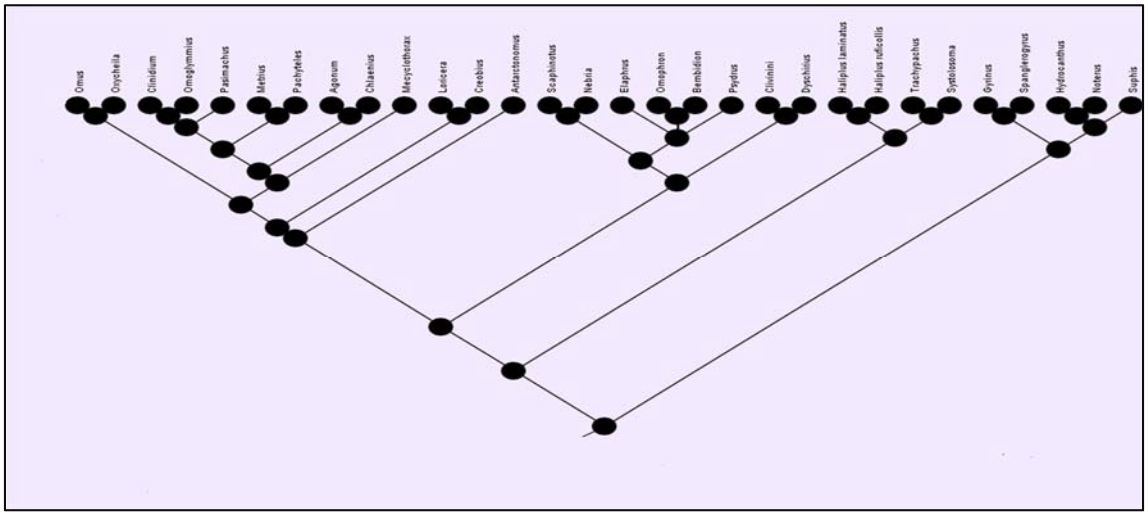


Figure 2: An example of a phylogenetic tree.

Our work seeks to detect the rules for neighborhood-based mutations that may have been resulted in the evolutions of sequences. Here, we used linear rules as a promising tool for analyzing mutation rates [19]. In the status of linear evolution rules, it takes a matrix format as follows:

$$\begin{bmatrix} \vdots \\ S_{i-1}^{t+1} \\ S_i^{t+1} \\ S_{i+1}^{t+1} \\ \vdots \end{bmatrix} = \begin{bmatrix} \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & M_{i-1,j-1} & M_{i-1,j} & M_{i-1,j+1} & \cdots \\ \cdots & M_{i,j-1} & M_{i,j} & M_{i,j+1} & \cdots \\ \cdots & M_{i+1,j-1} & M_{i+1,j} & M_{i+1,j+1} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix} \begin{bmatrix} \vdots \\ S_{i-1}^{t} \\ S_i^{t} \\ S_{i+1}^{t} \\ \vdots \end{bmatrix} + \begin{bmatrix} \vdots \\ V_{i-1}^{t+1} \\ V_i^{t+1} \\ V_{i+1}^{t+1} \\ \vdots \end{bmatrix} \quad (14)$$

where the velocity is updated using the following:

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$$\begin{bmatrix} \vdots \\ V_{i-1}^{t+1} \\ V_i^{t+1} \\ V_{i+1}^{t+1} \\ \vdots \end{bmatrix} = w \begin{bmatrix} \vdots \\ V_{i-1}^{t} \\ V_i^{t} \\ V_{i+1}^{t} \\ \vdots \end{bmatrix} + C_1 r_1 \left(\begin{bmatrix} \vdots \\ P\_best_{i-1}^{t} \\ P\_best_i^{t} \\ P\_best_{i+1}^{t} \\ \vdots \end{bmatrix} - \begin{bmatrix} \vdots \\ S_{i-1}^{t} \\ S_i^{t} \\ S_{i+1}^{t} \\ \vdots \end{bmatrix} \right) + C_2 r_2 \left(\begin{bmatrix} \vdots \\ g\_best_{i-1}^{t} \\ g\_best_i^{t} \\ g\_best_{i+1}^{t} \\ \vdots \end{bmatrix} - \begin{bmatrix} \vdots \\ S_{i-1}^{t} \\ S_i^{t} \\ S_{i+1}^{t} \\ \vdots \end{bmatrix} \right) \quad (15)$$

The states of all CA cells at time step t are represented by the column array at the right-hand side of Eq. (14). This array is multiplied by the array M that denotes the evolution rule. The array elements $M_{i,j}$ may hold only two values (0 and 1). The output array at the left-hand side of Eq. (14) contains the states of all CA cells at time step $t+1$. Figure 3 shows the necessary steps of our innovative hybridization algorithm, which are represented in detail through a simplified flow diagram.

