Processing Very Large Genomic Files

Michael Robinson
School of Computer Information Science
Florida International University
Miami, Florida, USA
michael.robinson@cs.fiu.edu

***Abstract*—**We have developed a computational framework called GenoPro that allows us to process very large genomic files from multiple formats such as NGS, fasta, and GBK, extracting DNA, RNA and Protein sub-sequences of any length, limited by the storage size or operating system.

GenoPro creates a data structure that contains sub-sequences of any size used to develop new tools that allow us to understand not only our human genome but any life form's genome.

Keywords – Gigabytes, Terabytes, Genome, Sub-Sequence.

I. Introduction

The Genome Project started in 1990 and completed in April 2003 sequenced the human genome producing files of about 3.2 Gigabytes. During the next 10 years a new sequencing method called Next Generation Sequencing (NGS) was created producing files of over 1.4 Terabytes. It is expected to produce files larger than 10 Terabytes in size. In addition there are many other current projects such as the 1000 Genomes Project and the manufactures of the NGS machines sequence many genomes with data in the Petabytes. Just like in many other disciplines, Biology has very large amounts of Genomic data that needs to be studied.

Out of the 3.2 billion nucleotides in the human genome over 95% is unknown and we call it dark matter[1]. The recent completed five-year project called Encyclopedia of DNA Elements (ENCODE) concluded that about 80 percent of the human genome is active and it is not just junk DNA as it was previously called. If ENCODE was presented in graphical form, the data it has generated so far would fill a poster 30 kilometers long and 16 meters high[2]. There is a great amount of information and knowledge to be discovered from these results.

In this modern age of genomics vast quantities of information are generated but are associated with uncertainty in significance. Biologists do not have the wherewithal to identify computationally the portions of the genome that are significant to biologic disease processes and the information science experts do not have enough knowledge of these biologic processes to accurately model how to extract the relevant information. This difficulty is present in a setting of ever increasing production of data. Therefore, any efforts between biologists/clinicians and computational experts to better understand genomic data will be of significant potential benefit to patient care and survival. It is our goal to provide a data structure that will allow scientists with little computational knowledge to find patterns and discover new information that can be proven in their biological labs. GenoPro will also help Computational Scientists develop sophisticated new tools to solve more complex tasks.

II. Previous Work

One the major challenges in studying the vast amount of Genomic data available is to arrange it in a way that is easy to use by non-Computational Scientists, our goal has been to create a framework that will do that.

Discovery and Annotation of Repeats, Signatures, and Patterns in Genomic Sequences[3] uses the new biobuckets sort method to produce indexed data files of extracted sub-sequences of length up to 5 bases, to find Signatures among genomes.

Finding Repeats and Signatures in Genomic Sequences[4] is able to find exact signatures between five strains of Pseudomonas Aeruginosas bacteria utilizing MPI Clusters, totaling 20 comparisons. Each set of genomes had pre-processed indexes which we created using Suffix Arrays.

Discovery and Annotation of Repeats, Signatures, and Patterns in Genomic Sequences[5] uses own framework can find exact Signature of any size from Genomes of any size.

Discovery and Annotation of Repeats, Signatures, and Patterns in Genomic Sequences[6]. Creation of new Framework to extract sub-sequences of any size from Genomes of any size, applied to Signatures and Repeats.

Applied Genomic Signatures and Patterns[7] shows how to find signatures of any length from input data files of any size.

Discovering Unknown Genes[8] uses the GenoPro framework to processes raw Genomic Data of any size.

III. Methods

Input Data Files: Our framework's algorithms have been implemented in Java using genomic data text files of multiple

formats and sizes as input as seen in TABLES I-III. Some of these input files can be larger than one Terabyte in size.

TABLE I. FASTA GENOMIC DATA SAMPLE

Fasta Format: >gi|88944472|ref|NW_921350.1| Homo sapiens chromosome 1 genomic contig, alternate assembly Hs_Celera 211000035831330, whole genome shotgun sequence TCACCTGGGTGTGTGGGTGCCGTTCCAGGCTGTCAGAGCT CGCGTGGGGGTGTGGGTGCTGCTCCAGGCT

TABLE II. NGS GENOMIC DATA SAMPLE

NGS Format: @ERR030881.107 HWI-BRUNOP16X_0001:2:1:13663:1096#0/2 CGGATTTCAGCTACTGCAAGCTCAGTACCACACAAGCTCG ATGTG + HH;HHHHHGHHHHHHHHHHGHDHE

TABLE III. GBK GENOMIC DATA SAMPLE

GBK Format: LOCUS NC_020912 6421010 bp DNA circular CON 11-JUN-2013 DEFINITION Pseudomonas aeruginosa B136-33, complete genome.

