A Design of Approximation Algorithm for Efficient DNA Mapping using Hadoop with GPU Acceleration

[1] Prashant Chaturvedi, [2] Aditya Kumar Sinha
[1][2] Computer Science & Engineering (High Performance Computing Solution) Vel-Tech University, Chennai, Tamil Nadu, India, C-DAC, Pune, Maharashtra, India
[1]prashantchaturvedi1987@gmail.com, [2] saditya@cdac.in

***Abstract*: DNA Sequencing is a process where we determine and identify every single DNA base and element that is in the genome of an individual. There are six billion of those in every normal cell in every person. When we apply DNA sequencing in the Cancer project. We first figure out what those six billion DNA bases are in the normal cells in that person and then we take some of the tumor in that person and figure out what the DNA bases are in the tumor. In this paper, we discuss impediments and future works about Hadoop in bioinformatics. We study the Map Reduce algorithm from algorithm lay by point and demonstrate the appropriates of our approach by tracing and analyzing efficient Map Reduce algorithms for sorting and simulation problem of parallel algorithms specified in the help of pigeonhole principle. The big data is a great computational challenge to statistical analysis of DNA big data. We can get general statistical analysis through R language. After studying the survey paper various approaches of using GPU and Map Reduce. We adopted the best solution to using R with Map Reduce. An R package is created to shift a set of critical R functions on GPU card. It allows users to run R code with GPU spread that enable much faster large data set of computation.**

***Index Terms*— Big Data, Approximation Algorithm, Pigeonhole Principle, NVIDIA card.**

I. INTRODUCTION

Bioinformatics department is help of solve the biologists computational problems on purpose confront large amount of data. Recently, computing and sequencing ability has improved throughout the processor. DNA is the hereditary building in all organisms.

DNA resides in every cell in the body of organism. The double helix looks an immensely long step twisted into a helix.

The sides of the step are formed by a backbone of sugar and join many phosphate molecules and the pole consist of nucleotide bases join in the middle by the hydrogen bonds. This is four types of nucleotides. Each nucleotide contains a base:

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

Base pairing form naturally between A and T and between G and C at normal data pattern therefore that basic sequence is a single strand of DNA each parts of DNA which can be simply deduced from that of its partner strand.

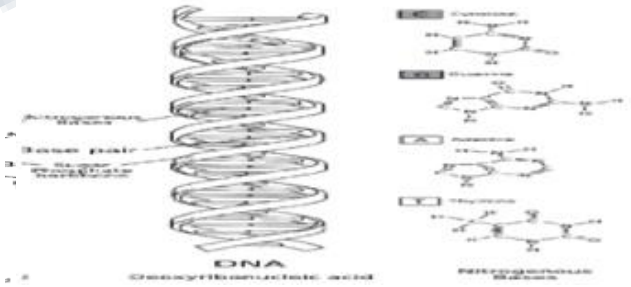


Figure 1 DNA Structure with Molecular Structure (source: google Image)

DNA is the chemical liquid which carrying instructions to cell. When those instructions have mistaken that cells function normal or not. These cancer causing changed to DNA sequence. It is called mutations. Cancer are changes DNA Sequence. It can cause cells to produce the wrong amount of a certain protein. In cancer many tissues are damaged. If mutation is changed in our DNA building. A gene mutation is a change in or damage to a gene. These changes in your DNA can result in genetic disorders. Mutations can lead to missing or deformed proteins and that can lead to disease.

- ❖ Deletion
- ❖ Substitution
- ❖ Inversion
- ❖ Insertion

In this disease this caused changes the human DNA structure and mutated in an individual's DNA sequence. These mutations can be create some subsequence an error in DNA replication due to environmental factors such as cigarette smoke, alcohol and divestment to radiation which cause changes in the DNA sequence. Our DNA provides the code for making proteins.

Another type of genes maintains the integrity of genes and provide the accuracy in the information transfer from one gene to another. XRCC3 gene being a DNA mismatch repair gene responsible for skin cancer.