4.1. Fitness Function

In this stage, the fitness function should be evaluated after representing each individual. The optimal solution is obtained based on the resulting fitness value. In the DNA sequence reconstruction problem, the optimum solution is the highest matching score of bases among sequences in the previous steps. First, we have to compute the evolution of DNA sequences to extract the proper rules for the reconstruction process shown in Fig. 1. The matching score is evaluated by enumerating the equivalent nucleotide of sequences. The matching score for a pair of sequences is computed using Eq. (16):

$$Score_{S_{i,j}\&\hat{S}_{i,j}} = \begin{cases} 0 & if\ bases\ do\ not\ match \\ Score_{S_{i,j}\&\hat{S}_{i,j}} + 1 & otherwise \end{cases} \quad (16)$$

where, $Score_{S_{i,j}\&\hat{S}_{i,j}}$ is a matching score of two consecutive sequences, i and j are the indices of the cell. After calculating the score of each two consecutive sequences, a total score is evaluated for a specific individual of PSO by using Eq. (17).

$$\max f(x) = \sum_i \sum_j Score_{S_{i,j}\&\hat{S}_{i,j}} \quad (17)$$

Algorithm 1: The steps of the PCA-PSO rules extraction

```
Initialize swarm-size, D search space dimension, N number of
neighbors and iterM max iteration times.
For i=1 to swarm-size do
    Particle[i].position=random(0,1,2,3)
    Particle[i].velocity=0,
    Particle[i].fitness=0,
Endfor
For i=1 to swarm-size do
   P-best[i]=particle[i]
Endfor
G-best=particle[1] //best global position
For i=1 to swarm-size do
    P-best[i]=fitness(X_i)  //Evaluate fitness for position i
Endfor
While (stop criterion) is not satisfied do
      For p=1 to swarm-size do
          //apply CA rule for each particle
  For each evolution step do
      Compute next rule
  Endfor
            Update particle's velocity V_i(t+1)
 Update cell's state S_i(t+1)
 Evaluate fitness
       Endfor
      //comparison of fitness
      For i=1 to swarm-size do
         If particle[i].fitness >P-best[i].fitness then
            P-best[i]=particle[i]
         Endif
        If G-best[i].fitness >particle[i].fitness then
            G-best[i]=particle[i]
        Endif
       Endfor
       For i=1 to swarm-size do
          Update PSO parameters: position and velocity
       Endfor
Endwhile
Return to step 2
```

