Design and Simulation of Hybrid de novo DNA Sequence Assembly for Large Eukaryotic Genomes

Wisnu Ananta Kusuma, Yutaka Akiyama
Department of Computer Science
Graduate School of Information Science and Engineering
Tokyo Institute of Technology, 2-12-1-W8-76, Ookayama, Meguro-ku, Tokyo 152-8552, Japan

Abstract - *We propose an approach for de novo DNA sequence assembly using very short reads of a large eukaryotic genome. Our assembler combines two main approaches in DNA sequence assembly, Overlap Layout Consensus (OLC) and Eulerian Path, with two types of reads to obtain significant contigs. This approach will be applied in a parallel environment. In this paper, we present the data and functional requirements for implementation in our assembler. We also present our hybrid assembler strategy and architecture. We simulate the core of the hybrid assembly and find that the hybrid strategy tends to produce long contigs. It is expected that this approach will generate an accurate and efficient DNA sequence assembly for large eukaryotic genomes.*

Keywords: de novo DNA sequence assembly; very short read; eukaryotic genome; high-performance computing

1 Introduction

DNA determines the characteristics of an organism. Therefore, sequencing DNA is very important. There is no technique for determining an organism's entire DNA sequence in a single process. Most of the technologies for sequencing genomes depend on the shotgun method, in which genomes are randomly cut into many fragments and computer programs are required to reconstruct the DNA sequence. Fragments of DNA sequences can be assembled in two ways: by mapping the DNA sequences to a reference sequence or by de novo DNA sequence assembly, which requires no reference sequence and can be used for newly sequenced organisms. De novo sequence assembly is conducted by detecting overlaps between fragments (reads) and merging them into longer fragments (contigs).

Many researchers have been successful in developing DNA sequence assemblers for long reads (500 bp) based on the Overlap Layout Consensus approach [1], [2], [3], [4].

This work is supported in part by the Global Centre of Excellence (GCOE) CompView, Tokyo Institute of Technology (email: ananta@bi.cs.titech.ac.jp).

This approach represents the sequence assembly problem as an overlap graph. In this graph, each node represents a read, and each edge represents overlap between two reads.

However, the OLC approach is not suitable for assembling the very short reads (25 – 50 bp) generated by recent sequencers. Applying the OLC approach to very short reads makes the overlapping stage more difficult. The abundant repeats in very short reads increase the complexity of the graph. Pevzner [5] attempted to solve this problem by introducing an Eulerian Path approach using the de Bruijn Graph. In this approach, elements are not organized around reads but around words of k nucleotides, called k-mers. This approach has encouraged researchers to develop genome assemblers for very short reads [6], [7].

Unlike sequence assemblers for long reads, which use eukaryotic genome datasets, most of the assemblers for very short reads are aimed at assembling prokaryotic genomes. The size of dataset limits researchers from using eukaryotic genomes as datasets because constructing the graphs is expensive in terms of memory and time. However, some researchers have succeeded in implementing parallelization to reduce the execution time in constructing and manipulating the graph during eukaryotic genome assembly [8], [9], [10].

In this paper, we contribute the design of a high-performance de novo DNA sequence assembly method for large eukaryotic genomes. We introduce a hybrid assembler strategy and architecture designed by combining two DNA sequence assembly approaches and two types of reads to obtain an accurate and efficient DNA sequence assembly for large eukaryotic genomes.

2 de novo DNA Sequence Assembly

2.1 A common problem in DNA sequence assembly

The problem of assembling DNA sequences can be formulated into the problem of finding the shortest common superstring. Suppose we have a DNA sequence from an unknown source of A= a_1, a_2,..,a_L. Shotgun

sequencing of A produces a set of reads (or fragments) $F = \{f_1, f_2,\ldots,f_R\}$ that are sequences over the alphabet $\sum = \{A, C, G, T\}$. To reconstruct the sequence A from the set of its fragments $F = \{f_1, f_2, \ldots,f_R\}$, we should find the shortest possible string S such that for every $f \in F$, S is a superstring of *f*.