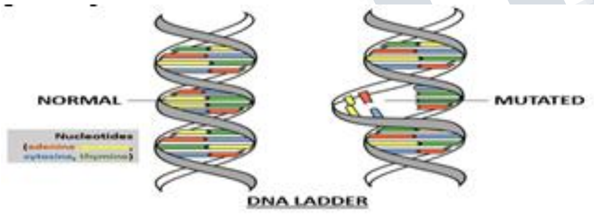


Figure 2 compare normal DNA structure to mutation DNA structure (source: google Image)

II. PROCEDURE FOR PAPER SUBMISSION

A. Implementation and Technique

We will do Approximation algorithm concept for more Efficient Mapping using Includes algorithm like Pigeonhole Principle and Hadoop. This can be widely used in healthcare for Treatment e.g. DNA. It will predict the disorderness, sorts and fits. The missing bits in the extremely complex, unstructured and semi structured data.

Pigeonhole principle is a fundamental mathematics tools which combined the big element. Unlike other strong theorems t is hard to have a glimpse of its elegance and useful applications in mathematics. The Pigeon Hole Principle (PHP) is one of the important mathematical concepts.

$$1 - [(m)_n / m^n]$$

Where (m)n is the falling factorial m(m - 1)(m - 2) ... (m - n + 1). For n = 0 and for n = 1 (and m > 0) that probability is 0 it means that if there are only one pigeon it is manifest that here cannot be a conflict. For n > m (more pigeons than pigeonholes) in which it coincides with the usual pigeonhole principle. But even if the number of pigeons does not increase the number of pigeonholes (n).

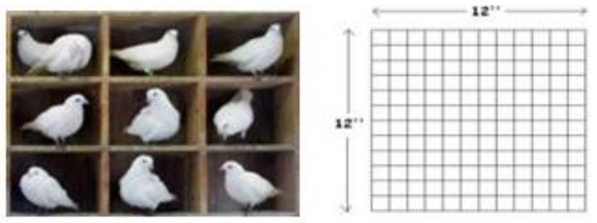


Figure 3 piegonhole algorithm and block with dimension (source: google Image)

Map Reduce is a JAVA programming model and a associated parallel and distributed computing for implementation, developing and process of parallel algorithms to massive datasets on clusters of commodity machines. The input is seen as a sequence of ordered key-value pairs (k,v). The map function take input one such (key, value) pair at a time and can processor a finite multiset of pairs {(k1, v1),(k2, v2), · · · }. The reduce function takes as input a key k and value v pair produce another set {(k1, v1),(k2, v2), · · · }. The output of reduce can be used as input for another map function to develop.

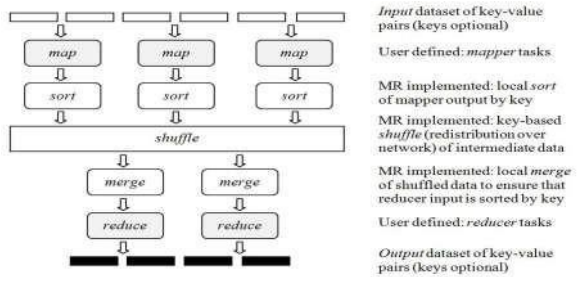

map
map
map
sort
sort
sort
shuffle
merge
merge
reduce
reduce
Input dataset of key-value pairs (keys optional)
User defined: mapper tasks
MR implemented: local sort of mapper output by key
MR implemented: key-based shuffle (redistribution over network) of intermediate data
MR implemented: local merge of shuffled data to ensure that reducer input is sorted by key
User defined: reducer tasks
Output dataset of key-value pairs (keys optional)

Figure 4 MapReduce algorithm (source: google Image)

B. Parallel Programming Model

NVIDIA has conduct new parallel programming called Compute unified device architecture (CUDA) in 2007.It can create the number of applications in GPU that are highly parallel in nature and run on thousand of GPU core processor in parallel. CUDA builds by threads which fast shared memory and carry out parallel execution.

The CUDA programming language is much similar in C language and has a high learning curve. CUDA calls some API function which increases various libraries in order to access the GPU specific features. It has some specific functions called kernels.

CUDA is a parallel computing architecture that enables stage increases in computing performance by harnessing the power of the GPU with millions of CUDA program enabled in GPUs sold to date, software developers, scientists and researchers are finding broad-ranging uses for CUDA when including image and video processing, computational biology and computational chemistry, fluid dynamics simulation, CT scan image reconstruction, seismic analysis, ray tracing, and much more.

