e-ISJN: A4372-3114
p-ISJN: A4372-3115

ISSN: 2321-7782 (Online)
ISSN: 2347-1778 (Print)
Impact Factor: 7.529

Volume 11, Issue 11, November 2023

International Journal of Advance Research in Computer Science and Management Studies

Research Article / Survey Paper / Case Study

Available online at: www.ijarcsms.com

A Monthly Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories

Application of Finite State Automaton in a Random DNA Sequence and Pattern Analysis

Dr. Asoke Nath[1]
St. Xavier's College (Autonomous),
Kolkata, India.

Priyanshu Acharya[2]
St. Xavier's College (Autonomous),
Kolkata, India.

DOI: https://doi.org/10.61161/ijarcsms.v11i11.2

Abstract: This paper presents a computational method that can be used to convert a given Non-Deterministic Finite State Automata (NFA) to a Deterministic Finite State Automata (DFA). Different classes of automata are used because it is often seen that one kind of automaton can achieve something that other cannot. However, DFA and NFA are equally powerful but DFA is often preferred for scientific operations. The same computational method is applied to long DNA for sequence matching. A given DNA Sequence is passed through a DFA to check for the presence of a required matching sequence. This makes it useful in the fields where large volumes of DNA are to be scanned to check for the presence of any given pattern, and it will also help in checking for mutated DNA and the presence or absence of any genetic disease in the sequence.

Keywords: Non-Deterministic Finite Automata, Deterministic Finite Automata, DNA Sequence Matching, Nitrogenous Bases, Python.

I. INTRODUCTION

A Deterministic Finite State Automata is abbreviated as DFA. There is only one path that can be traversed from the current state to the next state under one input symbol. This uniqueness of the path traversed makes this automata deterministic. It does not accept null moves and can contain multiple final states.

A DFA is represented as a set of quintuple. It is represented as: -

$$(Q,\sum,q_0,F,\delta)$$

where,

Q = Set of states

$\sum$ = Set of symbols

q_0 = Initial State

F = Set of Final States

δ = Transition Function

The transition function for DFA is defined as: -

$$\delta: Q \times \sum \rightarrow Q$$

A Non-Deterministic Finite State Automata is abbreviated as NFA. There exists multiple path from current state for a particular symbol which results in different next states. So, this automata is termed as Non-Deterministic. NFAs also consider null moves. For a given regular language, it is easier to construct the NFA than the DFA. The NFA can also be represented by a quintuple and it is of the same form as that of the DFA.

A NFA is represented as a set of quintuple. It is represented as: -

$$(Q,\sum,q_0,F,\delta)$$

where,

Q = Set of states

$\sum$ = Set of symbols

q_0 = Initial State

F = Set of Final States

δ = Transition Function

The transition function for NFA is defined as: -

$$\delta: Q \times \sum \rightarrow 2^Q$$

A DFA is a restricted NFA and a language which is accepted by a DFA will be accepted by some NFA. Every NFA is not DFA, however every NFA can be converted to a DFA. It is easy to convert a NFA to a DFA using standard algorithm such as Successor Table Method or using Epsilon (ε) Closure Method. In the present paper the authors have implemented a Python code to convert NFA to DFA which is going to make the laborious work easier and simpler. Although a NFA can perform the various operations that are performed over a DFA, often the DFA plays a more important role in performing scientific operations.

DNA or Deoxyribonucleic Acid is a polymer which consists of two polynucleotide chains coiling around each other to form a double helix. This contains various genetic information which is responsible for growth, development, functioning and reproduction. The double helical structure of polynucleotides is made up of simple monomeric units called nucleotides.

Each nucleotide is further made up of a phosphate group, a deoxyribose sugar and any one out of four nitrogenous bases. The four bases are Adenine (A), Thymine (T), Guanine (G) and Cytosine (C). The bases are bound by base pairing rules and A pairs with T and C pairs with G using hydrogen bonds.

The sequence of the four nitrogenous bases encodes the genetic information of an organism. They are responsible for RNA formation and Protein Generation. They also play a vital role in DNA Fingerprinting. So, the study of these sequences is of utmost importance for mankind, and various methods are used to study them.

II. Proposed Algorithm

A given NFA can be represented as a directed graph. So, to find a particular sequence in a DNA, prepare the corresponding NFA. For the algorithm, take this graph as an input. The following are the steps that can followed to convert NFA to DFA.

Step 1: From the given NFA draw the state transition table and mark the starting and final states accordingly.

Step 2: Try to eliminate the states which are similar. For this, check if the transition due to different inputs are the same. Also check if the state is a starting state or final states or none of them. States can only be eliminated if they are equivalent and keep one of the state for every such duplicacies.

Step 3: After eliminating the states, search for the transitions where those eliminated states are present. Replace those states with the one that has been kept. This is termed as the Reduced State Table.

Step 4: Build a Successor Table. Here, copy the first transition of the previous graph.

Step 5: Look for the state change from the first symbol and if the table does not have it as a state in it, put it in the table and then go for the next symbol till the last. If there is no such state, go to Step 8.

Step 6: Next go to next state and find its transition at various symbols and place them in the table.

Step 7: Repeat step 5 and step 6 till all the new states generated have been placed in the table.

Step 8: After getting the Successor Table, find the final states in it by selecting those states which have atleast one final state from the reduced state table.

Step 9: Again try to eliminate the equivalent states from the Successor Table.

Step 10: Finally, the Modified Table is achieved. It shows the transition of the required DFA.

Step 11: From the table we can easily draw the graph for the DFA by mapping the transitions corresponding to the input.

Step 12: Now, substitute the states with some names and make the transition table for the DFA accordingly. The final states are also marked.

Step 13: Next, feed the given DNA sequence in the DFA. If the resultant step is the one which is one of the final states, the given DNA sequence consists of the required pattern sequence in it. If the resultant step is not one of the final states, the given DNA sequence does not contain the required pattern sequence in it.

III. Results and Discussion

After applying the following algorithm to find a DNA sequence of the form,

$$\{ A^{m}T^{2n}C^{p} + AG^{2x}C \mid m, n, p, x >= 1 \}$$

The following results are obtained.

The corresponding NFA for the given language is shown in Figure 1.