Basically, this problem can be solved by finding the region of overlap among the fragments. The classical approach, Overlap Layout Consensus, applies an overlap graph to represent elements of this problem. Each node represents a read, while each edge represents the overlap between two reads. A solution can be obtained by finding a path that visits each node exactly once. This computation is a Hamiltonian Path problem, which is classified as an NP hard problem. Moreover, this problem will be more complicated by the presence of repeats and the existence of sequencing errors.

Repeats are multiple copies of identical substrings at different positions in the DNA. Repeats are much more prevalent in very short reads because the overlap region is shorter. It will increase the possibility of finding the same substring, not just in the overlap region but also in other parts of the read. Thus, it causes ambiguity and tends to generate mis-assembled contigs. Furthermore, sequencing errors produced by the sequencer increase the complexity of the graph topology because these errors yield erroneous edges in the graph.

2.2 DNA Sequence Assembly Approach

There are two main approaches to dealing with DNA sequence assembly: Overlap Layout Consensus (OLC) and the Eulerian Path approach. The OLC approach consists of three steps: overlap, layout, and consensus.

In the overlap step, we first find potentially overlapping reads by a greedy approach. This information is used to construct an overlap graph by the following procedure: construct a graph with n vertices, representing the n strings (reads) $s_1, s_2,\ldots, s_n$, and insert edges of length overlap (s_i, s_j) between the vertices s_i and s_j. For this purpose, we define weight as the length of the prefix of s_j that matches a suffix of s_i and overlap (si, s_j) as the length of the longest prefix of s_j that matches a suffix of s_i.

During the layout stage, the overlap graph is analyzed to find the single path in the overlap graph that has the maximum total weight. This path is the ideal solution. However, in fact, this overlap graph formulation is not suitable for finding the single path that represents the shortest superstring [11]. Therefore, current genome assemblers still produce many contigs as their outputs. In the layout stage, the set of contigs is merged to yield supercontigs. For this purpose, we need information on the mate pair lengths to estimate the distance between two contigs that are to be merged.

The final step in the OLC strategy is consensus. The goal of this step is to determine the DNA sequence by applying an alignment of all the reads covering the genome. The consensus sequence is determined by vote using quality values.

Currently, the Eulerian Path approach introduced by Pevzner [5] is very popular. It adopts de Bruijn Graphs to assemble the sequence by organizing (k-1)-mers as vertices and k-mers as edges. This approach can simplify the complexity of the layout problem in the OLC approach into an Eulerian Path problem that can be solved efficiently. The most important issue in the Eulerian approach is choosing the optimal value of k when constructing the de Bruijn Graph. Actually, the effectiveness of the de Bruijn Graph in producing significant contigs depends on the value of k. Smaller k-mers increase the connectivity of the graph, increasing sensitivity. However, smaller k-mers also increase the number of ambiguous repeats in the graph [6].

3 Requirement Analysis

3.1 Data Requirements

Based on the characteristics of the cell, there are two types of DNA sequence data: prokaryotic genomes and eukaryotic genomes. A distinction between prokaryotes and eukaryotes is that eukaryotic cells have a nucleus containing their DNA, whereas prokaryotic cells lack a nucleus. In general, working with prokaryotic genomes entails dealing with smaller data sets than working with eukaryotic genomes.

The length of read also affects DNA sequence assembly. There are three types of reads: long reads (500 bp) [1], short reads (100-200 bp), and very short reads (25-50 bp) [6]. Currently, some sequencers, such as the Illumina, 454, and SOLiD, produce very short reads. Eukaryotic genomes are usually assembled from long reads. On the other hand, prokaryotic genomes are usually assembled from shorter reads. Assembling large eukaryotic genomes from very short reads has yet to be exploited.

Beside DNA-sequence reads, we also require mate-pair reads as input for producing supercontigs. As explained in the previous section, representing the DNA sequence assembly problem as an overlap graph or a de Bruijn Graph could not meet the ultimate goal of producing a single contig that represents the original DNA sequence. We require mate-pair reads to provide information on the distance between two contigs that are to be merged. This information is used to concatenate the set of contigs into supercontigs.

