आई. ए. एस. आर. आई./पी. आर. -11/2019
I.A.S.R.I./P.R.-11/2019
Institute Project Code: AGENIASRISIL201700087
संस्थान परियोजना कोड: AGENIASRISIL201700087

परियोजना रिपोर्ट

PROJECT REPORT

एक बेहतर हाइब्रिड डी-नोवो पूर्ण जीनोम असेंबलर का विकास

Development of an Improved Hybrid *De-novo* Whole Genome Assembler

हर कदम, हर डगर
किसानों का हमसफर
भारतीय कृषि अनुसंधान परिषद

Agrisearch with a human touch

शशि भूषण लाल	Shashi Bhushan Lal
अनु शर्मा	Anu Sharma
संजीव कुमार	Sanjeev Kumar
द्विजेश चन्द्र मिश्र	Dwijesh Chandra Mishra
नीरज बुढ़लाकोटि	Neeraj Budhlakoti

कृषि जैवसूचना केन्द्र

CENTRE OF AGRICULTURAL BIOINFORMATICS

भा कृ अनु प –भारतीय कृषि सांख्यिकी अनुसंधान संस्थान
लाइब्रेरी एवेन्यू, पूसा नई दिल्ली–110012

ICAR-Indian Agricultural Statistics Research Institute
Library Avenue, Pusa, New Delhi – 110012

भाकृअनुप
ICAR

2019

आमुख

कंप्यूटर विज्ञान में उन्नति ने भारी कम्प्यूटेशनल कार्यों को आसान और तेज करना संभव बना दिया है। जैविक शोधकर्ताओं ने पाया है कि आधुनिक उच्च कम्प्यूटेशनल क्षमता विश्लेषण और प्रसंस्करण को तेज करती है। जैविक डेटा का आकार तेजी से बढ़ रहा है जो उच्च-प्रदर्शन कंप्यूटिंग की आवश्यकता को अधिक से अधिक बढ़ाता है। जीवों की जीनोम असेंबली प्रक्रिया जटिल संगणना के निषेध के कारण एक बहुत ही जटिल कम्प्यूटेशनल कार्य हो गया है। असेंबली के लिए उपलब्ध विविध प्लेटफार्मों की अपनी खूबियां और अवगुण हैं। ये अनुक्रमण मंच एक उच्च खंडित परिणाम उत्पन्न करते हैं। डी-नोवो जीनोम असेंबली में सैकड़ों गीगाबाइट मेमोरी और लाखों सीपीयू घंटे की आवश्यकता होती है।

अधिकांश उपलब्ध जीनोम असेंबलर शॉर्ट-रीड्स के लिए डी-ब्रूजन ग्राफ (डीबीजी) और लंबी रीडिंग के लिए ओवरलैप लेआउट कंसेंसस (ओएलसी) पर आधारित हैं। यह सर्वविदित है कि शॉर्ट रीड असेंबली में कम त्रुटियां हैं लेकिन भारी कम्प्यूटेशनल हैं, हालांकि, ओवरलैप लेआउट सर्वसम्मति (ओएलसी) विधि महंगा है और इसमें **30%** तक त्रुटियां होती हैं। कई मामलों में जीनोम असेंबलर विभिन्न अंतर्निहित जैविक और कम्प्यूटेशनल मुद्दों के कारण उच्च गुणवत्ता वाले इकट्ठे जीनोम का उत्पादन करने में विफल होते हैं। ये असेंबलर ज्यादातर प्लेटफॉर्म पर निर्भर करते हैं, और विभिन्न सीक्वेंसिंग प्लेटफॉर्म से आने वाले डेटा के संयोजन को संभालने में असमर्थ होते हैं। विभिन्न उच्च थ्रूपुट अनुक्रमण प्लेटफार्मों से आने वाले हाइब्रिड डेटा को संभालना इस परिस्थिति को और कम्प्यूटेशनल जटिलता को कई गुना बढ़ा देता है। अगर इन्हें संयुक्त रूप से प्रयुक्त किया जा सके तो, तो ये अनुक्रमण मंच बहुत उपयोगी हो सकते हैं।

एक मॉड्यूलर एवं कम्प्यूटेशनल रूप से कुशल हाइब्रिड डी-नोवो पूर्ण जीनोम असेंबलर, परिणाम के रूप में बेहतर जीनोम का उत्पादन करने के लिए, डीबीजी और ओएलसी दोनों तरीकों के गुणों का लाभ उठा सकते हैं। असेंबली का एक हाइब्रिड तरीका है, जिसमें लंबी रीडिंग पर छोटे रीड्स को संरेखित करके लंबी रीडिंग पर त्रुटियों को ठीक किया जा सकता है, जो जीनोम असेंबली के लिए उपयोगी हो सकता है। एक बेहतर डी-नोवो जीनोम असेंबलर क्रॉस प्लेटफॉर्म से आने वाले डेटा को कुशलतापूर्वक और सटीक रूप से इकट्ठा करता है। लोकल एरिया नेटवर्क पर उपयोग के लिए एक वेब-आधारित सॉफ्टवेयर जो कि स्टेप वाइज प्रक्रिया जैसे - अनुक्रमों का पूर्व प्रसंस्करण, लंबी रीडिंग पर त्रुटि सुधार के लिए संरेखण, सही लंबी रीडिंग की असेंबली और स्कैफ़ोल्डिंग कर सकता है, विकसित किया गया है। यह सॉफ्टवेयर जैव सूचना विज्ञानियों के लिए उपयोगी होने की सम्भावना है।

लेखकगण

PREFACE

The advancement in the computer science has made it possible to perform heavy computational tasks easier and quicker. Biological researchers have found that modern high computational capability makes the analysis and processing very fast. Biological data size is growing exponentially which enhances the need of high-performance computing more and more. Genome assembly of organisms is a very complex computational task because of inhibition of complex computation in the process. Varied platforms available for assembly have their own merits and demerits. These sequencing platforms produce a highly fragmented result. *De-novo* genome assembly requires hundreds of gigabytes of memory and millions of CPU hours.

Most of the available genome assemblers are based on de-Bruijn Graph (DBG) for short reads and Overlap Layout Consensus (OLC) for long reads. It is well known that short read assembly has less errors but heavily computational, however, overlap layout consensus (OLC) method is costlier and having errors up to 30%. Genome assemblers in many cases fail to produce high quality assembled genome due to various inherent biological and computational issues. These assemblers are mostly platform dependent, but unable to handle combination of data coming from various sequencing platforms. Handling hybrid data coming from various high throughput sequencing platforms further exacerbate this situation and increase computational complexity manifolds. If combined, these sequencing platforms can be very useful.

A modular, computationally efficient hybrid *de-novo* whole genome assembler can take advantage of merits of both DBG and OLC based methods to produce better genome as result. A hybrid method of assembly that can correct errors on long reads by aligning short reads over long reads can be useful for genome assembly. An improved *de-novo* genome assembler can efficiently and accurately assemble the data coming from cross platforms. A web-based software for use on LAN that can do the stepwise process namely – pre-processing of sequences, alignment for error correction on long reads, assembling corrected long reads and scaffolding have been developed. This software is expected to be very useful for bioinformaticians.

AUTHORS

LIST OF FIGURES

LIST OF TABLES

LIST OF LISTINGS

ABBREVIATIONS

1.	API	Application Program Interface
2.	APL	Application Layer
3.	ASHOKA	Advanced Super Hub for Omics Knowledge in Agriculture
4.	ASP	Active Server Pages
5.	BAM	Binary Alignment Map
6.	bp	Base pair
7.	BSD	Berkeley Software Distribution
8.	CPU	Central Processing Unit
9.	CSS	Cascading Style Sheet
10.	CUDA	Compute Unified Device Architecture
11.	DBG	de-Bruijn Graph
12.	DBL	Database Layer
13.	DDL	Data Definition Language
14.	DML	Data Manipulation Language
15.	DNA	Deoxy Ribonucleic Acid
16.	GB	Giga Byte
17.	HPC	High Performance Computing
18.	HTML	Hyper Text Markup Language
19.	HTTP	Hyper Text Transfer Protocol
20.	IDE	Integrated Development Environment
21.	IEEE	Institute of Electrical and Electronics Engineers
22.	JDBC	Java Database Connectivity
23.	JIT	Just-in-Time
24.	JSP	Java Server Pages
25.	JVM	Java Virtual Machine
26.	KB	Kilo Byte
27.	LAN	Local Area Network

28.	MPI	Message Passing Interface
29.	OLC	Overlap Layout Consensus
30.	PCR	Polymerase Chain Reaction
31.	POP	Post Office Protocol
32.	RAM	Random Access Memory
33.	SAM	Sequence Alignment Map
34.	SIMD	Single instruction, multiple data
35.	SMP	Symmetric Multiprocessing
36.	SMRT	Single Molecule Real-Time
37.	SMTP	Simple Mail Transfer Protocol
38.	SQL	Structured Query Language
39.	SSH	Secure SHell
40.	SSL	Secure Socket Layer
41.	TCP/ IP	Transmission Control Protocol/Internet Protocol
42.	UIL	User Interface Layer
43.	XML	Extensible Markup Language

CHAPTER I

INTRODUCTION

De-novo genome assembly of any sequenced organism is very important for researchers but this task is very complex because of inhibition of heavy computation required in the process. There are varied platforms available for *de-novo* assembly with their own merits and demerits. These sequencing platforms produce highly fragmented result. Table 1 shows the most popular sequencing platforms with respect to their read lengths, accuracy, advantages and disadvantages. But none of the available sequencing platforms ensure all these parameters. Assembling any genome requires proper combination of high coverage, long read length, and good read quality.

Table 1.1: Description of the available sequencing platforms

Platform	Read length (bp)	Accuracy (single read not consensus)	Advantages	Disadvantages
Pacific Biosciences	> 20kb	85% single-read accuracy	Longest read length Faster run time	Low sequence yield High error rate (15%) Very expensive
Ion Torrent	up to 400 bp	98%	Less expensive Faster run time	Homopolymer errors Moderate error rate
454/Roche	700 bp	99.9%	Long read size Low sequence error. Low PCR duplication	Runs are expensive Homopolymer errors
Illumina	50-500 bp	99.9% (Phred30)	Potential for high sequence yield Low cost per base	High percentage of PCR duplication More run time
SOLiD	50+35 or 50+50 bp	99.9%	Low cost per base	Slower than other methods Has issues sequencing palindromic sequences

Assembly can be successful if an appropriate algorithm and an efficient software is available. Combining the data coming from various sequencing platforms can produce improved results. At present, available assemblers are mostly platform dependent and are unable to handle combination of data coming from across platforms. *De-novo* genome assembly is highly computationally intensive and requires hundreds of gigabytes of memory and millions of CPU hours. Handling hybrid data coming from various high throughput sequencing platforms further exacerbate this situation and increase computational complexity manifolds. Most of the available genome assemblers are based on de-Bruijn Graph (DBG) for short reads and Overlap Layout Consensus (OLC) for long reads. It is well known that short read assembly has less errors but requires huge computational resources. However, Overlap Layout Consensus (OLC) method is costlier with errors up to 30%. Therefore, there is an urgent need for a modular, computationally efficient and hybrid *de-novo* whole genome assembler which can take advantage of merits of both the methods to produce better genome as result. In this project, we proposed to build an improved *de-novo* genome assembler that can efficiently and accurately assemble the data coming from cross platforms. It's modular design will also allow user intervention at appropriate stages of sequence assembly.

1.1 Motivation and Objectives

There are many platforms available for genome sequencing. Genome assemblers in many cases fail to produce high quality assembled genome due to various inherent biological and computational issues. These assemblers are mostly platform dependent. But these assemblers are unable to handle combination of data coming from various sequencing platforms. The combination of various sequencing platforms' data can be very useful. In terms of computational complexity, its handling increases manifold. So, there was need to develop computationally efficient method for hybrid *de-novo* whole genome assembly. In view of the above, the following objectives have been formulated for this study.

Objectives

i. To develop an alternate algorithm for hybrid de-novo whole genome assembly
ii. To design and develop a hybrid de-novo assembler
iii. To evaluate the performance of developed hybrid de-novo assembler

1.2 Plan of Report

This report presents the work done under a research project for development of a genome assembler program. The web-based software was developed using JSP and other open source tools. This report is divided into five chapters. Chapter-I of this report presents a brief introduction to the problem and objective of the study. Chapter-II deals with review of literature on the various latest work on hybrid assembler. Software Architecture and Design is given in Chapter-III. Chapter–IV deals with the stepwise assembly process that includes pre-processing of input sequences, alignment for error correction on long reads, assembling long reads, and scaffolding. At last, the references are mentioned that have been referred during the investigation.

CHAPTER II

REVIEW OF LITERATURE

The genome assembly from sequence reads is an algorithm-driven automated process. DNA-sequence-assembly programs have utilized sequence overlaps for sequence assembly in correct order. The computational aspect of assembly algorithms become important while actually an assembler is to be developed to get the results in reasonable amount of time. There are two major classes of assembly algorithms: Overlap Layout Consensus and de-Bruijn Graph, from how they match the Lander–Waterman model, to the required sequencing depth and reads length. The computational efficiency of each class of algorithm, the influence of repeats and heterozygosity and points of note in the subsequent scaffold linkage and gap closure steps. Given below are few reviews of various algorithm variants for developing hybrid assemblers. These are the recent years findings by the researchers to assemble genome in a specific way to efficiently produce assembled genome.

1. A new strategy for assembling genomic shotgun and EST sequence data was developed by Chevreux (2005). It combines novel enhancements like repeat detection and on-the-fly automatic editing with strengths of existing assemblers. The strategy also provides the assembler with the ability to use and - more importantly - to acquire by itself additional knowledge present in the assembly data. Furthermore, the knowledge acquisition was combined with the ability to resolve potential conflicts - like long term repeats in genome sequencing projects or different mRNA transcripts in EST projects - during the assembly by falling back to trace signal analysis routines.

2. Koren *et al.*, 2012 have developed a method for hybrid error correction and *de-novo* assembly of single-molecule sequencing reads. *De-novo* assemblers use the long-read sequencing data with considerable error rate. This approach compliments with shorter, high-identity sequences resulting in long contigs, accurate transcripts and improved assemblies. Because the average contig size produced by this approach correlates with read length, assembly results are expected to improve as the read lengths of the technology improve.

3. Liu *et al.*, 2012 have developed CUSHAW: a Compute Unified Device Architecture (CUDA) compatible short read aligner to large genomes based on the Burrows-Wheeler Transform (BWT). It exploits CUDA-compatible graphics hardware as accelerators to achieve fast speed. Algorithm uses a quality-aware bounded search approach based on the BWT and the Ferragina-Manzini Index to reduce the search space and achieve high alignment quality.

4. Wang *et al.*, 2012 have developed a pipeline for hybrid assembly consisting of primary and secondary assembly steps. In primary assembly, the 454, GAIIx, or SOLiD reads were assembled into contigs with Newbler 2.0.01.14, Velvet1.1.04, or Denove2.2, respectively. In secondary assembly, contigs from primary assembly step were merged into consensus genome draft with Phrap.

5. Lee *et al.*, 2014 have developed a novel hybrid error correction algorithm for long PacBio sequencing reads that uses pre-assembled Illumina sequences for the error correction.

6. Salmela and Rivals, 2014 have developed LoRDEC, a hybrid error correction method that builds a succinct DBG representing the short reads, and seeks a corrective sequence for each erroneous region in the long reads by traversing chosen paths in the graph. LoRDEC is claimed to be six times faster and requires at least 93% less memory or disk space than available tools, while achieving comparable accuracy.

7. Utturkar *et al*., 2014 have developed a method of hybrid assembly of Illumina and Roche 454 data. 454-Illumina hybrid assembly approach involved merging the 454-only assembly with Illumina reads by PHRAP. They have also developed a method of hybrid assembly of Illumina, 454 and PacBio data.

8. Li *et al.*, 2015 have presented a new mapper, minimap, and a *de-novo* assembler, miniasm, for efficiently mapping and assembling Single Molecule Real-Time (SMRT) sequencing technology and Oxford Nanopore Technologies (ONT) reads without an error correction stage.

9. Lin and Liao, 2015 have developed a hybrid assembly approach, in which microbial genomes were realized by correcting PacBio long reads using ECTools and subsequently assembling, *de-novo*, the corrected long reads.

