FEATURE EXTRACTION AND ANALYSIS OF LARGE BIOLOGICAL DATA

Submitted By

Md. Imranul Islam

(ID: 2013-1-60-034)

Umme Kulsum Binti

(ID: 2013-1-60-035)

&

Md. Abdul Kuddus

(ID: 2013-1-60-049)

Supervised by

Dr. Shamim H Ripon

Associate Professor

Department of Computer Science & Engineering

East West University

Spring 2017

FEATURE EXTRACTION AND ANALYSIS OF LARGE BIOLOGICAL DATA

Submitted By

Md. Imranul Islam

(ID: 2013-1-60-034)

Umme Kulsum Binti

(ID: 2013-1-60-035)

&

Md. Abdul Kuddus

(ID: 2013-1-60-049)

Department of Computer Science and Engineering

East West University

In Particular Fulfillment of the Requirements for the degree of

B.Sc.in Computer Science & Engineering

Supervised by

Dr. Shamim H Ripon

Associate Professor

Department of Computer Science and Engineering

East West University

Declaration

We hereby declare that this submission is our own work and that to the best of our knowledge and belief it contains neither material nor facts previously published or written by another person. Further, it does not contain material or facts which to a substantial extent has been accepted for the award of any degree of university or any other institution of tertiary education except where an acknowledgement.

(Md. Imranul Islam)

(Umme Kulsum Binti)

(Md. Abdul Kuddus)

Letter of Acceptance

The Thesis paper entitled "**Feature Extraction and Analysis of Large Biological Data**" submitted by Md. Imranul Islam (ID: 2013-1-60-034) , Umme KulSum Binti (ID: 2013-1-60-035) and Md. Abdul Kuddus (ID: 2013-1-60-049), to the Department of Computer Science & Engineering, East West University, Dhaka, Bangladesh is accepted by the department in partial fulfillment of requirements for the Award of Degree of Bachelor of Computer Science & Engineering on Spring 2017.

Board of Examiners

Dr. Shamim H Ripon
Associate Professor
Department of Computer Science & Engineering
East West University, Dhaka-1212, Bangladesh

Dr. Ahmed Wasif Reza
Associate Professor & Chairperson (Acting)
Department of Computer Science & Engineering
East West University, Dhaka-1212, Bangladesh

ABSTRACT

DNA sequencing is the process of determining the sequence of nucleotide bases (adenine, guanine, cytosine, and thymine) in a piece of DNA that is represented as A, G, C and T respectively. Today, with the right equipment and materials, sequencing a short piece of DNA is relatively straightforward. The advent of rapid DNA sequencing methods has greatly accelerated biological and medical research and discovery.

Knowledge of DNA has become indispensable for basic biological research, and in numerous applied fields such as medical diagnosis, biotechnology, forensic biology, virology and biological systematics. The rapid speed of sequencing and searching attained with modern DNA pattern searching technology has been instrumental. Nowadays analysis of large biological dataset using searching a pattern from DNA needs faster and cost effective machines to attain more accurate result within a short time. It is so much difficult to handle large set of biological data. Searching a specified pattern is now automated and works faster within a short piece of sequenced DNA. In this paper, modified version of Clustering and KMP algorithm is used to search a specific pattern in a large DNA dataset. Overall process also includes the total number of matching pattern found in DNA dataset.

TABLE OF CONTENT

Chapter 1

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Chapter 6

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENT

Firstly, I would like to express my gratitude to omnipotent and almighty Allah, whose invisible guidance helped me to complete this thesis paper, I would like to express my gratitude towards my supervisor, Dr. Shamim H Ripon, Associate Professor of Department of Computer Science and Engineering, East West University for being a perfect guide with incredible insight in all aspects regarding my thesis topic. Without his proper guidance, encouragement and support this thesis would have remained dream. I am also indebted to my parents, other professors of the department and friends for the support and encouragement regarding my thesis work.

CHAPTER 1

INTRODUCTION

1.1 Introduction

Manual biological data analysis, processing are an active research domain. Massive amount of the biological data are produced and stored continuously in biological studies. Information retrieving and analysis from these large sizes of generated data is challenging issues as well as key factor for successful knowledge discovery.

Diversity of human, animal and plant life/behavior is one of the basic reasons behind the availability of the large biological dataset. DNA sequence is a big part of biological data. The length of DNA sequence of data can be huge.

DNA is a molecule that contains the instructions an organism needs to develop, live and reproduce. These instructions are found inside every cell, and are passed down from parents to their children. DNA is made up of molecules called nucleotides. Each nucleotide contains a phosphate group and a nitrogen base. The four types of nitrogen bases are adenine (A), thymine (T), guanine (G), and cytosine (C). The order of these bases is what determines DNA's instructions [1].

For searching DNA sequence we applied here two methods such as Naïve search and clustering the data using modified K-means clustering algorithm. The naïve method for DNA sequence works by checking each element of the sequence and the large dataset. The naïve method actually checks the n-m position where "n" is the length of the dataset and "m" is the length of the DNA sequence. It checks the first character of the sequence against the first of the dataset; if there is a match, the search continues with the second character of each string, but if there is no match, the sequence is shifted along the dataset, and the attempt to match begins again (with the first of the pattern against the second of the text). This process continues until either the whole DNA sequence is found in the dataset, or until it has come to the end of the dataset (actually, the 'end' described

here would be the (n-m) position, as explained earlier). After finding an entire match with the DNA sequence, the whole process begins again, at the next position in the dataset.

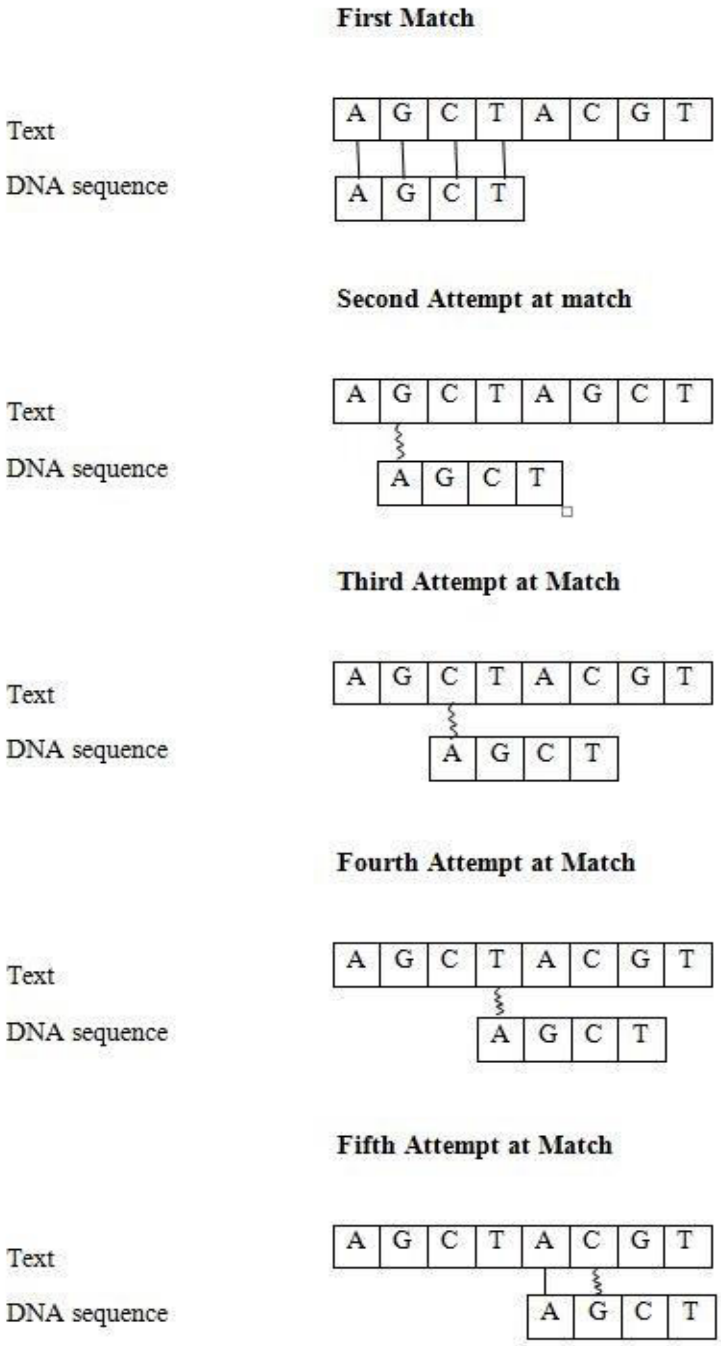


Figure 1.1: Steps of Naïve search method

Initially, the first of the part tern is matched to the first of the string, then the second, third, fourth and fifth is also matched. Here, the DNA sequence AGCT matches, so it is then shifted along one. No match: sequence is shifted along again. This continues until the other match is found.

Therefore, the worst case running time of this algorithm is achieved when each of the characters in the sequence must be compared to each of the characters in the dataset, except for the last few of length of the sequence. For example, if both sequence and original data are include only one letter such as searching for GGGG in GGGGGGGGG. In this case, it is – as expected – quite long: Θ ((n-m+1) m). ["Θ" denotes the worst-case running time, and "n" and "m" are again the lengths of the dataset and sequence respectively.] This means that the running time is effectively proportional to the length of the text multiplied by the length of the pattern [2].

The naïve search-matching algorithm should take considerably longer than other methods. It is so inefficient because it does not take note of the information it finds when matching for the first time, when matching subsequent times.

Searching DNA sequences from a large data is usually time-consuming because of redundant data. For this purposes we used feature extraction which is a process of extracting new set of reduced features from original features based on some attributes transformation. In real world large DNA dataset consists of irrelevant, redundant, noisy data, so before applying clustering algorithms on these data set. One must consider reduction as pre-processing step. Many different feature extraction methods exist and they are being widely used. All these methods aim to remove redundant and irrelevant features. So that clustering and occurrence of new instances will be more accurate [3].

In our proposed module we use modified K-means clustering algorithm to cluster the data in different group. K-means clustering algorithm is an unsupervised learning which will create different cluster of inputs and will be able to put any new input in appropriate cluster. K-means clustering is a popular clustering algorithm based on the partition of data. Though K-means cluster works better for numerical data but it plays an important role for DNA data clustering that overcomes the searching time complexities and occurrences of DNA sequence [4].