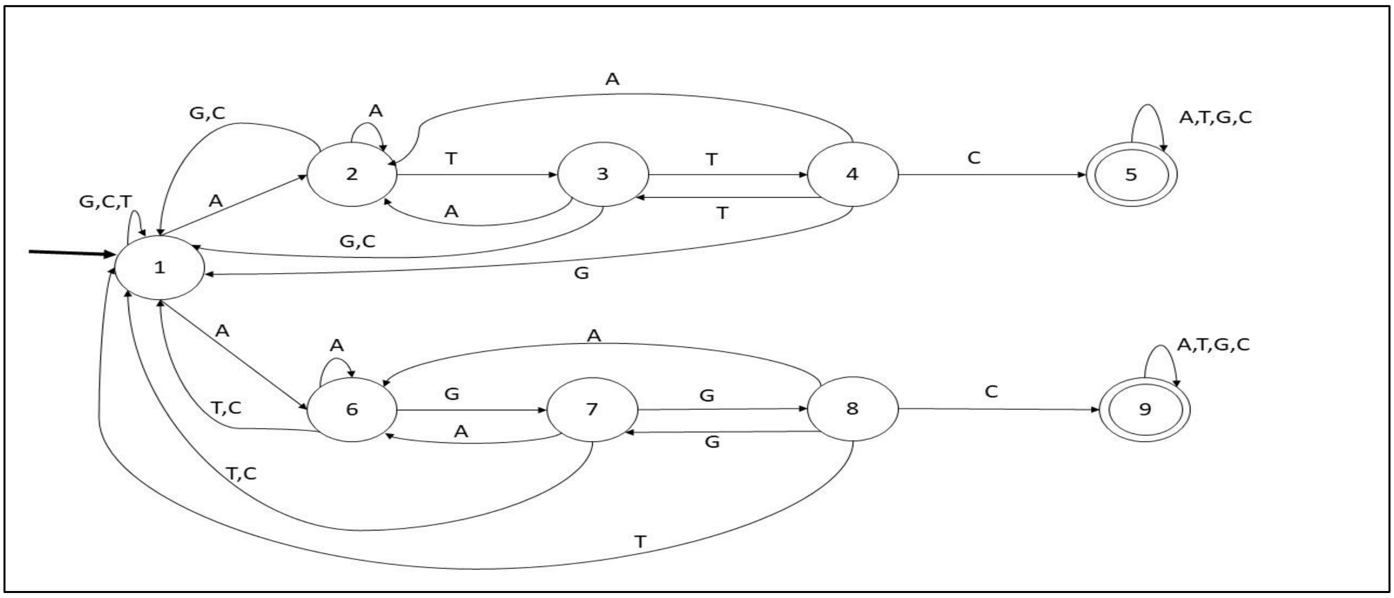
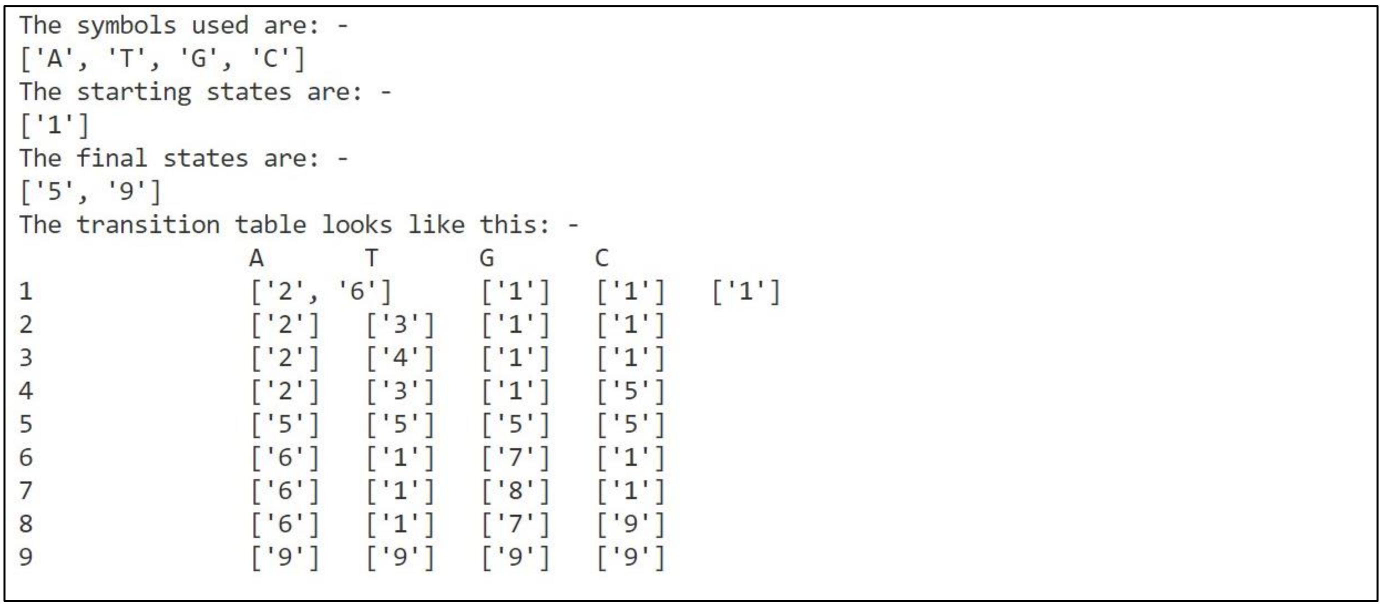


Figure 1: NFA for the required DNA sequence pattern

The starting state is marked by the bold arrow and the final states are marked using concentric circles. The transitions are depicted using arrows in Figure 1.

The state transition table for this NFA is shown in Figure 2.

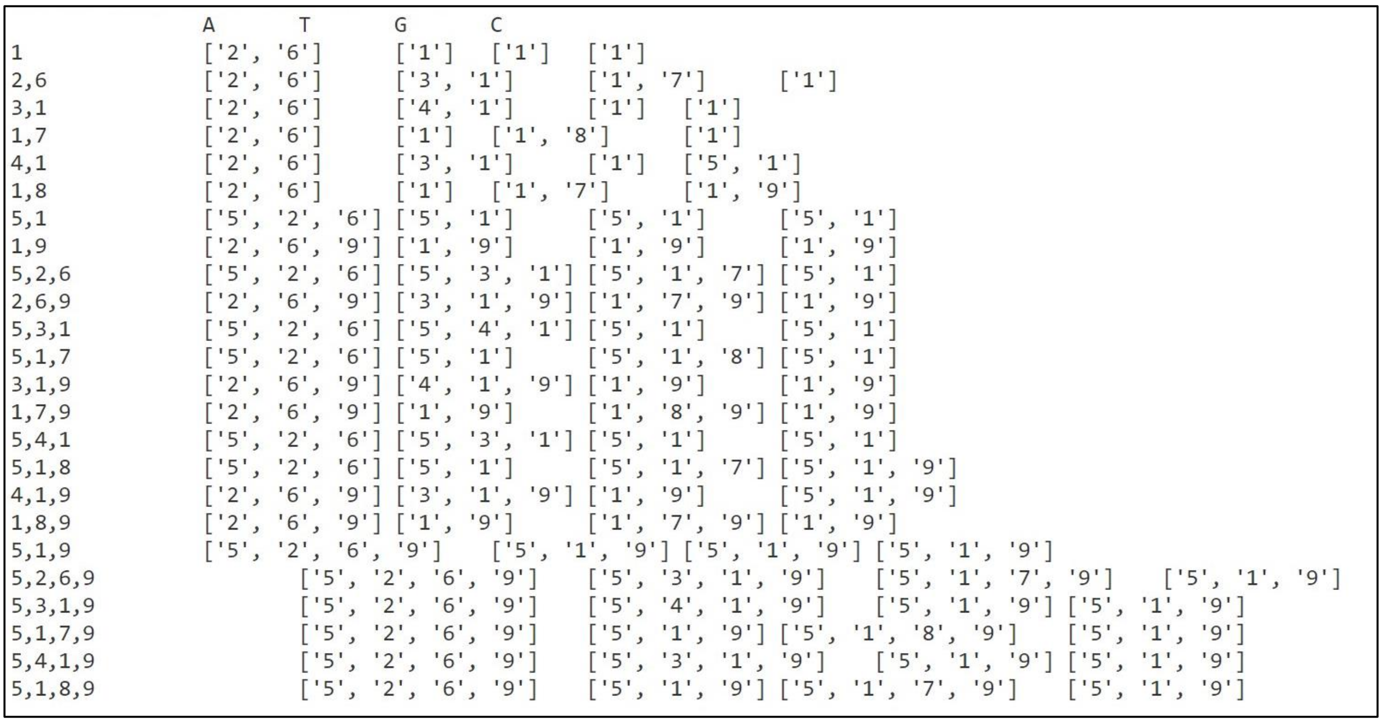


```
The symbols used are: -
['A', 'T', 'G', 'C']
The starting states are: -
['1']
The final states are: -
['5', '9']
The transition table looks like this: -
                A       T       G       C
1               ['2', '6']      ['1']   ['1']   ['1']
2               ['2']   ['3']   ['1']   ['1']
3               ['2']   ['4']   ['1']   ['1']
4               ['2']   ['3']   ['1']   ['5']
5               ['5']   ['5']   ['5']   ['5']
6               ['6']   ['1']   ['7']   ['1']
7               ['6']   ['1']   ['8']   ['1']
8               ['6']   ['1']   ['7']   ['9']
9               ['9']   ['9']   ['9']   ['9']
```

Figure 2: State Transition Table for the given NFA

The reduced table for this remains the same, as it cannot be further reduced.