User Input Batch Files: The files described in TABLE IV are created by the users with any editor like notepad or VI and they can have any name. These batch files will contain the location and names of the input DNA, RNA or Protein files to be processed, the length of the subsequences to be extracted from these files, the amount of RAM needed to process each Genomic input file, and all necessary documentation information about the job being processed. It can contain unlimited amount of lines written in any sequence. Documentation lines start with one or more spaces or double slashes //. These lines will be ignored by our framework and are used to describe the job at hand.

TABLE IV. User Batch File

//===== Hospital ABC inv#4523 ===== // data received on dd/mm/yy /hs/brain_1.fastq 36 500,000,000 /hs/brain_1.fastq 200 900,000,000 /hs/brain_2.fastq 87 500,000,000 /hs/brain_2.fastq 100 750,000,000

Our framework first accepts the previous User Input Batch File and then parses its contents to determine how to process each input data file. By examining each Genomic input data file, we determine the data type each file holds, such as fasta, fastq/ngs, or gbk, and then we process them accordingly.

First we extract the DNA, RNA or Proteins from these files as shown in TABLE V.

TABLE V. EXTRACTED INPUT DATA

TCACCTGGGTGTGTGGGTGCCGTTCCAGGC AGAGCTCGCGTGGGGGTGTGGGTGCTGCTG CTTCACCTGGGTGTGTGGGTGCCGTTCCAG AGAGCTCGCGTGGGGGTGTGGGTGCTGCTG

Then we partition this data into sub-sequences of any user selected length.

TABLE VI. SUB-SEQUENCES OF LENGTH 15

TCACCTGGGTGTGTG CACCTGGGTGTGTGG ACCTGGGTGTGTGGG CCTGGGTGTGTGGGT CTGGGTGTGTGGGTG TGGGTGTGTGGGTGC GGGTGTGTGGGTGCC GGTGTGTGGGTGCCG…..

Validation: Using the following formula we validate the total amount of extracted sequences from each file, where SeqLen denotes the length of the user selected sequence and n the length of the original genomic sequence.

$$\text{SeqLen} * \sum_{n=1}^{n+1-\text{SeqLen}} \text{SeqLen}$$

In TABLE VII we show the small sequence “ACGTACG“ being partitioned into sub-sequences of length 2. When we apply the previous formula to this small sequence we validate that the total amount of bases/bytes in the sub-sequence file corresponds to its source file, in this case the small sequence “ACGTACG“.

TABLE VII. SUB-SEQUENCES OF LENGTH 2

ACGTACG	7
AC	2
CG	2
GT	2
TA	2
AC	2
CG	2

As we produce these sub-sequences we create a new text file with their locations in the original input file. We name this new file using the genome's name, data format, and sub-sequence length. If the genome's file name is example, of fasta data type and sub-sequences of length 2, its name will be sample_fa_2

TABLE VIII. SUB-SEQUENCES OUTPUT FILE SAMPLE_FA_2

SUB-SEQUENCES	Location
AC	0
CG	1
GT	2
TA	3
AC	4
CG	5

Sorting: Our previous unsorted file becomes the input for the next step which sorts it contents alphabetically creating a new text file. We call this method “external sort” which allows us to sort files of any size, limited only by the size of the external storage devices used.

The name for the new output file is the name of the input file plus a dot and the word “sorted".

TABLE IX. SORTED SUB-SEQUENCES OUTPUT FILE SAMPLE_FA_2.SORTED

SUB-SEQUENCES	Location
AC	0
AC	4
CG	1
CG	5
GT	2
TA	3

These sorted files can be in the thousands of gigabytes in size.

Our external sort needs two additional external files, each with the maximum space equal to the original sub-sequences file. The following describes in detail our external sort algorithm used in our framework.

In the initial user input file called “User Input Batch Files” and used to pass information to our framework, we pass the amount of RAM requested for each input data file. If the RAM size requested is 2 gigs, we find the length of each record in each input file and divide 2 gigs by the sub-sequence length finding the amount of records we can have in RAM at one time in each sorting step.

The first process in our external sort, see Figure I, reads the amount of records that fit inside the reserved RAM, placing them into an Array List and then using the Java Collection.Sort, an in-place sort, we sort this Array List and proceed to write this sorted data into a new external file, see TABLE IX, named using the genome's name plus the sub-sequence length, a period and the word sorted, from now on referred as the *.sorted file. This file will contain all the sorted data. This process is used only once.