1.2 Motivation

With the presence of modern biotechnology, researchers have been able to determine the actual sequence of the roughly three billion bases of DNA (A,T,C,G) that make up the human genome. They have sequenced the genomes of many other types of creatures as well. Scientists have tried to use this new DNA data to find similarities in the DNA sequences of creatures. While the genome of each created kind is unique, many animal kinds share some specific types of genes that are generally similar in DNA sequence.

Another important aspect to search DNA sequence fast from large biological data is to ensure that the number of occurrences of a sequence is more than the number of sequences in a random dataset of the same length. This would make the sequence statistically significant.

Using naïve search, finding similarities and number of occurrences of DNA sequence will not be a good solution because it takes long time to search the sequence as we mentioned before. So because of that reasons we proposed a model based on modified K-means clustering algorithm. Modified K-means partitioned the dataset in small group of data so that it helps to search the sequence fast.

1.3 Objectives

The specific objectives of the project are as follows:

- Partitioning the data into pre-defined number of cluster using modified K-means clustering algorithm.
- Improving the searching time by reducing and find out the number of occurrences of DNA sequences.
- Comparisons the searching time difference between Naïve search and modified K-means algorithm.

1.4 Contribution

Contributions in this project are as follows:

We have used the concepts of clustering for large biological data. Here an algorithm has proposed based on clustering for searching the sequence of DNA data fast.

K-means clustering algorithm is efficient because it cluster the large biological data in different group so the size of data is reduced. Initially we clustered the DNA data in 4 segments. But the segment can be increased as 8 segments and 16 segments so that the data will be grouped in small sizes to search easily.

But before clustering the data, we remove the irrelevant data to extract the features so that the length of the data will be reduced.

After that we compared the searching time of Naive search and modified K-means clustering algorithm. We also compared the number of occurrences of specified pattern.

1.5 Outline

Chapter 1: This chapter represents about the motivation to work, specify the objectives and then the contribution that we have made.

Chapter 2: This chapter illustrated about the background of DNA data and K-means clustering algorithm.

Chapter 3: Proposed model of our project. Here we will show the architectural view of the project.

Chapter 4: Implementation and flow chat of the project. The process of implementation some short code and algorithm are described.

Chapter 5: Result and analysis. This chapter is about analysis of time and occurrence of DNA sequence and comparisons time and occurrence between Naive search and K-means clustering algorithm.

Chapter 6: At last in this chapter summarization of the work and give definitions about future plan of the project.

CHAPTER 2

BACKGROUND

2.1 DNA Sequencing

We know that the nucleus is a small spherical, dense body in a cell of any body. It is often addressed as the "control center" because it has every control of the cell which includes cell reproduction, and heredity. Recall that chromosomes which look like microscopic, threadlike strands structured of the chemical DNA (short for deoxyribonucleic acid). In a word, DNA manages the production of a body's protein. These proteins actually form the basic units of cells and manage all chemical activities within the cell. We can think of a building made by bricks one by one rearranged, like that proteins form your skin, your hair, parts of individual cells. Even how you will be looked like also determined by the proteins. These proteins have a unique sequence which defines the structure of the body.

There are some researches ongoing for DNA sequencing as it is very complex process. DNA sequencing involves the determination of the order of DNA bases. In a strand of DNA, there are some simple units known as nucleotides. They include sugar, phosphate and base. They stay together as the whole part of the DNA. There is an order of the bases of protein which determines the genetic code of a body structure [1].

There are four kinds of bases that hold the total information of our body. It actually holds the blueprint of the body. It has the very important and unique genetic information. These bases are:

- Adenine (A)
- Thymine (T)
- Cytosine (C)
- Guanine (G)

There are bonds between the base pair which pair holds the information of genome system of a body. There are billions of base pair in a body so it is quite a challenging task to sequence the dataset more efficiently. [5]

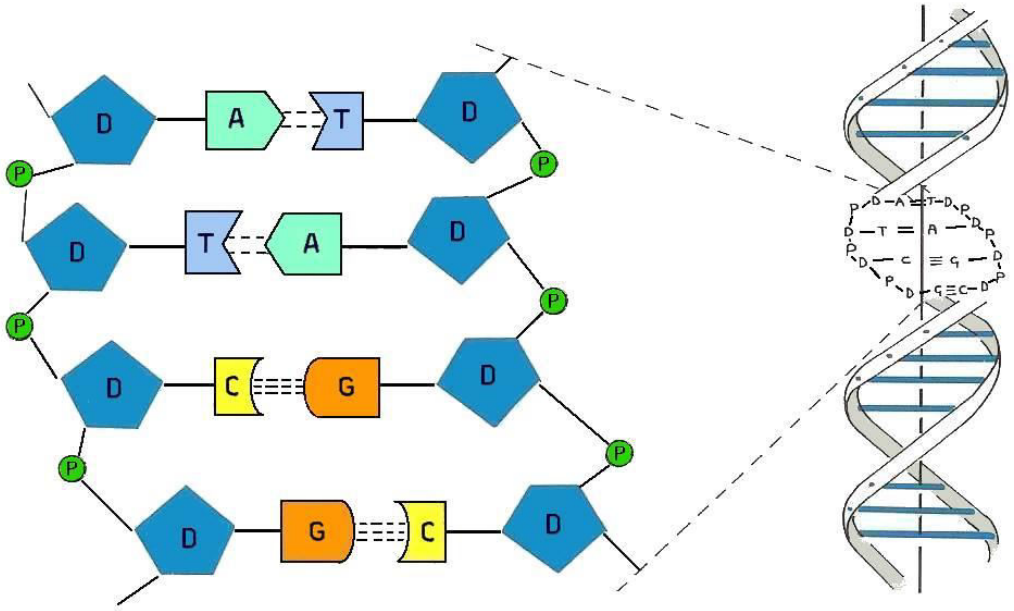


Figure 2.1: DNA structure with four nucleotides

This is a very complex task sequencing the genome. For the process it is needed to break the genome in smaller parts for the sequences and assembling those sequences into a single sequence "consensus." For the new technologies and methods it is now very useful and efficient for the time and speed where genome sequencing is getting very easier from the old methods mostly for the human genome. For these kinds of sequence it is extracted as a string from the entire dataset. Then it will form how the sequences are holding together to structure the whole body and hold the information of the protein of the body.

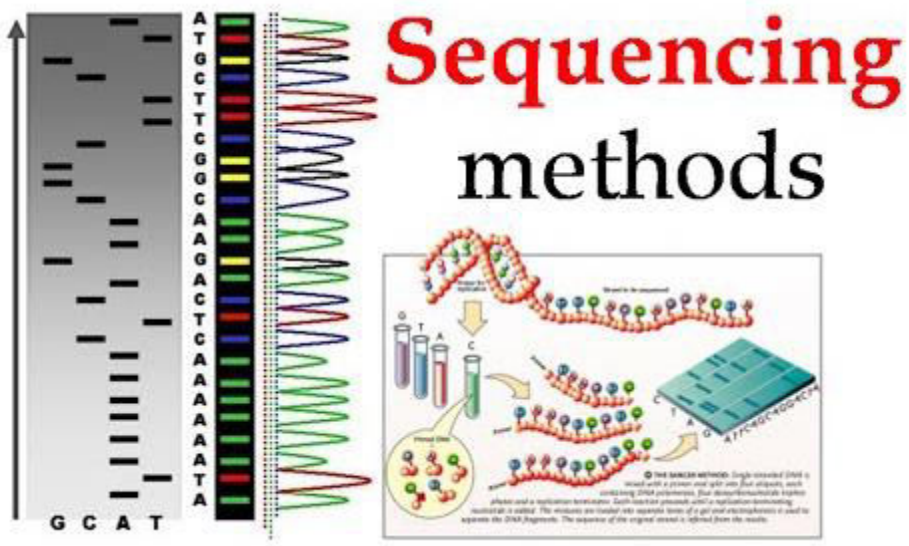


Figure 2.2: Sequencing methods demonstration

Early efforts at sequencing genes were painstaking, time consuming, and labor intensive, such as when Gilbert and Maxam reported the sequence of 24 base pairs using a method known as wandering-spot analysis (Gilbert &Maxam, 1973). With many technologies and methods some processes of sequencing the DNA is being processed under some practical solutions. For the large biological dataset of DNA it may be quite complex to acquire the actual information but the sequencing of DNA is now becoming easy for first generation sequencing and other technologies. However, it is quite important what is the actual speed of parallelization and improvement of automation and speed [6].

2.2 Cluster Analysis

Cluster analysis divides some data into some specific group of dataset to earn some goal which should be meaningful. If these clusters hold the goal, then there must be the nature of the data in those clusters. In summarization, these cluster analysis is becoming very important issue for handling large dataset. Understanding the basic

terms of utility cluster analysis maintain some specific units like psychology and other social sciences, biology, statistics, pattern recognition, information, machine learning, and data mining [7].

Actually cluster analysis divides the data into some groups for identical information to extract some specific nature from the dataset. In those groups, each group must contains specific characteristics which is different from the other groups. If the similarity is very much less for every group and there are more differences among them then the groups are actually distinct on the clustering.