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

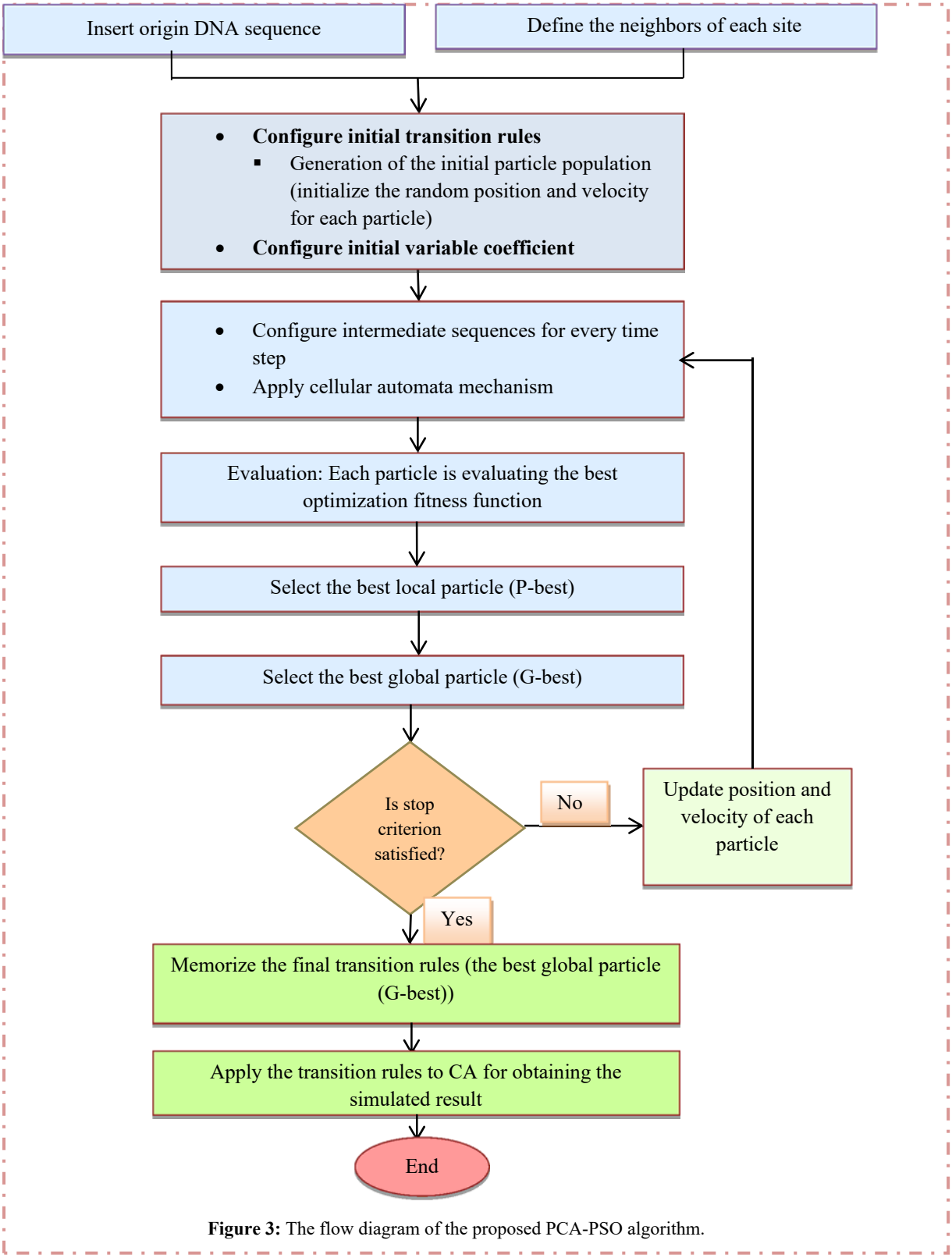


Figure 3: The flow diagram of the proposed PCA-PSO algorithm.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

where, $f(x)$ indicates the fitness value for individual i,j of PSO also, max indicates that our target is to get the maximum value of $f(x)$. The optimal solution is the one that has the largest value of $f(x)$. The fitness function is determined by summing all scores calculated by Eq. (16). The main goal is to find symmetry relations among various sequences. Algorithm 1 is the proposed PCA-PSO algorithm for the selection of CA evolution rules.

5. SIMULATIONS AND RESULTS

This section depicts the experimental setup and outcomes achieved after the simulation. Our enhanced PSO-PCA algorithm starts by inserting the following parameters to obtain the desired probabilistic rules:

- The native sequence of DNA,
- The sequences of DNA belong to in-between evolution steps,
- The eventual sequences of DNA,
- The maximum iterations number.

We used phylogenetic trees for the reconstruction process and to represent the samples of species used for the simulation. For each branch, we applied a set of CA rules for altering a predecessor sequence into the offspring sequence. The proposed technique works as follows. First, it compares the current sequence, with the produced sequence at the next time step using a similarity score. We define this score as the proportion of the number of matching base pairs in two consecutive sequences. Then, the proposed algorithm randomly chooses a CA rule in each time step and applies it to the immediate sequence.

We then observe the progression of the resemblance score, meanwhile evolution. From the concept that not all the base pairs mutate in each time step, we used for simulation non-uniform probabilistic rules. In our evolution, we attempt to dynamically generate a CA rule utilizing a sequence achieved within the evolution and the consecutive sequence. Figure 4 clarifies the dynamic structure of a rule.

While forming CA rules, we aim to generate a CA rule using a sequence that resulted in the evolution process and its next-step sequence. The transitions on the left-hand side are established by applying a rule to the immediate sequence. The next-step sequence configures the right-hand direction of the transitions. For instance, in Figure 3, the left-hand side of a transition is outlined by the premier three base-pairs of the immediate sequence, labeled, CAT. As for the right-hand side of the transition, it is outlined by the corresponding base-pair in the consecutive sequence, 0.

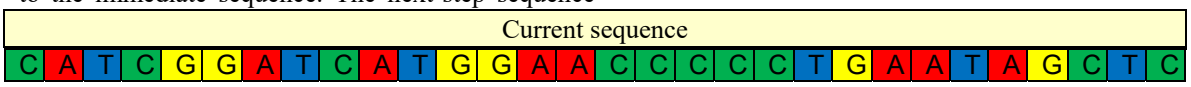
We used sequences from a 100-base size, as shown in Figure 5 that are resulted from the simulation software (Mesquite). From that pattern, we deduced the formation rules, as clarified in Table 2. This means that if we have the evolution pattern, we can detect its formation rules and vice versa.