This new computing technology, NVIDIA invented the CUDA parallel computing architecture that is now shipping in GeForce, ION Quadro, and Tesla GPUs, representing a significant installed base for application developer.

C. *Highest Delivered Performance*

High-through DNA sequence alignment using GPUs

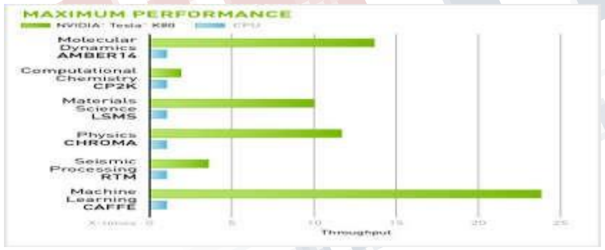


Figure 5 max. Performance (source: gpu website)

D. *Design Model*

This designed PHA algorithm is encoded by floating numbers and consists of special genetic operators including initialization, mutation, selection and termination. No crossover operation is implemented in this design. The fitness function is derived from the cost function. The initialization and mutation are designed with built-in constraints as defined in below.

Each chromosome op is a 2 × Nc Array matrix with N element with the equation of first row representing δv and the second row representing δφ in the control horizon. The total number of chromosome in the population is Npop.

$$op = \begin{bmatrix} \delta v_1\, \delta v_2 \ldots \delta v_{N_c} \\ \delta\varphi_1\, \delta\varphi_2 \ldots \delta\varphi_{N_c} \end{bmatrix}, p = 1,2,\ldots,Npop$$

A suitable initialization procedure at the beginning of each MPS cycle is essential for a better and faster optimization result. At first each gene in every chromosome is assigned by randomly picking a floating number within the intervals are only enforced in fitness evaluation process. The best chromosome from the previous optimization MPS cycle also plays a role in the initialization process of the current cycle. It is shifted to the left by one gene and patched randomly at the end. 20% of the population will be picked randomly and replaced by this chromosome. This facilities the PHA algorithm to exploit based on the best knowledge so that it may reach at the optimum faster. This also enforces the stability of the PHA algorithm results. Whereas the rest of the population which is generated randomly creates diversity and possibility in the searching space to prevent the PHA algorithm from being trapped at local optimum.

After three stages of selection are required one is the selection for mutation and the other is survivor selection. Data will go to Hadoop technology which are process to the map, shuffle, sort and reduce the size of data which faster and better speed. The selection for mutation selects individuals or chromosomes which go to the mutation process in Hadoop technology. To prevent the good individuals from taking over the entire population rapidly and maintain a suitable selection pressure deterministic q selection is implemented. Firstly q individuals are selected at random from the population. Then the fitness is compared and the best individual out of the q is selected and copied to the intermediate pool on which the mutation will act. The process is repeated for Npop times until the population in the intermediate pool reaches Npop. The survivor selection is to select Npop chromosomes out of the 2Npop after mutation. The 2Npop chromosomes consist of the original ones before selection for the mutation (or parents) and the mutated ones (or children). In order to maintain the diversity in the subsequent generations all parents will be replaced by children.

Mutation acts on the gene of individuals in intermediate pool. At each gene, a random number between 0 and 1 will be picked uniformly and compared with the mutation rate pm. If it is less than pm, that gene undergoes mutation meaning the gene will be replaced by a new random number picked uniformly within the range of higher. If the mutated gene is δφ, only its value will change but its sign remains. To prevent good individuals from being destroyed by mutation and expedite the optimization process. The new mutation rate ***p m*** is given by-

$$pm=(1+(1/pm \quad 1)e \quad \gamma N(0,1)) \quad 1$$

where γ is learning rate and N(0, 1) represents a random number from normal distribution (0 and 1).

The fitness of an individual is defined based on the cost function

$$F=1/(1+J)$$

The special feature about this process is enforced here to save computation time. Otherwise additional loops of condition checking have to be carried out in initialization and mutation process.

This determines when the GA should stop and return the best individual leading to the minimum of J. The termination condition is a balance-off between the best optimal solution and computation time. In particular the PHA algorithm will terminate when the optimal solution is found or the time elapsed is reaching 1.1T s whichever is earlier.