3.2 Functional Requirements

The functional features of DNA sequence assembly actually depend on the approach used. However, there are

several functions that are implemented by both approaches, such as simplification and sequence error removal. The other functions are adjusted to the characteristics of each approach. OLC consists of conducting the overlapping phase, processing the layout, and obtaining a consensus. On the other hand, the Eulerian Path approach usually contains several main functions, such as constructing de Bruijn Graphs and generating contigs by finding Eulerian paths. In addition, some assemblers introduce new functional features that distinguish them from other assemblers to obtain significant results, such as reducing the read redundancy [12], finding all paths in the de Bruijn Graph and localization [7], and clustering reads to obtain accurate contigs when running a distributed de Bruijn graph in a parallel environment [10].

To define a set of functional requirements for our assembler, we consider two aspects. First, we did a functional analysis of previous assemblers, taking into account the success of each assembler in applying its approach. Second, we focused on large eukaryotic genomes as our target. This strategy will guide us in developing an accurate and efficient DNA sequence assembler.

Table 1 DNA sequence assembler

Assembler	Strategy/Approach	Type of reads
Celera	OLC	long read
ARACHNE	OLC	long read
PCAP	OLC	long read
Allpath	Eulerian Path	very short read
Velvet	Eulerian Path	very short read
Edena	OLC	very short read

Table 2 Execution time of Edena and Velvet, in CPU 3.2 GHz and 2 Gb Memory using the *Staphylococcus Aureus* strain MW2

Assembler	Process	Execution time
Edena	Eliminating Redundant Reads and Detecting Overlap	11 m 30 s
	Layout and Consensus	17 s
Velvet (k=23)	Hashing Reads	59 s
	Constructing and Manipulating de Bruijn Graph	1 m 40 s

Table 1 lists DNA sequence assemblers and their approaches. All assemblers for long reads applied the OLC approach. On the other hand, most of the assemblers for very short reads, except Edena, adopted the Eulerian Path approach. From this fact, we gather that the OLC approach is more suitable for assembling long reads, while the Eulerian Path approach is appropriate for assembling very short reads. Although the authors of Edena tried to prove that the OLC approach could also give accurate results for very short reads [12], our experiment found it to be less efficient than Velvet (see Table 2). Edena requires a preprocessing stage to reduce the number of redundant reads, while redundant reads in Velvet are intrinsically handled by the de Bruijn Graph.

Another important issue in DNA assembly is related to sequencing error removal. Some assemblers, such as Velvet [6] and Edena [12], remove errors after constructing the graph. In the case of Velvet, sequencing errors are removed based on topological features, such as erroneous edges and bubbles. The topological approach is effective for small genomes, such as prokaryotic genomes. However, for large eukaryotic genomes, removing errors after constructing the graph is not efficient. Constructing a de Bruijn Graph for a eukaryotic genome will itself consume much memory and time. Therefore, removing sequencing errors before constructing the graph, as a preprocessing step, is more efficient. The Euler assembler [5] applies preprocessing sequencing error correction based on spectral alignment.

For efficiency, some researchers introduced parallelization to accelerate the eukaryotic genome assembly process. Some of the parallelization strategies that have been used are parallelizing hierarchical data [8], distributing and manipulating a bi-directed string graph using multiprocessor computers [9], and representing very short reads in distributed de Bruijn graphs [10]. However, to obtain accurate contigs, Simpson et al. [10] applied clustering to their data before constructing the distributed de Bruijn graph in a parallel environment. This clustering method is based on the k-mer information.

Table 3 Functional requirements

Functional Requirement	Description
Clustering of very short reads	Very short reads will be clustered based on their k-mer frequency to obtain groups of reads that share as many of the same k-mer as possible
Sequencing error correction	Sequencing errors will be corrected using the spectral alignment approach during a preprocessing step.
Construction, simplification, and processing of the de Bruijn Graph to produce contigs	Each group of reads will be distributed to each processor to implement the Eulerian Path approach
Application of the OLC approach for long reads	Contigs produced by the Eulerian approach in previous processes will be assembled using the OLC approach to generate supercontigs.

Furthermore, based on the above functional analysis of previous assemblers, we identified the main functional requirements of our assembler, as presented in Table 3.

4 Design and Simulation

4.1 Hybrid DNA Sequence Assembly

We propose a hybrid assembly process strategy to assemble very short reads of large eukaryotic genomes (Figure 1). The previous research by Reinhardt et al. [13] had adopted a hybrid approach by combining two assemblers, VCAKE and NEWBLER, and two types of reads, Illumina very short reads and 454 long reads. In our case, we would not apply the assembler directly; we want to combine the OLC and Eulerian Path assembler approaches in our framework to obtain significant contigs.