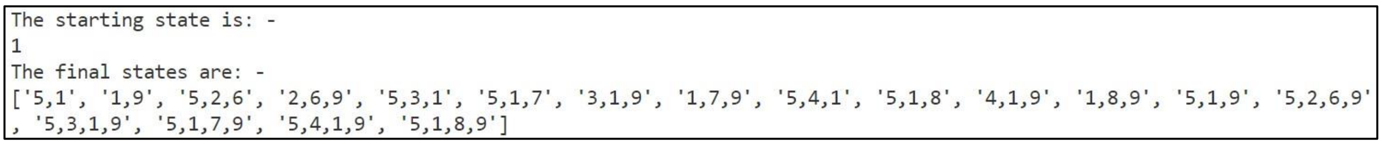
So, the Successor Table is prepared. The Successor Table is shown in Figure 3.



```
                A       T       G       C
1               ['2', '6']      ['1']   ['1']   ['1']
2,6             ['2', '6']      ['3', '1']      ['1', '7']      ['1']
3,1             ['2', '6']      ['4', '1']      ['1']   ['1']
1,7             ['2', '6']      ['1']   ['1', '8']      ['1']
4,1             ['2', '6']      ['3', '1']      ['1']   ['5', '1']
1,8             ['2', '6']      ['1']   ['1', '7']      ['1', '9']
5,1             ['5', '2', '6'] ['5', '1']      ['5', '1']      ['5', '1']
1,9             ['2', '6', '9'] ['1', '9']      ['1', '9']      ['1', '9']
5,2,6           ['5', '2', '6'] ['5', '3', '1'] ['5', '1', '7'] ['5', '1']
2,6,9           ['2', '6', '9'] ['3', '1', '9'] ['1', '7', '9'] ['1', '9']
5,3,1           ['5', '2', '6'] ['5', '4', '1'] ['5', '1']      ['5', '1']
5,1,7           ['5', '2', '6'] ['5', '1']      ['5', '1', '8'] ['5', '1']
3,1,9           ['2', '6', '9'] ['4', '1', '9'] ['1', '9']      ['1', '9']
1,7,9           ['2', '6', '9'] ['1', '9']      ['1', '8', '9'] ['1', '9']
5,4,1           ['5', '2', '6'] ['5', '3', '1'] ['5', '1']      ['5', '1']
5,1,8           ['5', '2', '6'] ['5', '1']      ['5', '1', '7'] ['5', '1', '9']
4,1,9           ['2', '6', '9'] ['3', '1', '9'] ['1', '9']      ['5', '1', '9']
1,8,9           ['2', '6', '9'] ['1', '9']      ['1', '7', '9'] ['1', '9']
5,1,9           ['5', '2', '6', '9']    ['5', '1', '9'] ['5', '1', '9'] ['5', '1', '9']
5,2,6,9                 ['5', '2', '6', '9']    ['5', '3', '1', '9']    ['5', '1', '7', '9']    ['5', '1', '9']
5,3,1,9                 ['5', '2', '6', '9']    ['5', '4', '1', '9']    ['5', '1', '9'] ['5', '1', '9']
5,1,7,9                 ['5', '2', '6', '9']    ['5', '1', '9'] ['5', '1', '8', '9']    ['5', '1', '9']
5,4,1,9                 ['5', '2', '6', '9']    ['5', '3', '1', '9']    ['5', '1', '9'] ['5', '1', '9']
5,1,8,9                 ['5', '2', '6', '9']    ['5', '1', '9'] ['5', '1', '7', '9']    ['5', '1', '9']
```

Figure 3: Successor Table for the NFA

The NFA has now been converted to a DFA. The starting and the final states for this DFA is shown in Figure 4.



```
The starting state is: -
1
The final states are: -
['5,1', '1,9', '5,2,6', '2,6,9', '5,3,1', '5,1,7', '3,1,9', '1,7,9', '5,4,1', '5,1,8', '4,1,9', '1,8,9', '5,1,9', '5,2,6,9'
, '5,3,1,9', '5,1,7,9', '5,4,1,9', '5,1,8,9']
```

Figure 4: The starting and final states are shown

The Modified Table for this Successor Table remains the same, as it cannot be further reduced. So, Figure 3 shows the Modified Table as well and the starting and final states are shown by Figure 4.

The conversion of the NFA to the DFA has been achieved. Now, the DNA sequence is applied on the DFA. Prior to that, some modifications are performed for easier understanding.

The states are labelled suitably for easier identification. The labels are shown in Table 1.

The starting state is q1, and the final states are highlighted in yellow.

q1	1
q2	2,6
q3	3,1
q4	1,7
q5	4,1
q6	1,8
q7	5,1
q8	1,9
q9	5,2,6
q10	2,6,9
q11	5,3,1
q12	5,1,7
q13	3,1,9
q14	1,7,9
q15	5,4,1
q16	5,1,8
q17	4,1,9
q18	1,8,9
q19	5,1,9
q20	5,2,6,9
q21	5,3,1,9
q22	5,1,7,9
q23	5,4,1,9
q24	5,1,8,9

Table 1: Labels for the states of DFA

The state transition table for the DFA is shown in Table 2.

Present State	**A**	**T**	**G**	**C**
q1	q2	q1	q1	q1
q2	q2	q3	q4	q1
q3	q2	q5	q1	q1
q4	q2	q1	q6	q1
q5	q2	q3	q1	q7
q6	q2	q1	q4	q8
q7	q9	q7	q7	q7
q8	q10	q8	q8	q8
q9	q9	q11	q12	q7
q10	q10	q13	q14	q8
q11	q9	q15	q7	q7
q12	q9	q7	q16	q7
q13	q10	q17	q8	q8
q14	q10	q8	q18	q8
q15	q9	q11	q7	q7
q16	q9	q7	q12	q19
q17	q10	q13	q8	q19
q18	q10	q8	q14	q8
q19	q20	q19	q19	q19

q20	q20	q21	q22	q19
q21	q20	q23	q19	q19
q22	q20	q19	q24	q19
q23	q20	q21	q19	q19
q24	q20	q19	q22	q19