E. Code Flow of GPU and CUDA

At the very beginning of the CUDA code's execution, code is compiled just like other python code. Its primary execution takes place in CPU. As the execution started all non-kernel functions getting executed on CPU and the execution of kernel code is being transferred to GPU. This way we get parallel execution on CPU and GPU. Once the memory transfer between CPU to GPU is done, without any impediments the rest of the execution is carried well otherwise execution will be halted. Pattern matching gives out the search results for presence of specific gene in DNA sequence. Different gene sequences are taken from well-known databases and genome project. This can be applied in cancer diagnosis by matching several gene patterns in input sequence and draw inferences from result. The simplest approximate algorithm for pigeonhole technique is used for the matching so as to cope up with complexity and to prevent possible overhead occurring due to parallelization. There will be two input files, one carrying gene code for each gene; another will be set of files carrying the information. To extract pattern match from a large sequence it takes more time. In order to reduce searching time matching is carried out parallel that reduces the search time with accurate retrieval. The basic idea behind using the parallel approach for cancer diagnosis is quite simple. The genes that are responsible for particular type of cancer are being organized (data is collected from well-known database NCBI and other genome projects). DNA is declared to be cancer prone unless each is gene pattern is exactly present in given human DNA.

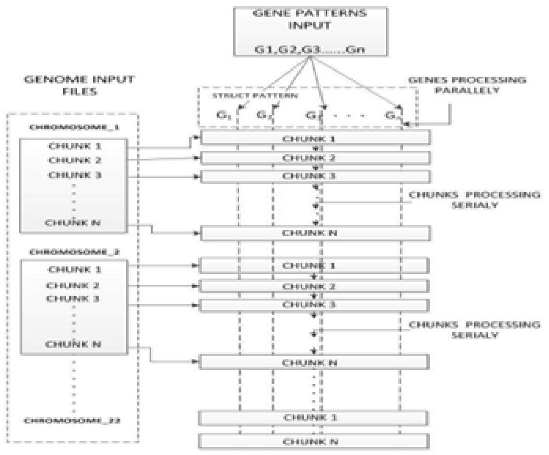


Figure 6 gpu coding process (source: Snehal [15])

```
import pycuda.compiler  as  comp
import pycuda.driver  as drv
import numpy
import pycuda.autoinit

Mod= comp.SourceModule("""

__global___void multiply_them(float  *dest,
float *a, float *b)
{
Const int I = threadIdx.x;
Dest [i] = a[i] * b[i];
}
""")
Multiply_them  =
mod.get_function("multiply_them")

a=
numpy.random.randh(400).astype(numpy.float
32)
b=
numpy.random.randn(400).astype(numpy.float
32)

dest = numpy.zeros_like(a)
multiply_them(
 drv.Out(dest),  drv.In(a),  drv.In(b),
 block= (400,1,1))
print dest-a*b
```

F. Flow Chart

To further improve the computation time since the chromosomes are independent of each other all the individual processes are expedited by parallel programming.

Flow Chart of approximation algorithm processes is shown in Figure-5

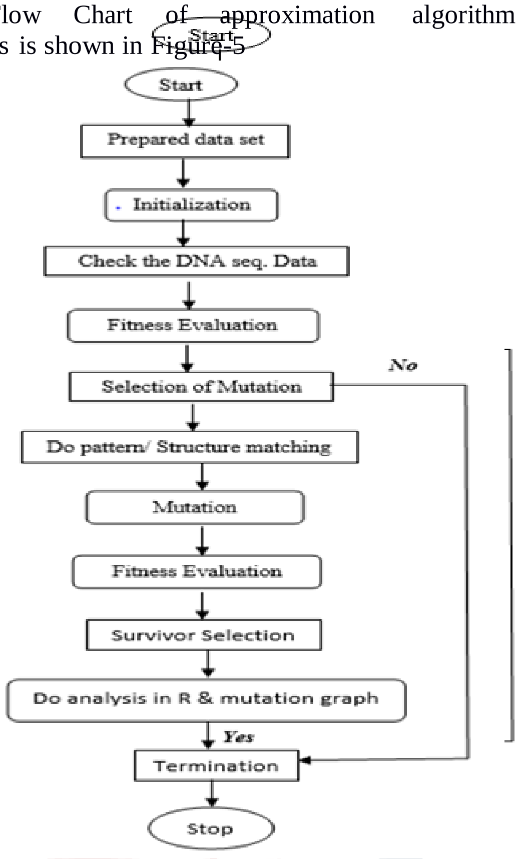


Figure 7 approximation algorithm processes (source: own)

