IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727
PP 01-05
www.iosrjournals.org

Towards Development of Genome Assembly Framework using improved Clustering Techniques

Prof. Virendra Bagade[1], Dr. K Dhanasekaran[2]
[1]*Computer Engineering, PICT/SPPU, Pune, Maharashtra, India*
[2]*Computer Science and Engineering, Jain College of Engineering, VTU, Belgaum, Karnataka, India*

Abstract: *Genome assembling has been an active research area since the DNA structure was discovered and has gained much attention after the Human Genome project was launched. A large number of genomes have been assembled and many more are in the pipeline. A number of full-scale assemblers and other special-purpose modules have been reported. Since the volume of data involved in the genome assembly process is very large and requires significantly large computational power and processing time, many assemblers have utilized parallel computing to achieve faster and more efficient reconstruction of the DNA. A genome assembler has a multistep process which includes different components that may be partly or fully parallelized. Assembling of large genomes from tens of millions of short genomic fragments are computationally demanding, requires hundreds of gigabytes of memory and tens of thousands of CPU hours. Therefore, new gene-enrichment sequencing strategies are expected to further exacerbate this situation. In this paper, we present a massively parallel genome assembly framework. The unique features of our approach include space-efficient and on-demand algorithms that consume only linear space, and heuristic strategies that reduce the number of expensive pair-wise sequence alignments while maintaining assembly quality.*

Keywords: *Parallel algorithms, genome assembly framework, high performance bioinformatics*

I. Introduction

Each cell in a living organism contains one or more long DNA sequences called chromosomes, collectively known as genome. The genome contains DNA sequences called genes that encode instructions for producing proteins and RNA molecules, which perform various cellular functions in an organism. Deciphering an entire genome sequence and identifying regions that include genes and regulatory elements is of fundamental importance in molecular and functional genomics.

Genome assembly is defined as the process of grouping reads into contigs and then contigs into scaffolds. Contigs can be represented by a string of four letters namely, adenine (A), guanine (G), cytosine(C), and thymine (T). A scaffold linkage is represented by a directed graph.The rapid pace of development in the field of sequencing technology has laid a solid foundation for the whole genome shotgun assembly approach. Genomes span multiple length scales from a few tens of thousands of nucleotides in viruses to millions of nucleotides in microbes to billions of nucleotides in complex eukaryotic organisms such as plants and animals. The biochemical procedure of determining the nucleotide sequence of a DNA molecule is called sequencing. To extend the reach of sequencing to genomic scales, long genomic stretches are sampled at uniform random locations by a procedure called shotgun sequencing, this results in numerous short DNA fragments that can be sequenced using conventional techniques. If this procedure is directly applied to an entire genome, it is called Whole Genome Shotgun (WGS) sequencing. After generating and sequencing such fragments, the target genome is computationally assembled from them.

Related to sequencing strategies, many genome assembly programs have been developed, for example, Arachne[3], Atlas[10], CAP3[11], Celera Assembler [13], Euler[6], Gig Assembler[14], PCAP [12], Phrap[4], Phusion [5] and TIGR Assembler[11]. In this paper, we present a parallel genome assembly framework for human genome and other large-scale genome sequencing applications.

II. Overview Of Genome Assembly

DNA material is present in all living things and it contains all relative information. DNA material is very long in nature and contains more than 10,000 base pairs. These base pairs contain mainly four elements (i.e. A, G, C, T). To study DNA directly is very difficult task. DNA fragmentation is the task of separating or breaking of DNA strands into pieces. DNA fragmentation helps in analyzing or processing on DNA fragments. After DNA fragmentation, it requires reconstruction to form original DNA to identify the living thing. Research has shown that the problems like Computational errors, unknown origin of DNA, repetition of similar sequence occur while reconstructing the fragments into the original one. To solve or minimize these problems, many

researchers have proposed different methods [1]. The interleaving problem with complex scaffold linkage has been classified as a non-deterministic polynomial time hard problem [7]. Further, existing assemblers are not optimized for reads of 80-120 bases. And, no current sequencing process is capable of directly producing an organism's entire string of millions or billions of bases. Instead, the process produces a large number of random substrings of the sequence known as reads. Reads can vary in length from 25 to 1,000 bases, depending on the sequencing technology [8]. Genome assembly is the computational process of arranging reads in the correct order to produce the largest possible contiguous strings known as contigs.