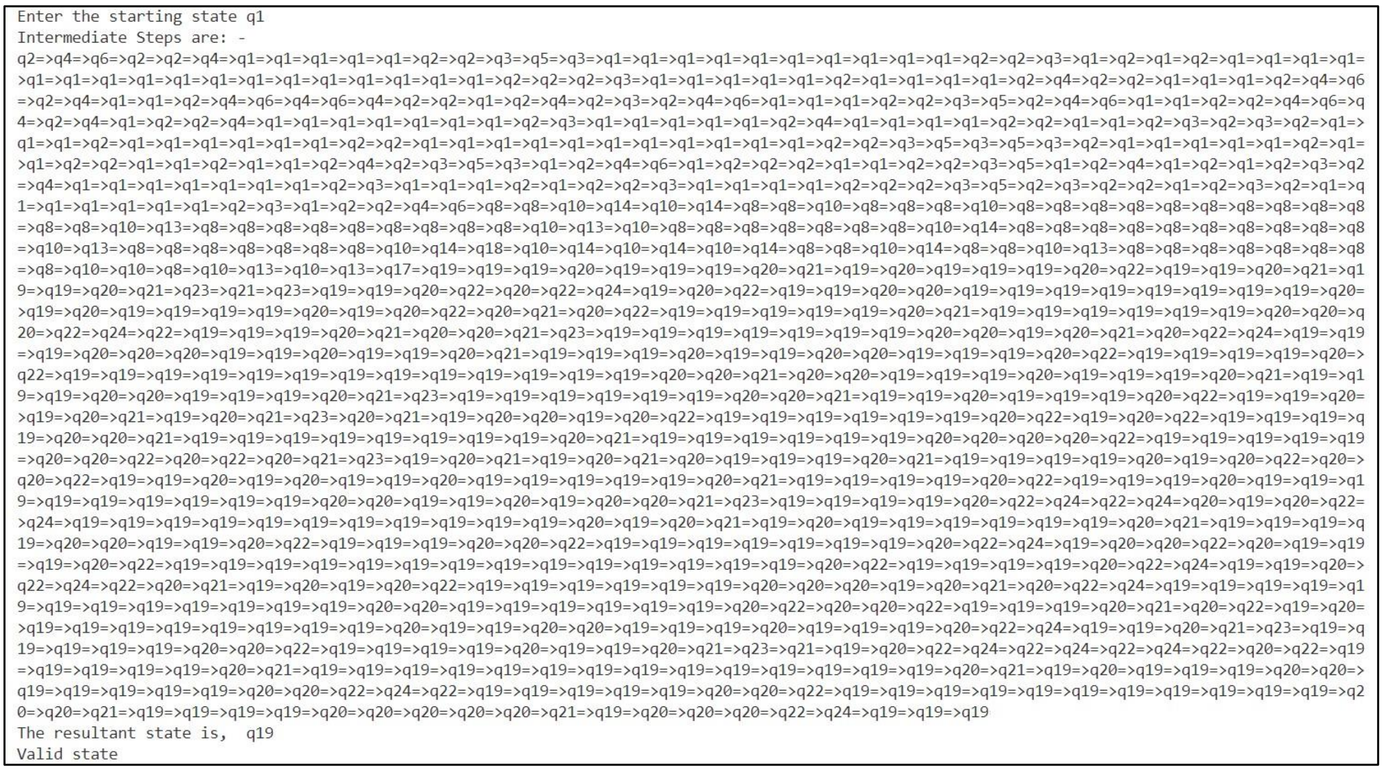
Table 2: Transition Table for the DFA

The DFA has been achieved. Now, this resultant DFA has been tested on 4 examples of DNA Sequences with 1000 base pairs, with varied GC Contents. The results are produced below.

Example 1: The DNA sequence with GC content of 0.5 used for testing is: -

AGGAAGTCTCCAATTTCTTGTTTCCGAATGACACGCGTCTCCTTGCGGGTAAATCGCCGACCGCAGAACTTAGGA
GCCAGGGGGAACAGATAGGTCTAATTAGGTTAAGGGAGTAAGTCCTCGGATGGTTCAGTTGTAACCATATACTTA
CGCTGGAACTTCTCCGGCGAATTTTTACTGTCACCAACCACGAGATTTGAGGTAAACCAATTGAGCACATAGTCG
CGCTATCCGACAATCTCCAAATTATAACATACCGTTCCATGAAGGCCAGAGTTACTTACCGGCCCTTTCCATGCGC
GCGCCATACCCCCCCAGTTCCCCGGTTATCTCTCCGAGGAGAGAGTGAGCGATCCTCCGTTAACATATTCTTACGT
ATGACGTAGCTATGTATTTTGCAGAGGTAGCGAACGGGTTTGACACTTCACAGATAGTGGGGATCCGGGCAAAG
GGCGTATAATTGCGGTCCAACATAGGCGTAAACTACGATGGCACCAACTCAGTCGCAGCTCGTGCGCCGTGAATA
ACGTACTCATCCCAACTGATTCTCGGCAATCTACGGAGCGACATGATTATCAACAGCTGTCTAGCAGTTCTAATCT
TTTGCCATCGTCGTAAAAGCCTCCAAGAGATTGATCATACCTATCGGCACAGAAGTGACACGACGCCGATGGGTA
GCGGACTTTTGGTCAACCACAATTCGCTAGGGGACAGGTCCTGCGGCGTACATCACTTTGTATGTGCAACCAGCC
CAAGTGGGGCCAGGCAAGACTCAGCTGGTTCCTGTGTTAGCTCGAGGCTAGGGATGACAGCTCTTTAAACATAGG
CTGGGGGCGTCGAACGGTCGAGAAGCTCATAGTACCTCGGGTACCAACTTACTCAGGCTATTGCTTGAAGCTGTA
CTATTTCAGGGGGGGAGCGCTGATGGTCTCTTCTTCTGATGACTCAACTCGCAAGGGTCGTGAAGTCGGTTCCTTC
AATGGTTAAAAATCAAAGGCTC

The output from the DFA is shown in Figure 5.



```
Enter the starting state q1
Intermediate Steps are: -
q2=>q4=>q6=>q2=>q2=>q4=>q1=>q1=>q1=>q1=>q1=>q2=>q2=>q3=>q5=>q3=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q2=>q2=>q3=>q1=>q2=>q1=>q2=>q1=>q1=>q1=>q1=
>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q2=>q2=>q2=>q3=>q1=>q1=>q1=>q1=>q1=>q2=>q1=>q1=>q1=>q1=>q2=>q4=>q2=>q2=>q1=>q1=>q1=>q2=>q4=>q6
=>q2=>q4=>q1=>q1=>q2=>q4=>q6=>q4=>q6=>q4=>q2=>q2=>q1=>q2=>q4=>q2=>q3=>q2=>q4=>q6=>q1=>q1=>q1=>q2=>q2=>q3=>q5=>q2=>q4=>q6=>q1=>q1=>q2=>q2=>q4=>q6=>q
4=>q2=>q4=>q1=>q2=>q2=>q4=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q2=>q3=>q1=>q1=>q1=>q1=>q1=>q2=>q4=>q1=>q1=>q1=>q1=>q2=>q2=>q1=>q1=>q2=>q3=>q2=>q3=>q2=>q1=>
q1=>q1=>q2=>q1=>q1=>q1=>q1=>q1=>q1=>q2=>q2=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q1=>q1=>q1=>q1=>q1=>q2=>q1=
>q1=>q2=>q2=>q1=>q1=>q2=>q1=>q1=>q2=>q4=>q2=>q3=>q5=>q3=>q1=>q2=>q4=>q6=>q1=>q2=>q2=>q2=>q1=>q1=>q2=>q2=>q3=>q5=>q1=>q2=>q4=>q1=>q2=>q1=>q2=>q3=>q2
=>q4=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q2=>q3=>q1=>q1=>q1=>q2=>q1=>q2=>q2=>q3=>q1=>q1=>q1=>q1=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q1=>q2=>q3=>q2=>q1=>q
1=>q1=>q1=>q1=>q1=>q1=>q2=>q3=>q1=>q2=>q2=>q4=>q6=>q8=>q8=>q10=>q14=>q10=>q14=>q8=>q8=>q10=>q8=>q8=>q8=>q10=>q8=>q8=>q8=>q8=>q8=>q8=>q8=>q8=>q8=>q8
=>q8=>q8=>q10=>q13=>q8=>q8=>q8=>q8=>q8=>q8=>q8=>q8=>q8=>q10=>q13=>q10=>q8=>q8=>q8=>q8=>q8=>q8=>q8=>q10=>q14=>q8=>q8=>q8=>q8=>q8=>q8=>q8=>q8=>q8=>q8
=>q10=>q13=>q8=>q8=>q8=>q8=>q8=>q8=>q8=>q10=>q14=>q18=>q10=>q14=>q10=>q14=>q10=>q14=>q8=>q8=>q10=>q14=>q8=>q8=>q10=>q13=>q8=>q8=>q8=>q8=>q8=>q8=>q8
=>q8=>q10=>q10=>q8=>q10=>q13=>q10=>q13=>q17=>q19=>q19=>q19=>q20=>q19=>q19=>q19=>q20=>q21=>q19=>q20=>q19=>q19=>q19=>q20=>q22=>q19=>q19=>q20=>q21=>q1
9=>q19=>q20=>q21=>q23=>q21=>q23=>q19=>q19=>q20=>q22=>q20=>q22=>q24=>q19=>q20=>q22=>q19=>q19=>q20=>q20=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q20=
>q19=>q20=>q19=>q19=>q19=>q19=>q20=>q19=>q20=>q22=>q20=>q21=>q20=>q22=>q19=>q19=>q19=>q19=>q19=>q20=>q21=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q20=>q
20=>q22=>q24=>q22=>q19=>q19=>q19=>q20=>q21=>q20=>q20=>q21=>q23=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q20=>q19=>q20=>q21=>q20=>q22=>q24=>q19=>q19
=>q19=>q20=>q20=>q20=>q19=>q19=>q20=>q19=>q19=>q20=>q21=>q19=>q19=>q19=>q20=>q19=>q19=>q20=>q20=>q19=>q19=>q19=>q20=>q22=>q19=>q19=>q19=>q19=>q20=>
q22=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q20=>q21=>q20=>q20=>q19=>q19=>q19=>q20=>q19=>q19=>q19=>q20=>q21=>q19=>q1
9=>q19=>q20=>q20=>q19=>q19=>q19=>q20=>q21=>q23=>q19=>q19=>q19=>q19=>q19=>q20=>q20=>q21=>q19=>q19=>q20=>q19=>q19=>q19=>q20=>q22=>q19=>q19=>q20=
>q19=>q20=>q21=>q19=>q20=>q21=>q23=>q20=>q21=>q19=>q20=>q20=>q19=>q20=>q22=>q19=>q19=>q19=>q19=>q19=>q20=>q22=>q19=>q20=>q22=>q19=>q19=>q19=>q
19=>q20=>q20=>q21=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q21=>q19=>q19=>q19=>q19=>q19=>q20=>q20=>q20=>q20=>q22=>q19=>q19=>q19=>q19=>q19
=>q20=>q20=>q22=>q20=>q22=>q20=>q21=>q23=>q19=>q20=>q21=>q19=>q20=>q21=>q20=>q19=>q19=>q19=>q20=>q21=>q19=>q19=>q19=>q19=>q20=>q19=>q20=>q22=>q20=>
q20=>q22=>q19=>q19=>q20=>q19=>q20=>q19=>q19=>q20=>q19=>q19=>q19=>q19=>q19=>q20=>q21=>q19=>q19=>q19=>q19=>q20=>q22=>q19=>q19=>q19=>q20=>q19=>q19=>q1
9=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q20=>q19=>q19=>q20=>q19=>q20=>q20=>q21=>q23=>q19=>q19=>q19=>q19=>q20=>q22=>q24=>q22=>q24=>q20=>q19=>q20=>q22=
>q24=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q19=>q20=>q21=>q19=>q20=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q21=>q19=>q19=>q19=>q
19=>q20=>q20=>q19=>q19=>q20=>q22=>q19=>q19=>q19=>q20=>q20=>q22=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q22=>q24=>q19=>q20=>q20=>q22=>q20=>q19=>q19
=>q19=>q20=>q22=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q22=>q19=>q19=>q19=>q19=>q20=>q22=>q24=>q19=>q19=>q20=>
q22=>q24=>q22=>q20=>q21=>q19=>q20=>q19=>q20=>q22=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q20=>q20=>q19=>q20=>q21=>q20=>q22=>q24=>q19=>q19=>q19=>q19=>q1
9=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q20=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q22=>q20=>q20=>q22=>q19=>q19=>q19=>q20=>q21=>q20=>q22=>q19=>q20=
>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q19=>q19=>q20=>q20=>q19=>q19=>q19=>q20=>q19=>q19=>q19=>q20=>q22=>q24=>q19=>q19=>q20=>q21=>q23=>q19=>q
19=>q19=>q19=>q19=>q20=>q20=>q22=>q19=>q19=>q19=>q20=>q19=>q19=>q20=>q21=>q23=>q21=>q19=>q20=>q22=>q24=>q22=>q24=>q22=>q24=>q22=>q20=>q22=>q19
=>q19=>q19=>q19=>q19=>q20=>q21=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q20=>q21=>q19=>q20=>q19=>q19=>q19=>q20=>q20=>
q19=>q19=>q19=>q19=>q19=>q20=>q20=>q22=>q24=>q22=>q19=>q19=>q19=>q19=>q19=>q20=>q20=>q22=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q19=>q2
0=>q20=>q21=>q19=>q19=>q19=>q19=>q20=>q20=>q20=>q20=>q20=>q21=>q19=>q20=>q20=>q20=>q22=>q24=>q19=>q19=>q19
The resultant state is,  q19
Valid state
```