10. H. Backman and Girke, 2016 developed R/Bioconductor package systemPipeR. It is an extensible environment for both building and running end-to-end analysis workflows with automated report generation for a wide range of NGS applications. Its unique features include a uniform workflow interface across different NGS applications, automated report generation, and support for running both R and command-line software on local computers and computer clusters. A flexible sample annotation infrastructure efficiently handles complex sample sets and experimental designs. To simplify the analysis of widely used NGS applications, the package provides pre-configured workflows and reporting templates for RNA-Seq, ChIP-Seq, VAR-Seq and Ribo-Seq.

11. Miclotte *et al.*, 2016 have developed Jabba, in which hybrid method is used to correct long third generation reads by mapping them on a corrected DBG that was constructed from second generation data. It uses pseudo alignment approach with a seed-and-extend methodology, using maximal exact matches seeds.

12. Vaser *et al.*, 2016 have developed a fast and accurate *de-novo* genome assembly from long uncorrected reads. Here high-quality consensus sequences were generated without additional error correction steps, efficiently with a Single Instruction Multiple Data (SIMD) accelerated, partial order alignment based stand-alone consensus module called Racon.

13. Yeo *et al.*, 2017 developed ARCS, an application that utilizes the barcoding information contained in linked reads to further organize draft genomes into highly contiguous assemblies. It has been shown that how the contiguity of an ABySS H.sapiens genome assembly can be increased over six-fold, using moderate coverage (25-fold) Chromium data.

14. Wang *et al.*, 2018 developed a novel method leveraging a multi-string Burrows-Wheeler Transform with auxiliary FM-index to correct errors in long read sequences using a set of complementary short reads. We demonstrate that our method efficiently produces

significantly more high quality corrected sequence than existing hybrid error-correction methods. This method produces more contiguous assemblies, in many cases, than existing state-of-the-art hybrid and long-read only *de-novo* assembly methods.

1. Ye *et al.*, 2016 have developed DBG2OLC, an efficient assembly of large genomes using long erroneous reads of the third-generation sequencing technologies. This approach first maps DBG contigs to the long reads. Each long read is converted into an ordered list of contigs, termed compressed reads. Then it calculates overlaps between the compressed reads. The alignment is calculated using the anchors. Contained reads are removed and the reads are chained together in the best-overlap fashion, then it constructs the assembly backbone from the best overlaps. Further align all related reads to the backbone and calculate the most likely sequence as the consensus output.

2. Zimin *et al.*, 2016 have developed a method of hybrid assembly of the large and highly repetitive genome of Aegilops tauschii, a progenitor of bread wheat, with the mega-reads algorithm. It exploits the mega-reads algorithm. In this approach Low-error rate Illumina reads are used to build longer super-reads, which in turn are used to construct a database of all 15-mers in those reads. PacBio reads and super-reads are then aligned, using the 15-mer database. Inconsistent super-reads are discarded and the remaining super-reads are merged, using the PacBio read as a template, to produce pre-mega-reads. These are further merged to produce the final mega-reads and to generate linking mates across gaps.

CHAPTER III

SOFTWARE ARCHTECTURE AND DESIGN

The Software for hybrid assembly of genome sequences has been developed that can be used currently on LAN for convenience of the users. The programming language used for the development of web interface are Java Server Pages (JSP), Cascading Style Sheets (CSS) and Java. It has been developed on Intel Xeon based 64-bit computer with 3.20 GHz-clock speed, Microsoft Windows 7 Operating System and 16.0 GB RAM. NetBeans 8.0 Integrated Development Environment (IDE) with java development kit 1.8 has been used as a platform for development of the software.

3.1 Software Architecture

The web application has been developed using client-server architecture. It is broken into logical chunks called "tiers", where every tier is assigned a role. Traditional applications consist of single tier, which resides on the client machine, but web applications lend themselves to an n-tiered approach. In its most common form, the three tiers are presentation, application and database layers. A web browser is the first tier (presentation), an engine using dynamic web content technology (such as ASP, JSP) is the middle tier (application logic), and a database is the third tier (storage). The web browser sends requests to the middle tier, which services them by making queries and updates against the database and generates a user interface. The advantage of web application over windows-based application is that it is platform independent and doesn't need to be installed on individual machine. In this software, heavy computations are carried out over high-end Symmetric Multiprocessing (SMP) server or Advanced Supercomputing Hub for OMICS Knowledge in Agriculture (ASHOKA) cluster to get the computation faster and achieve the results in least possible amount of time. Data size of the input files are very big and they need to be uploaded separately to divide the process in separate activities.

3.2 Three-Tier Architecture of Assembler Software

This software is implemented as a layered structure with each layer corresponding to a different functionality. The three-tier architecture of the software is given in Fig. 3.1.

- User Interface Layer (UIL)

- Application Layer (APL)
- Database Layer (DBL)

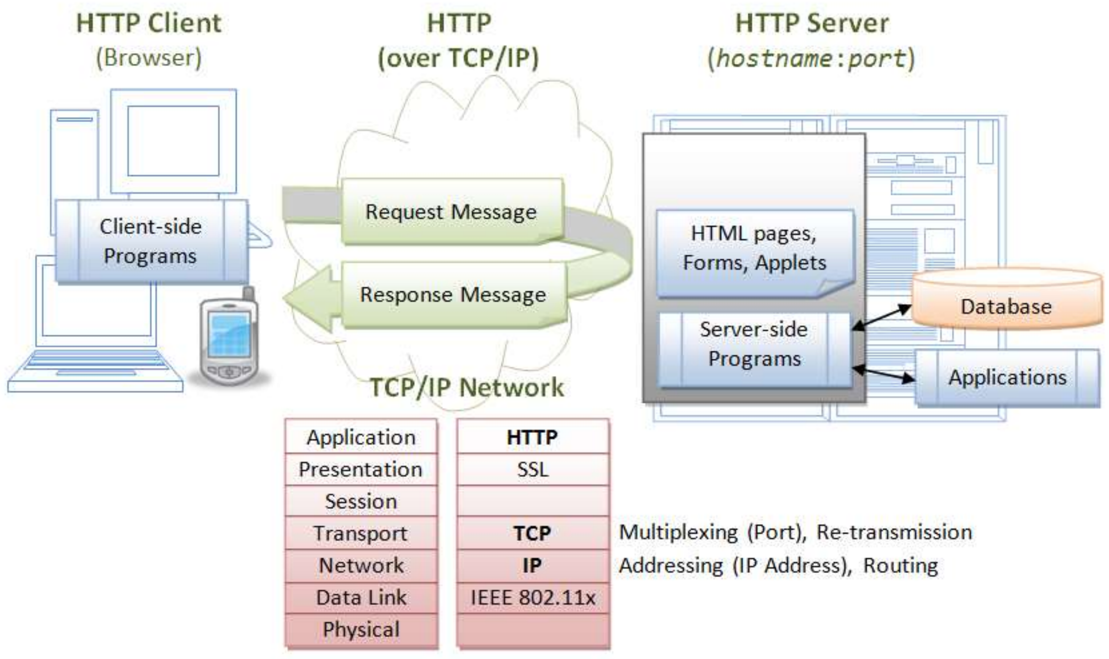


Fig. 3.1 Three Tier Architecture of software

User Interface layer (UIL)

The User Interface Layer has been implemented using Hyper Text Markup Language (HTML) and JavaScript. The UIL consists of forms for accepting information from the user and validating those forms using JavaScript.

Application Layer

Server-Side Application Layer has been implemented using Java Server Pages (JSP). JSP technology is the Java platform technology for delivering dynamic content to web clients in a portable, secure and well-defined way. The JSP specification extends the Java Servlet Application Programming Interface (API) to provide web application developers with a robust framework for creating dynamic web content on the server using HTML, and Extensible Markup Language (XML) templates, and Java code, which is secure, fast, and independent of server platforms. JSP has been built on top of the Servlet API and utilizes Servlet semantics. JSP has become the preferred request handler and response mechanism.

Database Layer (DBL)

Database Layer has been implemented using MySQL. It is used for designing the tables for storing set of preferred codons for some organisms and users' profiles. The relational database approach has been used to design the database. The fundamentals of normalization theory have been used to normalize the different tables of the database. All tables have proper interaction among themselves via primary key - foreign key relationship.

3.3 Technologies Used for Development

3.3.1 Java Technology

Initially the language java was called as "oak" but it was renamed as "Java" in 1995. The primary motivation of this language was the need for a platform-independent language that could be used to create software to be embedded in various consumer electronic devices:

- Java is a programmer's language.
- Java is cohesive and consistent.
- Except for those constraints imposed by the Internet environment, Java gives the programmer, full control.
- Finally, Java is for Internet programming whereas C was for system programming.

3.3.1.1 *Importance of Java to the Internet Platform*

Java has had a profound effect on the Internet. This is because; Java expands the universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the Server and the Personal computer. They are: Passive information and Dynamic active programs. The Dynamic, Self-executing programs cause serious problems in the areas of Security and probability.

Applications and Applets: An application is a program that runs on our computer under the operating system of that computer. It is more or less like one creating using C or C++. Java's ability to create Applets makes it important. An Applet is an application designed to be transmitted over the Internet and executed by a Java –compatible web browser.

3.3.1.2 *Security*

Every time you that you download a “normal” program, you are risking a viral infection. Prior to Java, most users did not download executable programs frequently, and those who did scan them for viruses prior to execution. Most users still worried about the possibility of infecting their systems with a virus. In addition, another type of malicious program exists that must be guarded against.

3.3.1.3 *Portability*

For programs to be dynamically downloaded to all the various types of platforms connected to the Internet, some means of generating portable executable code is needed. As we can see, the same mechanism that helps ensure security also helps create portability. Indeed, Java’s solution to these two problems is both elegant and efficient.

3.3.1.4 *The Byte Code*

The key that allows the Java to solve the security and portability problems is that the output of Java compiler is Byte code. Byte code is a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine (JVM). That is, in its standard form, the JVM is an interpreter for byte code. Although Java was designed for interpretation, there is technically nothing about Java that prevents on-the-fly compilation of byte code into native code. Sun has just completed its Just in Time (JIT) compiler for byte code. When the JIT compiler is a part of JVM, it compiles byte code into executable code in real time, on a piece-by piece, demand basis. It is not possible to compile an entire Java program into executable code all at once, because Java performs various run-time checks that can be done only at run time. The JIT compiles code, as it is needed, during execution.

3.3.1.5 *Java Virtual Machine (JVM)*

Beyond the language, there is the Java virtual machine. The Java virtual machine is an important element of the Java technology. The virtual machine can be embedded within a web browser or an operating system. Once a piece of Java code is loaded onto a machine, it is verified. As part of the loading process, a class loader is invoked and does byte code verification makes sure that the code that's has been generated by the compiler will not corrupt the machine that it's loaded on. Byte code verification takes place at the end of the compilation process to make sure that is all accurate and correct. So, byte code verification is integral to the compiling and executing of Java code.

3.3.1.6 *Java Architecture*

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual
Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

3.3.1.7 *Compilation of Code*

When you compile the code, the Java compiler creates machine code (called byte code) for a hypothetical machine called Java Virtual Machine (JVM). The JVM is supposed to execute the byte code. The JVM is created for overcoming the issue of portability. The code is written and compiled for one machine and interpreted on all machines. This machine is called Java Virtual Machine.

3.3.1.8 *Simple*

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ programmer, learning Java will be even easier. Because Java inherits the C/C++ syntax and many of the object-oriented features of C++.

3.3.1.9 *Object-Oriented*

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank slate. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

3.3.1.10 *Robust*

The multi-platform environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and run time.

3.3.1.11 *Java Database Connectivity*

JDBC is a Java API for executing SQL statements. (As a point of interest, JDBC is a trade-marked name and is not an acronym; nevertheless, JDBC is often thought of as standing for Java Database Connectivity. It consists of a set of classes and interfaces written in the Java programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API. Simply put, JDBC makes it possible to do three things:

- Establish a connection with a database.
- Send SQL statements.
- Process the results.

3.4 Structured Query Language (SQL)

Structured Query Language (SQL) is the language used to manipulate relational databases. SQL is tied very closely with the relational model. In the relational model, data is stored in structures called relations or tables. SQL statements are issued for the purpose of:

Data Definition: Defining tables and structures in the database (DDL used to create, alter and drop schema objects such as tables and indexes).

Data Manipulation: Used to manipulate the data within those schema objects (DML Inserting, Updating, Deleting the data, and Querying the Database).

3.4.1 Data Definition

Defining tables and structures in the database (DDL used to create, alter and drop schema objects such as tables and indexes).

3.4.2 Data Manipulation

Used to manipulate the data within those schema objects (DML Inserting, Updating,
Deleting the data, and Querying the Database). A schema is a collection of database objects that can include: tables, views, indexes and sequences.

3.5 Java Server Pages (JSP)

Java server Pages is a simple, yet powerful technology for creating and maintaining dynamic-content web pages. Based on the Java programming language, Java Server Pages offers proven portability, open standards, and a mature re-usable component model. The Java Server Pages architecture enables the separation of content generation from content presentation. This separation not eases maintenance headaches; it also allows web team members to focus on their areas of expertise. Now, web page designer can concentrate on layout, and web application designers on programming, with minimal concern about impacting each other's work.

3.6 Features of JSP

3.6.1 Portability

Java Server Pages files can be run on any web server or web-enabled application server that provides support for them. Dubbed the JSP engine, this support involves recognition, translation, and management of the Java Server Page lifecycle and its interaction components.

3.6.2 Components

It was mentioned earlier that the Java Server Pages architecture can include reusable Java components. The architecture also allows for the embedding of a scripting language directly into the Java Server Pages file. The components current supported include Java Beans, and Servlets.

3.6.3 Processing

A Java Server Pages file is essentially an HTML document with JSP scripting or tags. The Java Server Pages file has a JSP extension to the server as a Java Server Pages file. Before the page is served, the Java Server Pages syntax is parsed and processed into a Servlet on the server side. The Servlet that is generated outputs real content in straight HTML for responding to the client.

3.6.3 Access Models

A Java Server Pages file may be accessed in at least two different ways. A client's request comes directly into a Java Server Page. In this scenario, suppose the page accesses reusable Java Bean components that perform particular well-defined computations like accessing a database. The result of the Beans computations, called result sets is stored within the Bean as properties. The page uses such Beans to generate dynamic content and present it back to the client.

Steps in the execution of a JSP Application:

- The client sends a request to the web server for a JSP file by giving the name of the JSP file within the form tag of a HTML page.
- This request is transferred to the Java Webserver. At the server-side Java Webserver receives the request and if it is a request for a jsp file server gives this request to the JSP engine.
- JSP engine is program which can under stands the tags of the jsp and then it converts those tags into a Servlet program and it is stored at the server side. This Servlet is loaded in the memory and then it is executed and the result is given back to the Java Webserver and then it is transferred back to the result is given back to the Java Webserver and then it is transferred back to the

3.7 Software Design

The design of the assembler software has been shown in the Fig. 3.2 that schematically shows the modules developed under the software. This software has four modules for file and user management, Pre-processing of input sequence data, alignment for error correction, assembly of corrected sequence and Scaffolding generation.

A separate library has been developed for connectivity to the ASHOKA supercomputing platform and the computations thereon. Reference of the library has been added in the main application for subsequent usage. These are developed as reusable components that can be utilized in other Java based applications of various types like windows, web application and web services.