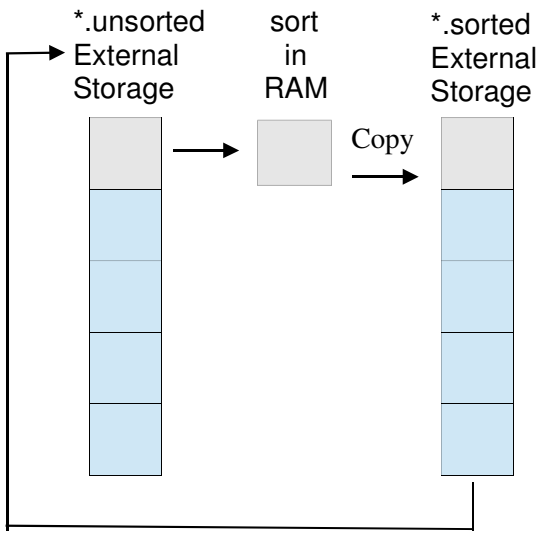

Figure I. First Step External Sort

We now start repeating the following steps until we process the entire input file, see Figure II.

1 - Clear the Array List and re-load it with the next batch of records from the input file and sort them.

2 - Create an empty file called temp. At this time we have three files: the input file, the sorted file, and the temp file in addition to the sorted data in RAM.

3 - Merge the data in RAM with the data in the *.source file into the temp file as follows:

a) Read the first record from RAM and the first record in the sorted file, compare them and write the smallest record into the temp file. These records cannot be equal because the input file does not have duplicate records; if the sub-sequences are equivalent the locations are different.

b) Read the next record from the file of the record written to the temp file, compare the two records and write the smallest one to the temp file until one of the two files is totally processed.

c) Add the remaining records of the other file to the temp file.

d) Close both files.

e) Rename the temp file with the name of *.sorted file. At this time the *.sorted file will contain the sorted records from the temp file and the temp file will not exist.

f) Continue this process until the input file is totally processed.

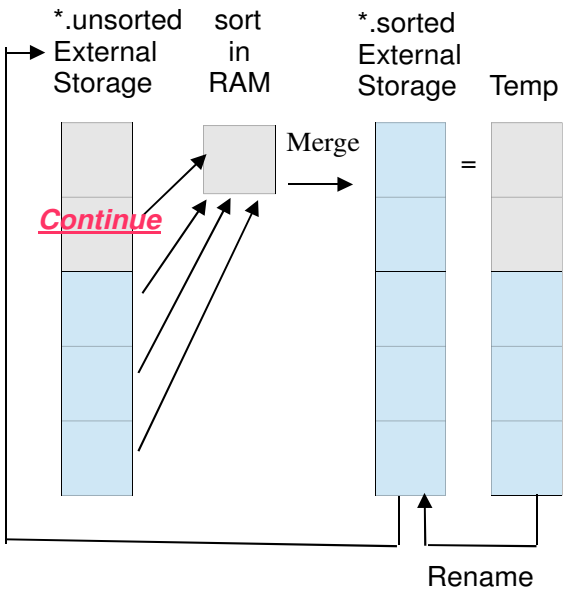

Figure II. Merge Step External Sort

Validation: The unsorted input file must have the same amount of records of the sorted output file.

The sorted output file must not have duplicate records. The record is made of two fields, the sub-sequence and its index. Indexes are unique therefore there should not be duplicate records.

Reduction Step: This last step generates the final files that will be used to create multiple applications to be used for further research.

In Figure III the *.sorted column represents the *.sorted External Storage column of Figure II. The *.sorted column represents a data files with two columns. The first column, called the key column, contains the sorted sub-sequences and the second column, called the values column, is the location, in the original DNA, RNA, or Protein file, where each sub-sequence is found.

The Reduced Sorted column in Figure II represents a file where we create unique entries/records placing in the first column, the key column, each unique sub-sequence, and in the second column, the values column, we merge all locations sorted in ascending order.

The Reduced Sorted Counted column represents one of the three final data structures. At the beginning of each record in the values column we have added the total amount of locations for each unique sub-sequences found in the source input DNA, RNA or Protein file.