III. Clustering based parallel framework for genome assembly

This section describes various stages of DNA sequencing in detail. Also, it should be noted that this study is focused mainly on the genome assembly framework using OLC, DBG and proposed hybrid method which is shown below in Figure1 illustrates the process flow and the data produced by framework using improved clustering techniques.

In this paper, we present first the massively parallel genome assembly framework. Our approach guarantees a worst-case O(n) total space complexity despite gene-enrichment, repeats, and uses heuristic strategies to significantly reduce run-time. The result demonstrates the effectiveness of our massively parallel framework for the assembly of the human genome and other impending large-scale genome sequencing project.

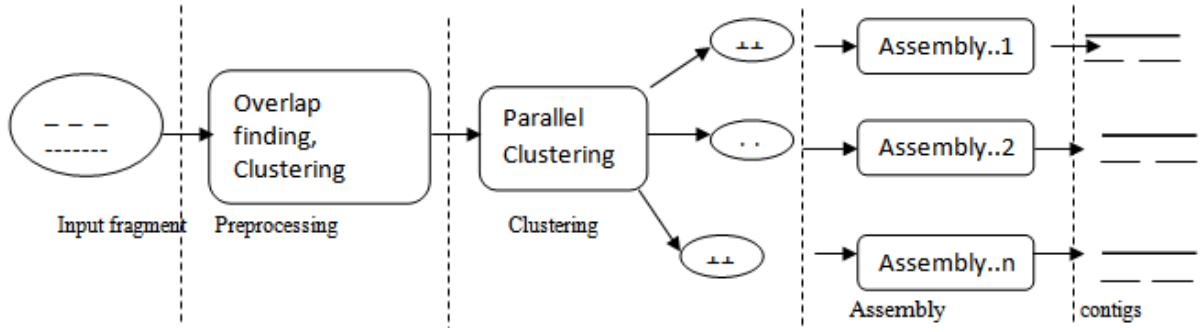

Fig1: The system diagram with preprocessed fragments and parallel clusters

3.1 High performance

The original algorithm is sequential and each step in the iteration depends upon the previous one. A number of attempts were made to parallelize this algorithm to attain a significant speedup. Here, we look into an improved approach presented by Yap et al. [9, 10] who studied other similar sequential algorithms that were parallelized using "speculative computation". This approach led to a higher speedup and a more scalable implementation in addition to guaranteeing the same results as achieved by the original Burger–Munson algorithm. The original Burger–Munson algorithm does not lend itself well to a parallel programming environment due to the high degree of dependence among different steps. The algorithm divides the sequences into two groups. It then compares sequences from one group to the second group and attempts to align the sequences by inserting gaps. A score is calculated for every alignment, as described earlier. The alignment with higher score is accepted and kept and the one with a lower score is rejected.

IV. Proposed Methodology

The proposed methodology consists of following major tasks:

4.1. Designing a Parallel Architecture for Genome Assembly using improved clustering mechanism

Parallel-MPI is targeted for workstation clusters with distributed memory architecture, which consist of divide and conquer strategy to reduce execution time.

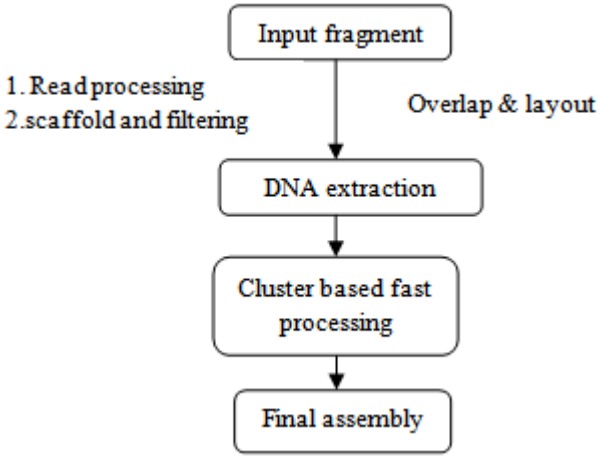

Fig 2: The system diagram with process flow

Dispatcher (Master) in our work partitions subject database or MSA tasks into a number of chunks in preprocessing steps and sends them to compute nodes.
An algorithm on a Single Node (Worker) is designed to receive sequence chunks from master and performs the corresponding DP calculations.
Result Collector (Master) performs additional operations which are required to further process the returned results.