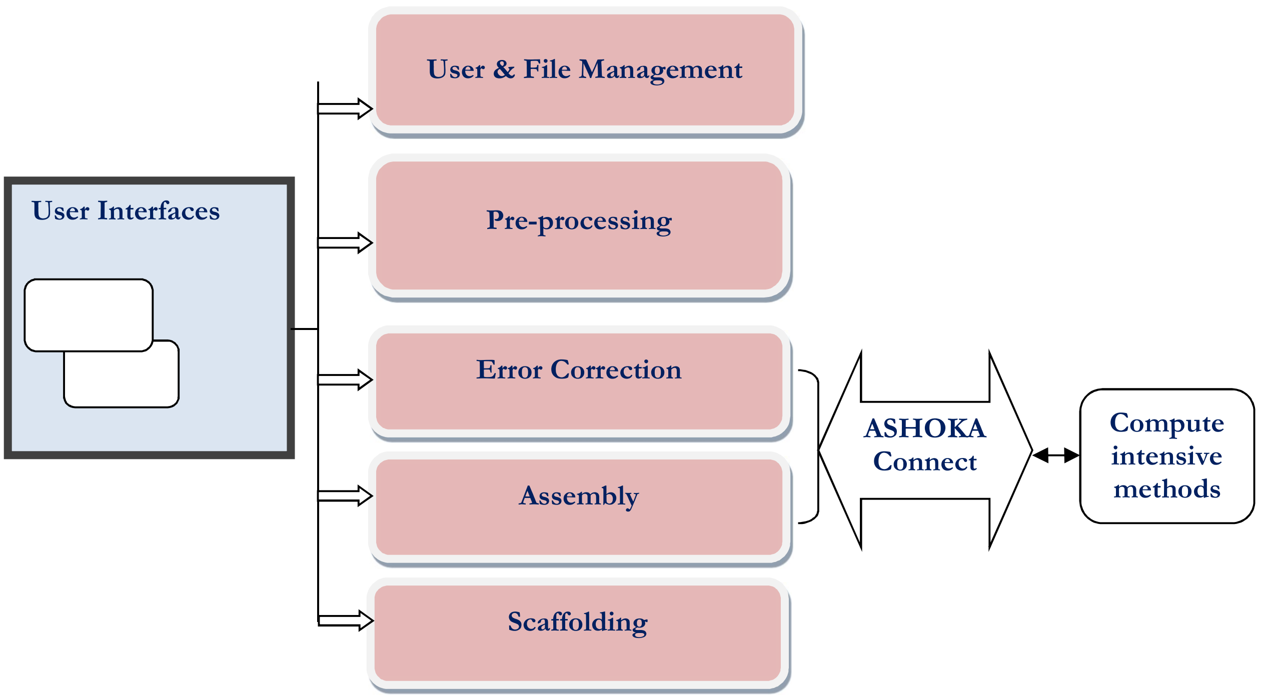


Fig. 3.2: Design of the Assembler Software

Table 3.1: Identification of modules for computation

Module Name	Description
Login	Provide facility of login to users

Assembler	The main module, which provide **Assembler** (including dependent variables and independent variables)
Help	Provide online help about software
Contact Us	Contact details of developer team
Sample Data Download	Download sample data to understand format of input data.
Signup	Provide facility of sign up to new user
Changed Password	An option for change of password

3.7.1 User Profile

The user of the system is divided into two categories:

1) **Registered users**: All register users can access the results of codon usage indices and correspondence analysis.

2) **Administrator**: - Administrator has rights to make any change in data base and other access polices.

3.7.2 Database Design

Database for the system is maintained using MySQL at server level. Database contains independent table namely login table. The schema design for table is presented in Table 3.2.

Table 3.2: Database table and its fields for storing profile details of the users

Attribute Name	**Constraint**	**Attribute Type**
uname	Primary Key	Varchar(30)
pass	Not Null	Varchar(30)
address	None	Varchar(100)
desig	None	Varchar(45)
deptt	None	Varchar(45)
org	None	Varchar(45)
city	None	Varchar(45)
state	None	Varchar(45)
country	None	Varchar(45)
phone	None	Varchar(15)
email	Not Null	Varchar(45)
name	Not Null	Varchar(45)
sex	Not Null	Varchar(6)

3.8 Java Secured Channel: JSch is a pure Java implementation of SSH2. JSch allows to connect to an sshd server and use port forwarding, X11 forwarding, file transfer, etc., and you can integrate its functionality into your own Java programs. JSch is licensed under BSD style license. Motive to develop was to allow users of pure java X servers, WiredX, to enable secure X sessions. So, our efforts had mostly targeted to implement the SSH2 protocol for X11 forwarding. Of course, however, we have also added other functionality like port forward, file transfer, terminal emulation, etc.

This is a utility to connect to remote linux server through an authorized login and password. This can be utilized run an application on the remote server and get the result back. It is a secured channel to connect to any high end linux server. The utility was found useful for connecting to ASHOKA supercomputing system.

3.9 Message Passing Interface: MPI is a specification for the developers and users of message passing libraries. By itself, it is not a library - but rather the specification of what such a library should be. MPI primarily addresses the message-passing parallel programming model: data is moved from the address space of one process to that of another process through cooperative operations on each process. Simply stated, the goal of the Message Passing Interface is to provide a widely used standard for writing message passing programs. The interface attempts to be:

- practical
- portable
- efficient
- flexible

The MPI standard has gone through a number of revisions, with the most recent version being MPI-3. Interface specifications have been defined for C and Fortran90 language bindings:

- C++ bindings from MPI-1 are removed in MPI-3
- MPI-3 also provides support for Fortran 2003 and 2008 features

Actual MPI library implementations differ in which version and features of the MPI standard they support.

This chapter mainly explained the various web application development technologies, three-tier architecture of the software and MPI libraries.

CHAPTER –IV

METHODOLOGY

The assembly process includes many steps to carry out as mentioned in Fig 4.1. In the first step, the sequence data received from any sequencing platform needs to be pre-processed. The long read pre-processed sequence data are aligned using short read sequences for correcting errors. These corrected long read sequences are then assembled using an assembly program. Finally, these assembled sequences are combined together to form scaffolds using another available program. The following Fig. 4.1 describes the whole process. These processes have also been described one by one in detail.

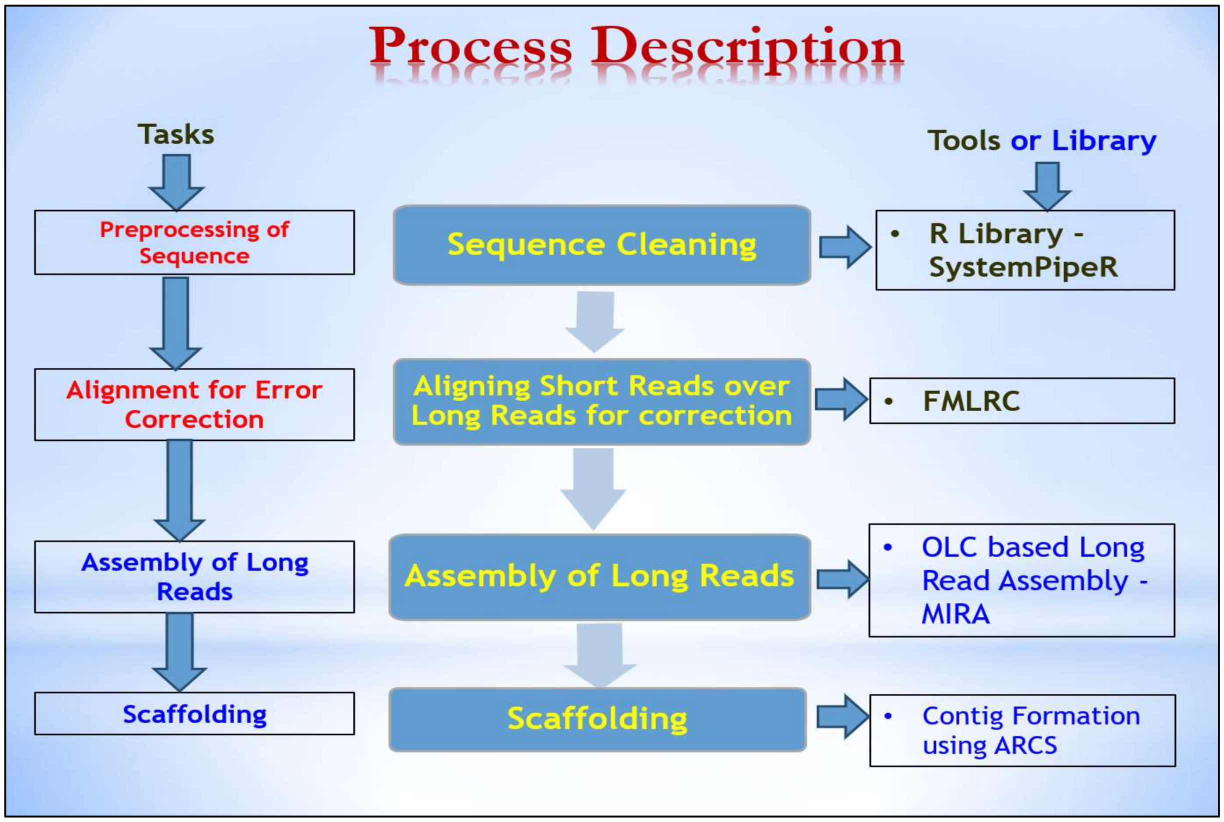

Process Description
Tasks
Tools or Library
Preprocessing of Sequence
Sequence Cleaning
• R Library - SystemPipeR
Alignment for Error Correction
Aligning Short Reads over Long Reads for correction
• FMLRC
Assembly of Long Reads
Assembly of Long Reads
• OLC based Long Read Assembly - MIRA
Scaffolding
Scaffolding
• Contig Formation using ARCS

Fig 4.1: Workflow of the developed assembler

4.1 Pre-processing: Pre-processing of next generation sequencing data is carried out for quality checking. The following activities are carried out for quality checking.

- Import of data from BAM, SAM or FastQ files (any variant)
- Providing a quick overview to tell the areas where there may be problems

- Summary graphs and tables to quickly assessment data
- Export of results to an HTML based permanent report
- Offline operation to allow automated generation of reports without running the interactive application

A variety of useful trimming tasks for paired-end and single ended data are also carried out. The selection of trimming steps and their associated parameters are supplied on the command line. The trimming steps are given below:

- Cut adapter and other platform-specific sequences from the read.
- Perform a sliding window trimming, cutting once the average quality within the window falls below a threshold.
- Cut bases off the start of a read, if below a threshold quality
- Cut bases off the end of a read, if below a threshold quality
- Cut the read to a specified length
- Cut the specified number of bases from the start of the read
- Drop the read if it is below a specified length
- Convert quality scores to Phred-33
- Convert quality scores to Phred-64

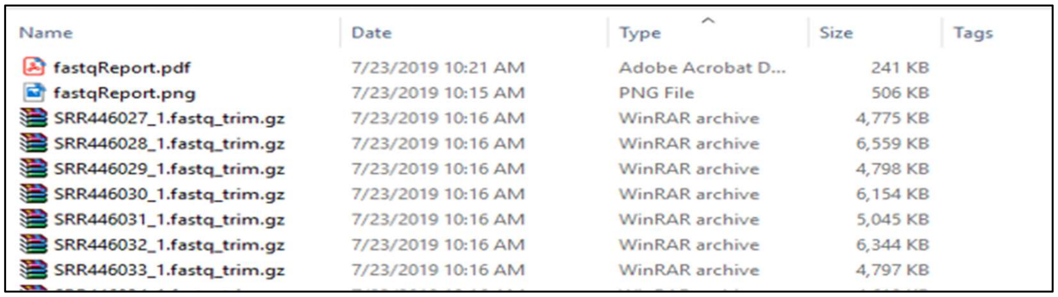
It uses the **FastqStreamer** function from **ShortRead** package to stream through large FASTQ files in a memory efficient manner. It performs adapter trimming with the **trimLRPatterns** function from the **Biostrings** package of R. After the trimming step, a new targets file is generated (targets_trim.txt) containing the paths to the trimmed FASTQ files.

Read quality filtering and trimming: The function preprocessReads allows to apply predefined or custom read pre-processing functions to all FASTQ files referenced in a SYSargs container, such as quality filtering or adaptor trimming routines. Fig 4.2 and 4.3 show the files received after pre-processing of reads and FASTQ Quality Report respectively.

Source Code: The following source code (Listing 4.1) has been used for running the pre-processing of sequence data.

Listing 4.1: Source code for pre-processing of sequences

```
#Library Requirement
library(systemPipeRdata)
library(systemPipeR)
#Experiment definition provided by targets file
targetspath <- system.file("extdata", "targets.txt", package="systemPipeR")
targets <- read.delim(targetspath, comment.char = "#")[,1:4]
#Read pre-processing
#Read quality filtering and trimming
args <- systemArgs(sysma="param/trim.param", mytargets="targets.txt")
writeTargetsout(x=args, file="targets_trim.txt", overwrite=TRUE)
#FASTQ quality report
pdf("./results/fastqReport.pdf", height=18, width=4*length(fqlist))
seeFastqPlot(fqlist)
dev.off()
```



Name	Date	Type	Size	Tags
fastqReport.pdf	7/23/2019 10:21 AM	Adobe Acrobat D...	241 KB	
fastqReport.png	7/23/2019 10:15 AM	PNG File	506 KB	
SRR446027_1.fastq_trim.gz	7/23/2019 10:16 AM	WinRAR archive	4,775 KB	
SRR446028_1.fastq_trim.gz	7/23/2019 10:16 AM	WinRAR archive	6,559 KB	
SRR446029_1.fastq_trim.gz	7/23/2019 10:16 AM	WinRAR archive	4,798 KB	
SRR446030_1.fastq_trim.gz	7/23/2019 10:16 AM	WinRAR archive	6,154 KB	
SRR446031_1.fastq_trim.gz	7/23/2019 10:16 AM	WinRAR archive	5,045 KB	
SRR446032_1.fastq_trim.gz	7/23/2019 10:16 AM	WinRAR archive	6,344 KB	
SRR446033_1.fastq_trim.gz	7/23/2019 10:16 AM	WinRAR archive	4,797 KB	

Fig 4.2: Showing pre-processed files

FASTQ quality report (Zoomed)

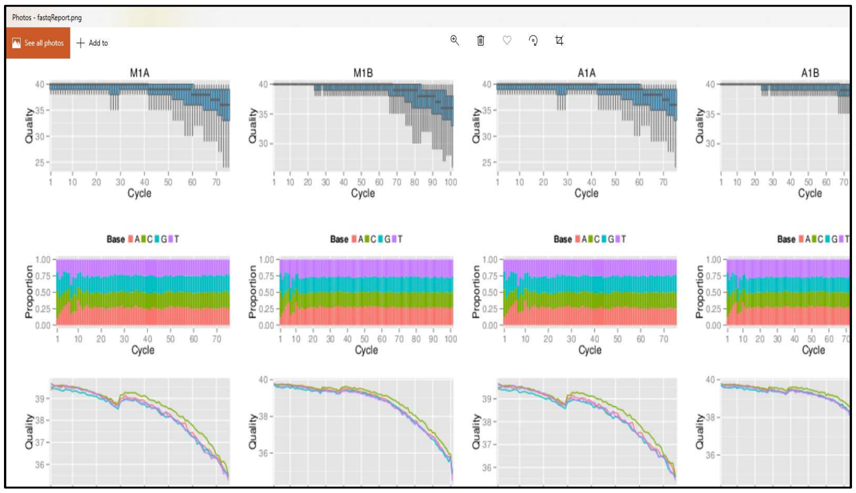


Fig 4.3: Showing FASTQ Quality Report

4.2 Read Alignment and Correction

Long read sequencing is changing the landscape of genomic research, especially *de-novo* assembly. Despite the high error rate inherent to long read technologies, increased read lengths dramatically improve the continuity and accuracy of genome assemblies. However, the cost and throughput of these technologies limit their application to complex genomes. One solution is to decrease the cost and time to assemble novel genomes by leveraging "hybrid" assemblies that use long reads for scaffolding and short reads for accuracy. Alignment of short reads over long reads has been carried out using a computer program called FMLRC for correcting errors present in long reads.

FMLRC: A novel method has been adopted leveraging a multi-string Burrows-Wheeler Transform with auxiliary FM-index to correct errors in long read sequences using a set of complementary short reads (Wang *et al.,* 2018). This method efficiently produces significantly more high-quality corrected sequence. It produces more contiguous assemblies, in many cases, than existing state-of-the-art hybrid and long-read only *de-novo* assembly methods. This method accurately corrects long read sequence data using complementary short reads and has improved throughput and computational efficiency. The FM-index enables arbitrary length *k*-mer searches through the dataset, allowing for FMLRC to retrieve *k*-mer frequencies from the short-read dataset. FMLRC uses the FM-index to *implicitly* represent *all* de Bruijn graphs of the short-read sequencing dataset. These de Bruijn graphs are then used to correct regions in the long reads that are not supported by the short-read sequencing dataset.

The Listing 4.2 shows the stepwise commands to be executed for long read sequence alignment.