Table 2 lists some of the most common CA rules for mutations regarding the effect of the neighborhood. The rules achieved using the hybridized technique, are used instantly to foretell of next-step sequences.

Detecting the most probable rules for mutations that take into consideration the impact of the neighborhood is the major target of this work. Table 2 demonstrates some of the resultant rules that were stratified to some of the phylogenetic tree branches from random DNA sequences.

It is apparent that PCA-PSO proved to be a successful procedure for reconstructing the evolution paradigm of the given DNA sequences. We developed a proper PCA-PSO algorithm that efficiently elicits the CA rule, which controls the evolution of sequence. During these random experiments, the innovative procedure dictated the possible rules that produced the given evolution pattern, and the algorithm showed itself to be a successful simulation tool. As a result, we demonstrated that by having a series of DNA sequences that represent a set of evolutionary steps, this procedure could be utilized for generating the probabilistic rules of this evolution pattern. These rules are capable of reconstructing DNA sequences properly. Our attempt of incorporating probabilistic components in our model produces a system capable of predicting the likelihood of definite mutations. Also, our technique showed itself to be a promising tool for properly simulating the evolution of large sequences, as seen in Figure 6.

Testing the data clarified in Figure 6, with the help of the Mesquite simulation software, we obtained the evolution in Figure 7.



ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

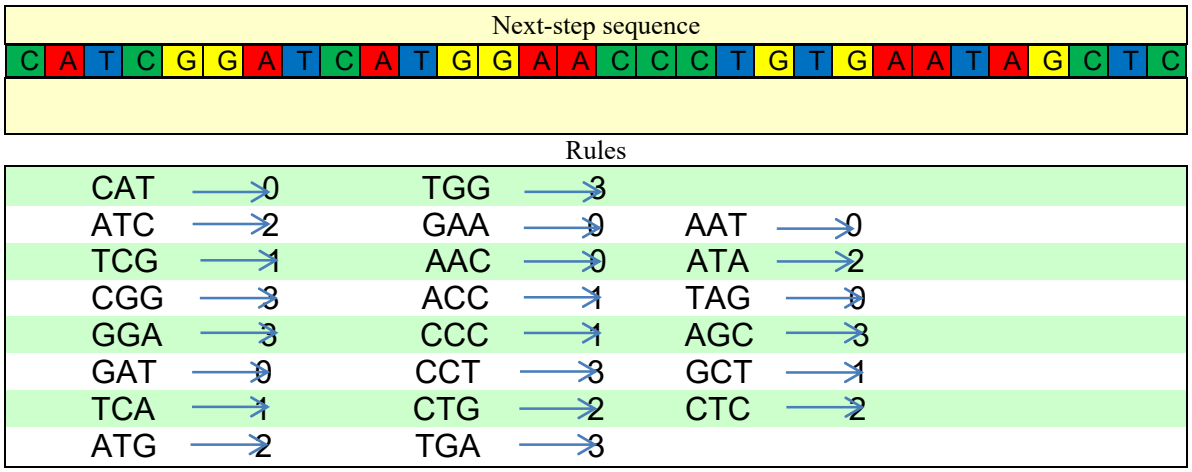


Figure 4: An example of the formation of CA rules.

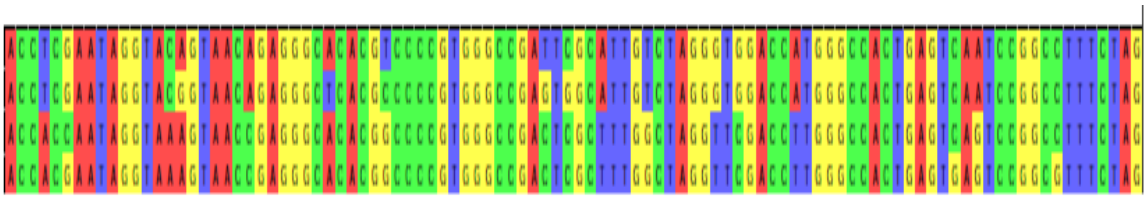


Figure 5: The DNA sequences corresponding to the radix, the eventual, and two in-between evolution steps.