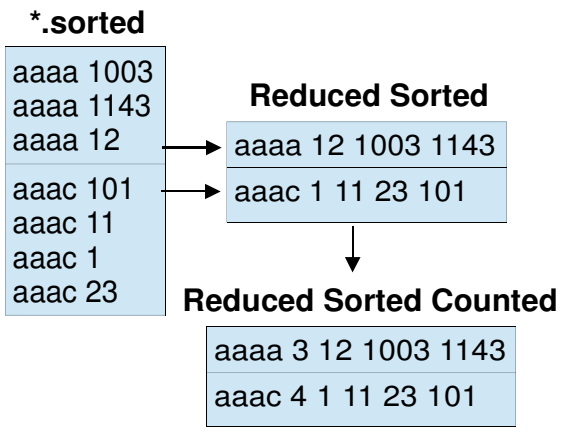

Figure III. Reduction Step

Splitting File: In our final step we created two new files Sub-sequences/keys and Locations/Values, shown in Figure IV. These two files are very useful in developing additional applications to extend our research.

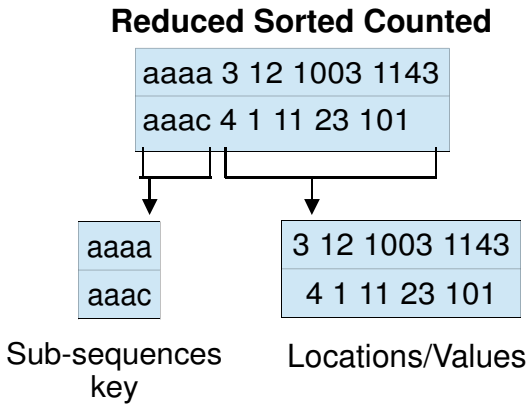

Figure IV. Splitting File Step

Using the file containing the sub-sequences key we have developed an application called Signatures that allows us to find the differences (signatures) between any types of life forms using sub-sequences of any size. The Locations/Values file helps us to develop an application to find the distances between repeats of unique sub-sequences and predict where the repeats can be found.

IV. Results

Our Framework converts raw data files into files containing large amount of well-organized information, which is very easy to use in the development of new applications that will further our research. We created several applications using our framework. We created several applications using our framework.

Signatures: Computationally we performed exact comparisons of DNA, RNA and amino acid sequences of any length, using multiple biological life forms. Finding signatures in multiple genomes, bacteria or viruses is the process of discovering what sub-sequences exist in one sample but not in the other, which showed us the differences between samples. We selected many types of Genomes and extracted sub-sequences of multiple lengths creating their corresponding data structures, then using our signatures application we compared each sample against the others finding their Signatures.

Gene Prediction: By examining the dark matter areas in the above bacteria, we can predict the location of currently unknown genes.

Disease Progress: We were able to compare sub-sequences from the same biological subjects at different times to track changes.

Genomic Fingerprints Libraries: Using genomes of organisms that can potentially be used as biological weapons, we can create libraries of genetic fingerprints to be used in the rapid detection of bio-warfare threats and take corrective and immediate medical measures.

Creating genetic fingerprints libraries of known bacteria, viruses, and tumors, which are currently cured with known treatments or medications, will allow us to search for those fingerprints in other genomes and apply the known cures to new diseases. These fingerprints can also be useful to identify the source of drug resistance.

Predict Repeat Locations: Since we know the locations of all sequence repeats, we were able to mathematically predict the possible repeat locations in the above bacteria.

V. REFERENCES

[1] B. Franklin Pugh, Bryan Venters. The Origins of Genomic Dark Matter, 2013

[2] Malcolm Ritter, Associated Press DNA research may offer clues into disease, 2012

[3] Michael Robinson, Discovery and Annotation of Repeats, Signatures, and Patterns in Genomic Sequences. Michael Robinson et al. 5th International Symposium on Bioinformatics Research and Applications (ISBRA09), Nova Southeastern University, Ft. Lauderdale, Florida, USA. May 2009.

[4] Michael Robinson, Guangyuan Liu, Camilo Silva, Masoud Sadjadi, Hector Duran4 and Giri Narasimhan. Finding Repeats and Signatures in Genomic Sequences. ISBRA May 2009.

[5] Michael Robinson, S.S.Iyengar and Puneeth Iyengar Discovery and Annotation of Repeats, Signatures, and Patterns in Genomic Sequences. SBEC 2013 29th Southern Biomedical Engineering Conference. 2013

[6] Michael Robinson. Discovery and Annotation of Repeats, Signatures, and Patterns in Genomic Sequences Journal “unpublished”

[7] Michael Robinson. Applied Genomic Signatures and Patterns. “unpublished”

[8] Michael Robinson. Discovering Unknown Genes. “unpublished”