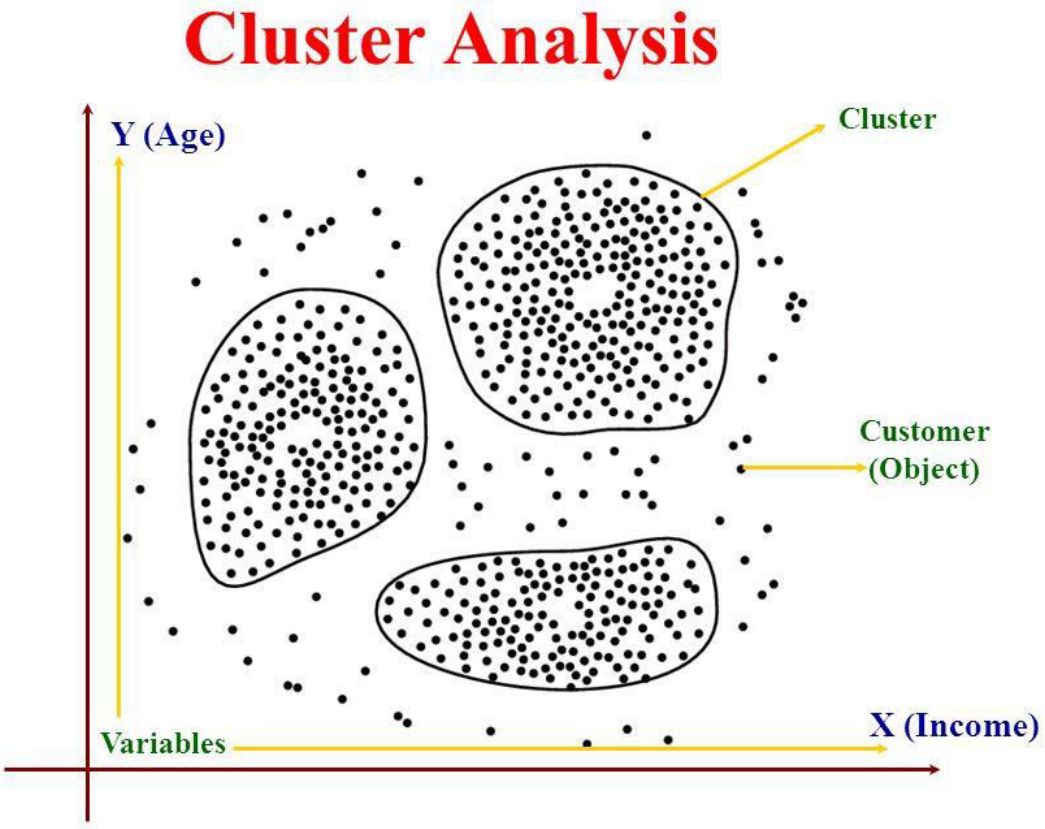


Figure 2.3: Illustration of a cluster analysis example

Actually clustering is very helpful for sorting out some meaningful group from the data. It helps us to get clear about the natural information about the data set. Used either as a

stand-alone tool to get insight into data distribution or as a preprocessing step for other algorithms.

The cluster analysis can't be defines an algorithm itself but it is the basic phrase. There are different kind of clustering algorithm by which the grouping or clustering the data set is very much efficient. Popular notions of clusters include groups with small distances among the cluster members, dense areas of the data space, intervals or particular statistical distributions. Clustering can therefore be formulated as a multi-objective optimization problem. There are different parameters for different clustering algorithm, some are threshold based some are time based which are actually depend on the result or output of clustering. Cluster analysis as such is not an automatic task, but an iterative process of knowledge discovery or interactive multi-objective optimization that involves trial and failure. It is often necessary to modify data preprocessing and model parameters until the result achieves the desired properties [8].

2.3 K-means Clustering

K-means clustering is a traditional, simple machine learning algorithm that is trained on a test data set and then able to classify a new data set using a prime, number of clusters defined a priori.

K-means algorithm groups the data set into some specific identical clusters which are unique from each other. Data is separated in different clusters, which are usually chosen to be far enough apart from each other spatially, in Euclidian Distance, to be able to produce effective data mining results. Each cluster has a center, called the centroid, and a data point is clustered into a certain cluster based on how close the features are to the centroid [8-9].

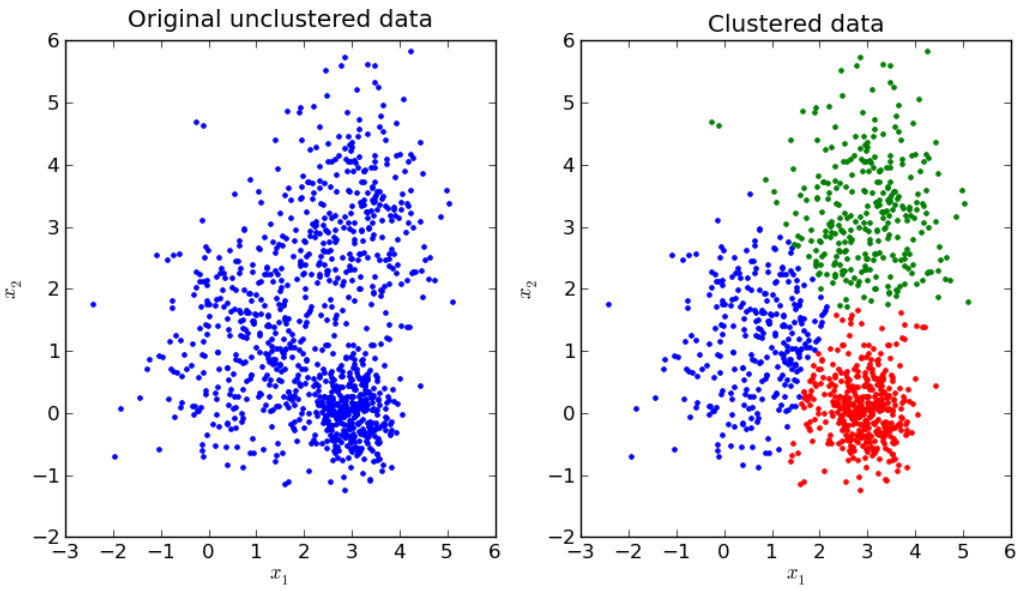


Figure 2.4: A computer-generated program showing k-means clustering

K-means algorithm iteratively minimizes the distances between every data point and its centroid in order to find the most optimal solution for all the data points.

1. K random points of the data set are chosen to be centroids.
2. Distances between every data point and the centroids are calculated and stored.
3. Based on distance calculates, each point is assigned to the nearest cluster
4. New cluster centroid positions are updated: similar to finding a mean in the point locations
5. If the centroid locations changed, the process repeats from step 2, until the calculated new center stays the same, which signals that the clusters' members and centroids are now set [6].

Algorithm 1 Basic K-means Algorithm.

1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** The centroids don't change

Figure 2.5: Algorithm of basic K-means algorithm

Finding the minimal distances between all the points implies that data points have been separated in order to form the most compact clusters possible, with the least variance within them. In other words, no other iteration could have a lower average distance between the centroids and the data points found within them [9].

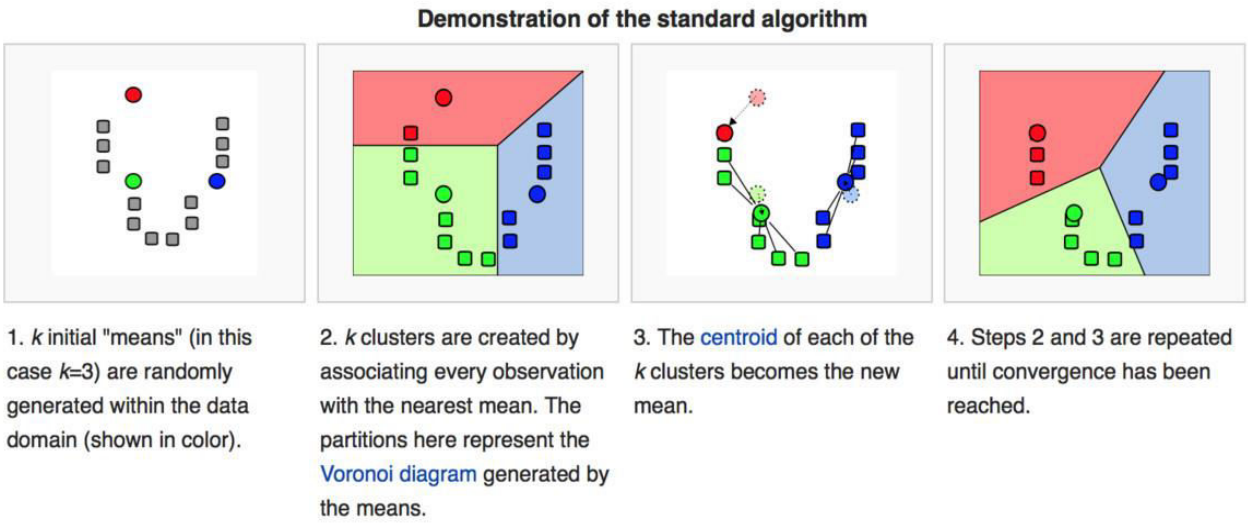


Figure 2.6: A graphical look at k-means clustering

K-Means clustering is a fast, robust, and simple algorithm that gives reliable results when data sets are distinct or well separated from each other in a linear fashion. It is best used when the number of cluster centers, is specified due to a well-defined list of types shown in the data. However, it is important to keep in mind that K-Means clustering may not perform well if it contains heavily overlapping data, if the Euclidean distance does not measure the underlying factors well, or if the data is noisy or full of outliers.

CHAPTER 3

PROPOSED MODEL

3.1 Overview of the System

Now-a-days managing large dataset to find some features for some specific reasons is very challenging. It is a major issue for handling DNA datasets to identify some criteria on it. We are working with the DNA datasets and we mentioned that DNA consists a chain of nucleotide (A, G, C, T) for cell that is different for different creatures. Most importantly, any human body has more than hundred million bases of pairs. There are different modules or algorithms to identify the features on the body cells and there are some tools used for this problem.

Here is a model proposed for representing our works by some steps. This architecture has three phases. Different algorithmic approaches are used in every phases.