Table 2: *The rule analysis for DNA sequences.*

Current sequence-next step sequences	**Resultant Rule**
(1-2)	1121300203320103200030333101321113231022302312030112312331021322
(2-3)	1101020320020011303331010131132331012312223120320011212330212220
(1-30)	1121300203320323030331013111323311302111223022330112112310023122

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

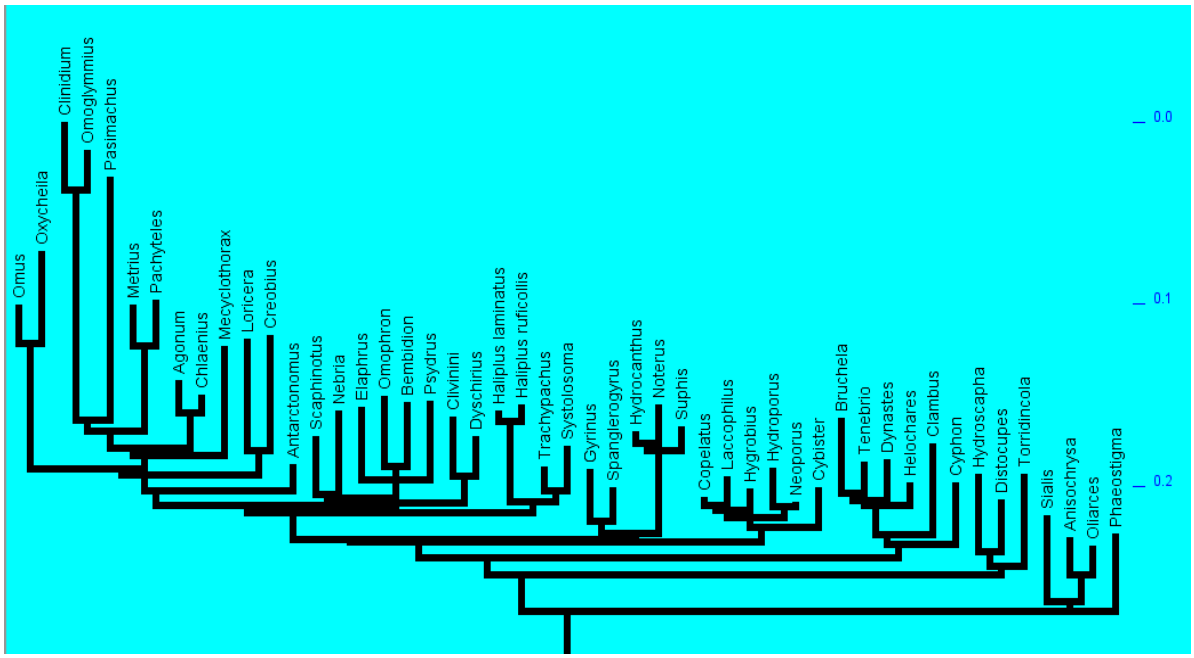


Figure 6: Example of a phylogenetic tree with 49 samples.