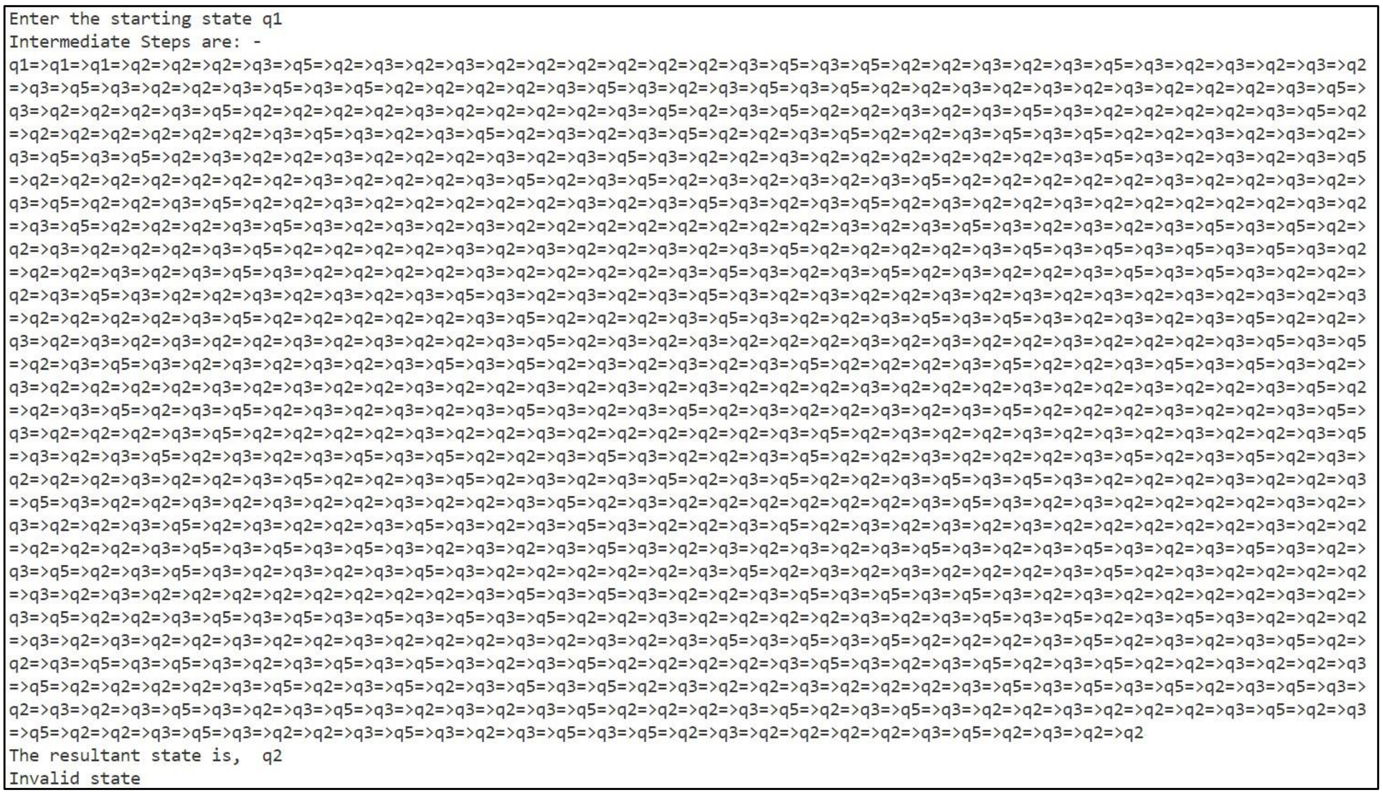
Figure 5: The output generated by the DFA on Example 1

The resultant state achieved is q19. Since, q19 is a final state, the DNA sequence contains the language for which the DFA is created and it is a valid string.