G. Code Flow of approximation algorithm

Let ARR is an array with N elements. Num is the searching element and LOC is the location of searching element.

```
1. Begin
2. Real ARR and NUM
3. Set found = 0
4. Set i=0, Loc=0
5. Repeat slep(6) while i<=N-1
6.     if ARR[i]= NUM
       than Set found
       =1 and Set loc
       =i
       and break the loop.
7. if found =1
     then Write : Element <NUM>
   found at position <Loc>
     else Write : Element <NUM> not present
     in
   ARR
8. return
```

H. Implementation

The Hadoop used in this research work which is process through Pigeonhole and Hadoop Algorithm (PHA). The DNA data and sequencing is a very large sequencing process. So this process cannot by manually. Many processes are existing in research center. But that technology takes more time. Therefore, we use the New Hadoop Technology.

The input data will follow the mentioned process. Firstly data will be prepared for the initialization. This initialization process check the DNA sequencing data with C coding in pigeonhole algorithm. Then data fit on block one by one. Then again check the mutation if mutation available in that data so it is select the mutation data then send data another technology which are map the data through the Mapper tasks are select data to keep keys and values pairs which is defined mapper tasks then sorting the data in pieces in local host. That output is go to shuffle and merge the data means this process is keep separate key and value pairs It is keep number and alphabetic key separate. Then next process will reduce the data in reduce tasks process. The reduce task process is the reduce size of data and delete the unwanted sequence and get output. If there is not any mutation data so process already terminates here and stop the process. Ahead process is not do in pigeonhole algorithm. Then data go to check the DNA sequence pattern and do matching. The mutations keep separate and again fitting in the block. Then do the analysis in R language.

III. SYSTEM REQUIREMENTS

The GPU used in this research work is TESLA C2070 on a
HP Z420 VMwave workstation i7 multicore processor having
7 GB RAM operated on a 64 bit Cloudera Linux operating system. It is the NVIDIA computing processor designed to redefine high performance computing. It is based on the next-generation CUDA "Fermi" architecture. CUDA programming model is used in this implementation work. CUDA makes the computing engines of graphics processor units accessible to general purpose software developers through a standard programming language C, with an API to explicit the architecture parallelism. GPU allows thousands of threads to run in parallel.

IV. RESULTS AND DISCUSSION

Figure 6 are shown about Normal DNA and mutation DNA. There are two axis which consists totalcin versus stage Here a first figure is shown normal DNA and second figure shown mutation DNA.

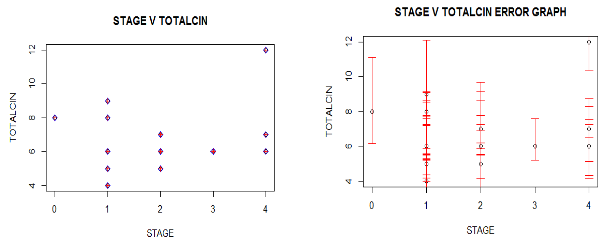
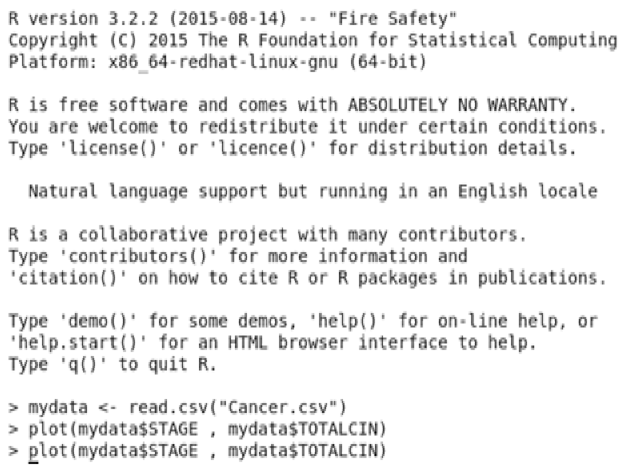


Figure 6 generate the graph normal DNA and mutation D by R Language (source: own)