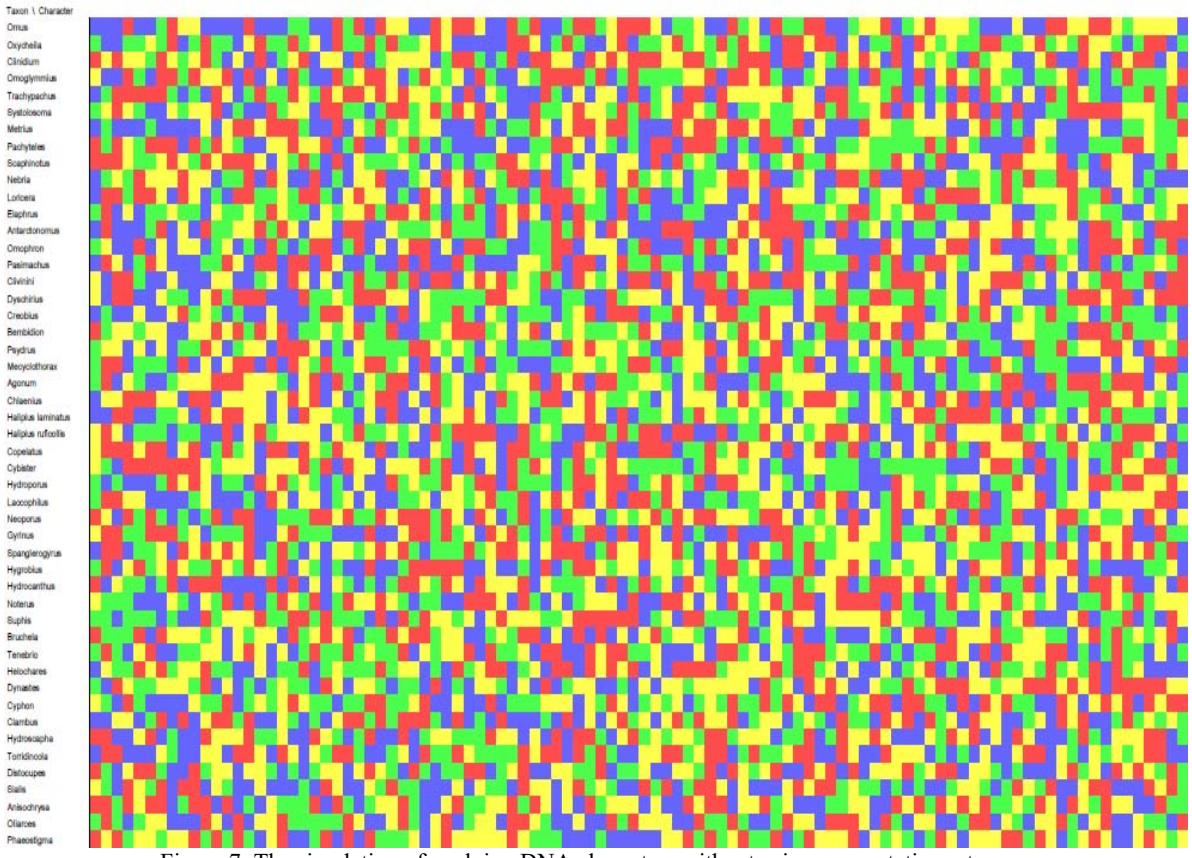


Figure 7: The simulation of evolving DNA characters without using our mutation rates.

Journal of Theoretical and Applied Information Technology
15th March 2020. Vol.98. No 05

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

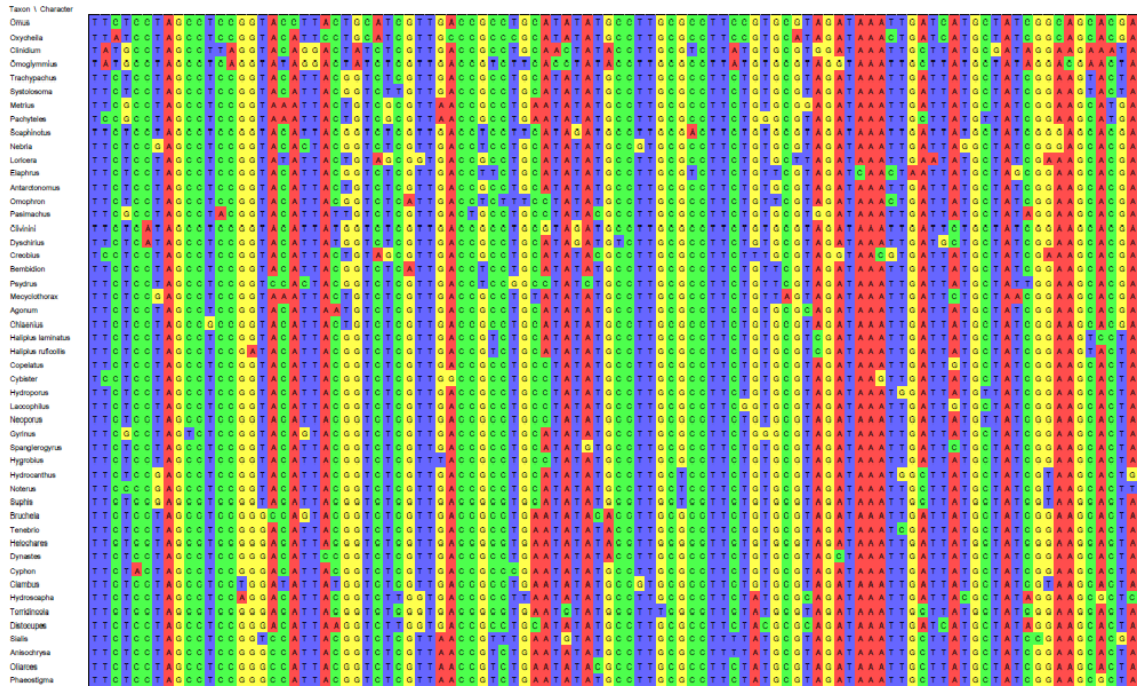


Figure 8: The simulation of evolving DNA characters after using our rates and the probabilistic rules.

Moreover, after applying our mutation rates besides using our innovative PCA-PSO algorithm, we obtained the results shown in Figure 8.

The results in Figure 7 and Figure 8 proved that, even for long sequences, our model leads to an evolution pattern with a high similarity score among sequences, which in turn, can be very useful to biologists. And this is because analyzing DNA sequences for various species has many practical applications including drug detection, producing new hybridized breeds and discovering why a disease attacked a particular part of the body.