Example 2: The DNA sequence with GC content of 0 used for testing is: -

TTTAAATTATATAAAAAATTTTAATATTTATATATTTAATTTTAAAATTTATTTTAATATATAAATTTAAATTAAAA
TAAAATTATTAATATTTAAAATTAAAAAAATTTATTATATTAATTAATTTTAATATATTTTATAATAAATATTTAAT
AAAAAATTTATATTAAAAAAATAAATTATTATATATTAAAAATAATATTAATTAATAAAAATATTTATTATAATA
AAAATATTAAATTTATATAAAAAAAATATTTATATTTTAATAAATTAAAATATAATATTAAAATTTTTTTTTAAAT
ATTTAAAATAAAATTTATTATAATTTTTAAATTTAATATATTTATATTTATAATATATATATATAAAATTAAAAATT
AAATTTAATTTTTATATTAATATATAATATAATTATATAAATATAATAAATTTTATTTATATATTTTATATATTAAA
TTAATTTTTATAAAATATAATAATATATAAATAAATAATAATTAATTATTATATATTTATTATAATATTAAATAAT
TTAAATTAAAATAATAAAAATTATAATATATAATTTATTATATTTTAATTTAATTAATAAATTATTATAAATAATT
AATTATATTATTAATTTTTAAATAATTTAATATAATAATTATAAAAAATTTATAAAATATAATTATAATTTATTTA
ATTATATATAAAAATAAAAATTTTTTTATATTTATATATTTATTTATTTATTATTTATATTTAAAAATTATATAAAT
TATAAAATATAAAAAAAATTTTTAATTTTTTTATAAAATATTAATTTTTTTTTTAATAAAATATTTTATTTAAATAT
AATAATAAATATATTTTTTAAATTATATTAATTTTTATTTTTATTAAAATTTATTATTAATAATTAAAATTATTATTT
TATAATAAATTTTTTATTTATATTTATTTATATTAAATTATTTAATAAATTATTAATTTAATTTATTTTATAAAATTA
TAA

The output from the DFA is shown in Figure 6.



```
Enter the starting state q1
Intermediate Steps are: -
q1=>q1=>q1=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q5=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q2
=>q3=>q5=>q3=>q2=>q2=>q3=>q5=>q3=>q5=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q3=>q5=>q2=>q2=>q3=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q3=>q5=>
q3=>q2=>q2=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q3=>q2=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q2=>q2=>q2=>q3=>q5=>q2
=>q2=>q2=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q3=>q5=>q2=>q2=>q3=>q2=>q3=>q2=>
q3=>q5=>q3=>q5=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q5
=>q2=>q2=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q2=>q3=>q2=>
q3=>q5=>q2=>q2=>q3=>q5=>q2=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q3=>q2
=>q3=>q5=>q2=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q5=>q3=>q5=>q2=>
q2=>q3=>q2=>q2=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q2=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q5=>q3=>q5=>q3=>q2
=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q2=>q2=>q2=>q3=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q2=>
q2=>q3=>q5=>q3=>q2=>q2=>q3=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q2=>q3=>q2=>q3=>q2=>q3=>q2=>q3=>q2=>q3=>q2=>q3
=>q2=>q2=>q2=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q2=>q3=>q5=>q2=>q2=>q2=>q3=>q5=>q3=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>
q3=>q2=>q3=>q2=>q3=>q2=>q2=>q3=>q2=>q3=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q3=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q2=>q3=>q5=>q3=>q5
=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q2=>q3=>q5=>q3=>q5=>q2=>q3=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>
q3=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q3=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q3=>q2=>q2=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>q2
=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>
q3=>q2=>q2=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q3=>q2=>q3=>q2=>q3=>q2=>q2=>q3=>q5
=>q3=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q5=>q3=>q5=>q2=>q2=>q3=>q5=>q3=>q2=>q2=>q3=>q5=>q2=>q2=>q3=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q3=>
q2=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q2=>q2=>q3=>q2=>q2=>q3
=>q5=>q3=>q2=>q2=>q3=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q3=>q2=>
q3=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q3=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q2
=>q2=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q3=>q2=>
q3=>q5=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q2=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q2=>q2
=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q5=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q3=>q2=>
q3=>q5=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q5=>q3=>q5=>q3=>q5=>q2=>q2=>q3=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q5=>q2=>q3=>q5=>q3=>q2=>q2=>q2
=>q3=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q2=>q3=>q2=>q3=>q2=>q3=>q5=>q3=>q5=>q3=>q5=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q5=>q2=>
q2=>q3=>q5=>q3=>q5=>q3=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q2=>q3=>q2=>q2=>q3
=>q5=>q2=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q3=>q5=>q3=>q5=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q5=>q2=>q3=>q5=>q3=>
q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q5=>q3=>q2=>q2=>q3=>q2=>q2=>q2=>q3=>q5=>q2=>q3
=>q5=>q2=>q2=>q3=>q5=>q3=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q3=>q5=>q2=>q3=>q2=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q2
The resultant state is,  q2
Invalid state
```

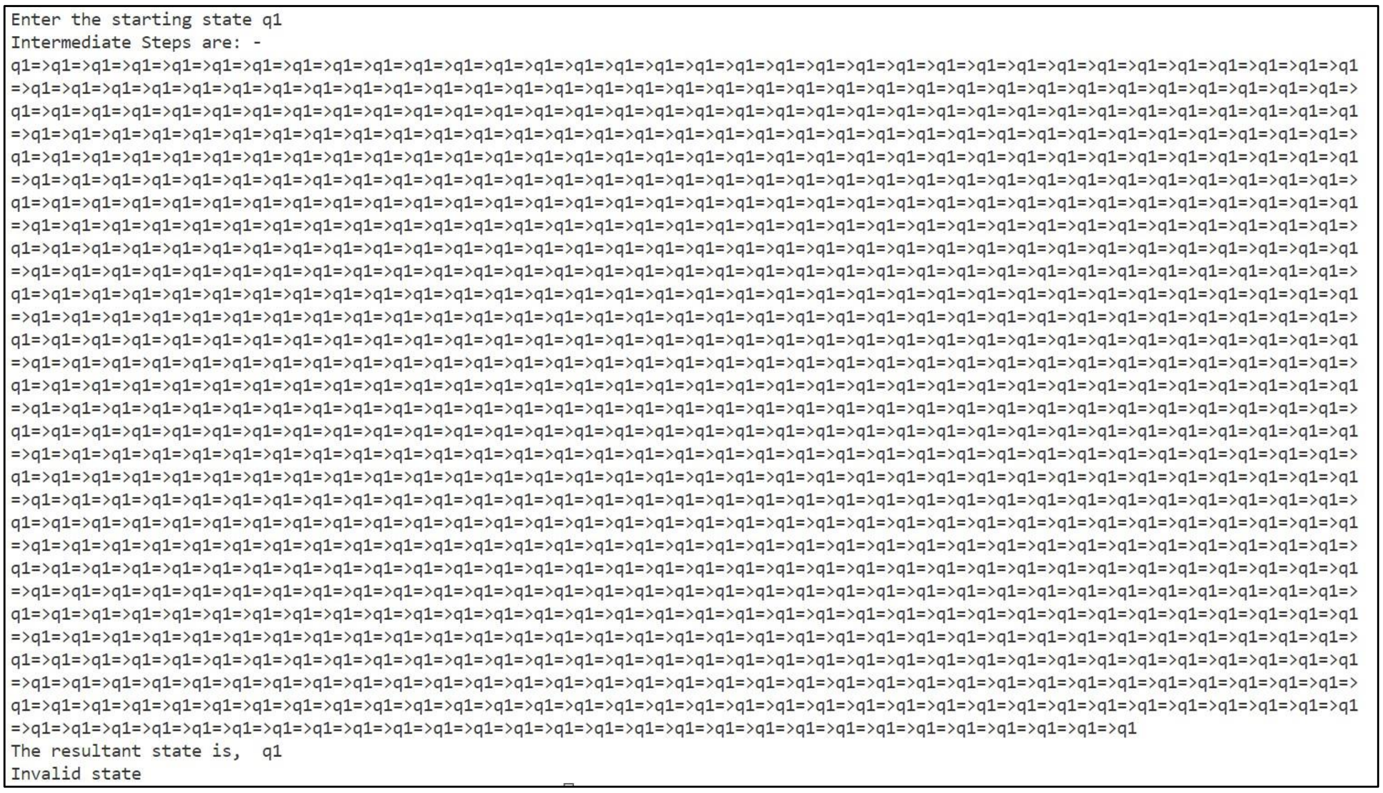
Figure 6: The output generated by the DFA on Example 2

The resultant state achieved is q2. Since, q2 is not a final state the DNA sequence does not contain the language for which the DFA has been created and so it is an invalid string.

