

An Exhaustive Genome Assembly Algorithm Using K-Mers to Indirectly Perform N-Squared Comparisons in O(N)

Maulik K. Shah^{1,2}, HoJoon Lee¹, Stephanie A. Rogers¹, Jeffrey W. Touchman^{2,3}

¹Computational Biosciences Program, Arizona State University, Tempe, Arizona

²Translational Genomics Research Institute, Phoenix, Arizona

³School of Life Sciences, Arizona State University, Tempe, Arizona

Email: maulik.k.shah@asu.edu

Abstract

We present an algorithm that indirectly makes N^2 sequence comparisons in $O(N)$ with respect to the size of the genome. This algorithm is very applicable in assembling whole genomes from the thousands of DNA sequence fragments that are generated in shotgun sequencing. First, we assume that fragments that share k -mers should overlap in the final assembly. We then catalog all k -mers that exist in the shotgun library and infer links between fragments that share k -mers. These links are then used to represent edges in a graph. This graph is generated in $O(N)$ yet represents the result of comparing every fragment to every other fragment.

1. Introduction

The complete sequence of an organism's genome is an incredibly valuable tool for biological research. Using today's technology, the most common method of elucidating the sequence is shotgun sequencing followed by in-silico assembly.

An example description of sequencing a large microbial genome is as follows. Approximately five million base pairs are assembled from 70,000 sequence fragments, where each is approximately 800 base pairs in length. The most straightforward approach would be to compare each fragment to all others, identify fragments that overlap, and then rebuild the original sequence. However, the time to directly make all comparisons will grow exponentially with the size of the genome. We have developed an algorithm to indirectly make all comparisons in linear running time. Because our approach effectively makes all of these comparisons and does not abandon the traditional 'overlap-layout-consensus' approach, we feel it will make more accurate assemblies than non-

exhaustive approaches such as those used in ARACHNE [1], PHRAP [2], and EULER [3].

2. Methods

2.1. K-Mer Library

To make all N^2 comparisons, we begin by building a library of k -mers (sequences that consist of nucleotides of length k). Scanning through each fragment in the dataset by using a sliding window of size k , we catalog each of these k -mers and the fragment in which it occurred. Since we assume that fragments that neighbored each other in the complete genomic sequence would share unique k -mers, cataloging the k -mers in this fashion would indirectly identify neighboring fragments. Unlike ARACHNE, we do not attempt to minimize memory requirements, but rather evaluate all k -mers.

2.2. Adjacency Table

The k -mer library shows which fragments share k -mers. However, because we used a sliding window, a pair of neighboring fragments would share as many k -mers as the length of their overlap. However, the only relevant information is that these two fragments overlap by at least k base pairs and therefore belong next to one another in the complete genome. Therefore, we built a new table that contained only one link per pair of overlapping fragments, no matter the number of k -mers shared. This table is particularly significant as it represents all N^2 comparisons and was generated in $O(N)$

2.3. Identifying Contigs

The adjacency table is the most valuable data structure in this algorithm. We viewed the adjacency

table as a set of disjoint, undirected, cyclical graphs; a similar approach to EULER. Identifying and separating these graphs would result in a representation of smaller segments of the genome. Traditional assemblers call these segments contigs and each disjoint graph would represent a single contig. We employed multiple breadth first searches to identify these disjoint graphs. Since this process explores each fragment only once, it also runs in linear time.

3. Results

Two datasets were used in this experiment. The first was a simulated shotgun library derived from the complete genome of the CO92 strain of *Y.Pestis*. We artificially generated a shotgun library containing 41,400 random fragments which represents nine-fold coverage. We also used a real shotgun library of the FV-1 strain of *Y.Pestis* which contained 70,176 fragments. Using 24 as the size of k, the CO92 dataset showed that 39,809 of the reads were linked into a single graph and the FV-1 dataset showed that 41,539 fragments were linked into a single graph. We had initially hypothesized that this approach would yield similar results to ARACHNE and PHRAP. However, it seems to have connected more fragments than expected.

The following figures were generated by running the algorithm on data from the FV-1 *Y.Pestis*. shotgun library. Figure 1 shows how the occurrence of k-mers were distributed.

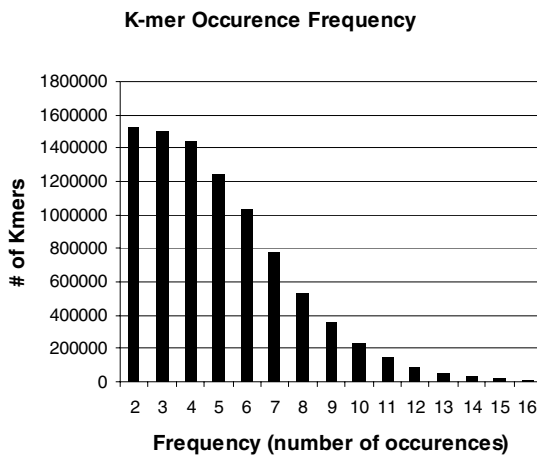


Figure 1 – K-mer distribution

Note that Figure 2 shows that on average, a node in the graph had 9.5 neighbors. This is consistent with

our expectations as the shotgun library had over nine-fold coverage.

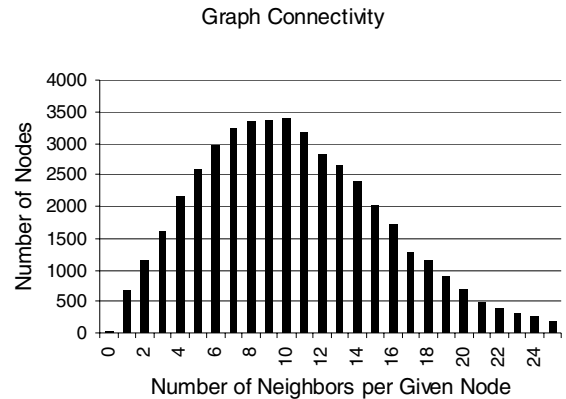


Figure 2 – Connectivity of the nodes in the graph

4. Conclusions

By combining concepts found in both ARACHNE and EULER, we plan to unite these methods of using overlap-layout-consensus with building graphs to assemble a genome with higher accuracy. Although we have yet to make use of mate-pair information, error-correction or repeat-masking to help in finding the solution, initial results are promising.

We are grateful to Steve Mastrian, Debbie Benitez, Erin Clark, Jicheng Hao, Dave Youngkin, Andrzej Czygrinow and Rosemary Renaut for providing guidance and sequencing data.

5. References

- [1] Batzoglu, S., Jaffe, D., 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.* 2002 Jan 1; 12(1): 177–189.
- [2] Green P. (1994) Documentation for Phrap (<http://www.phrap.org/phrap.docs/phrap.html>)
- [3] Pevzner PA, Tang H, Waterman MS. An Eulerian path approach to DNA fragment assembly. *Proc Natl Acad Sci USA.* 2001 Aug 14; 98(17): 9748-9753