- Preprocessing
- Clustering
- Feature Extraction and Analysis

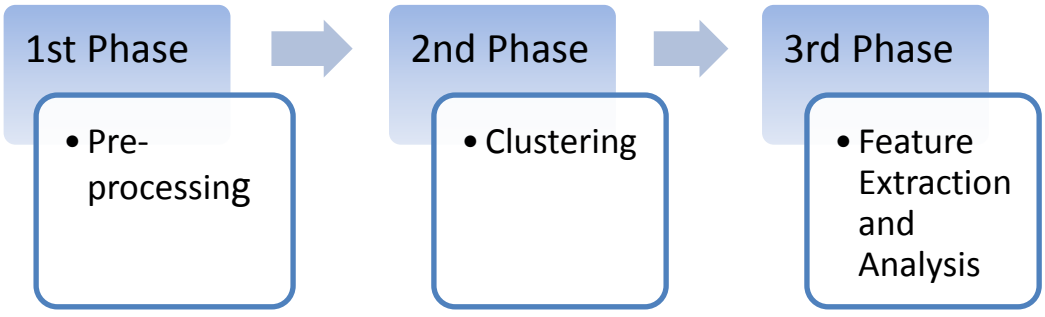


Figure 3.1: Phases of proposed architecture

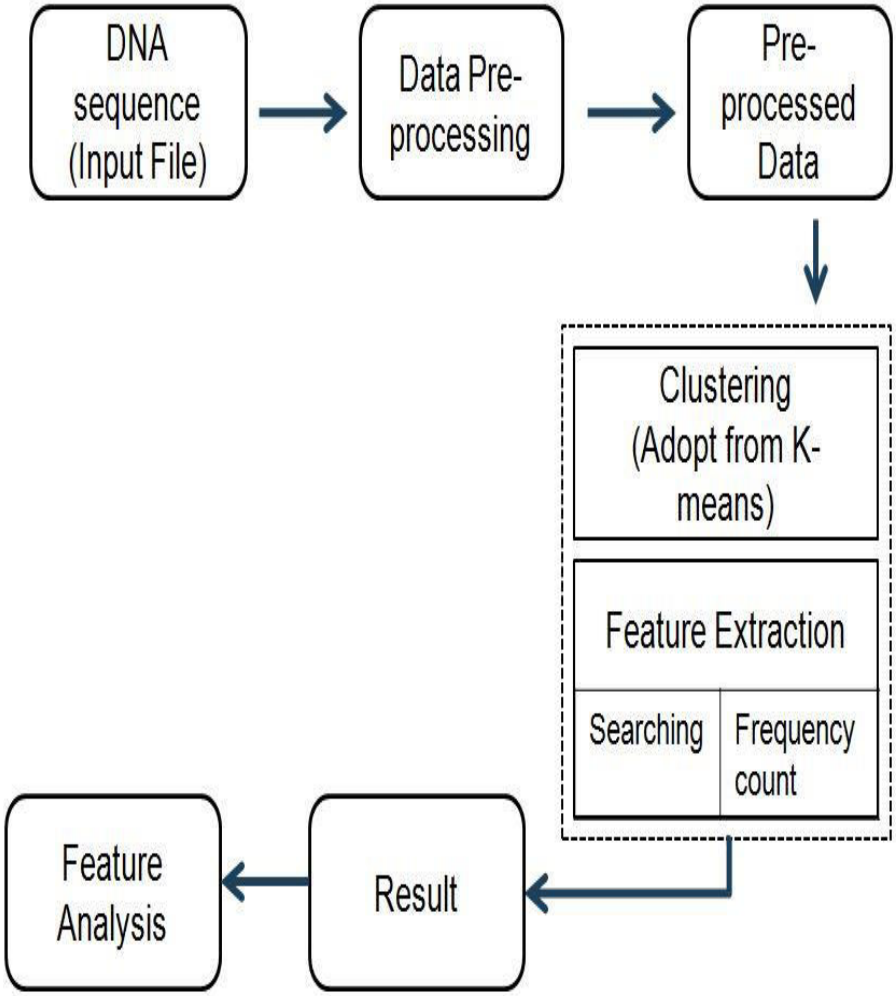


Figure 3.2: Architectural approach for clustering DNA sequence feature extraction and analysis

3.2 Biological Data

DNA data are data collected from biological sources, which are often stored or exchanged in a digital form. These kinds of files are often stored in files or databases. In our project DNA data are DNA base-pair sequences. These kinds of data are represented as digital data in A T G C form.

3.3 Data Preprocessing

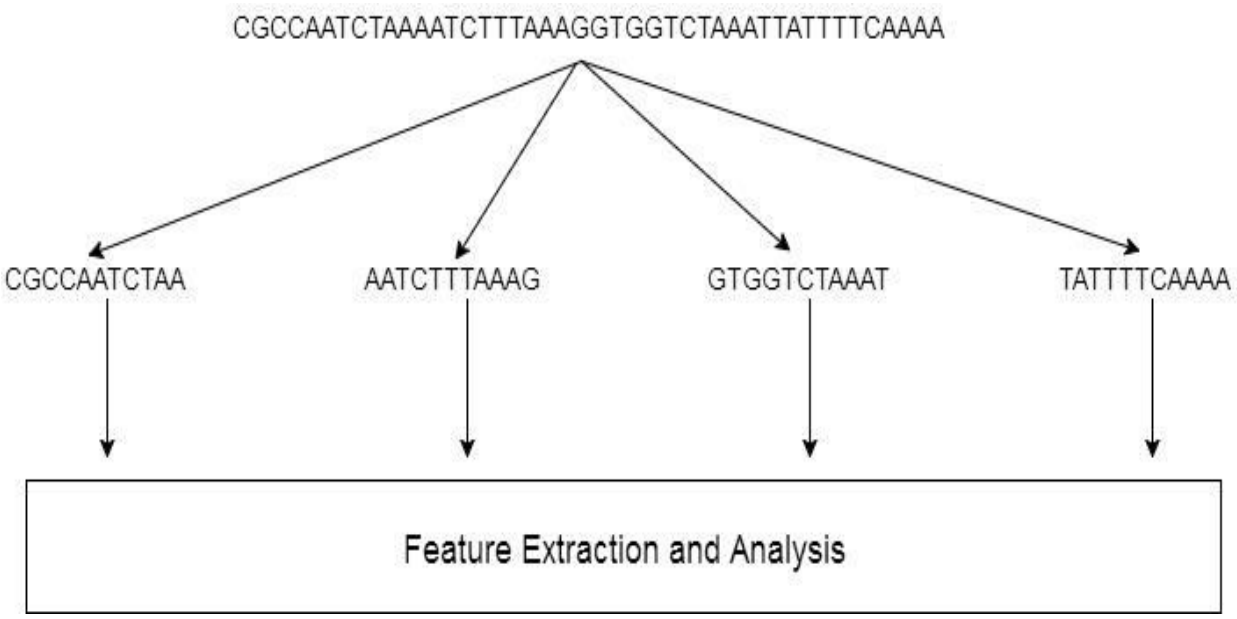
Data preprocessing is very important for eliminating noise from the dataset. There are some common techniques for this preprocessing. After preprocessing the noise data are removed and there will be only the fresh dataset where it has only the DNA base-pair sequences.

Data preprocessing also generates extra DNA base-pair sequence from the original dataset. In this project we have used Java functions to preprocess the dataset. This preprocessing technique removes large number of noise data from the original dataset.

3.4 DNA Data Clustering

It is easy to search the whole dataset by clustering it in some specific cluster size. When the dataset is clustered into the clusters, the system will check the feature DNA sequence cluster wise. There are different cluster sizes to search faster and check the occurrence of the sequence how many times it occurs in the dataset to identify some specific features.

In this project, we take the K-means algorithm for the basic clustering techniques and modified it for our projects. We clustered the DNA dataset with some specific length of cluster and regroup the clusters for some critical situations.



3.5 Feature Extraction

In this phase, we search the specific features for checking whether the dataset has the sequence or not and check how many times the sequence occurs on it. This searching technique works cluster by cluster. For finding the specific feature, it will work faster than the normal search technique where every DNA base pair is checked until the end of the file and on the other hand our proposed technique will be processed until it find the match and for the counting the occurrence the feature on the dataset, it will be processed until the end of the dataset file.

3.6 Feature Analysis

After extracting the features are analyzed for various conclusions. In this phase, we tried to show different perspectives and different view how the statistics look like and the total comparison between our methods in compared to the existing one. We here showed different results we got from the above analysis and make them clear to reach to any destination needed. Here the occurrence of the pattern are computed and we demonstrated these calculations and the comparison and also illustrated how the feature extraction works on our method and reach for the actual analysis of the features and how the large biological datasets are handled and the time consumption of our processes are demonstrated clearly.

CHAPTER 4

IMPLEMENTATION

4.1 Flow Chart of Searching Nucleotide Pattern in DNA

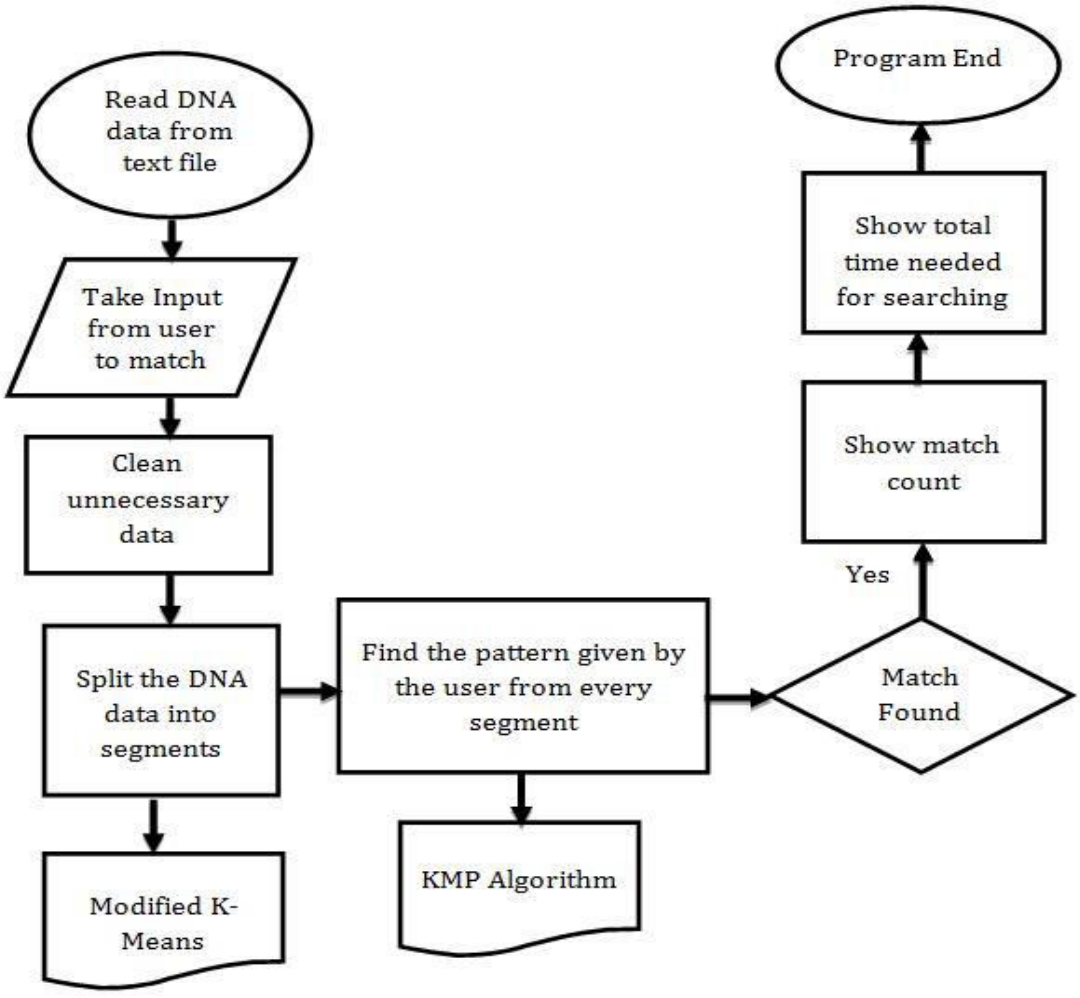


Figure 4.1: Flow Chart of DNA Sequencing

4.2 Setting up the Interface

We have used JAVA Window Builder platform and Swing Designer to design the interface and JAVA for demonstrating our whole work. Overall workflow of our program is illustrated below.