4.2 Determining The Validity Of Assembly

A complete and correct genome sequence is a fundamental requirement to a wide variety of analyses and is a key to unlocking the health and biology of the organism [17]. Distance-matrix calculation which is a mixed fine and coarse-grained approach is used for final validity.

Table1: Basic Assembly Metrics

Metric	Description
Assembly size	N50/NG50
N50/NG50	Length of contigs ,scaffolds
Coverage	Percentage of the genome/gene that is contained in the assembly. Genome coverage of 90–95% is generally considered to be good.
Errors	Three kinds of graph errors: tips; bubbles; and erroneous connections.

The proposed flow of the system is as under.

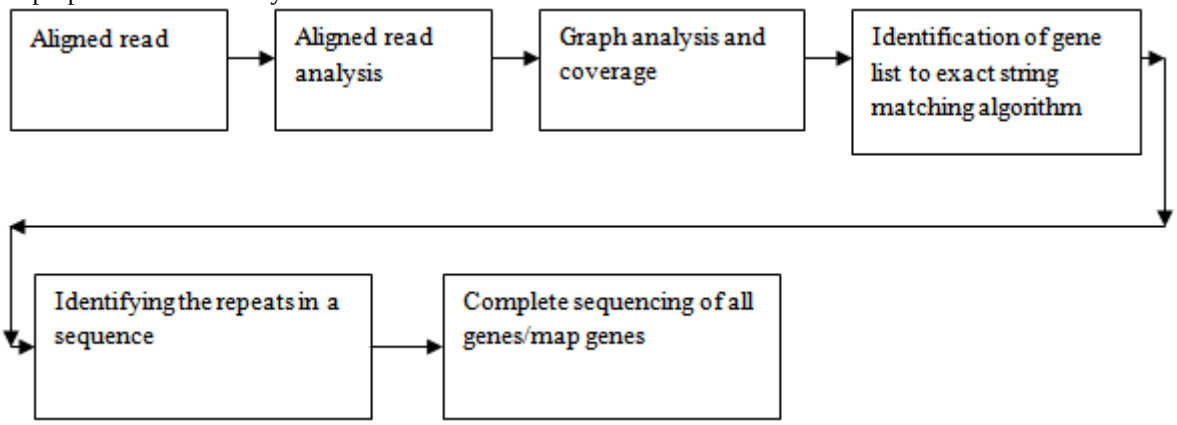

Fig 3: Framework component start to end

Algorithmic enhancement includes,

a) Reducing computational power and processing time
b) Quantifying accuracy or reliability of weak bases and
c) De novo assembly, transcriptome and reference genome assembly (for high throughput).

There are times during parallel processing when a processor has to wait and synchronize with other processors before proceeding with the next steps. A well-designed algorithm can effectively use such an idle time, which can not only provide extra functionality, but also enhance the utilization and therefore the efficiency of the entire system.

V. Quality and performance evaluation

From computer science perspective, solving bioinformatics problems require the design and development of intelligent algorithms, which can provide speedy, accurate, and scalable solutions that are economically viable to implement. Since there are many different algorithms and various implementations of each of them, it becomes necessary to evaluate such techniques and the corresponding implementations using a high-level frame of reference and a common set of metrics to allow a comparative analysis. Furthermore, for various algorithms in the realm of parallel and distributed computing, it is essential to have the ability to compare them both quantitatively and qualitatively, using the well-established metrics. Table1 shows a list of such metrics that should be used to distinguish one algorithm from another [16]. It should be noted that it is not possible to declare one approach to be the best of all, as different problems require their unique criteria to be met.

Table 2: Key measures to evaluate algorithms

Metric/Measure	Algorithm(A)/Implementation(I)
Time Complexity	A
Processing time	I
Accuracy	A,I
Efficiency	I
Scalability	A,I

Table 3: Feature and Challenges

Sequencing Technology Feature	**Assembly Challenges**
Short reads	Difficult assembling repeats
Mate-pairs absent or difficult/expensive to obtain	Difficult assembling repeats, Lack of scaffolding information
New types of errors	Need to modify existing software and/or incorporate technology- specific features in assembly software
Large amounts of data (number of reads and size of auxiliary information)	Efficiency issues Require parallel implementations or specialized hardware when applied to large genomes

There is generally a trade-off between performance and quality where performance can be defined as the faster processing of the data and the quality can be assessed using certain standards to measure accuracy and reliability of the solution. Depending upon the need at hand, a faster, easy to use, and more economically viable program may be more desirable than a slow and complex.