Listing 4.2: Source code for alignment of long read sequences

```
Step 1 – Get Short Reads
wget http://spades.bioinf.spbau.ru/spades_test_datasets/ecoli_mc/s_6_1.fastq.gz
OR
wget http://spades.bioinf.spbau.ru/spades_test_datasets/ecoli_mc/s_6_2.fastq.gz
Step 2- Get Long Reads
wget                    http://files.pacb.com/datasets/secondary-analysis/e-coli-k12-de-
novo/1.3.0/Ecoli_MG1655_pacBioToCA.tgz
tar -xvzf Ecoli_MG1655_pacBioToCA.tgz
```

```
awk 'NR%4==1||NR%4==2' ./PacBioCLR/PacBio_10kb_CLR.fastq | tr "@" ">" > ./PacBioCLR/PacBio_10kb_CLR.fasta
```

Step 3 - build the bwt

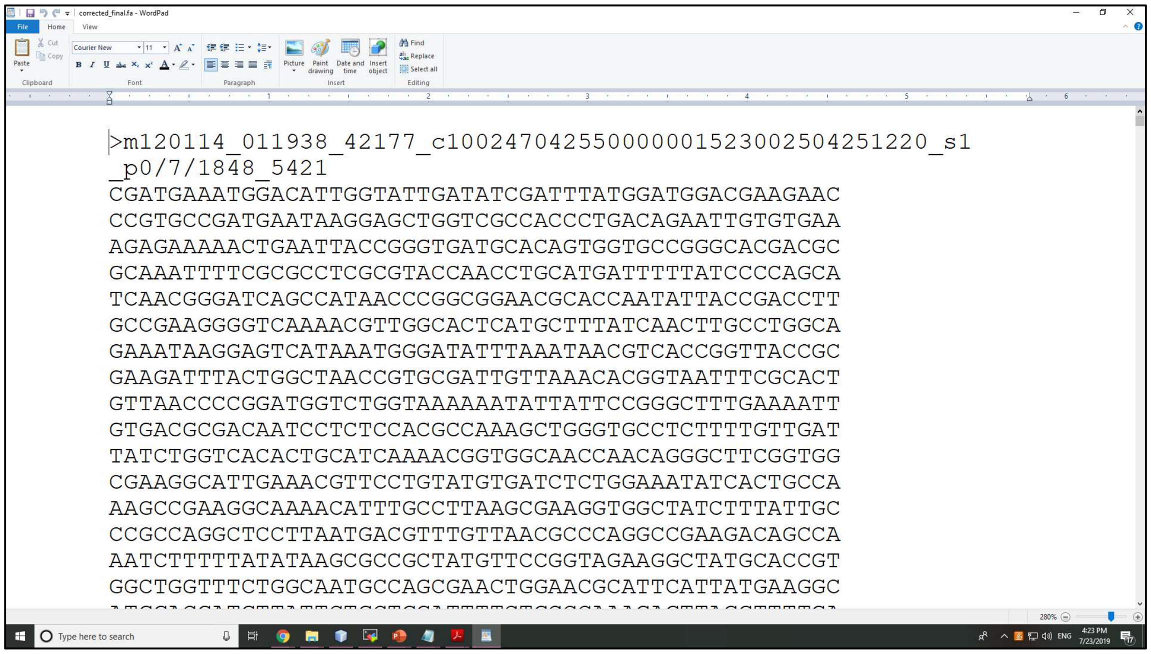
```
gunzip -c s_6_1.fastq.gz | awk "NR % 4 == 2" | sort -T ./temp | tr NT TN | /opt/software/ropebwt2/bin/ropebwt2 -LR | tr NT TN | msbwt convert ./ecoli_mc_msbwt
```

Step 4 - run fmlrc

```
NUM_PROCS=4
/opt/software/fmlrc-0.1.2-h2d50403_0/bin/fmlrc -p $NUM_PROCS -V ./ecoli_mc_msbwt/comp_msbwt.npy ./PacBioCLR/PacBio_10kb_CLR.fasta ./corrected_final.fa
```

Corrected Sequence

The following Fig 4.4 shows the corrected long read sequence file.



```
>m120114_011938_42177_c100247042550000001523002504251220_s1
_p0/7/1848_5421
CGATGAAATGGACATTGGTATTGATATCGATTTATGGATGGACGAAGAAC
CCGTGCCGATGAATAAGGAGCTGGTCGCCACCCTGACAGAATTGTGTGAA
AGAGAAAAACTGAATTACCGGGTGATGCACAGTGGTGCCGGGCACGACGC
GCAAATTTTCGCGCCTCGCGTACCAACCTGCATGATTTTTATCCCCAGCA
TCAACGGGATCAGCCATAACCCGGCGGAACGCACCAATATTACCGACCTT
GCCGAAGGGGTCAAAACGTTGGCACTCATGCTTTATCAACTTGCCTGGCA
GAAATAAGGAGTCATAAATGGGATATTTAAATAACGTCACCGGTTACCGC
GAAGATTTACTGGCTAACCGTGCGATTGTTAAACACGGTAATTTCGCACT
GTTAACCCCGGATGGTCTGGTAAAAAATATTATTCCGGGCTTTGAAAATT
GTGACGCGACAATCCTCTCCACGCCAAAGCTGGGTGCCTCTTTTGTTGAT
TATCTGGTCACACTGCATCAAAACGGTGGCAACCAACAGGGCTTCGGTGG
CGAAGGCATTGAAACGTTCCTGTATGTGATCTCTGGAAATATCACTGCCA
AAGCCGAAGGCAAAACATTTGCCTTAAGCGAAGGTGGCTATCTTTATTGC
CCGCCAGGCTCCTTAATGACGTTTGTTAACGCCCAGGCCGAAGACAGCCA
AATCTTTTTATATAAGCGCCGCTATGTTCCGGTAGAAGGCTATGCACCGT
GGCTGGTTTCTGGCAATGCCAGCGAACTGGAACGCATTCATTATGAAGGC
```

Fig 4.4: Corrected long read sequences

4.3 Assembling Corrected Long Reads

Long read sequencing is changing the landscape of genomic research, especially *de-novo* assembly. Despite the high error rate inherent to long read technologies, increased read lengths dramatically improve the continuity and accuracy of genome assemblies. However, the cost and throughput of these technologies limit their application to complex genomes. One solution is to decrease the cost and time to assemble novel genomes by leveraging "hybrid" assemblies that use long reads for scaffolding and short reads for accuracy.

MIRA - Mimicking Intelligent Read Assembly - A multi-pass DNA sequence data assembler/mapper for whole genome were used. It assembles/maps reads into contiguous sequences (called contigs) gained by the following:

- electrophoresis sequencing (aka Sanger sequencing)
- 454 pyro-sequencing (GS20, FLX or Titanium)
- Ion Torrent
- Solexa (Illumina) sequencing
- (in development) Pacific Biosciences sequencing

The MIRA code snippet for assembly and manifest file has been shown in Listing 4.3:

Listing 4.3: MIRA Code and manifest file

Code Snippet

```
String cmdline = "mira manifest.conf";
Cmdline = RLT.Execute_CommandOn_HPC(pscpFolderPlusPathOnServer, FileNameWithPath, ASHOKALogin1_IP, ASHOKA_L1_ID, ASHOKA_L1_PW, cmdline);
```

Manifest File

```
project=SBLal
job=genome, denovo, accurate
parameters= -GE:not=4
readgroup=SomeReads
data=/home/sblall/PacBio1.fastq
technology=pcbiohq
```

4.4 Scaffolding

A scaffold is a portion of the genome sequence reconstructed from end-sequenced whole-genome shotgun clones. Scaffolds are composed of contigs and gaps. A contig is a contiguous length of genomic sequence in which the order of bases is known to a high confidence level. Gaps occur where reads from the two sequenced ends of at least one fragment overlap with other reads in two different contigs (as long as the arrangement is otherwise consistent with the contigs being adjacent). Since the lengths of the fragments are roughly known, the number of bases between contigs can be estimated.

The goal of whole-genome shotgun assembly is to represent each genomic sequence in one scaffold; however, this is not always possible. One chromosome may be represented by many scaffolds (e.g., Chlamydomonas reinhardtii) or just a single scaffold (e.g., Human chromosome 19), depending on how completely the genome can be reconstructed, or assembled, from the available reads. The relative locations of scaffolds in the genome are unknown.

Scaffolds are normally numbered approximately from largest to smallest. Some scaffolds may ultimately be filtered out of the assembly, resulting in skipped scaffold numbers.

In some cases, scaffolds can overlap. For example, in polymorphic genomes, regions with a high density of allelic differences between haplotypes may be split into separate sets of scaffolds, each representing one allele. Thus, a sequence that exists in only one location in the genome may appear on more than one scaffold.

ARCS - Scaffolding genome drafts with linked read

It is an application that utilizes the barcoding information contained in linked reads to further organize draft genomes into highly contiguous assemblies. It harnesses the barcoding information contained in linked read data for connecting high-quality sequences in genome assembly drafts. For layout building, ARCS' gv file is converted to a tab-separated value (tsv) file listing all possible oriented sequence pairs, the number of supporting barcodes with gap sizes arbitrarily set at 10 bp. This is facilitated by the supplied python script (makeTSVfile.py).

Listing 4.4: Steps and codes to be executed for scaffolding process

1. Downloading sample Chromium read alignment .bam file

```
wget
http://www.bcgsc.ca/downloads/supplementary/ARCS/testdata/NA24143_genom
e_phased_namesorted.bam1.sorted.bam
wget    http://www.bcgsc.ca/downloads/supplementary/ARCS/testdata/hsapiens-
8reformat.fa
```

2. Running ARCS

```
/Backup/arcs/Arcs/arcs -f hsapiens-8reformat.fa -a alignments.fof -s 98 -c 5 -l 0 -d 0 -
r 0.05 -e 30000 -m 20-10000 > ARCSlog_c5r0.05e30000.txt
```

3. Converting graph for LINKS

```
/Backup/arcs/Examples/makeTSVfile.py                                        hsapiens-
8reformat.fa.scaff_s98_c5_l0_d0_e30000_r0.05.dist.gv  test_checkpoint.tsv  hsapiens-
8reformat.fa
```

4. Running LINKS

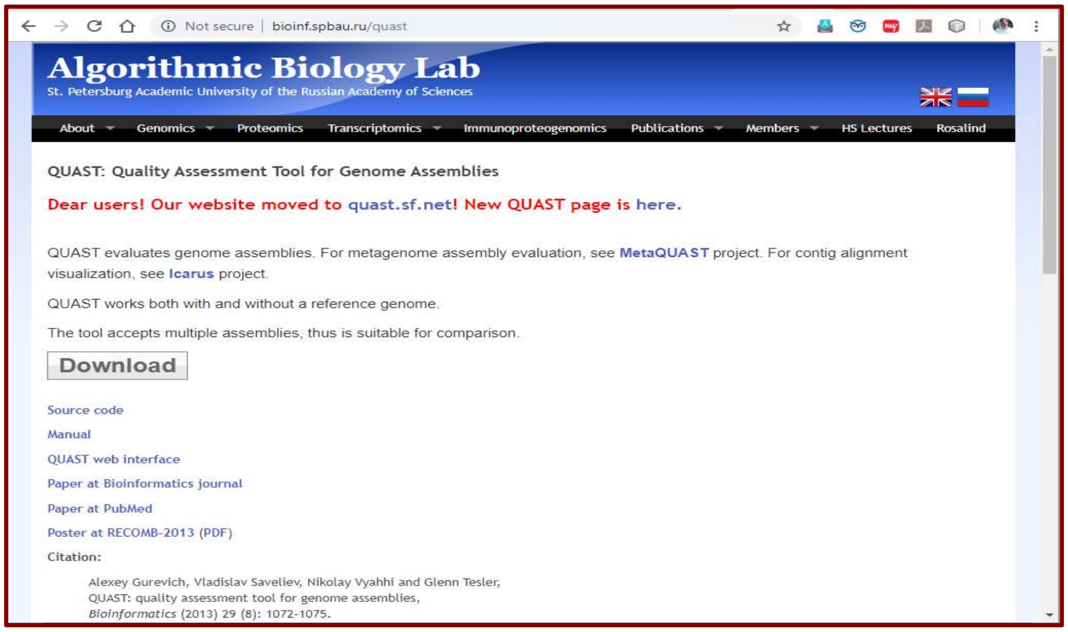
```
/opt/links_v1.8.6/LINKS   -f   hsapiens-8reformat.fa   -s   empty.fof   -k   20   -b
links_c5r0.05e30000-l5-a0.9 -l 5 -t 2 -a 0.9 -x 1
```

Since positional information of reads within the molecule of origin is not known, estimation of gap sizes is not a straightforward problem, and would require more sophisticated approaches.

ARCS first pairs sequences within a draft assembly, then lays out the pairing information for scaffolding. Input alignments in BAM format are processed for sets of read pairs from the same barcode that align to different sequences. A link between the two sequences is formed. Each link represents evidence that one barcode/molecule connects the sequences.

4.5 Quality Assessment of results

The quality assessment of the generated results from the developed assembler was done using QUAST tool available. The following figures show the available web page of the tool and the result generated after running this tool respectively. Table 4.1 shows the result of QUAST tool.



QUAST: Quality Assessment Tool for Genome Assemblies

Dear users! Our website moved to quast.sf.net! New QUAST page is here.

QUAST evaluates genome assemblies. For metagenome assembly evaluation, see MetaQUAST project. For contig alignment visualization, see Icarus project.

QUAST works both with and without a reference genome.

The tool accepts multiple assemblies, thus is suitable for comparison.

Source code
Manual
QUAST web interface
Paper at Bioinformatics journal
Paper at PubMed
Poster at RECOMB-2013 (PDF)
Citation:

Alexey Gurevich, Vladislav Saveliev, Nikolay Vyahhi and Glenn Tesler,
QUAST: quality assessment tool for genome assemblies,
Bioinformatics (2013) 29 (8): 1072-1075.

Fig 4.4: QUAST tool web page

All statistics are based on contigs of size >= 500 bp, unless otherwise noted (e.g., "# contigs (>= 0 bp)" and "Total length (>= 0 bp)" include all contigs).

Table 4.1: Quality Assessment result from QUAST

Statistics without reference	Assembly_out.padded
# contigs	32
# contigs (>= 0 bp)	36
# contigs (>= 1000 bp)	27
# contigs (>= 5000 bp)	10
# contigs (>= 10000 bp)	4
# contigs (>= 25000 bp)	0
# contigs (>= 50000 bp)	0
Largest contig	38230
Total length	138701
Total length (>= 0 bp)	8066
Total length (>= 1000 bp)	3241
Total length (>= 5000 bp)	439

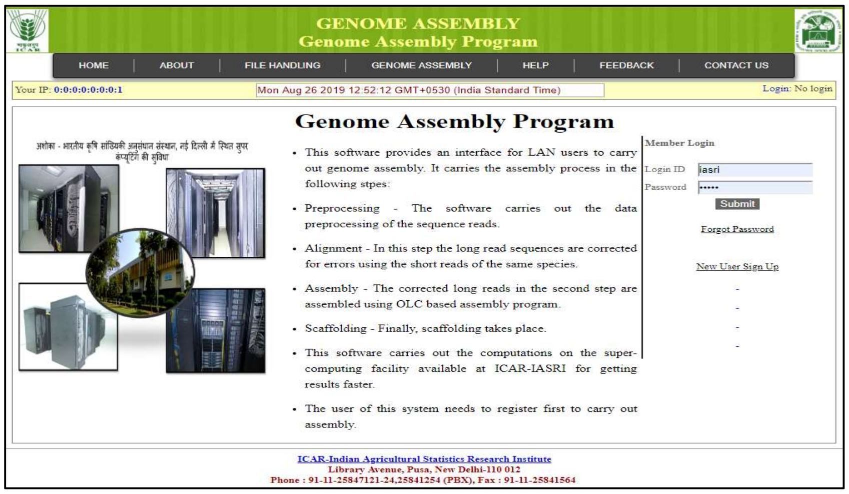
Total length (>= 10000 bp)	688
Total length (>= 25000 bp)	0
Total length (>= 50000 bp)	0
N50	**18531**
N75	**14452**
L50	7
L75	12
GC (%)	48.73
Mismatches	
# N's	0
# N's per 100 kbp	0

CHAPTER –V
SOFTWARE DESCRIPTION

The Software for genome assembly has been developed for web platform and programming has been done with the Java Server Pages (JSP), Cascading Style Sheets (CSS) and Java programming language. It has been developed on Intel Xeon based 64 bit computer with 3.20 GHz-clock speed, Microsoft Windows 7 Operating System and 16.0 GB RAM. NetBeans 8.0 Integrated Development Environment (IDE) with java development kit 1.8 has been used as a platform for development of the software.