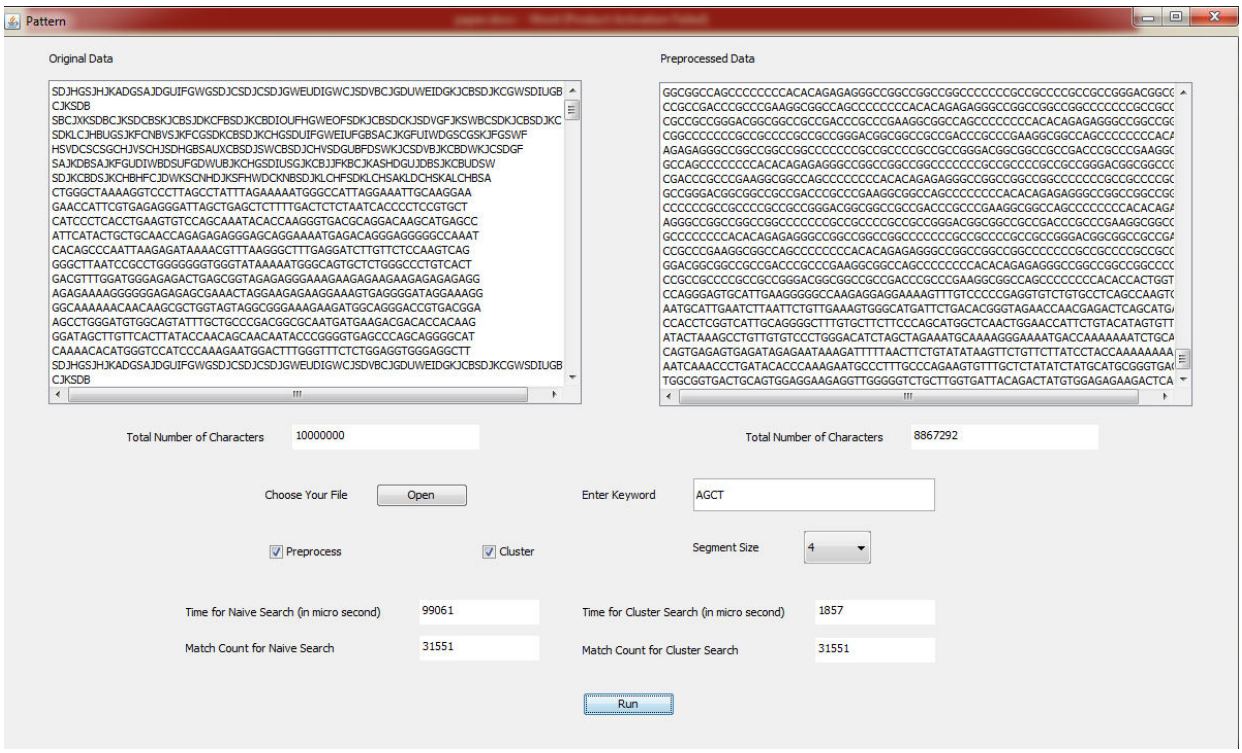


Figure 4.2: Main Interface of the Program

4.3 DNA data read from Large DNA Dataset

In bioinformatics, DNA dataset that consists of A (Adenine), C (Cytosine), G (Guanine) &T (Thymine) to represent the nucleotide sequence. There are one or more lines may follow describing the sequence. Each line of a sequence should have fewer than 80

characters. Sequences are expected to be represented in the standard IUB/IUPAC amino acid and nucleic acid codes.

In this program, we have used this DNA file format for preprocessing as well as clustering using modified K-Means algorithm.

4.4 DNA data preprocessing

As we know DNA dataset consists only (A, G, C, T), there is a probability of noise or irrelevant data that can also come up with the dataset unexpectedly. Therefore, we need to preprocess the data by "data cleaning" method. In this program, we have used JAVA functions for cleaning the irrelevant data and we keep only relevant (A, G, C, T) data. The overall process is as follows:

Preprocess (Text)

```
1. text[]= AGCTBMKCTNGC
2. n = length of text
2. for i = 0 to n
3. if text[i] is equals to ('A'|| 'G' || 'C' ||'T')
4.   subStr[i] = text[i]
6.   end if
9. end for
10.subStr[] = AGCTCTGC
```

4.5 Searching a Pattern from Large DNA Data Using Naive Approach

In our program, we have used Naive Pattern Searching Approach for searching a specific pattern. The reason behind using this approach is to determine the actual match count and time (in microsecond) needed for searching the specified pattern. Then we compare this count and time with our own algorithm to understand the difference between them. [9] We have used JAVA built-in functions for implementing Naive Search in our program. The overall process is as follows:

Check (Text, Pattern)

```
1. text = ACCTACTGC
2. n = length of text
2. pattern = ACTG
3. m = length of pattern
3. for i = 0 to n
4.    for j = 0 to m
5. if pattern matches
6.      count++
8.   end if
9. end for
10.end for
11.count = 1
```

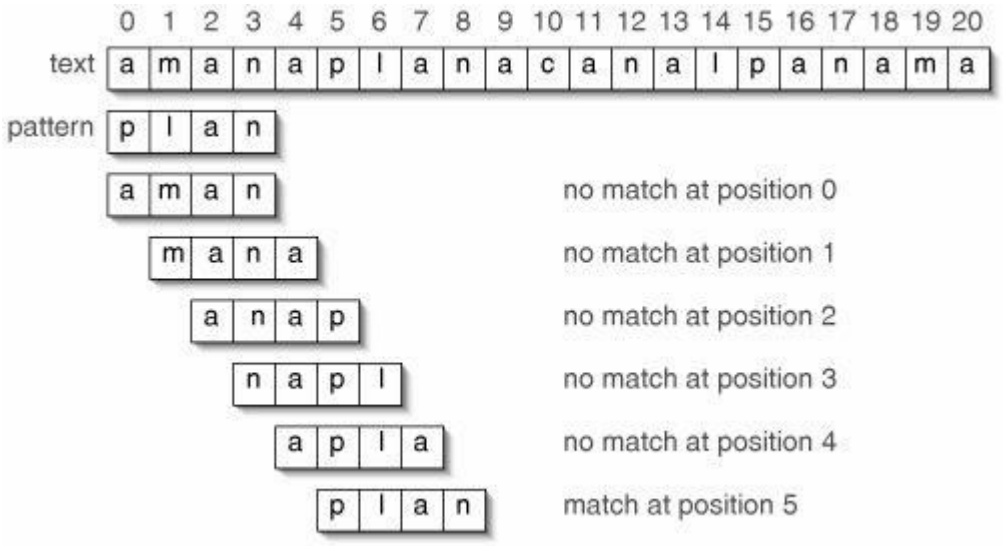


Figure 4.3: Naive Approach for pattern searching

4.6 Splitting the DNA Data into Segments using Modified K-Means Algorithm

K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K. The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Data points are clustered based on feature similarity. The algorithm inputs are the number of clusters K and the data set. The data set is a collection of features for each data point. The algorithm starts with initial estimates for the K centroids, which can be either randomly generated or randomly selected from the data set [10-11].

From K-means algorithm, we have used the idea of clustering the DNA dataset by several segments that is given as an input before clustering. We have used 4,8,16

segments respectively to cluster the string data of DNA. Each cluster consists of multiple characters. The overall process is as follows:

Cluster (Text, Input, Segment)

```
1. define start time
2. length = length of text / segment
3. last cluster index =  length of text / length
4. extra elements of last index =  length of text % segment
5. cluster = n
6. for i = 0 to n
7. if n
8. length =  length + extra elements of last index
9. endif
10.   cluster the text at the size of length
11.   for find pattern using KMP algorithm
12.   count ++
13.   if(count = 1)
14.   flag1 = 1
15.   end if
16.   flag2 = 1
17.   end for
18.   if flag1 not equal 1
19.   calculate the end time
20.   end for
21.   if(flag2 = 1)
22.   count
23.   calculate the elapsed time
```

From the above discussion, we can illustrate the clustering process into a figure. In this figure, we take AGCT as the pattern and AGCTTGGCACAGCTAC as an input text. Then it is segmented into 4 segments. Illustration of segmentation is given below:

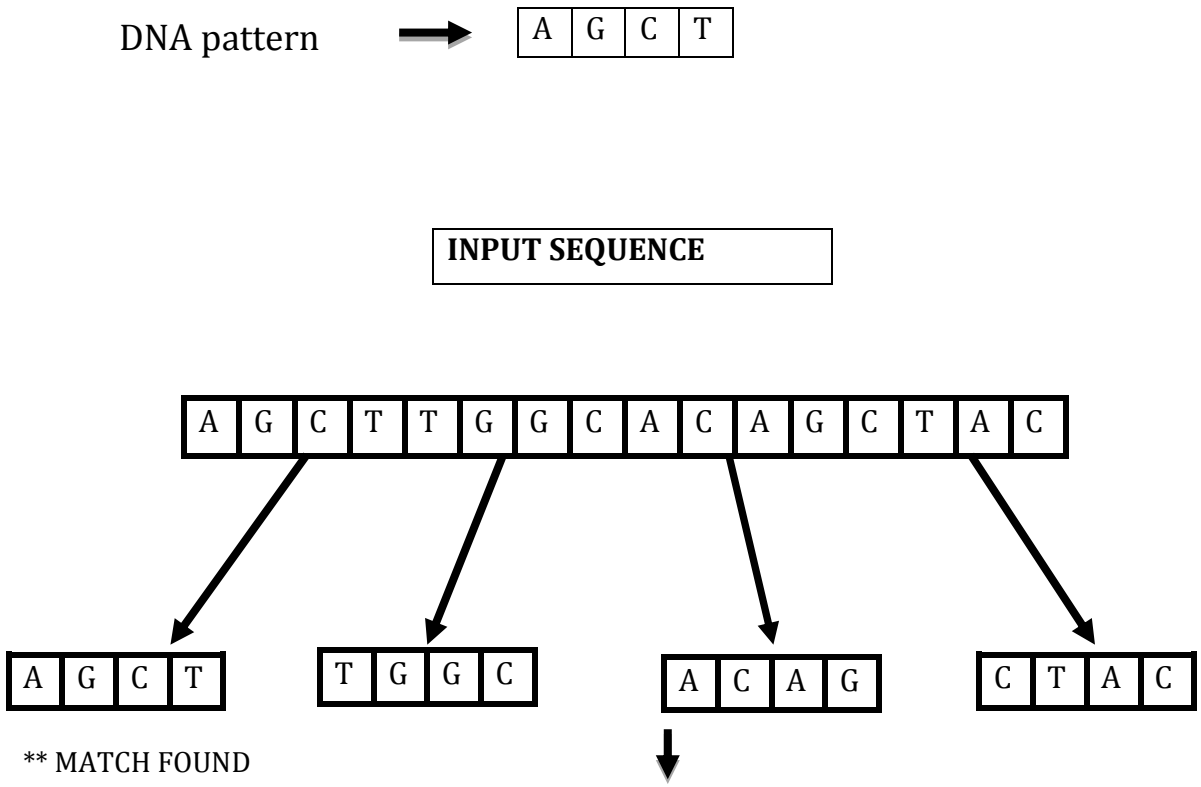


Figure 4.4: Splitting the DNA Data into Segments

4.7 Searching a pattern using KMP (Knuth-Morris-Pratt) Algorithm from segmented data

The KMP matching algorithm uses degenerating property (pattern having same sub-patterns appearing more than once in the pattern) of the pattern and improves the worst-case complexity to O (n). The basic idea behind KMP's algorithm is: whenever we detect a mismatch (after some matches), we already know some of the characters in the text of next window. We take advantage of this information to avoid matching the characters that we know will anyway match [12].

Process of KMP Algorithm:

- KMP algorithm does pat[] and constructs an auxiliary lps[] of size m (same as size of pattern) which is used to skip characters while matching.
- lps indicates longest proper prefix which is also suffix. A proper prefix is prefix with whole string not allowed. For example, prefixes of "ABC" are "", "A", "AB" and "ABC". Proper prefixes are "", "A" and "AB". Suffixes of the string are "", "C", "BC" and "ABC".
- For each sub-pattern pat[0..i] where i = 0 to m-1, lps[i] stores length of the maximum matching proper prefix which is also a suffix of the sub-pattern pat[0..i].
- We start comparison of pat[j] with j = 0 with characters of current window of text.
- We keep matching characters sub[i] and pat[j] and keep incrementing i and j while pat[j] and sub[i] keep matching.

When we see a mismatch,

- We know that characters pat[0..j-1] match with sub[i-j+1…i-1] (Note that j starts with 0 and increment it only when there is a match).
- We also know (from above definition) that lps[j-1] is count of characters of pat[0…j-1] that are both proper prefix and suffix.
- From above two points, we can conclude that we do not need to match these lps[j-1] characters with sub[i-j…i-1] because we know that these characters will anyway match.

CHAPTER 5

RESULT AND ANALYSIS

In the Present era biological data is getting big day by day. Nobody doubts that the biological data will create huge amount of values. It's getting tough for the bio-scientist to find any sequence of DNA data in a very short time.

So to find any sequence of data faster we evaluated the proposed model on large biological data set. In our implementation, we found that the data is divided into no. of clusters which we have given. The data is divided into different groups or clusters using the concept of clustering from k means algorithm. We have found comparative good results. We take AGCT DNA sequence to compare the time difference between the naive search and clustering. We take fixed segment size to cluster the different size of data. After clustering if the DNA sequence AGCT is in first cluster then the data will be found fast.

Table 5.1: Searching time of Clustering with pre-process and naive search with pre-process

Data Size	Clustering with Pre-process (µs)	Naive search with Pre-process (µs)
1000000	388	14433
2000000	794	27243
3000000	1134	41430
4000000	1409	55502
5000000	1795	69572
6000000	2189	83422
7000000	2472	96821
8000000	2836	110673
9000000	3221	124414
10000000	3668	144620

Here the table 5.1 presents the searching time difference between Clustering with pre-processing data and Naive search with pre-processing data. Pre-processing data there will be no noisy data in the data set. In Naive search it checks the whole length of data

set to find any DNA sequence such as AGCT. But in Clustering algorithm, we clustered the data in the 4 segment so that the size of the data reduces. If the size of a whole DNA can be reduces the operational time of search is also reduced. For 1million length of data the time of Clustering is only 388 µs but in Naïve search the time is 14433 µs. Another example is for 10 million lengths of data the time of Clustering is only 3668 µs but in Naïve search the time is 144620 µs. From the table we can see the time of Clustering is much faster than the naive search for different size of biological data set.

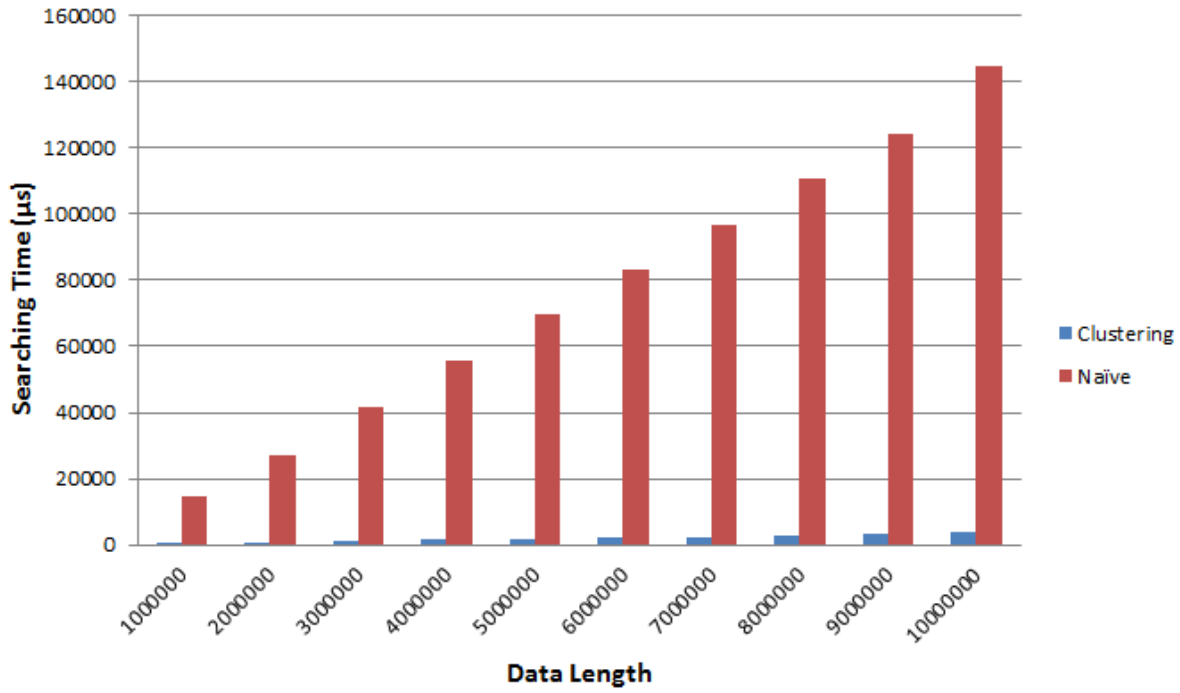


Figure 5.1: Comparisons between Clustering with Pre-process and Naive search with pre-process based on searching time

Figure 5.1 depicts that the Clustering achieves better relationship between time and data length than with pre-processing results. It takes least time compared to Naive search. Here time has been represented in micro second to understand the difference. Since the data size become larger, then the searching time of Naive search become high. Here we can see the huge time difference between the Naïve search and Clustering.

The preceding results established the efficiency of the proposed model for different length of data of large biological dataset.

Table 5.2: Searching time of Clustering with pre-process and Clustering without pre-process

Data Size	Clustering with Pre-process (**µs**)	Clustering without Pre-process (**µs**)
1000000	401	683
2000000	787	896
3000000	1120	1230
4000000	1409	1607
5000000	1813	2072
6000000	2255	2468
7000000	2511	2724
8000000	2827	3198
9000000	3290	3497
10000000	3562	3941

Here the table 5.2 presents the average searching time difference between Clustering with pre-processing data and Clustering without pre-processing data. Here the Clustering with pre-process need less searching time compared to the Clustering without pre-process for seeking the occurrence of AGCT sequence. In the table 2 for 1 million lengths of data searching time of Clustering with pre-process is 401 µs and the searching time of Clustering without pre-process is 683 µs. So the time difference between them is 282 µs. So the searching time of modified clustering with pre-process is faster than the clustering without the pre-process.

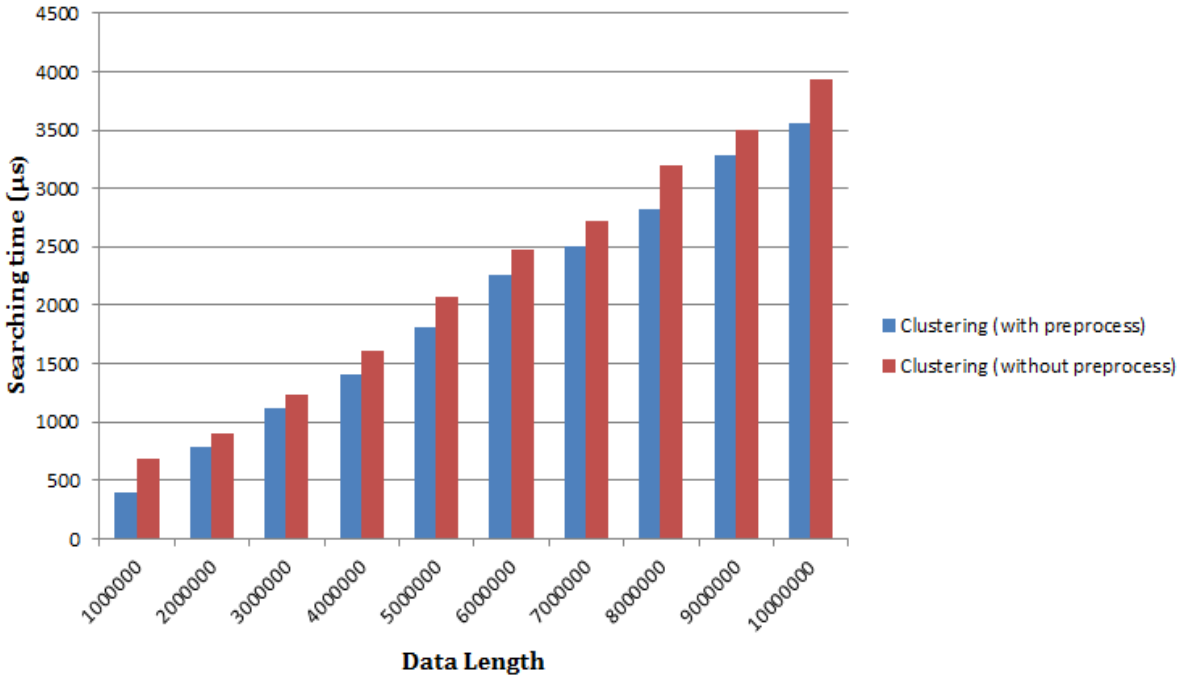


Figure 5.2: Comparisons between Clustering without Pre-process and Clustering without pre-process based on searching time

Figure 5.2 depicts that the modified Clustering with pre-processing achieves better relationship between time and data length than without pre-processing results. It takes least time compared to Clustering without pre-processing. After pre-processing we are getting AGCT sequence faster than before for different large data set.