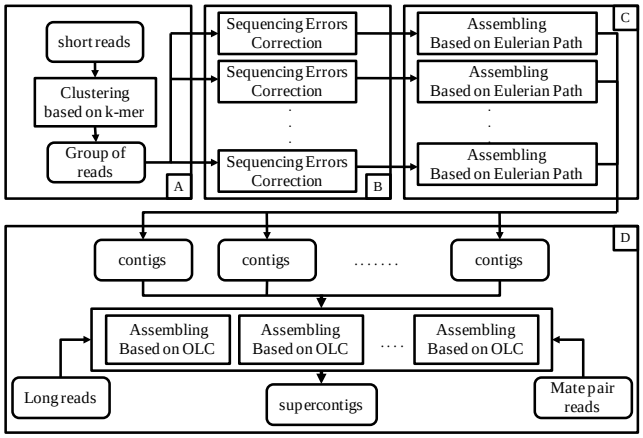

Figure 1 Hybrid DNA sequence assembly

In our proposed assembler, first, Illumina very short reads will be clustered based on k-mers (A). If the same k-mer is found in multiple sequences, those sequences are likely to come from the same region of the genome. Each group of reads will be distributed to each processor, which will simultaneously execute a sequencing error correction process based on spectral alignment (B). This process is aimed at reducing the size of the de Bruijn graph that will be constructed. Then, the Eulerian Path approach will be applied in a parallel environment to assemble very short reads, starting with constructing and simplifying the graph, and then finding the Eulerian path to produce contigs (C). These contigs will be treated as long reads and combined with long reads and mate pair reads to generate supercontigs using the OLC approach in parallel environment (D). This hybrid assembly process is outlined in Figure 1.

At the implementation level, we would like to consider a new programming model, MapReduce. MapReduce has a map function that can be used to extract all shifts of a predefined length k of k-mers in the input sequences. Thus, we could develop clustering based on k-mers more simply than by using another programming model. Furthermore, we would like to apply MPI to perform the error correction process and execute the core assembly process for both the Eulerian and OLC approaches. MPI is a distributed memory programming model; therefore, it is suitable for representing the distributed de Bruijn graph.

4.2 Simulation and Results

In this simulation, we would like to focus on the assembly process as the core of our approach. Our objective is to show the effectiveness of our hybrid assembler. Therefore, in this paper, we were not concerned with clustering, sequencing error correction, or even parallelization. We believe that if we can prove that our hybrid assembler approach is effective, then we can implement it in a parallel environment. We applied three assemblers, Velvet (based on the Eulerian Path), Edena (based on OLC), and Minimus (based on OLC) [15], as proposed by Hernandez [12], to simulate our approach.

Table 4 Results of Hybrid assembly using the *Staphylococcus Aureus* strain MW2 dataset

Assembler	Number of contigs	N50 (kbp)	Maximum (kbp)	Total (Mbp)
Velvet	1152	5.3	2.3	2.78
Edena	1175	5.4	2.3	2.76
Hybrid	890	7.4	3.3	2.77

Table 5 Results of Hybrid assembly using the *Helicobacter pullorum* NCTC 12824 dataset

Assembler	Number of contigs	N50 (kbp)	Maximum (kbp)	Total (Mbp)
Velvet	3981	0.58	4.4	1.77
Edena	4330	0.35	4.1	1.25
Hybrid	2570	0.86	9.1	1.66

Table 6 Results of Hybrid assembly using the *Saccharomyces cerevisiae* chromosome 5 dataset

Assembler	Number of contigs	N50 (kbp)	Maximum (kbp)	Total (Mbp)
Velvet	105	22.6	48.9	0.56
Edena	131	17.9	41.8	0.56
Hybrid	78	27.2	48.9	0.56