5.1 Client interface

This web-based assembler software that is freely accessible for LAN users. User authentication is needed to ensure security. It is accessible only after entering valid user name and password. For getting user name and password, any user may signup by clicking on appropriate link on home page. The home page (Fig 5.1) of the software presents the user with a brief welcome note on the software.



Your IP: 0:0:0:0:0:0:0:1

Mon Aug 26 2019 12:52:12 GMT+0530 (India Standard Time)

Login: No login

Genome Assembly Program

अशोका - भारतीय कृषि सांख्यिकी अनुसंधान संस्थान, नई दिल्ली में स्थित सुपर कंप्यूटिंग की सुविधा

- This software provides an interface for LAN users to carry out genome assembly. It carries the assembly process in the following stpes:
- Preprocessing - The software carries out the data preprocessing of the sequence reads.
- Alignment - In this step the long read sequences are corrected for errors using the short reads of the same species.
- Assembly - The corrected long reads in the second step are assembled using OLC based assembly program.
- Scaffolding - Finally, scaffolding takes place.
- This software carries out the computations on the super-computing facility available at ICAR-IASRI for getting results faster.
- The user of this system needs to register first to carry out assembly.

Login ID: iasri

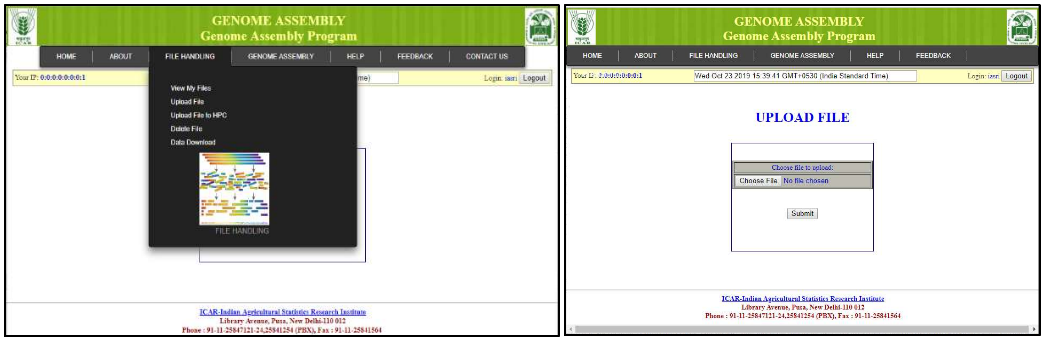
Password: •••••

ICAR-Indian Agricultural Statistics Research Institute
Library Avenue, Pusa, New Delhi-110 012
Phone : 91-11-25847121-24,25841254 (PBX), Fax : 91-11-25841564

Figure 5.1: Home page of Assembler Software

The home page has links in the form of horizontal menu bar which has links for “Home”, “About”, “File Handling”, “Genome Assembly”, “Help”, “Feedback” and “Contact Us”. The links “About”, “File Handling” and “Genome Assembly” have submenus too. “About” has link for CABin (Centre for Agricultural Bioinformatics). File Handling has links for “View My Files”, “Upload File”, “Upload File to HPC”, Delete File”, “My Files”, “Download Files” and “Data Download”.

As the name suggests the functionalities for upload or download a data file to or from the server, deleting a file from the server, viewing the files uploaded by the logged in user are facilities provided (Figure 5.2 and 5.3). “Data Download” provides facility to download sample input files for three workflows provided in the software. “Contact Us” page provides the details of the primary contact and project team. “Help” gives the user the detailed description of how to use this software. Figure 5.4 shows the “Contact Us” page of the software.



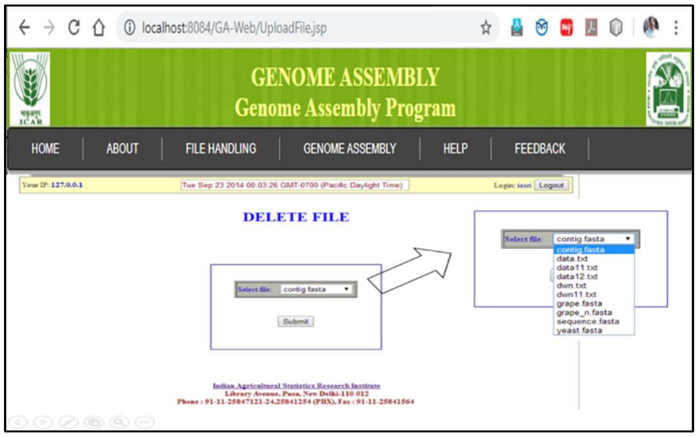
Your IP: 0:0:0:0:0:0:0:1

Wed Oct 23 2019 15:39:41 GMT+0530 (India Standard Time)

Login: iasri

ICAR-Indian Agricultural Statistics Research Institute
Library Avenue, Pusa, New Delhi-110 012
Phone : 91-11-25847121-24,25841254 (PBX), Fax : 91-11-25841564

Figure 5.2: Upload file on Assembler Software



Your IP: 127.0.0.1

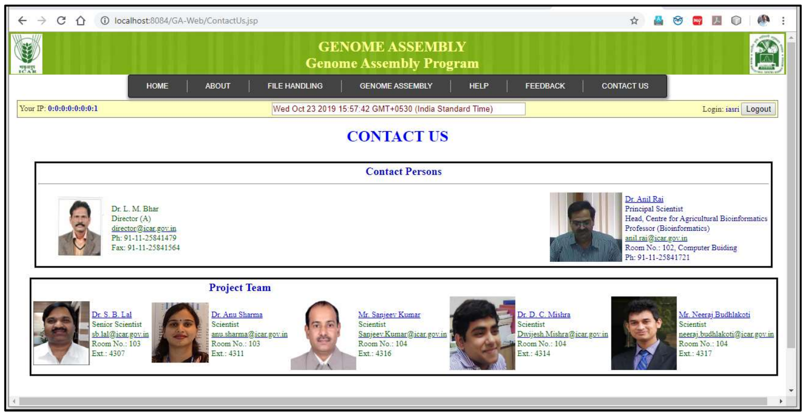
Tue Sep 23 2014 00:03:26 GMT-0700 (Pacific Daylight Time)

Login: iasri

contig.fasta
data.txt
data11.txt
data12.txt
dwn.txt
dwn11.txt
grape.fasta
grape_n.fasta
sequence.fasta
yeast.fasta

Indian Agricultural Statistics Research Institute
Library Avenue, Pusa, New Delhi-110 012
Phone : 91-11-25847121-24,25841254 (PBX), Fax : 91-11-25841564

Figure 5.3: Delete file from Assembler Software



Your IP: 0:0:0:0:0:0:0:1

Wed Oct 23 2019 15:57:42 GMT+0530 (India Standard Time)

Login: iasri

CONTACT US

Contact Persons

Dr. L. M. Bhar
Director (A)
director@icar.gov.in
Ph: 91-11-25841479
Fax: 91-11-25841564

Dr. Anil Rai
Principal Scientist
Head, Centre for Agricultural Bioinformatics
Professor (Bioinformatics)
anil.rai@icar.gov.in
Room No.: 102, Computer Buiding
Ph: 91-11-25841721

Project Team

Dr. S. B. Lal
Senior Scientist
sb.lal@icar.gov.in
Room No.: 103
Ext.: 4307

Dr. Anu Sharma
Scientist
anu.sharma@icar.gov.in
Room No.: 103
Ext.: 4311

Mr. Sanjeev Kumar
Scientist
Sanjeev.Kumar@icar.gov.in
Room No.: 104
Ext.: 4316

Dr. D. C. Mishra
Scientist
Dwijesh.Mishra@icar.gov.in
Room No.: 104
Ext.: 4314

Mr. Neeraj Budhlakoti
Scientist
neeraj.budhlakoti@icar.gov.in
Room No.: 104
Ext.: 4317

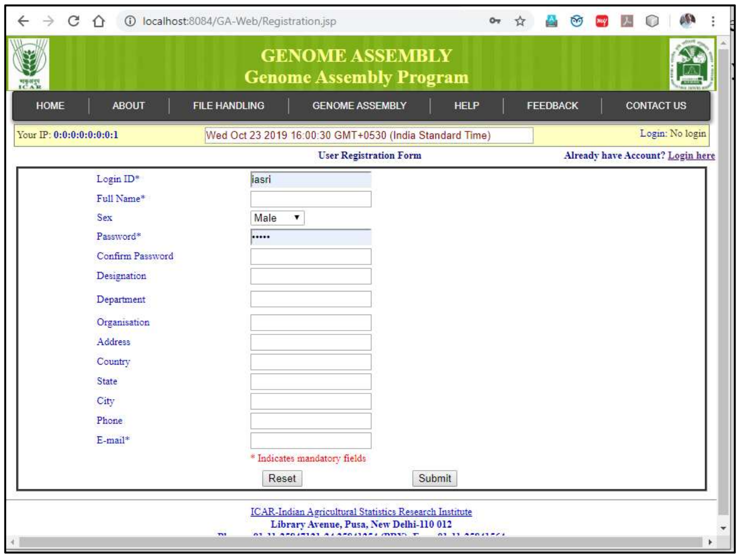
Figure 5.4: Contact Us page of Assembler Software

5.1.1 *User management*

User management module of the software provides the following facilities to the users:

Creating a new user

For a new user registration, clicking on the "New User Sign Up" link takes the user to the registration page (Figure 5.5) where the username and password can be set by filling in all the required details for registration process.



Your IP: 0:0:0:0:0:0:0:1 Wed Oct 23 2019 16:00:30 GMT+0530 (India Standard Time) Login: No login

User Registration Form Already have Account? Login here

Login ID*	iasri
Full Name*	
Sex	Male
Password*	•••••
Confirm Password	
Designation	
Department	
Organisation	
Address	
Country	
State	
City	
Phone	
E-mail*	

* Indicates mandatory fields

ICAR-Indian Agricultural Statistics Research Institute
Library Avenue, Pusa, New Delhi-110 012

Figure 5.5: Sign Up page of Assembler Software

After entering the details user can click on "Submit" button for submitting the information into the user table of the database. After authentication of username and password in login page. Logged in users have access to all facilities of the software with a Graphical User Interface (GUI).

Changing and retrieving user password

Options are also provided to change the existing user password and to retrieve the password in the situations when user forgets the password.

5.2 Data Management

- ✓ Registration: User profile is stored in MYSQL database, Email notification
- ✓ Folder Management: User's folder created with signup

- ✓ File Management: Separate module for upload file, file viewer, file download, file delete and sample data download

Input selection: Select file from user’s own folder

Output:

- ✓ Can be viewed on browser
- ✓ Can be downloaded on user’s disk
- ✓ Copied to the user’s own folder

Input data handling

Short Read or Long Read sequences are nucleic acid separated by at least one header line. A header line is defined as any line whose first character is a right-angled bracket ‘>’. There may be any number of header lines but they must precede each sequence, and the second or subsequent header lines are ignored. Those lines whose first character is not ‘>’ are considered to be sequence data. Sequences must be in the correct reading frame, and should not contain untranslated 5’ or 3’ sequence. The format of each line of sequence data is relaxed; sequences can be either upper- or lower-case characters. Input lines may be any width and contain spaces and/or numbers.

Input data handling module has been designed and developed for reading data for computation on the assembler software. Client is required to upload the input data in ‘fasta’ or ‘fastq’ format in each section.

5.3 Cluster Connectivity

During the process of assembly, programs installed on ASHOKA HPC have been used for faster results. The JSP based web application connects to the HPC through available java library called “JSch”. It needs user credentials to login to the HPC. File management and folder creation are done using different methods developed in java. Command line arguments are passed to the java method to copy input file and run program on HPC (cluster). The output files generated on the HPC are copied back to web server the same way. The files generated on HPC login are deleted after the process of computation and file copying is complete.

Assembly Options

The options available for assembly process have been shown in the following Fig 5.6.

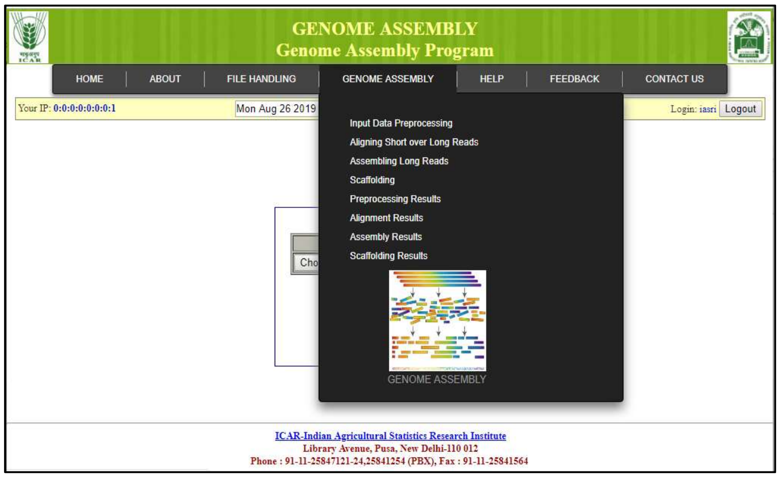
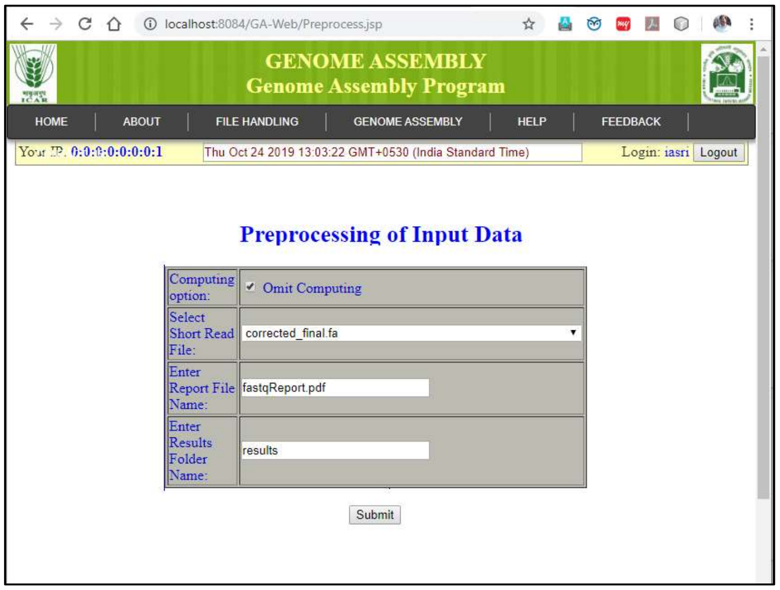


Figure 5.6: Options for Assembly Process

5.4 Pre-processing

On the menu of "Genome Assembly" there are many submenu options available. For pre-processing of input sequences, the "Input Data Pre-processing" needs to be chosen. Clicking on this option opens the page as given in Fig 5.7. On this page read file needs to be chosen from the user's folder shown on a dropdown. We can also mention result file and the folder name for result file generation.

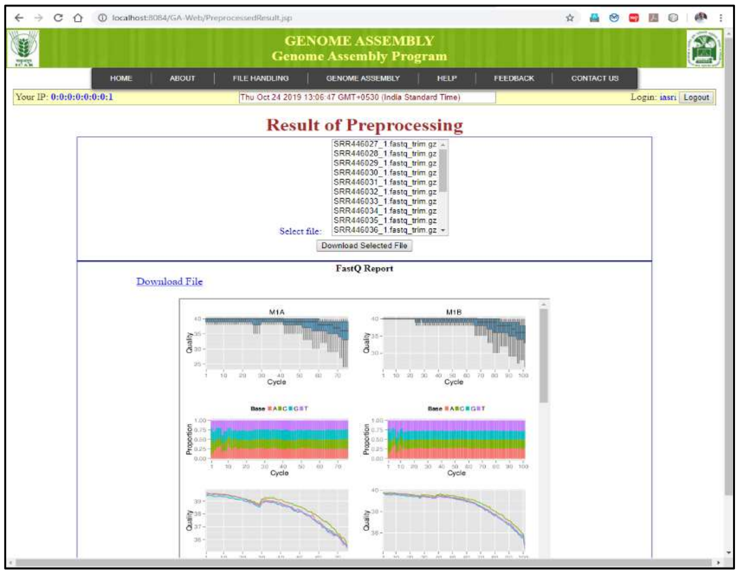


Preprocessing of Input Data

Computing option:	✓ Omit Computing
Select Short Read File:	corrected_final.fa
Enter Report File Name:	fastqReport.pdf
Enter Results Folder Name:	results

Figure 5.7: Pre-processing of Input Data

After clicking on submit button, computation starts and after it is finished the following result page (Fig. 5.8) is shown.