Thus, the proposed clustering algorithm is applied on different table block with different data size. Figure 5.2 illustrates the relation between the dataset of different size and the searching time of DNA sequence. The figure depicts that the difference between Clustering with pre-process and Clustering without pre-process on searching time. It depicts that Clustering with pre-processing data searching fast a sequence AGCT of DNA data than Clustering without pre-processing data.

Table 5.3: Searching time of Clustering with pre-process in different segment size

Data Size	4 Segment(µs)	8 Segment(µs)	16 Segment (µs)
1000000	328	191	104
2000000	770	367	223
3000000	1068	554	255
4000000	1443	743	393
5000000	1730	897	487
6000000	2087	1065	542
7000000	2431	1207	632
8000000	2784	1432	714
9000000	3215	1593	781
10000000	3483	1764	887

Here the table 5.3 represents searching time of Clustering with pre-process in micro second (µs) for different segment size. For 5 million length of data if the data is partitioned in 4 segments then the average searching time is 1730 µs but for 8 segment the average searching time is 897 µs. for 16 segment the average time is 487 µs. The searching time difference between 4 and 8 segments is 833 µs and for 8 and 16 segments is 410 µs. For 10 million length of data if the data is partitioned in 4 segments then the average searching time is 3483 µs but for 8 segment the average searching time is 1764 µs. for 16 segment the average time is 887 µs. The searching time difference between 4 and 8 segments is 1719 µs and for 8 and 16 segments is 877 µs.

From this comparison we can understand that partitioning large data in large segment size can reduce the data size and alleviate the searching time to find out number of occurrences of DNA sequence such as AGCT.

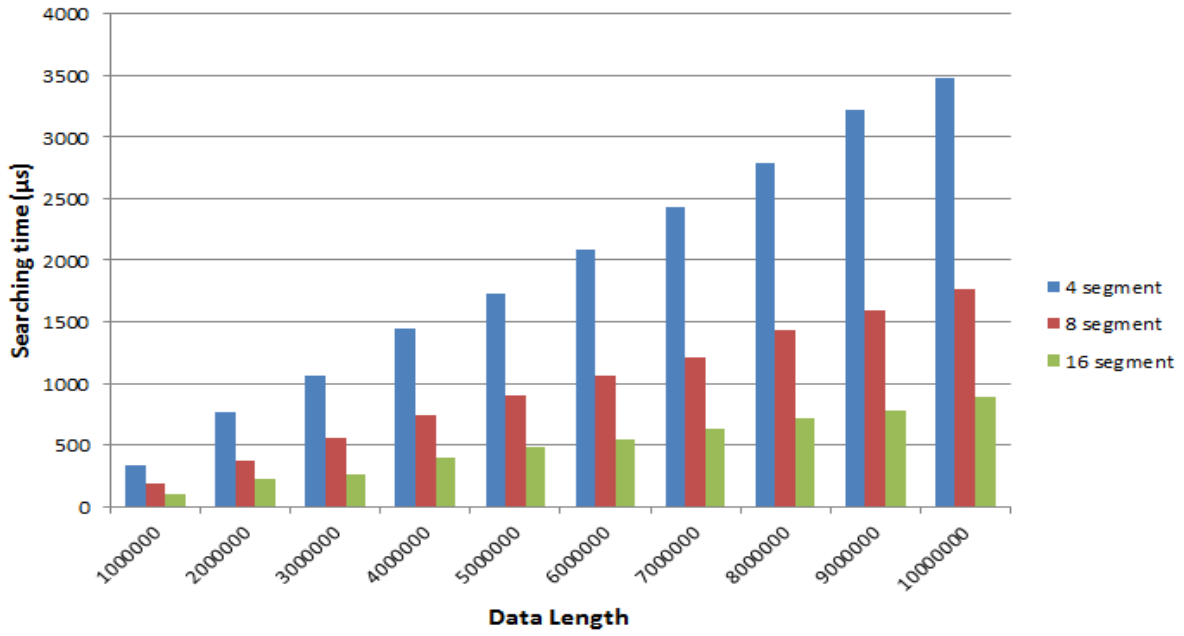


Figure 5.3: Comparisons of Clustering with Pre-process on searching time in different segment size

Figure 5.3 depicts that after segmentation searching achieves better relationship between time and data length with preprocess DNA sequence of data. Using the concepts of segmentation from clustering, it reduces the searching time of the required DNA pattern. As we can see smaller the segment sizes larger the searching time and as well as larger the segment size smaller the searching time.

Table 5.4: Number of occurrences in Clustering with pre-process and naïve search without pre-process in different segment size

Data Size	Clustering with Pre-process (no of Occurrence)	Naïve search with Pre-process (no of Occurrence)
1000000	1127	1127
2000000	2254	2254
3000000	3380	3380
4000000	4508	4508
5000000	5635	5635
6000000	6760	6760
7000000	7888	7888
8000000	9016	9016
9000000	10143	10143
10000000	11270	11270

Table 5.4 represents the comparisons between number of occurrences of DNA sequence using Clustering with pre-process and naïve search with pre-process. In this experimental result data has been partitioned in 8 segments with pre-process data. From our experimental result for 1million length of data number of occurrences in both methods is 1127. For 10 million length of data number of occurrences of DNA sequence (ACTTT) in both methods are 11270. As we can see there is no data loss between Clustering and Naïve search after pre-processing.

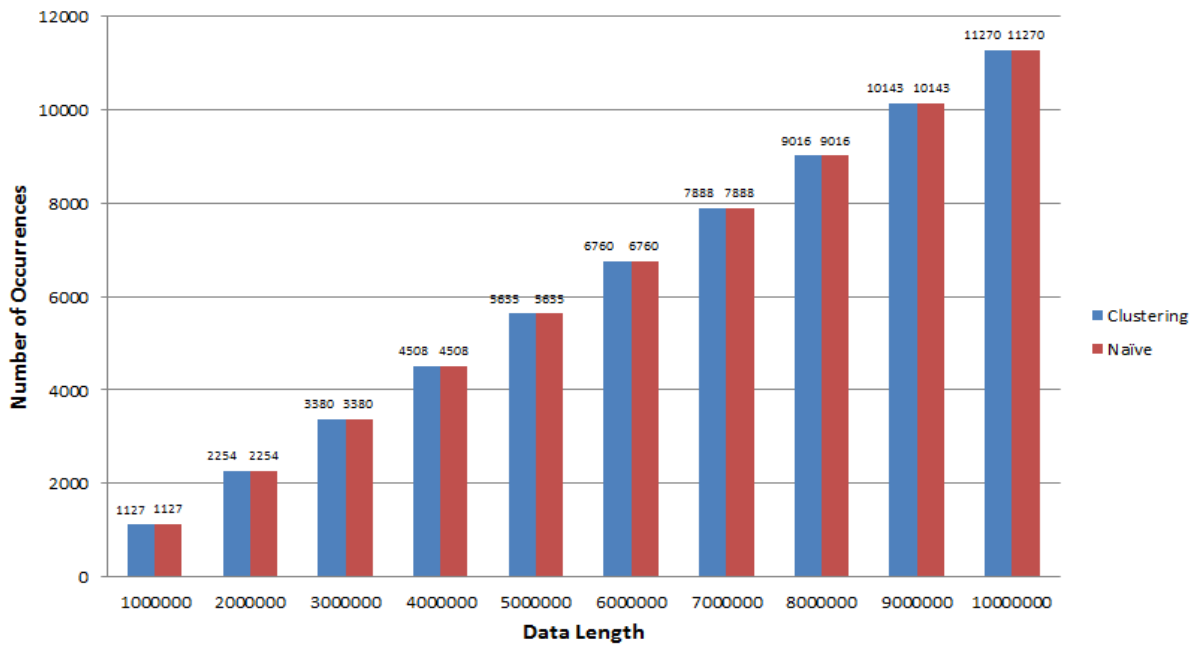


Figure 5.4: Comparisons between Clustering with Pre-process and Naïve search with pre-process based on number of occurrences of DNA sequence

Fig 5.4 depicts the comparisons of number of occurrences of DNA sequence between clustering with pre-process and naïve search with pre-process. In this experimental result segment size doesn't matter because the same number of occurrences of ACTTT sequence. In any size of segment such as 4, 8 and 16 there is no data loss for ACTTT sequence.

Most of the time number of occurrences in both methods is same. Our experimental results showing no difference between Clustering and Naïve search. So there is no data

loss in our experimental result. But applying Clustering and Naïve search in our experimental dataset we didn't get any data loss.

But somehow data can loss for another sequence of data in Naïve search because of noisy, irrelevant data. But using pre-processed dataset in Clustering we can get the lost data back.

Table 5.5: Number of occurrences in Clustering with Pre-process and naïve search without pre-process

Data Size	Naïve search without Pre-process (no of Occurrence)	Clustering with Pre-process (no of Occurrence)
1000000	1126	1127
2000000	2252	2254
3000000	3373	3380
4000000	4504	4508
5000000	5630	5635
6000000	6754	6760
7000000	7881	7888
8000000	9008	9016
9000000	10134	10143
10000000	11259	11270

In our experimental dataset we experiment difference between Naïve search and Clustering for ACTTT. We found difference of number of occurrences between Naïve search and Clustering. For 1 million length of dataset there is only one DNA sequence has lost in Clustering. For 3 million there is only three DNA sequence has been lost. If we see in the results of 6 million and 10 million lengths of data there only 6 and 11 DNA sequence (ACTTT) has been lost. In that case in Naïve search we will not get the exact number of occurrences.