6. CONCLUSION

In this paper, we presented a new technique based on PCA to investigate the rules regarding neighborhood effects on DNA mutations. CA proved to be a robust system for analyzing DNA mutations. CA rules generated by simulating DNA mutations can give us beneficial insights about the influence of neighboring base pairs on the evolution of DNA sequences. The proposed tool for simulation is based on the usage of PSO because it extracts the PCA evolution rules very efficiently. While using this hybridized system of both PSO and CA, we observed that they have many common properties:

- Both of them are consisted of a group of individuals, which are represented as cells in CA and particles in PSO.
- Each individual communicates with each other to transfer definite essential information. In CA, each cell interacts with its neighborhoods and modifies its state regarding the immediate states of the cell itself and its neighborhood. Similarly, each particle interacts with other particles and updates such essential information as position, fitness, speed, local best location, and global best location in PSO.
- In CA, a transition rule is applied to control the evolution of the cell, while PSO uses two equations representing position and velocity for updating particles.
- Finally, both PSO and CA evolve in a discrete-time step. PSO simulates a swarm system that involves communication among diverse particles in the swarm.

In order to improve the performance of the communication, we concentrate on exploiting the notion of PCA to investigate the interaction configuration and information inheriting and diffusing techniques of the swarm system of PSO. So in the suggested PCA-PSO, we inspect PSO with a PCA model, where individuals can only exchange information through their neighborhood.

There are powerful concerns that neighboring base pairs influence mutations of DNA base pairs. We try to reveal this correlation by modeling DNA as a CA model, where the CA rules govern the DNA mutations. Due to the enormous rule space comprised of our simulation, we adopted the PSO algorithm for extracting these rules efficiently. Then, we attempt to use the resultant rules and the mutation rates for predictions of future DNA sequences in phylogenetic trees. Simulating DNA as a PCA model facilitates viewing, analyzing, and comparing various sequences. Also, our method is suitable for the comparison of many very long sequences. We plan to popularize this study as a way to handle distinct

rules for diverse structures of the cell neighborhood regarding mutation effects. Particularly, diverse neighborhood sizes larger than one can be discussed.

ACKNOWLEDGMENT

We thank Prof. Dr. E. Ahmed, Mathematics Department, Faculty of Science, Mansoura, Egypt, for his guidance and comments.

REFERENCES

[1] W. M. Elsayed, M. Elmogy, and B. El-Desouky, "Evolutionary behavior of DNA sequences analysis using non-uniform probabilistic cellular automata model," *Cincia e Tcnica Vitivincola,* vol. 32, pp. 137-148, 2017.

[2] T. Gingeras, J. Milazzo, D. Sciaky, and R. Roberts, "Computer programs for the assembly of DNA sequences," *Nucleic Acids Research,* vol. 7, no. 2, pp. 529-543, 1979.

[3] H. Peltola, H. Söderlund, and E. Ukkonen, "SEQAID: A DNA sequence assembling program based on a mathematical model," 1984.

[4] Y. Fengxia and L. Gang, "The simulation and improvement of particle swarm optimization based on cellular automata," *Procedia Engineering,* vol. 29, pp. 1113-1118, 2012.

[5] S. Djemame and M. Batouche, "Combining cellular automata and particle swarm optimization for edge detection," *International Journal of Computer Applications,* vol. 57, no. 14, 2012.

[6] S. K. Dewangan, G. S. Chhabra, And A. Trivedi, "Dna Sequence Assembly Using Particle Swarm Optimization And Naïve Crossover," *Indian J. Sci. Res,* vol. 14, no. 2, pp. 358-365, 2017.

[7] M. Li, "Towards a DNA sequencing theory (learning a string)," in *Foundations of Computer Science, 1990. Proceedings., 31st Annual Symposium on*, 1990, pp. 125-134: IEEE.

[8] G. B. Fogel, K. Chellapilla, and D. B. Fogel, "Reconstruction of DNA sequence information from a simulated DNA chip using evolutionary programming," in *International Conference on Evolutionary Programming*, 1998, pp. 427-436: Springer.

[9] J.-H. Zhang, L.-Y. Wu, and X.-S. Zhang, "Reconstruction of DNA sequencing by hybridization," *Bioinformatics,* vol. 19, no. 1, pp. 14-21, 2003.

[10] G. C. Sirakoulis, I. Karafyllidis, R. Sandaltzopoulos, P. Tsalides, and A. Thanailakis, "An algorithm for the study of DNA sequence evolution based on the genetic code," *BioSystems,* vol. 77, no. 1-3, pp. 11-23, 2004.