Your IP: 0:0:0:0:0:0:0:1 Thu Oct 24 2019 13:06:47 GMT+0530 (India Standard Time) Login: iasri

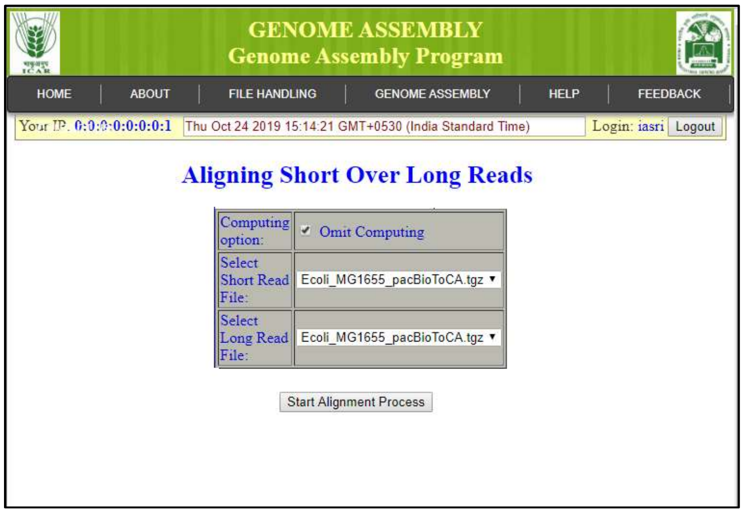
Result of Preprocessing

Select file:
SRR446027_1.fastq_trim.gz
SRR446028_1.fastq_trim.gz
SRR446029_1.fastq_trim.gz
SRR446030_1.fastq_trim.gz
SRR446031_1.fastq_trim.gz
SRR446032_1.fastq_trim.gz
SRR446033_1.fastq_trim.gz
SRR446034_1.fastq_trim.gz
SRR446035_1.fastq_trim.gz
SRR446036_1.fastq_trim.gz

Figure 5.8: Result of Pre-processing

5.5 Alignment

The pre-processed long reads sequences are corrected for error using short read sequences by aligning. The alignment process can be started by choosing submenu “Aligning Short Over Long Reads” from menu option “GENOME ASSEMBLY”. This opens a new page as given in Fig 5.9. The user needs to choose short read and long read files in zipped form. After selecting these files “submit” button is clicked to start the process. The alignment process takes some time to complete.



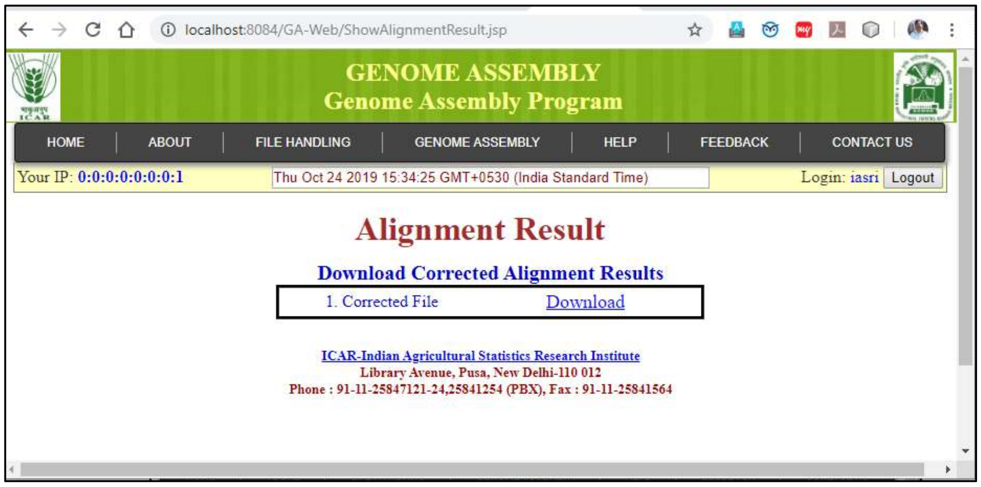
Your IP: 0:0:0:0:0:0:0:1 Thu Oct 24 2019 15:14:21 GMT+0530 (India Standard Time) Login: iasri

Aligning Short Over Long Reads

Computing option:	✓ Omit Computing
Select Short Read File:	Ecoli_MG1655_pacBioToCA.tgz
Select Long Read File:	Ecoli_MG1655_pacBioToCA.tgz

Figure 5.9: Aligning Short over Long Reads

The corrected file is generated on HPC which is then put on the web server for providing a link to download by the user as given in Fig 5.10. The content of a corrected sequence file has been shown in Fig 5.11 as an example.



Your IP: 0:0:0:0:0:0:0:1 Thu Oct 24 2019 15:34:25 GMT+0530 (India Standard Time) Login: iasri

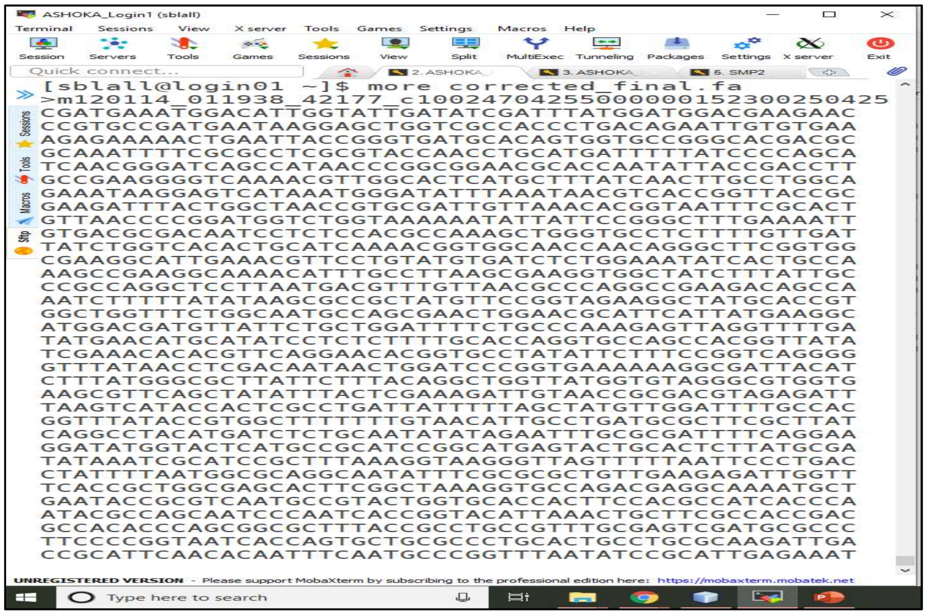
Alignment Result

Download Corrected Alignment Results

1. Corrected File	Download

ICAR-Indian Agricultural Statistics Research Institute
Library Avenue, Pusa, New Delhi-110 012
Phone : 91-11-25847121-24,25841254 (PBX), Fax : 91-11-25841564

Figure 5.10: Assembling Short over Long Reads


ASHOKA_Login1 (sblall)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings X server Exit
Quick connect...
2. ASHOKA
3. ASHOKA
5. SMP2
Sessions
Tools
Macros
Sftp
Type here to search

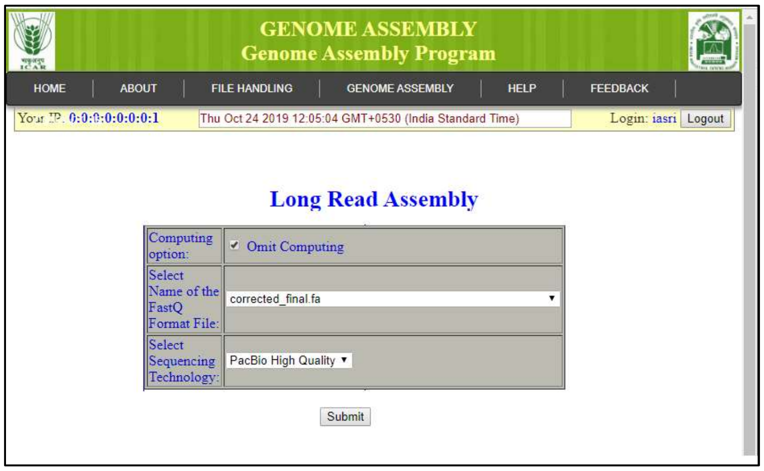
```
[sblall@login01 ~]$ more corrected_final.fa
>m120114_011938_42177_c100247042550000001523002504 25
CGATGAAATGGACATTGGTATTGATATCGATTTATGGATGGACGAAGAAC
CCGTGCCGATGAATAAGGAGCTGGTCGCCACCCTGACAGAATTGTGTGAA
AGAGAAAAACTGAATTACCGGGTGATGCACAGTGGTGCCGGGCACGACGC
GCAAATTTTCGCGCCTCGCGTACCAACCTGCATGATTTTTATCCCCAGCA
TCAACGGGATCAGCCATAACCCGGCGGAACGCACCAATATTACCGACCTT
GCCGAAGGGGTCAAAACGTTGGCACTCATGCTTTATCAACTTGCCTGGCA
GAAATAAGGAGTCATAAATGGGATATTTAAATAACGTCACCGGTTACCGC
GAAGATTTACTGGCTAACCGTGCGATTGTTAAACACGGTAATTTCGCACT
GTTAACCCCGGATGGTCTGGTAAAAAATATTATTCCGGGCTTTGAAAATT
GTGACGCGACAATCCTCTCCACGCCAAAGCTGGGTGCCTCTTTTGTTGAT
TATCTGGTCACACTGCATCAAAACGGTGGCAACCAACAGGGCTTCGGTGG
CGAAGGCATTGAAACGTTCCTGTATGTGATCTCTGGAAATATCACTGCCA
AAGCCGAAGGCAAAACATTTGCCTTAAGCGAAGGTGGCTATCTTTATTGC
CCGCCAGGCTCCTTAATGACGTTTGTTAACGCCCAGGCCGAAGACAGCCA
AATCTTTTTATATAAGCGCCGCTATGTTCCGGTAGAAGGCTATGCACCGT
GGCTGGTTTCTGGCAATGCCAGCGAACTGGAACGCATTCATTATGAAGGC
ATGGACGATGTTATTCTGCTGGATTTTCTGCCCAAAGAGTTAGGTTTTGA
TATGAACATGCATATCCTCTCTTTTGCACCAGGTGCCAGCCACGGTTATA
TCGAAACACACGTTCAGGAACACGGTGCCTATATTCTTTCCGGTCAGGGG
GTTTATAACCTCGACAATAACTGGATCCCGGTGAAAAAAGGCGATTACAT
CTTTATGGGCGCTTATTCTTTACAGGCTGGTTATGGTGTAGGGCGTGGTG
AAGCGTTCAGCTATATTTACTCGAAAGATTGTAACCGCGACGTAGAGATT
TAAGTCATACCACTCGCCTGATTATTTTTAGCTATGTTGGATTTTGCCAC
GGTTTATACCGTGGCTTTTTTTGTAACATTGCCTGATGCGCTTCGCTTAT
CAGGCCTACATGATCTCTGCAATATATAGAATTTGCGCGATTTTCAGGAA
GGATATGGTACTCATGCCGCATCCGGCATGAGTACTGCACTCTTATGCGA
TATAAATCGCATCCGCTTTAAAGGTAAGGGTTAGTTTTTAATTCCCTGAC
CTATTTTAATGGCGCAGGCAATATTTCGCGCGCTGTTGAAGAGATTGGTT
TCACCGCTGGCGAGCACTTCGGCTAAAGGTGCCAGACGAGGCAAAATGCT
GAATACCGCGTCAATGCCGTACTGGTGCACCACTTCCACGCCATCACCCA
ATACGCCAGCAATCCCAATCACCGGTACATTAAACTGCTTCGCCACCGAC
GCCACACCCAGCGGCGCTTTACCGCCTGCCGTTTGCGAGTCGATGCGCCC
TTCCCCGGTAATCACCAGTGCTGCGCCCTGCACTGCCTGCGCAAGATTGA
CCGCATTCAACACAATTTCAATGCCCGGTTTAATATCCGCATTGAGAAAT
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

Figure 5.11: Content of a corrected sequence file

5.6 Assembling Long Reads

Next step in the genome assembly process is now assembling of corrected sequence using MIRA program. The menu option to run this program is “Assembling Long Reads”. The following web page will be shown (Fig 5.12) after choosing this option.


GENOME ASSEMBLY
Genome Assembly Program

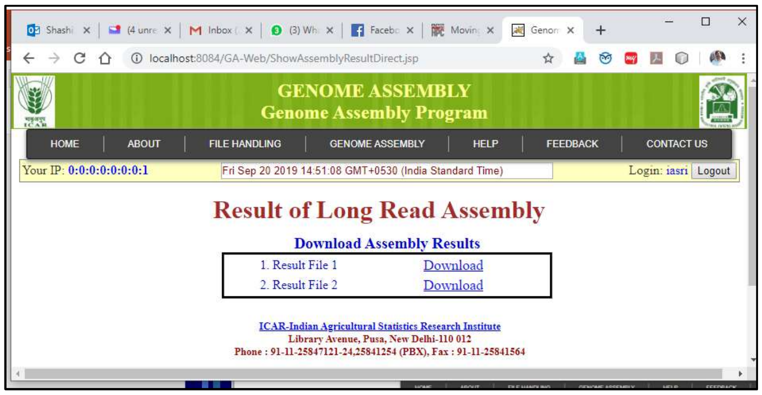
Your IP: 0:0:0:0:0:0:0:1 | Thu Oct 24 2019 12:05:04 GMT+0530 (India Standard Time) | Login: iasri

Long Read Assembly

Computing option:	✓ Omit Computing
Select Name of the FastQ Format File:	corrected_final.fa
Select Sequencing Technology:	PacBio High Quality

Figure 5.12: Assembling Short over Long Reads

This web page asks two parameters – Name of the FastQ file and the name of sequencing technology. Clicking on “Submit” button below starts the assembly program MIRA installed on HPC. After the assembly is complete, the result page as given in Fig 5.13 is shown. The fasta file as shown in Fig 5.14 and quality file as given in Fig 5.15 is generated on the server.


localhost:8084/GA-Web/ShowAssemblyResultDirect.jsp

GENOME ASSEMBLY
Genome Assembly Program

Your IP: 0:0:0:0:0:0:0:1 | Fri Sep 20 2019 14:51:08 GMT+0530 (India Standard Time) | Login: iasri

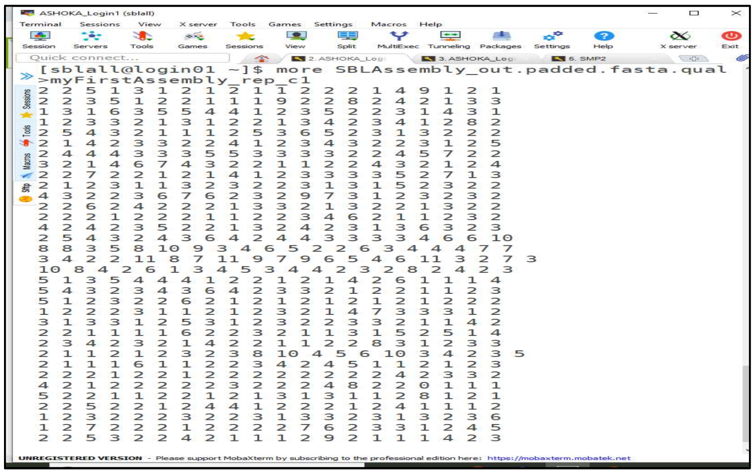
Result of Long Read Assembly

Download Assembly Results

1. Result File 1	Download
2. Result File 2	Download

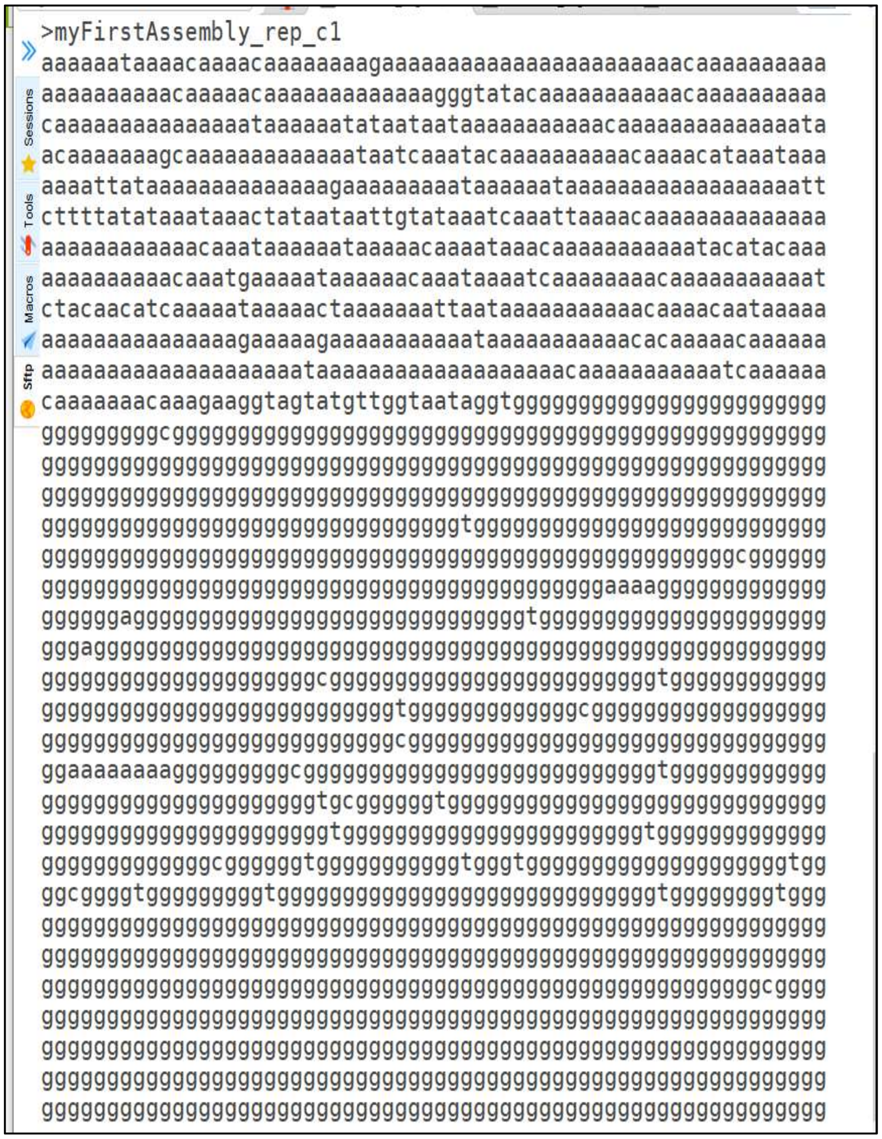
ICAR-Indian Agricultural Statistics Research Institute
Library Avenue, Pusa, New Delhi-110 012
Phone : 91-11-25847121-24,25841254 (PBX), Fax : 91-11-25841564

Figure 5.13: Result of Assembly