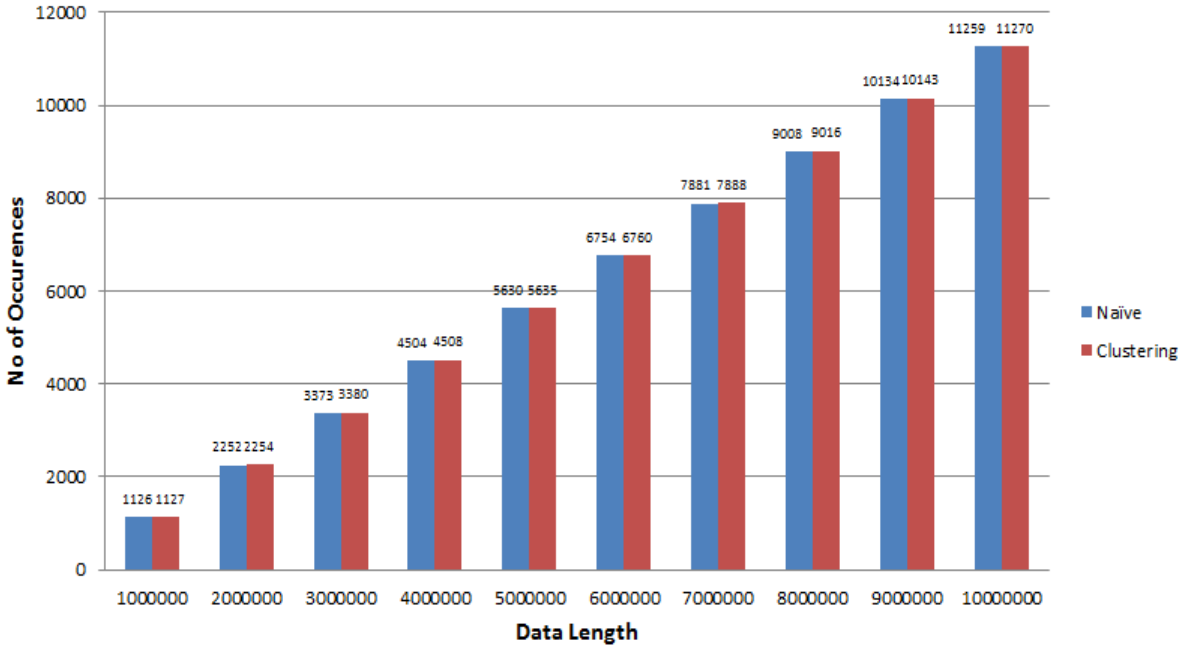


Figure 5.5: Comparisons between Clustering with Pre-process and Naïve search without pre-process based on number of occurrences of DNA sequence

Figure 5.5 depicts the comparisons between the Clustering with pre-process and Naïve search without pre-process. After pre-processing the length of dataset is reduced because it removes the irrelevant data from the original dataset. After cleaning original data we get accurate number of occurrences in clustering rather than Naïve search. Naïve search without irrelevant data causes data loss.

Table 5.6: Searching time of Clustering with pre-process in different segment size for searching all occurrences

Data Size	4 Segment **(µs)**	8 Segment **(µs)**	16 Segment**(µs)**
1000000	7005	7106	7586
2000000	12828	12736	12249
3000000	19936	19617	20215
4000000	26579	25968	25957
5000000	33256	32813	31920
6000000	38175	36551	38416
7000000	45874	46568	45507
8000000	51634	53211	50631
9000000	56754	57722	57104
10000000	64106	61842	63067

Table 5.6 represents the searching time of all occurrences of DNA sequence with different segment size to check which segment size is good for which length of data. Here we can see for 8 million lengths of data searching time of all occurrences in segment 4 is 51634 µs but for 8 segments searching time is 53211 µs and for 16 segment time searching time is 50631 µs. There is difference between 4,8,16 segments. For 10 million length of data searching time of all occurrences in segment 4 is 64106 µs but for 8 segment searching time is 61842 µs and for 16 segment time searching time is 63067 µs. So if we split the data in 8 segments for 8 million lengths of data then it takes more time than other segment size. But for 10 million lengths of data searching time is less than other segments. With other length of data same changes So searching time of all occurrences for different length of data with different segment size is different.

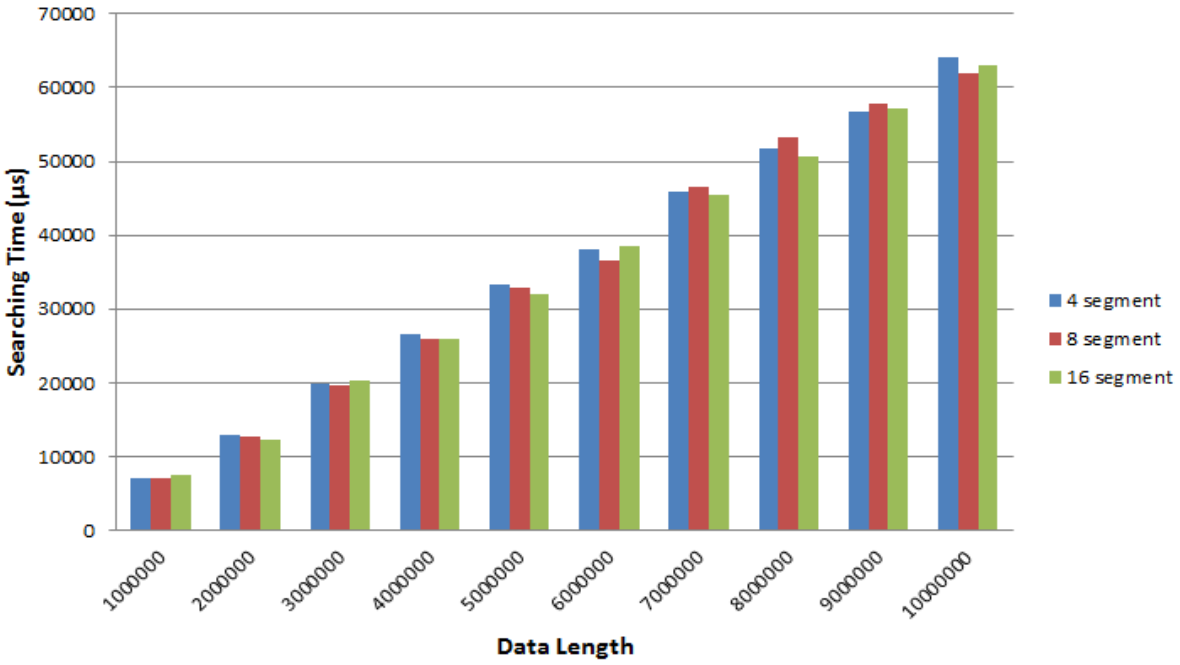


Figure 5.6: Searching time of clustering with different segments for searching all occurrences

Figure 5.6 illustrates the comparisons between different segments for searching all occurrences. But in this graph searching time is different for different length of data. We couldn't differentiate between 4, 8, 16 segments for searching time of all occurrences to find out that which segment size is good for which length of data.

The preceding results established the efficiency of the proposed system for different dataset of large biological data. Actually we tested for different sequence pattern for our

modified Clustering approach with or without preprocess and compute the elapsed time for each comparison. These comparisons depicts that our proposed approach for clustering the DNA dataset are very efficient for extracting the expected features. We tested the dataset for two factors: elapsed time for extracting the features and how many times the expected pattern occurs on the DNA sequence. There may be some loss of data for our process but at the end we can say that our proposed approach are not much but enough efficient for getting the outcome on these biological data.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this era of technology, we can now do whatever we want using technology. It is possible to make our life more comfortable through the blessings of technology. DNA sequencing technology opens the door of gaining more knowledge about all the living beings of the world. Our world is rapidly changing, so does our technology.

As we know the knowledge behind DNA is being revolutionized by the DNA sequencing industry as well as bioinformatics, it is obvious for us to improve the sequence and pattern matching methods. Just as computer technology and the internet created whole new industries and extraordinary benefits for people that extend into almost every realm of human endeavor from education to transportation to medicine, genetics will undoubtedly benefit people everywhere in ways we can't even imagine but know will surely occur.

6.2 Future Work

Some features of this program leaves the door open for further improvement in the future. In our program, we can also enhance the overall work of DNA pattern matching with some extra techniques that we can implement later. Here we are highlighting the key improvements that we can add in future:

- Using better algorithm approach for preprocessing data.
- Full implementation of K-Means algorithm in clustering for more efficiency.
- Thread based parallel searching on different clusters.
- Using better searching algorithms that works faster.

BIBLIOGRAPHY

1. Rachael Rettner, “DNA: definition, structure and discovery”, June 6, 2013.
2. Abigail Linton, “Pattern Searching in a Single Genome”.
3. Aarti.B.Sahitya, DR.Mrs.M.Vijayalakshmi, “Feature Extraction from Big Data”, International Journal of Advance Foundation and Research in Computer (IJAFRC) Volume 2, Issue 10, October - 2015. ISSN 2348 – 4853, Impact Factor – 1.317
4. Li W, Jaroszewski L, Godzik A: Clustering of highly homologous sequences to reduce the size of large protein databases. *Bioinformatics* 2001, 17(3):282
5. Jill U. Adams, “DNA Sequencing Technologies”, 2008.
6. Rosie Cornish, “Cluster Analysis”, 2007.
7. Andrea Trevino, “Introduction to K-means Clustering”, June 12, 2016.
8. URL: https://brilliant.org/wiki/k-means-clustering/ [Last accessed: 09-04-2017]
9. Ricardo A. Baeza-Yates. "Algorithms for string searching: A survey". ACM SIGIRForum,23, pages 34-58, 1989.
10. Selection of K in K-means clustering: D T Pham, S SDimov, and C D NguyenManufacturing Engineering Centre, Cardiff University, Cardiff, UK
11. Hartigan, J. A.; Wong, M. A. (1979). "Algorithm AS 136: A K-Means Clustering Algorithm". Journal of the Royal Statistical Society, Series C. 28 (1): 100–108. JSTOR 2346830.
12. Knuth, Donald; Morris, James H.; Pratt, Vaughan (1977)."Fast pattern matching in strings".SIAM Journal on Computing. 6 (2): 323–350. doi:10.1137/0206024.