First, we assembled very short reads with Velvet and Edena, separately. Next, we assembled contigs produced by Velvet and Edena using the Minimus assembler. In this case, the contigs produced by Edena represent long reads that would be combined with contigs produced by Velvet and then be assembled using Minimus. This simulation combines the Eulerian Path and OLC approaches, which

are represented by the Velvet and Minimus assemblers, respectively. Moreover, it also combines two types of reads: very short reads and long reads, represented by the contigs produced by Edena. Thus, we refer to our approach as a Hybrid assembler. To show the performance of our simulation, we use three datasets: Staphylococcus *Aureus* strain MW2 [12], *Helicobacter pullorum* NCTC 12824 from the NCBI Short Read Archive, and *Saccharomyces cerevisiae* chromosome 5 from simulated Solexa/Illumina-style datasets [14]. All the results of the assembly process showed that assembly using a hybrid strategy could produce longer contigs, especially on N50 and maximum contigs, than Velvet or Edena (Table 4-6).

5 Conclusions and Future Work

In this paper, we have presented data and functional requirements for an accurate and efficient hybrid DNA assembler for large eukaryotic genomes. Our proposed assembler uses a hybrid strategy that combines two main DNA assembly approaches and two types of reads for execution in a high-performance computing environment. Our simulation of the core assembly process has shown that the hybrid strategy can generate long contigs. Implementation of this hybrid approach with a large dataset, such as a eukaryotic genome, may produce similar results. However, the accuracy and efficiency of our assembler for large eukaryotic genomes will also depend on other processes, such as clustering of very short reads and correction of sequencing errors. Therefore, our future work includes implementing those two processes in a parallel environment.

6 References

[1] Myers, E. W., Sutton G. G., Delcher, A. L., Dew, I. M., Fasulo, D. P., Flanigan, M. J., Kravitz, S. A., Mobarry, C. M., Reinert, K.H.J., Remington, K.A. "A Whole-genome assembly of Drosophila". Science 287: 2196-2204. 2000.

[2] Batzoglou, S., Jaffe, D. B., Stanley, K., Butler, J., Gnerre, S., Mauceli, E., Berger, B., Mesirov, J.P., and Lander, E. S. "ARACHNE: A whole genome shotgun assembler". Genome Res. 12: 177 – 189. 2002.

[3] Huang, X., Wang, J., Aluru, S., Yang, S., and Hillier, D. "PCAP: A Whole-Genome Assembly Program". Genome Res. 13:2164-2170. 2003.

[4] Mullikin, J. C. and Ning, Z. "The Phusion Assembler". Genome Res. 13:81-90. 2003.

[5] Pevzner, P.A., Tang, H., and Waterman, M.S. "An Euler path approach to DNA fragment assembly". Proc. Natl. Acad. Sci. 98: 9478-9753. 2001.

[6] Zerbino, D.R. and Birney, E. "Velvet: Algorithms for de novo short read assembly using de Bruijn graphs". Genome Res. 18: 821-829. 2008.

[7] Butler, J., MacCallum, L., Kleber, M., et. al. "ALLPATHS: De novo assembly of whole-genome shotgun microreads". Genome Res. 18:810-820. 2008.

[8] Sundquist, A., Ronaghi, M., Tang, H., Pevzner, P, Batzoglou S. "Whole genome sequencing and assembly with high-throughput, short read technology". PloS One 2:e484. 2007.

[9] Jackson, B.G., Shnable, P.S., and Aluru, S. "Parallel short sequence assembly of transcriptomes". BMC Bioinformatics 10 (Suppl I):S14. 2009.

[10] Simpson, J. T., Wong, K., Jackman, S. D., et. al. "ABySS: A parallel assembler for short read sequence data". Genome Res. 19:1117-1123. 2009.

[11] Pop, M. "Genome assembly reborn: recent computational challenges". Oxford University Press. 2009.

[12] Hernandez, D., et. al. "De novo bacterial genome sequencing: Millions of very short reads assembled on a desktop computer". Genome Research 18:802 – 809. 2008.

[13] Reinhardt, J. A. et. al. "De novo using low-coverage short read sequence data from the rice pathogen Pseudomonas syringae pv. Oryzae". Genome Res. 19:294-305. 2009.

[14] Shi, H. et. al. "Accelerating Error Correction in High-Throughput Short-Read DNA Sequencing Data with CUDA". IEEE International Symposium on Parallel&Distributed Processing. 2009.

[15] Sommer, D. D., et. al. "Minimus: a fast, lightweight genome assembler". BMC Bioinformatics: 8:64. 2007.