```
[sblall@login01 ~]$ more SBLAssembly_out.padded.fasta.qual
>myFirstAssembly_rep_c1
2  2  5  1  3  1  2  1  2  1  1  2  2  2  1  4  9  1  2  1
2  2  3  5  1  2  2  1  1  1  9  2  2  8  2  4  2  1  3  3
1  3  1  6  3  5  5  4  4  1  2  3  5  2  2  3  1  4  3  1
1  2  3  3  2  1  3  1  2  2  1  3  4  2  3  4  1  2  8  2
2  5  4  3  2  1  1  1  2  5  3  6  5  2  3  1  3  2  2  2
2  1  4  2  3  3  2  2  4  1  2  3  4  3  2  2  3  1  2  5
2  4  4  4  3  3  3  5  5  3  3  3  3  2  2  4  5  7  2  2
3  2  1  4  6  7  4  3  2  2  1  1  2  2  4  3  2  1  2  4
2  2  7  2  2  1  2  1  4  1  2  3  3  3  3  5  2  7  1  3
2  1  2  3  1  1  3  2  3  2  2  3  1  3  1  5  2  3  2  2
4  3  2  2  3  6  7  6  2  3  2  9  7  3  1  2  3  2  3  2
2  2  6  2  4  2  2  2  1  3  3  2  1  3  2  2  1  3  2  2
2  2  2  1  2  2  2  1  1  2  2  3  4  6  2  1  1  2  3  2
4  2  4  2  3  5  2  2  1  3  2  4  2  3  1  3  6  3  2  3
2  5  4  3  2  4  3  6  4  2  4  4  3  3  3  3  4  6  6  10
8  8  3  5  8  10  9  3  4  6  5  2  2  6  3  4  4  4  7  7
3  4  2  2  11  8  7  11  9  7  9  6  5  4  6  11  3  2  7  3
10  8  4  2  6  1  3  4  5  3  4  4  2  3  2  8  2  4  2  3
5  1  3  5  4  4  4  1  2  2  1  2  1  4  2  6  1  1  1  4
5  4  3  2  3  4  3  6  4  2  3  3  2  1  2  2  1  1  2  3
5  1  2  3  2  2  6  2  1  2  1  2  1  2  1  2  1  2  2  2
1  2  2  2  3  1  1  2  1  2  3  2  1  4  7  3  3  3  1  2
3  1  3  3  1  2  5  3  1  2  3  2  2  3  3  2  1  1  4  2
2  2  1  1  1  1  6  2  2  3  2  1  1  3  1  5  2  5  1  4
2  3  4  2  3  2  1  4  2  2  1  1  2  2  8  3  1  2  3  3
2  1  1  2  1  2  3  2  3  8  10  4  5  6  10  3  4  2  3  5
2  1  1  1  6  1  1  2  2  3  4  2  4  5  1  1  2  1  2  3
2  2  2  1  2  2  1  2  2  2  2  2  2  2  2  4  2  3  3  2
4  2  1  2  2  2  2  2  3  2  2  2  4  8  2  2  0  1  1  1
5  2  2  1  1  2  2  1  2  1  3  1  3  1  1  2  8  1  2  1
2  2  5  2  2  1  2  4  4  1  2  2  2  1  2  4  1  1  1  2
1  2  3  2  2  2  3  2  2  3  1  3  3  2  3  1  3  2  3  6
1  2  7  2  2  2  1  2  2  2  2  7  6  2  3  3  1  2  4  5
2  2  5  3  2  2  4  2  1  1  1  2  9  2  1  1  1  4  2  3
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

Figure 5.14: Assembling Short over Long Reads

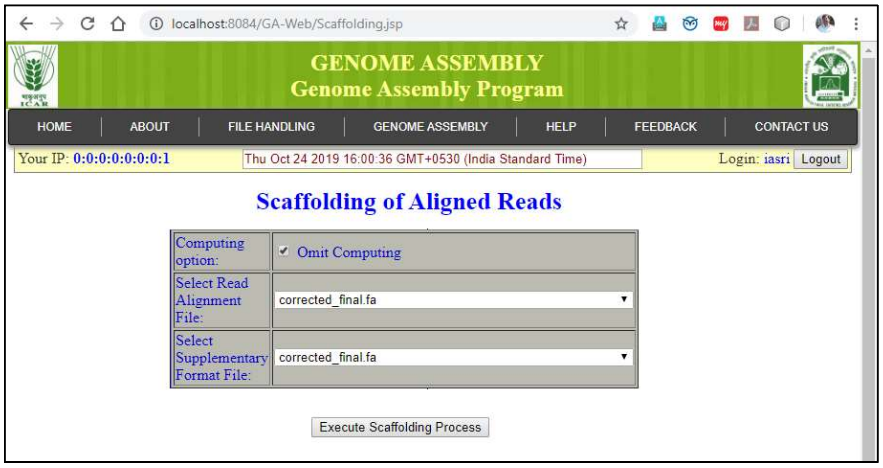


```
>myFirstAssembly_rep_c1
aaaaaataaaacaaaacaaaaaaaaagaaaaaaaaaaaaaaaaaaaaaacaaaaaaaaaa
aaaaaaaaaacaaaaacaaaaaaaaaaaaagggtatacaaaaaaaaaaaacaaaaaaaaa
caaaaaaaaaaaaaataaaaatataataataaaaaaaaaaaaacaaaaaaaaaaaaaata
acaaaaaaagcaaaaaaaaaaaaataatcaaatacaaaaaaaaaacaaaacataaataaa
aaaattataaaaaaaaaaaaaagaaaaaaaaataaaaaataaaaaaaaaaaaaaaaaatt
cttttatataaataaactataataattgtataaatcaaattaaaacaaaaaaaaaaaaaa
aaaaaaaaaaaacaaataaaaaataaaaacaaaataaacaaaaaaaaaaatacatacaaa
aaaaaaaaaacaaatgaaaaataaaaaacaaataaaatcaaaaaaaacaaaaaaaaaaat
ctacaacatcaaaaataaaaactaaaaaaattaataaaaaaaaaaacaaaacaataaaaa
aaaaaaaaaaaaaaagaaaaagaaaaaaaaaaataaaaaaaaaaacacaaaaacaaaaaa
aaaaaaaaaaaaaaaaaaaataaaaaaaaaaaaaaaaaaacaaaaaaaaaaatcaaaaaa
caaaaaaacaaagaaggtagtatgttggtaataggtgggggggggggggggggggggggg
gggggggggcgggggggggggggggggggggggggggggggggggggggggggggggggg
gggggggggggggggggggggggggggggggggggggggggggggggggggggggggggg
gggggggggggggggggggggggggggggggggggggggggggggggggggggggggggg
ggggggggggggggggggggggggggggggggtggggggggggggggggggggggggggg
gggggggggggggggggggggggggggggggggggggggggggggggggggggcgggggg
gggggggggggggggggggggggggggggggggggggggggggaaagggggggggggggg
ggggggaggggggggggggggggggggggggggggggtgggggggggggggggggggggg
gggagggggggggggggggggggggggggggggggggggggggggggggggggggggggg
gggggggggggggggggggggcgggggggggggggggggggggggggtgggggggggggg
gggggggggggggggggggggggggggtgggggggggggggcgggggggggggggggggg
gggggggggggggggggggggggggggcgggggggggggggggggggggggggggggggg
ggaaaaaaaagggggggggcgggggggggggggggggggggggggggtgggggggggggg
gggggggggggggggggggggtgcggggggtggggggggggggggggggggggggggggg
ggggggggggggggggggggggtggggggggggggggggggggggggtgggggggggggg
gggggggggggggcggggggtgggggggggggtgggtggggggggggggggggggggtgg
ggcggggtgggggggggtgggggggggggggggggggggggggggggtggggggggtggg
gggggggggggggggggggggggggggggggggggggggggggggggggggggggggggg
gggggggggggggggggggggggggggggggggggggggggggggggggggggggggggg
gggggggggggggggggggggggggggggggggggggggggggggggggggggggcgggg
gggggggggggggggggggggggggggggggggggggggggggggggggggggggggggg
gggggggggggggggggggggggggggggggggggggggggggggggggggggggggggg
gggggggggggggggggggggggggggggggggggggggggggggggggggggggggggg
gggggggggggggggggggggggggggggggggggggggggggggggggggggggggggg
```

Figure 5.15: Assembling Short over Long Reads

5.7 Scaffolding

Scaffolding option is chosen to start the scaffolding process. It shows the following page given in Fig 5.16. The assembled sequence file is chosen from the dropdown menu. The dropdown menu shows all the files available on user's specific folder. The chosen sequence file and supplementary file is needed to be specified to start the scaffolding process. Clicking on "Submit" button starts the computation on the HPC after copying necessary files on HPC. The generated result file is then sent to the server. The generated result files are shown in Fig 4.17. These files can be downloaded on the user's pc. The content of result file is shown in Fig 5.18.

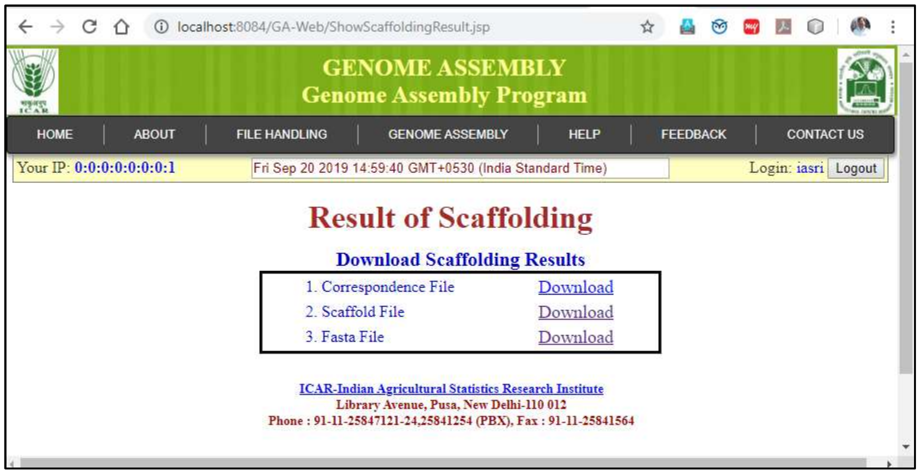


Your IP: 0:0:0:0:0:0:0:1 Thu Oct 24 2019 16:00:36 GMT+0530 (India Standard Time) Login: iasri

Scaffolding of Aligned Reads

Computing option:	✓ Omit Computing
Select Read Alignment File:	corrected_final.fa
Select Supplementary Format File:	corrected_final.fa

Figure 5.16: Scaffolding



Your IP: 0:0:0:0:0:0:0:1 Fri Sep 20 2019 14:59:40 GMT+0530 (India Standard Time) Login: iasri

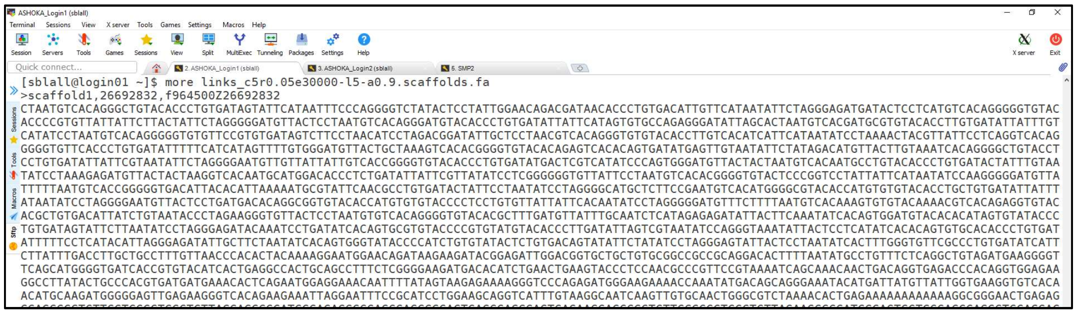
Result of Scaffolding

Download Scaffolding Results

1. Correspondence File	Download
2. Scaffold File	Download
3. Fasta File	Download

ICAR-Indian Agricultural Statistics Research Institute
Library Avenue, Pusa, New Delhi-110 012
Phone : 91-11-25847121-24,25841254 (PBX), Fax : 91-11-25841564