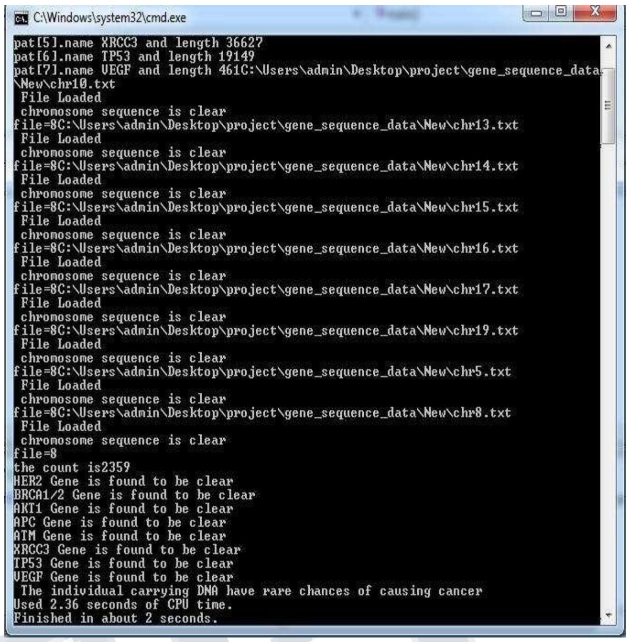
The figure 7 is shown about output command in R language using cloudera platform



```
R version 3.2.2 (2015-08-14) -- "Fire Safety"
Copyright (C) 2015 The R Foundation for Statistical Computing
Platform: x86_64-redhat-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> mydata <- read.csv("Cancer.csv")
> plot(mydata$STAGE , mydata$TOTALCIN)
> plot(mydata$STAGE , mydata$TOTALCIN)
```

Figure 7 output command (source: own)

Figure 8 shows the output of the execution of the parallel implementation of pattern matching algorithm which diagnose the input DNA sequence by detecting exact presence of provided genes. The specified output first displays the name of all the input genes with corresponding lengths. Then each pattern is simultaneously searched in the several files loaded into directory name new. As the input DNA sequence belongs to normal individual, it generates true negative diagnosis.



```
pat[5].name XRCC3 and length 36627
pat[6].name TP53 and length 19149
pat[7].name VEGF and length 461C:\Users\admin\Desktop\project\gene_sequence_data
\New\chr10.txt
 File Loaded
 chromosome sequence is clear
file=8C:\Users\admin\Desktop\project\gene_sequence_data\New\chr13.txt
 File Loaded
 chromosome sequence is clear
file=8C:\Users\admin\Desktop\project\gene_sequence_data\New\chr14.txt
 File Loaded
 chromosome sequence is clear
file=8C:\Users\admin\Desktop\project\gene_sequence_data\New\chr15.txt
 File Loaded
 chromosome sequence is clear
file=8C:\Users\admin\Desktop\project\gene_sequence_data\New\chr16.txt
 File Loaded
 chromosome sequence is clear
file=8C:\Users\admin\Desktop\project\gene_sequence_data\New\chr17.txt
 File Loaded
 chromosome sequence is clear
file=8C:\Users\admin\Desktop\project\gene_sequence_data\New\chr19.txt
 File Loaded
 chromosome sequence is clear
file=8C:\Users\admin\Desktop\project\gene_sequence_data\New\chr5.txt
 File Loaded
 chromosome sequence is clear
file=8C:\Users\admin\Desktop\project\gene_sequence_data\New\chr8.txt
 File Loaded
 chromosome sequence is clear
file=8
the count is2359
HER2 Gene is found to be clear
BRCA1/2 Gene is found to be clear
AKT1 Gene is found to be clear
APC Gene is found to be clear
ATM Gene is found to be clear
XRCC3 Gene is found to be clear
TP53 Gene is found to be clear
VEGF Gene is found to be clear
 The individual carrying DNA have rare chances of causing cancer
Used 2.36 seconds of CPU time.
Finished in about 2 seconds.
```

Figure 8 Resultant output of the pattern matching algorithm for DNA sequence (source: own)

V. CONCLUSION

From the six papers, presented for survey, it describes the advanced computational capability achieved through different algorithm. As the future work, we will use approximation algorithm with help of pigeonhole principle. The R analytics software will executive data fast and find DNA disorder into bits. These algorithms will sort extremely complex and unstructured data. Studies many papers presented for survey it describes the advanced computational capability achieved used different algorithm. In this Presentation we presented a string short sequence DNA data will separate alphabetic map the all element and read easy using Hadoop MapReduce algorithm. Approximate algorithm in Pigeonhole Principle to search and mismatch string of nucleotide in DNA structure to store pairing of element into each block. As of future work if molecular science has brought up new study and results about cancer genes for other types of cancer disease. This work can be extended for all types of cancers. So we will create new tools which will do fast alignment sequencing data with searching, insert and delete element which used base on pigeonhole algorithm. These algorithms will sorts extremely complex and unstructured data arrange easily and analysis the percentage of cancer disease.