VI. Conclusion

The primary contribution of this paper is to streamline and present in-depth the entire process of genome assembly framework in the realm of high-performance computing. It should be noted that there are many different methodologies and variations of processes and algorithms discussed. Clearly, there is a great potential for enhancing both the speed and quality of genome assembly by employing parallel and distributed computing throughout the process. This paper evaluates and emphasizes specific areas with such a potential. For certain areas, e.g. pairwise alignment and repeat finding, the paper has laid the foundation of further research work by proposing high-level parallel algorithms to enhance performance of some modules. As processors and memory modules become more cost-effective, many of the sequential algorithms can be parallelized to improve speed. This in turn will allow for more complex algorithms to be implemented, thereby resulting in an overall

improved quality of the assembly. This paper serves as a guiding step in the direction of exploring the specific research areas by furthering the work in the field of genome assembling.

References

Journal Papers

[1]. Yiming He, Zhen Zhang, Xiaoqing Peng, Fangxiang Wu, and Jianxin Wang , De Novo Methods for Next Generation Sequencing Data, *ISSNll1007-0214ll08/11llpp500-514 Volume 18, Number 5, October 2013*

[2]. A. Kalyanaramana, S.J. Emrichb,c, P.S. Schnablec,d, S. Alurub,c, Assembling genomes on large-scale parallel computers, *Journal of parallel and distributed computing,Elsevier 9 June 2007*

[3]. Serafim Batzoglou,David B. Jaffe,ARACHNE: A Whole-Genome Shotgun Assembler, *Cold Spring Harbor Laboratory Press ISSN 1088- 9051/01*

[4]. Melissa de la Bastide andW. Richard McCombie,Assembling Genomic DNA Sequences with PHRAP, *Current Protocols in Bioinformatics* (2007) 11.4.1-11.4.15

[5]. James C. Mullikin and Zemin Ning,The Phusion Assembler, *Cold Spring Harbor Laboratory Press ISSN 1088-9051/03*

[6]. Pavel A. Pevzner,Haixu Tang, and Michael S. Waterman,An Eulerian path approach to DNA fragment assembly,*August 14, 2001,Vol 98*

[7]. Higgins D, Sharp P (1988) CLUSTAL: a package for performing multiple sequence alignment on a microcomputer. Gene 73(1):237–44

[8]. Huang X, Wang J, Aluru S, Yang S, Hillier L PCAP: a whole-genome assembly program. Genome Res 13:2164–2170(2003)

[9]. A. Kalyanaramana, S.J. Emrichb,c, P.S. Schnablec,d, S. Alurub,c, Assembling genomes on large-scale parallel computers, *Journal of parallel and distributed computing,Elsevier 9 June 2007*

[10]. Yap T, Frieder O, Martino R , Parallel computation in biological sequence analysis, *IEEE Trans Parallel Distrib Syst 9(3) :283–294@1998*

[11]. Granger G.Sutton, Owen White, Mark D.Adams,TIGR Assembler:A New Tool for Assembling Large Shotgun Sequencing Projects, *Genome Science & Technology Volume 1, Number 1, 1995*

[12]. Xiaoqiu Huang1,2 and Anup Madan3 CAP3: A DNA Sequence Assembly Program, *Cold Spring Harbor Laboratory Press ISSN 1054- 9803/99 @1999*

[13]. Eugene W. Myers, Granger G. Sutton,A Whole-Genome Assembly of Drosophila ,The drosophila Genome, *Science vol 287 24March 2000*

[14]. W. James Kent and David Haussler,Assembly of the Working Draft of the Human Genome with GigAssembler, *Cold Spring Harbor Laboratory Press ISSN 1088-9051 @2001*

Books:

[15]. Ali Masoudi-Nejad,Zahra Narimani,Nazanin Hosseinkhan,*Next generation sequencing and sequence assembly methodologies and algorithms* (Springer New York Heidelberg Dordrecht London: Springer,2013).

Theses:

[16]. Michael Christopher Schatz, *High performance computing for sequence alignment and assembly*, doctoral diss., *University of Maryland*, 2010.

Proceedings Papers:

[17]. Yap T,Munson P, Frieder O, Martino R, Parallel multiple sequence alignment using speculative computation. *In*: *Proceedings of the international conference on parallel processing@1995*