[11] C. Mizas, G. C. Sirakoulis, V. Mardiris, I. Karafyllidis, N. Glykos, and R. Sandaltzopoulos, "Reconstruction of DNA sequences using genetic algorithms and cellular automata: Towards mutation prediction?," *Biosystems,* vol. 92, no. 1, pp. 61-68, 2008.

[12] P. F. Arndt, C. B. Burge, and T. Hwa, "DNA sequence evolution with neighbor-dependent mutation," *Journal of Computational Biology,* vol. 10, no. 3-4, pp. 313-322, 2003.

[13] Adamatzky, A., Identification Of Cellular Automata. London: CRC Press, https://doi.org/10.1201/9781482272543, (1994).

[14] M. Sipper, "Evolving uniform and non-uniform cellular automata networks," in *Annual Reviews Of Computational Physics V*: World Scientific, 1997, pp. 243-285.

[15] N. Ganguly, B. K. Sikdar, A. Deutsch, G. r. Canright, and P. P. Chaudhuri, "A Survey on Cellular Automata, Centre for High Performance Computing, Dresden University of Technology," Technical Report 92003.

[16] G. C. Sirakoulis, I. Karafyllidis, C. Mizas, V. Mardiris, A. Thanailakis, and P. Tsalides, "A cellular automaton model for the study of DNA sequence evolution," *Computers in biology and medicine,* vol. 33, no. 5, pp. 439-453, 2003.

[17] S. Zhou, B. Wang, X. Zheng, and C. Zhou, "Study and Application of DNA Cellular Automata Self-assembly," in *Bio-Inspired Computing-Theories and Applications*: Springer, 2014, pp. 654-658.

[18] M. Stoelinga, "An introduction to probabilistic automata," *Bulletin of the EATCS,* vol. 78, no. 176-198, p. 2, 2002.

[19] M. Jones, S. Thomas, and K. Clarke, "The application of a linear algebra to the analysis of mutation rates," *Journal of theoretical biology,* vol. 199, no. 1, p. 11, 1999.

[20] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, 1998, pp. 69-73: IEEE.

[21] R. S. Verma, V. Singh, and S. Kumar, "Dna sequence assembly using particle swarm optimization," *International Journal of Computer Applications,* vol. 28, no. 10, 2011.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[22] G. J. Olsen, H. Matsuda, R. Hagstrom, and R. Overbeek, "fastDNAml: a tool for construction of phylogenetic trees of DNA sequences using maximum likelihood," *Bioinformatics,* vol. 10, no. 1, pp. 41-48, 1994.

[23] A. Siepel and D. Haussler, "Phylogenetic estimation of context-dependent substitution rates by maximum likelihood," *Molecular biology and evolution,* vol. 21, no. 3, pp. 468-488, 2004.

[24] F. Delsuc, H. Brinkmann, and H. Philippe, "Phylogenomics and the reconstruction of the tree of life," *Nature Reviews Genetics,* vol. 6, no. 5, p. 361, 2005.

[25] N. Goldman, "Maximum likelihood inference of phylogenetic trees, with special reference to a Poisson process model of DNA substitution and to parsimony analyses," *Systematic Zoology*, vol. 39, no. 4, pp. 345-361, 1990.

[26] S. T. Hess, J. D. Blake, and R. D. Blake, "Wide variations in neighbor-dependent substitution rates," *J Mol Biol,* vol. 236, no. 4, pp. 1022-33, Mar 4 1994.

[27] M. Bulmer, "Neighboring base effects on substitution rates in pseudogenes," *Molecular biology and evolution,* vol. 3, no. 4, pp. 322-329, 1986.

[28] B. R. Morton, I. V. Bi, M. D. McMullen, and B. S. Gaut, "Variation in mutation dynamics across the maize genome as a function of regional and flanking base composition," *Genetics,* vol. 172, no. 1, pp. 569-577, 2006.

[29] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*, 1997, vol. 5, pp. 4104-4108: IEEE.

[30] V. Makarenkov, "T-REX: reconstructing and visualizing phylogenetic trees and reticulation networks," *Bioinformatics,* vol. 17, no. 7, pp. 664-668, 2001.

[31] D. Posada and K. A. Crandall, "Evaluation of methods for detecting recombination from DNA sequences: computer simulations," *Proceedings of the National Academy of Sciences,* vol. 98, no. 24, pp. 13757-13762, 2001.

[32] N. Saitou and M. Nei, "The neighbor-joining method: a new method for reconstructing phylogenetic trees," *Molecular biology and evolution,* vol. 4, no. 4, pp. 406-425, 1987.