REFERENCES

[1] Izzat Alsmadi and Maryam Nuser, String Matching Evaluation Methods for DNA Comparison, Vol. 47,

October, 2012, International Journal of Advanced Science and Technology

[2] LI Xu-bin, JIANG Wen-rui, JIANG Yi, ZOU Quan*, “Hadoop Applications in Bioinformatics”, 2012 7th Open Cirrus,Summit,978-0-7695-4908-8/12$26.00 © 2012 IEEE DOI 10.1109/OCS.2012.40

[3] Gang Liao, Longfei Ma, Guangming Zang, Lin Tang, Parallel DC3 Algorithm for Suffix Array Construction on Many-core Accelerators

[4] Aryan Arbabi, Milad Gholami, Mojtaba Varmazyar, Shervin Daneshpajouh, Fast CPU-Based DNA Exact Sequence Aligner, 978-1-4673-1313-1/12/$31.00 ©2012 IEEE

[5] Da Li, Michela Becchi, Multiple Pairwise SequencesAlignments with Needleman-Wunsch Algorithm on GPU

[6] Chad Nelson, Kevin Townsend, Bhavani Satyanarayana Rao, Phillip Jones, Joseph Zambreno, Shepard: A Fast Exact Match Short Read Aligner

[7] *Sophie Schbath,1 *VE´ Ronique Martin,1 Matthias Zytnicki,2 Julien Fayolle,1 Valentin Loux,1 and Jean-Franc¸ OIS Gibrat1, Mapping Reads on a Genomic Sequence: An Algorithmic Overview and a Practical Comparative Analysis, Journal of computational biology, Volume 19, Number6,2012# Mary Ann Liebert, Inc. Pp.796–813, DOI:10.1089/cmb.2012.0022

[8] Manber, “Finding similar files in a large file system [C/OL]”, In: Proceedings of the Winter USENIX Conference, (1994), pp. 1-10.

[9] Wei Wang, Juan Liu* “Distinguishing Single-Stranded and Double-Stranded DNA binding Proteins Based on Structural Information”, 978-1-4799-1310-7/13/$31.00 ©2013 IEEE, 2013 IEEE International Conference on Bioinformatics and Biomedicine.

[10] Ka Kit Lam and Nihar B. Shah, Towards Computation, Space, and Data Efficiency in de novo DNA Assembly: A Novel Algorithmic Framework.

[11] Gang Liao, Qi Sun, Longfei Ma, Zhihui Qin, GPUAccelerated Multiple Deoxyribose Nucleic Acid Sequence Parallel Matching , arXiv:1303.3692v1 [cs.DS] 15 Mar 2013

[12] Wei-Chun Chung †‡, Yu-Jung Chang , D. T. Lee ‡§, Jan-Ming Ho †, Using Geometric Structures to Improve the Error Correction Algorithm of High-Throughput Sequencing Data on MapReduce Framework, 2014 IEEE International Conference on Big Data, 978-1-4799-5666-1/14/$31.00 ©2014 IEEE

[13] LI Xu-bin, JIANG Wen-rui, JIANG Yi, ZOU Quan*, Hadoop Applications in Bioinformatics, 2012 7th Open Cirrus Summit, IEEE Computer Society, 978-0-7695-4908-8/12 $26.00 ©2012 IEEE, DOI 10.1109/OCS.2012.40

[14] Cancer Genomics: What Does It Mean for You?, The Cancer Genome Atlas (TCGA), NIH Publication no. 10-7556 Printed July-2010

[15] Snehal P. Adey, Dr. Vandana Inamdar, GPU Accelerated Pattern Matching Algorithm for DNA Sequences to Detect Cancer using CUDA,Department of Computer Engineering and Information Technology.