Example 3: The DNA sequence with GC content of 1 used for testing is: -

CCGCGCGCGGGGCCGGCGGGCCGGCGGGGGGCGCGGGGCGCCCCCGGGCGCGGCGGGCGGGCGGCCCGGCGCC
CCGGGCGCGGCCGCGCGCCCGGGCCCGGCCCCGCCCCGGGGGCCGCGGGCCCCGGCGCCCGCGGGCCGCGGGGC
CGGCCGGGGGCGCGCGCGGCGGCGCCCGCCGGCCGCCCCCGGGGCGCGGCCCCCCGCCCCGCCGGGGCCCGGCC
CCCGCCGCGGCCGGCCCCCCCCGCGGGCCGCGCCGCCCCGGCGCCCCGGCCGCGCCCCCGCCGCGGGGGCGGGC
CCGCCCGGCCCCGCGCCGCGGCCCCGGCGGGGGCCCGCGCGCCCCGCCCGGGGCGCCGCCGGGGGCCCGGGCC
CCCGGGCGGCGGGGGCCCCGGGCGCGGGCCCCCGCCCGCGGCCCGGCCCGGGCGCGGGCCCGCGCGCCGCCCCG
CCGGGCCCGCCCCGCGCGGGGCGGGCGCGGGGCGCGCCCGGCCCGGCCCCCGGCGCCCCGCGCCGCCGCGCGCC
CGCCGGCCGCCGGCCGGCGGCGCGCGGGCGGCGCCGCGGCCCCCGGGGCCCGGCCCGGCCGCCCCGGGCGCGGC
GCCCCGGGCGGCGCGGGGCGGCGGCGGCCGCCCGGCGCCGGGCGGCGCCCGGCCCGGCGGCGGGGGGCCCGCG
CGGCGCCCCCGCCGGCGCGCCCGGGGCGCGCCCGGCCGGCGCCGGGCGGGCCGGCGCGGGCCCCCGCGCCGGGG
CGGGGGCGGGCGCGCGGGCCGGCGGGGGGCGGGCGGGGGCCCCCGGCCCGCCCGGCGCCCCGCGGCCCCCCCG
GGGGCGGGCGGGGCGCCCCGCGGCCCGGGGGGGGGCCGCCCCGCGGGGCGGGCCCGCGCCGCCCCCCCCGCGG
GGGGCCCGGCGCGCCGGGGGGGCGGGGCGGCCCCGGGCGGCGGCCGCCGGCCGCGGCGGCGGGGCGCCGCCCG
GGGGGCGGGCCCGGCGGCGCGCGGCCCGGCCGGCCGCCCGCCG

The output from the DFA is shown in Figure 7.



```
Enter the starting state q1
Intermediate Steps are: -
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>
q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1=>q1
The resultant state is,  q1
Invalid state
```

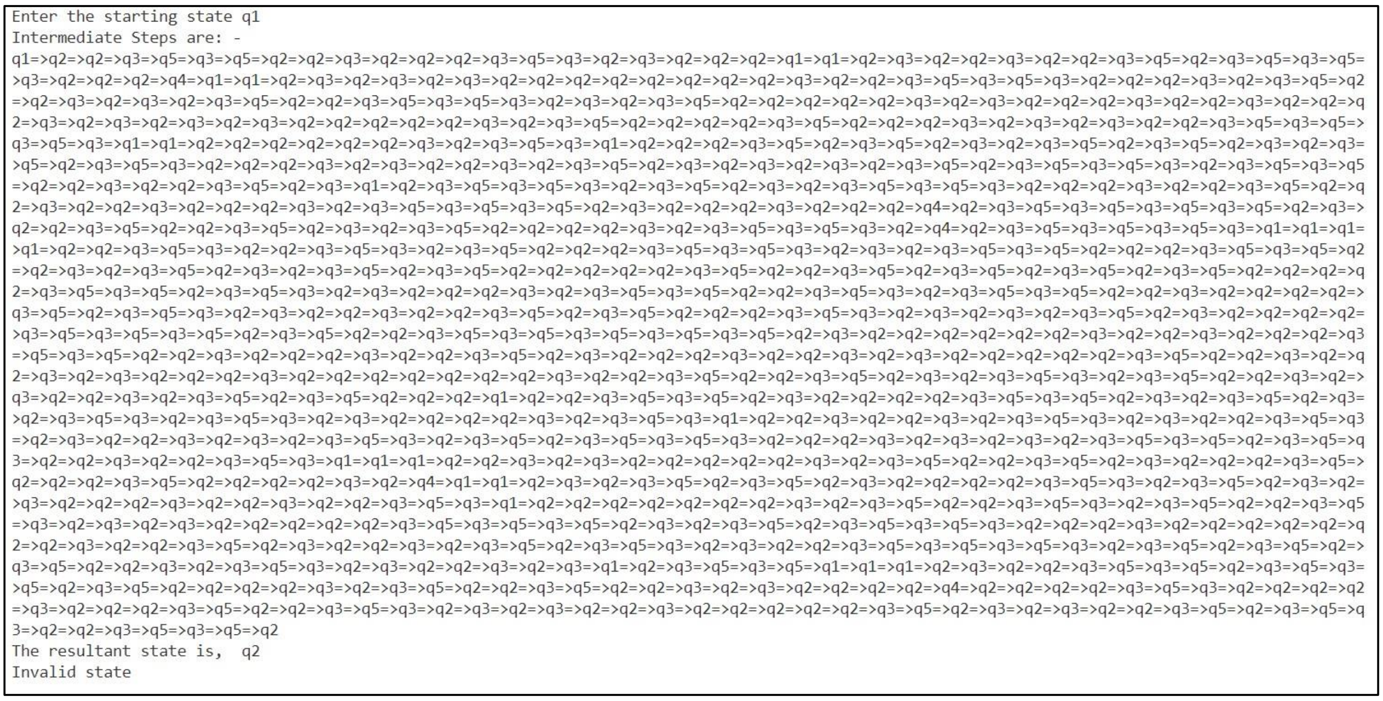
Figure 7: The output generated by the DFA on Example 3

The resultant state achieved is q1. Since, q1 is not a final state the DNA sequence does not contain the language for which the DFA has been created and so it is an invalid string.

Example 4: The DNA sequence with GC content of 0.11 used for testing is: -

TAATTTTAATAAATTTATAAACTATAATAATTATTTTTAAAGTTATATATAAAAAAAATAATTTTTAAATATTAAT
ATATTAATTTTTATATTAAAAATATAAATAATAAATATATATAAAAATATTAAAATTAAATATATAATTTTTTTGT
AAAAAATATTTCAAATTATTATATTATTATATTATTTAAATATAATATTATATATATTATTTTTATTTTAATAATTA
TCATTTTTATTATATTTTTAAATAATTAATAATAAATATTTTTTATAAATAAAGATTTTTTTTATAATTAATTATATT
AAAATATTTTTAGATTTTTTTGTTTAATTTAATTTATTAAATTTTTATATTTTAAATTTTAATATTATATTATTAAAA
ATTAATTATTATTATTAAAATTTTATTTATAAATATTTTAATTTATTTTAATAAAATTATTTATAATAATTATTAAA
TTTATATATTATAAAATTTTTTATTAATTTTTTTTTTATAAAAAATAATAAATTTTAATAAATAATTATAAATAATA
TAAAAATTAATAATATAAATAAAAAAATAATTAATTATATTTATTAATATAATATTATTAAACAATTTTATAAAAT
TTTATATTATATTTATTTATAAAATATTTCAATAATATTTATAATTTATAATATATTTATTATTTTTAAATATATATT
TTATTTAATAATTTCTTAATATAAAAATATTAATTATAAATTAAATTAAAATAGTTATATTATTATAAAATTTATTA
TATAAATAATAATTTGAAAAAAATATTAATTTATTAATTTATATAAAAATTTTTTATATTATTTTTAAATAAAAAA
ATAATTATAATATTATTTATAATTTTTTTTATTATTATTAATATTTATAATATGATTTTGTTATAATTTTATTTTATTA
AAATATTAATTAATATAAAAGAAAATTTAAAATAAATTAATTTATATAATAAAAATTATATAATTATTTAATTTT