Figure 5.17: Result of Scaffolding


ASHOKA_Login1 (sblall)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
X server Exit
Quick connect...
2. ASHOKA_Login1 (sblall)
3. ASHOKA_Login2 (sblall)
5. SMP2

```
[sblall@login01 ~]$ more links_c5r0.05e30000-l5-a0.9.scaffolds.fa
>scaffold1,26692832,f964500Z26692832
CTAATGTCACAGGGCTGTACACCCTGTGATAGTATTCATAATTTCCCAGGGGTCTATACTCCTATTGGAACAGACGATAACACCCTGTGACATTGTTCATAATATTCTAGGGAGATGATACTCCTCATGTCACAGGGGGTGTAC
ACCCCGTGTTATTATTCTTACTATTCTAGGGGGATGTTACTCCTAATGTCACAGGGATGTACACCCTGTGATATTATTCATAGTGTGCCAGAGGGATATTAGCACTAATGTCACGATGCGTGTACACCTTGTGATATTATTTGT
CATATCCTAATGTCACAGGGGGTGTGTTCCGTGTGATAGTCTTCCTAACATCCTAGACGGATATTGCTCCTAACGTCACAGGGTGTGTACACCTTGTCACATCATTCATAATATCCTAAAACTACGTTATTCCTCAGGTCACAG
GGGGTGTTCACCCTGTGATATTTTTCATCATAGTTTTGTGGGATGTTACTGCTAAAGTCACACGGGGTGTACACAGAGTCACACAGTGATATGAGTTGTAATATTCTATAGACATGTTACTTGTAAATCACAGGGGCTGTACCT
CCTGTGATATTATTCGTAATATTCTAGGGGAATGTTGTTATTATTGTCACCGGGGTGTACACCCTGTGATATGACTCGTCATATCCCAGTGGGATGTTACTACTAATGTCACAATGCCTGTACACCCTGTGATACTATTTGTAA
TATCCTAAAGAGATGTTACTACTAAGGTCACAATGCATGGACACCCTCTGATATTATTCGTTATATCCTCGGGGGGTGTTATTCCTAATGTCACACGGGGTGTACTCCCGGTCCTATTATTCATAATATCCAAGGGGGATGTTA
TTTTTAATGTCACCGGGGGTGACATTACACATTAAAAATGCGTATTCAACGCCTGTGATACTATTCCTAATATCCTAGGGGCATGCTCTTCCGAATGTCACATGGGGCGTACACCATGTGTGTACACCTGCTGTGATATTATTT
ATAATATCCTAGGGGAATGTTACTCCTGATGACACAGGCGGTGTACACCATGTGTGTACCCCTCCTGTGTTATTATTCACAATATCCTAGGGGGATGTTTCTTTTAATGTCACAAAGTGTGTACAAAACGTCACAGAGGTGTAC
ACGCTGTGACATTATCTGTAATACCCTAGAAGGGTGTTACTCCTAATGTGTCACAGGGGTGTACACGCTTTGATGTTATTTGCAATCTCATAGAGAGATATTACTTCAAATATCACAGTGGATGTACACACATAGTGTATACCC
TGTGATAGTATTCTTAATATCCTAGGGAGATACAAATCCTGATATCACAGTGCGTGTACCCCGTGTATGTACACCCTTGATATTAGTCGTAATATCCAGGGTAAATATTACTCCTCATATCACACAGTGTGCACACCCTGTGAT
ATTTTTCCTCATACATTAGGGAGATATTGCTTCTAATATCACAGTGGGTATACCCCATCTGTGTATACTCTGTGACAGTATATTCTATATCCTAGGGAGTATTACTCCTAATATCACTTTGGGTGTTCGCCCTGTGATATCATT
CTTATTTGACCTTGCTGCCTTTGTTAACCCACACTACAAAAGGAATGGAACAGATAAGAAGATACGGAGATTGGACGGTGCTGCTGTGCGGCCGCCGCAGGACACTTTTAATATGCCTGTTTCTCAGGCTGTAGATGAAGGGGT
TCAGCATGGGGTGATCACCGTGTACATCACTGAGGCCACTGCAGCCTTTCTCGGGGAAGATGACACATCTGAACTGAAGTACCCTCCAACGCCCGTTCCGTAAAATCAGCAAACAACTGACAGGTGAGACCCACAGGTGGAGAA
GGCCTTATACTGCCCACGTGATGATGAAACACTCAGAATGGAGGAAACAATTTTATAGTAAGAGAAAAGGGTCCCAGAGATGGGAAGAAAACCAAATATGACAGCAGGGAAATACATGATTATGTTATTGGTGAAGGTGTCACA
ACATGCAAGATGGGGGAGTTGAGAAGGGTCACAGAAGAAATTAGGAATTTCCGCATCCTGGAAGCAGGTCATTTGTAAGGCAATCAAGTTGTGCAACTGGGCGTCTAAAACACTGAGAAAAAAAAAAAAAGGCGGGAACTGAGAG
```

Figure 5.18: Content of Scaffolded file

5.8 Help

Finally, online help facility is provided to the users upon going to the “Help” menu option. It includes the details on how to use the software for signup, data management and using the process of assembly. User can click on the help option provided on the horizontal menu provided on each page of the software.

CHAPTER –VI

CONCLUSIONS

Researchers and developers have been working very hard to address the issue of complexity involved in the computational process of genome assembly. As presented in Chapter-II of this report, many algorithms have been developed to reduce the complexity involved in genome assembly. However, more efforts are needed to be put to address this problem so that genome assembly process may be simpler, less complex and faster with the available computational resources.

In this project, the team worked towards this problem to get a faster hybrid genome assembler. This assembler uses high-end computing resources for correction of errors on long read sequences by aligning them with short reads and then assemble the long reads using an efficient parallelized assembly program.

The genome assembler developed under the project carries out the assembly process in the form of pipeline using fast available algorithms. The pipeline carries out the processes such as pre-processing of short and long read sequences, correcting the errors in the long reads, assembly of corrected long reads using a fast and parallelized assembler to form contigs and finally scaffolding these contigs using a parallelized scaffolding tool. The whole process of computation can be carried out using a web browser for convenience of the user. The results of every computational steps are downloaded to the client machine for viewing. The output the of the developed software was also tested for its quality using a web-based tool.

It is known that new technologies in the computational and hardware resources are emerging very fast which is making heavy computational task easier. Therefore, the process of assembly may further be improved with availability of more high-end computational resources. Further, new and improved algorithms may be developed for error correction on long read sequences. Development of improved algorithms for assembly of long reads and scaffolding can also provide a better genome assembler in future.

सारांश

वर्तमान डी-नोवो असेम्बलर्स मुख्य रूप से वर्तमान एकल-अणु अनुक्रमण तकनीकों द्वारा उत्पन्न लंबे समय से पढ़े जाने वाले अनुक्रमण डेटा का प्रभावी रूप से उपयोग करने में असमर्थ हैं क्योंकि इसमें मुख्य रूप से काफी त्रुटि दर है। इस परियोजना में, कुशल असेंबली परिणामों के लिए लंबे और छोटे दोनों प्रकार के रीड्स की आवश्यकता होती है। लंबी रीडिंग पर त्रुटि सुधार लंबी रीडिंग पर छोटी रीडिंग को संरेखित करके लंबे रीडिंग पर त्रुटियों को कम करने और उन्हें असेंबली के लिए उपयोग करने के द्वारा किया गया था। हमारा दृष्टिकोण छोटी, उच्च-पहचान वाले अनुक्रमों के साथ पूरक करके इस तकनीक का उपयोग करता है, जिसके परिणामस्वरूप लंबे, सटीक टेप और बेहतर असेंबलियां होती हैं। हमारे हाइब्रिड दृष्टिकोण का परिणाम कम त्रुटियों और अंतराल के साथ उच्च गुणवत्ता की असेंबली है, जो जीनोम परिष्करण की महंगी लागत को कम कर देगा और अधिक सटीक डाउनस्ट्रीम विश्लेषण को सक्षम करेगा। जीनोमिक्स के सभी पहलुओं, विशेष रूप से जीनोम एनोटेशन और तुलनात्मक जीनोमिक्स के लिए उच्च-गुणवत्ता की असेंबली महत्वपूर्ण हैं। यह स्पष्ट है कि उच्च-गुणवत्ता वाली असेंबली, लंबी अखंडित स्पंजी के साथ, विस्तृत विषयों पर सकारात्मक प्रभाव डालेगी।

इस तरह, यह देखा गया है कि उच्च त्रुटि दर जीनोम असेंबली के लिए एक बाधा नहीं बन सकती है। हाई-एरर, लॉन्ग रीड्स को किसी भी पूर्व तकनीक के साथ असेंबलियों का उत्पादन करने के लिए पूरक शॉर्ट-रीड्स के साथ संयोजन में कुशलतापूर्वक इकट्ठा किया जा सकता है, जो हमें "एक गुणसूत्र, एक कंटिग के लक्ष्य के करीब एक कदम आगे लाएगा। पैकबायो और अन्य प्रौद्योगिकियां, जैसे कि आयन टोरेंट, एक बार आवश्यक होने के कुछ समय में उच्च गुणवत्ता वाले जीनोम असेंबलियों का उत्पादन करना संभव बनाती हैं।

जैव सूचना विज्ञान में कई उपकरण समानांतर कम्प्यूटेशनल बुनियादी ढांचे पर परिणाम प्राप्त करने के लिए समानांतर कम्प्यूटेशनल बुनियादी ढांचे पर चलते हैं क्योंकि भारी कम्प्यूटेशनल एल्गोरिदम या जॉब के आकार शामिल हैं। इस काम में, सुपरकंप्यूटिंग इन्फ्रास्ट्रक्चर पर स्थापित समानांतर उपकरणों का उपयोग तेज परिणामों के लिए किया गया था। जीनोम असेंबली को एचपीसी पर्यावरण पर एक पाइपलाइन रूप और चल औजारों में किया जाता है। यह अध्ययन असेंबली के विभिन्न घटकों के लिए एक वेब-आधारित सॉफ्टवेयर बनाने के उद्देश्यों के साथ शुरू किया गया था - प्रीप्रोसेसिंग, त्रुटि सुधार के लिए संरेखण, लॉन्ग रीड के लिए असेंबली और मचान। सॉफ्टवेयर को जे एस पी, जावा, एच टी एम एल और सी एस एस का उपयोग करके विकसित किया गया है। यह सॉफ्टवेयर शामिल सभी चरणों के लिए गणना की एक श्रृंखला करता है। इन गणनाओं को तेज परिणाम प्राप्त करने के लिए अशोका सुपरकंप्यूटिंग सिस्टम पर किया जाता है। परिणाम ब्राउज़र पर उपयोगकर्ता को दिखाए जाते हैं जिसे ग्राहक की स्थानीय हार्ड डिस्क पर भी डाउनलोड किया जा सकता है।

SUMMARY

Current *de-novo* assemblers are unable to effectively use the long-read sequencing data generated by present single-molecule sequencing technologies primarily because of the considerable error rate. In this project, both long and short reads have been required for efficient assembly results. The error correction on long reads were performed by aligning short reads over long reads to get reduced errors on the long reads and use them for assembly. Our approach exploits this technology by complementing it with shorter, high-identity sequences resulting in long, accurate transcripts and improved assemblies. The result of our hybrid approach is higher quality assemblies with fewer errors and gaps, which will drive down the expensive cost of genome finishing and enable more accurate downstream analyses. High-quality assemblies are critical for all aspects of genomics, especially genome annotation and comparative genomics. It is clear that higher-quality assemblies, with long unbroken contigs, will have a positive impact on a wide range of disciplines.

This way, it is noticed that high error rates do not become a barrier to genome assembly. High-error, long reads can be efficiently assembled in combination with complementary short-reads to produce assemblies not possible with any prior technology, bringing us one step closer to the goal of "one chromosome, one contig." The rapid turnaround time possible with PacBio and other technologies, such as Ion Torrent, can make it possible to produce high-quality genome assemblies at a fraction of the time once required.

Many tools in bioinformatics run on parallelized computational infrastructure for getting results in a comparatively less time because of heavy computational algorithms or job sizes involved. In this work, the parallelized tools installed on supercomputing infrastructure were utilized for faster results. The genome assembly is carried out in a pipeline form and running tools on HPC environment. This study was undertaken with the objectives to create a web-based software for various components of assembly namely – pre-processing, alignment for error correction, long read assembly and scaffolding. The software has been developed using JSP, Java, HTML and CSS. This software does a series of computations for all the steps involved. These computations are done on ASHOKA supercomputing system to get the faster results. The results are shown to the user on the browser which can also be downloaded to the client's local hard disk.

REFERENCES

1. Bushnell Brian. (2014). BBMap: A Fast, Accurate, Splice-Aware Aligner. Berkeley, CA: Ernest Orlando Lawrence Berkeley National Laboratory.

2. Chevreux, B. *et al.* (2004). Using the miraEST assembler for reliable and automated mRNA transcript assembly and SNP detection in sequenced ESTs. Genome Res. 14, 1147–1159.

3. H. Backman TW, Girke T. (2016). systemPipeR: NGS workflow and report generation environment. *BMC Bioinformatics* 17:388.

4. Koren, S., Schatz, M.C., Walenz, B.P., Martin, J., Howard, J.T., Ganapathy, G., Wang, Z., Rasko, D.A., McCombie, W.R., Jarvis, E.D., et al. (2012). Hybrid error correction and *de-novo* assembly of single-molecule sequencing reads. Nature Biotechnology 30, 693–700.

5. Lee, H., Gurtowski, J., Yoo, S., Marcus, S., McCombie, W.R., and Schatz, M. (2014). Error correction and assembly complexity of single molecule sequencing reads. bioRxiv.

6. Li, H. (2016). Minimap and miniasm: fast mapping and *de-novo* assembly for noisy long sequences. Bioinformatics.

7. Li, Z., Chen, Y., Mu, D., Yuan, J., Shi, Y., Zhang, H., Gan, J., Li, N., Hu, X., Liu, B., et al. (2012). Comparison of the two major classes of assembly algorithms: overlap-layout-consensus and de-bruijn-graph. Briefings in Functional Genomics 11, 25–37.

8. Lin, H., and Liao, Y. (2015). Evaluation and Validation of Assembling Corrected PacBio Long Reads for Microbial Genome Completion via Hybrid Approaches. PLOS ONE 10, e0144305.

9. Liu, Y., Schmidt, B., and Maskell, D.L. (2012). CUSHAW: a CUDA compatible short read aligner to large genomes based on the Burrows-Wheeler transform. Bioinformatics 28, 1830-1837.

10. Miclotte, G., Heydari, M., Demeester, P., Rombauts, S., Van de Peer, Y., Audenaert, P., and Fostier, J. (2016). Jabba: hybrid error correction for long sequencing reads. Algorithms for Molecular Biology 11.

11. Salmela, L., and Rivals, E. (2014). LoRDEC: accurate and efficient long read error correction. Bioinformatics 30, 3506–3514.

12. Utturkar, S.M., Klingeman, D.M., Land, M.L., Schadt, C.W., Doktycz, M.J., Pelletier, D.A., and Brown, S.D. (2014). Evaluation and validation of *de-novo* and hybrid assembly techniques to derive high-quality genome sequences. Bioinformatics 30, 2709–2716.

13. Vaser, R., Sovic, I., Nagarajan, N., and Sikic, M. (2016). Fast and accurate *de-novo* genome assembly from long uncorrected reads. bioRxiv.

14. Wang, Y., Yu, Y., Pan, B., Hao, P., Li, Y., Shao, Z., Xu, X., and Li, X. (2012). Optimizing hybrid assembly of next-generation sequence data from Enterococcus faecium: a microbe with highly divergent genome. BMC Systems Biology 6, S21.

15. Wang JR, Holt J, McMillan L, and Jones CD. (2018). FMLRC: Hybrid long read error correction using an FM-index. BMC Bioinformatics. 19(1):50. doi: 10.1186/s12859-018-2051-3. PubMed PMID: 29426289; PubMed Central PMCID: PMC5807796.

16. Ye, C., Hill, C.M., Wu, S., Ruan, J., and Ma, Z. (Sam) (2016). DBG2OLC: Efficient Assembly of Large Genomes Using Long Erroneous Reads of the Third Generation Sequencing Technologies. Scientific Reports 6, 31900-31906.

17. Yeo, S., Coombe, L., Chu, J., Warren, R. L. & Birol, I. (2017). ARCS: scaffolding genome drafts with linked reads. *Bioinformatics* https://doi.org/10.1093/bioinformatics/btx675.

18. Zimin, A.V., Puiu, D., Luo, M.-C., Zhu, T., Koren, S., Yorke, J.A., Dvorak, J., and Salzberg, S. (2016). Hybrid assembly of the large and highly repetitive genome of Aegilops tauschii, a progenitor of bread wheat, with the mega-reads algorithm. bioRxiv.

***