The output from the DFA is shown in Figure 8.


```
Enter the starting state q1
Intermediate Steps are: -
q1=>q2=>q2=>q3=>q5=>q3=>q5=>q2=>q2=>q3=>q2=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q2=>q2=>q1=>q1=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>q2=>q3=>q5=>q3=>q5=
>q3=>q2=>q2=>q2=>q4=>q1=>q1=>q2=>q3=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q2
=>q2=>q3=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q
2=>q3=>q2=>q3=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q3=>q5=>q2=>q2=>q2=>q3=>q2=>q3=>q2=>q3=>q2=>q2=>q3=>q5=>q3=>q5=>
q3=>q5=>q3=>q1=>q1=>q2=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q1=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q3=>q2=>q3=
>q5=>q2=>q3=>q5=>q3=>q2=>q2=>q2=>q3=>q2=>q3=>q2=>q2=>q3=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q2=>q3=>q2=>q3=>q5=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q3=>q5=>q3=>q5
=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>q2=>q3=>q1=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>q2=>q
2=>q3=>q2=>q2=>q3=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q5=>q3=>q5=>q2=>q3=>q2=>q2=>q2=>q3=>q2=>q2=>q2=>q4=>q2=>q3=>q5=>q3=>q5=>q3=>q5=>q3=>q5=>q2=>q3=>
q2=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q4=>q2=>q3=>q5=>q3=>q5=>q3=>q5=>q3=>q1=>q1=>q1=
>q1=>q2=>q2=>q3=>q5=>q3=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q5=>q3=>q5=>q2=>q2=>q2=>q3=>q5=>q3=>q5=>q2
=>q2=>q3=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q2=>q2=>q
2=>q3=>q5=>q3=>q5=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q5=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q3=>q5=>q2=>q2=>q3=>q2=>q2=>q2=>q2=>
q3=>q5=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q2=>q2=
>q3=>q5=>q3=>q5=>q3=>q5=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q5=>q3=>q5=>q3=>q5=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q2=>q3
=>q5=>q3=>q5=>q2=>q2=>q3=>q2=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q2=>q3=>q2=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q3=>q5=>q2=>q2=>q3=>q2=>q
2=>q3=>q2=>q3=>q2=>q2=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q2=>q3=>q2=>
q3=>q2=>q2=>q3=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q2=>q2=>q1=>q2=>q2=>q3=>q5=>q3=>q5=>q2=>q3=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q5=>q2=>q3=>q2=>q3=>q5=>q2=>q3=
>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q1=>q2=>q2=>q3=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q2=>q3=>q5=>q3
=>q2=>q3=>q2=>q2=>q3=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q2=>q2=>q3=>q2=>q3=>q2=>q3=>q2=>q3=>q5=>q3=>q5=>q2=>q3=>q5=>q
3=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>q3=>q1=>q1=>q1=>q2=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q2=>q3=>q5=>
q2=>q2=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q3=>q2=>q4=>q1=>q1=>q2=>q3=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q3=>q2=
>q3=>q2=>q2=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q3=>q5=>q3=>q1=>q2=>q2=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q2=>q3=>q5
=>q3=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q5=>q2=>q3=>q2=>q3=>q5=>q2=>q3=>q5=>q3=>q5=>q3=>q2=>q2=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q
2=>q2=>q3=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q2=>q3=>q2=>q3=>q5=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q2=>q3=>q5=>q3=>q5=>q3=>q5=>q3=>q2=>q3=>q5=>q2=>q3=>q5=>q2=>
q3=>q5=>q2=>q2=>q3=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q2=>q3=>q2=>q3=>q1=>q2=>q3=>q5=>q3=>q5=>q1=>q1=>q1=>q2=>q3=>q2=>q2=>q3=>q5=>q3=>q5=>q2=>q3=>q5=>q3=
>q5=>q2=>q3=>q5=>q2=>q2=>q2=>q2=>q3=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q2=>q2=>q3=>q2=>q3=>q2=>q2=>q2=>q2=>q4=>q2=>q2=>q2=>q2=>q3=>q5=>q3=>q2=>q2=>q2=>q2
=>q3=>q2=>q2=>q2=>q3=>q5=>q2=>q2=>q3=>q5=>q3=>q2=>q3=>q2=>q3=>q2=>q2=>q3=>q2=>q2=>q2=>q2=>q2=>q3=>q5=>q2=>q3=>q2=>q3=>q2=>q2=>q3=>q5=>q2=>q3=>q5=>q
3=>q2=>q2=>q3=>q5=>q3=>q5=>q2
The resultant state is,  q2
Invalid state
```
Figure 8: The output generated by the DFA on Example 4

The resultant state achieved is q2. Since, q2 is not a final state the DNA sequence does not contain the language for which the DFA has been created and so it is an invalid string.

IV. Conclusion and Future Scope

Thus, the NFA is converted to a DFA and then the DNA sequence is used as a string on the automata. This can have a variety of applications where there is a need to check for the presence of a particular sequence in a DNA pattern and where normal string-matching algorithm fails to operate due to its rigid nature.

The pattern matching of the DNA can be directly applied to the NFA as well but the algorithm would then have to traverse each and every new state that is generated from the previous state on a certain input. Due to its Non-Determinism, it would often give a large number of states and traversing each of them would increase the time complexity. However, upon converting it to DFA, although the number of states has increased while traversing the large sequence, due to its deterministic nature it traverses only the required states one by one, thereby making it efficient in comparison to the traversal on NFA.

In future, the NFA can be prepared in such a manner which checks for such sequences which are responsible for certain diseases. When real life sequences are used, it can easily assert the presence or absence of any genetic diseases in that sequence. Also, it can be extended to look for any mutations present in the DNA using suitable NFA. Moreover, this process can be applied on RNA as well.

At present the Python code that has been implemented to convert the NFA to DFA does not consider the lambda transitions that are possible in NFA, which in future can be updated. As a result, the presence of lambda transitions while making the NFA would definitely ease up the process of creating the automata for a given language.

References

1. An Introduction to Formal Languages and Automata, Author: - Peter Linz.
2. Introduction to Automata Theory, Languages, and Computation, Author: - John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman.
3. DNA Pattern Analysis using Finite Automata. Author: - Qura-Tul-Ein, , Yousaf Saeed, Shahid Naseem, Fahad Ahmad, Tahir Alyas, Nadia Tabassum in International Research Journal of Computer Science (IRJCS) Volume 1 Issue 2 (October 2014).
4. Blanton, M., & Aliasgari, M. (2010). Secure Outsourcing of DNA searching via Finite Automata. DBSec'10 Proceedings of 24th annual IFIP WG 11.3 working conference on Data and applications security and privacy (p. 16). Heidelberg: Springer.
5. Krogh, A. (1998). Gene finding: putting the parts together. Guide to Human Genome Computing, 13.
6. Burge, C., & Karlin, S. (1997). Prediction of complete gene structure in Human Genomic DNA. JMB, 16.
7. Lawson, M. V. (2009). Finite Automata. Edinburg: CRC Press.
8. https://www.javatpoint.com/deterministic-finite-automata
9. https://www.javatpoint.com/non-deterministic-finite-automata
10. https://en.wikipedia.org/wiki/DNA

AUTHOR(S) PROFILE

Dr. Asoke Nath, is Associate Professor in Department of Computer Science, St. Xavier's College (Autonomous), Kolkata. Apart from his teaching assignment he is involved with various researches in cryptography and network security, Visual Cryptography, Steganography, Mathematical modelling of Social Networks, Green Computing, Big data analytics, MOOCs, Quantum computing, e-learning. He has already published more than **272** papers in reputed Journals and conference proceedings. Dr. Nath is the life member of MIR Labs (USA), CSI Kolkata Chapter.

Priyanshu Acharya, is a student of B.Sc. Computer Science, St. Xavier's College (Autonomous), Kolkata. Currently he is doing his research work on Finite State Automata and Machine Learning.

How to cite this article?

Dr. Asoke Nath, Priyanshu Acharya. Application of Finite State Automaton in a Random DNA Sequence and Pattern Analysis. International Journal of Advance Research in Computer Science and Management Studies. 2023; 11(11): 10-19. DOI: https://doi.org/10.61161/ijarcsms.v11i11.2