

Improving Shotgun Sequencing Through Learning

Mark Goldberg,¹ Darren Lim²

Keywords: DNA assembly, shotgun sequencing, machine learning.

The presence of *repeats*, long identical disjoint segments of a DNA string, is one of the main difficulties in applying the shotgun approach to sequencing DNA strings. An assembler implementing the traditional *overlap-layout-consensus* paradigm ([1, 2]) can be “tricked” into merging fragments with large, similar ends that cover disjoint segments of the DNA. Thus, according to [5], the assemblers Phrap, CAP3, and TIGR make substantial errors while assembling bacterial DNAs. To overcome the problem³ several strategies were developed; see [3, 4, 5]. The experiments with a new assembler EULER, reported in [5], show a significant improvement in the accuracy of assembly on the bacterial DNAs. Still, no purely algorithmic strategy is reported which would perform satisfactorily assembling eukaryotic genomes.

We apply the machine learning methods to improve the accuracy and efficiency of the shotgun sequence assembly algorithms in DNA strings with a significant portion of repeats, such as in eukaryotic genomes. We describe a generic assembly algorithm, *VEDA*, whose parameters are “learned” by a procedure *L-VEDA*. The parameters of the search to be performed by *VEDA* are established by the analysis done by *L-VEDA* of the training DNA string; these parameters are then applied to sequence a new DNA. The first stage of *VEDA* is error correction; this is similar to the design of EULER, although our error correction is based on different ideas. During the next stage, *VEDA* performs preliminary sequencing by creating contigs that cover most of the “unique” parts of the DNA. The assembly is finished by sequencing the repeat areas. The main idea used for correcting and assembling is to establish, through learning, the “reliable” areas of the DNA and utilize those as “anchors” to expand the contigs to the areas containing repeats. Our experiments with prokaryotic and eukaryotic DNA strings up to ten million bp yield close to 97% successful assembling. Since *L-VEDA* supplies domain-specific parameters to *VEDA*, the usefulness of the resulting assembler is expected to depend on whether the new input belongs to the “correct” domain. However, our experiments show that there is, apparently, a great deal of commonality among the computational domains, even when the DNA strings themselves are different. Thus, the assembler trained on *C. Trachomatis* was successfully used for Human Chromosome 22.

Algorithms and Experimental Results. *VEDA* starts by constructing a hash table of all k -mers ($k = 20$) occurring in the input fragments. Each k -mer in the hash table is associated with a list of all input fragments containing the k -mer; the length of this list is called the k -mer’s *frequency*. The rest of *VEDA* is comprised of three basic stages: *Cleaning*, *Staircasing*, and *Expanding*.

Cleaning. This procedure locates and corrects errors that occur in the input fragments. First, *VEDA* extracts the k -mers that, with high probability, are error-free (the reliable k -mers), and then uses them as anchors in order to fix the errors located in other k -mers. Our experiments show that when the average cover ratio is close to ten, *VEDA* corrects most of the errors that are located in the k -mers of frequencies up to four.

Staircasing. During this stage, initial contigs are formed by merging reliable k -mers whose overlaps are of length $k - 1$. The reliability is determined by the frequency of the k -mers. The

¹Computer Science Dept., Rensselaer Polytechnic Institute, Troy, NY 12180. E-mail: goldberg@cs.rpi.edu

²Computer Science Dept., Rensselaer Polytechnic Institute, Troy, NY 12180. E-mail: limd@cs.rpi.edu

³If repeats are longer than the lengths of the fragments, the problem is, in principle, unsolved unless some additional information is used, for example mate-pairs.

specific bounds on the frequency that guarantee reliability are previously established by *L-VEDA*. According to our experiments, the resulting contigs mainly cover the unique areas of the DNA.

Expanding. Once initial contigs have been formed, the fragments that non-trivially overlap them (according to *L-VEDA*, which establishes the meaning of non-triviality during the learning stage) are used to extend those contigs. The contigs are first extended into regions with low coverage. After this phase, contigs are sequenced together by fragments that span repeated areas. *VEDA* returns the contig set as the final result.

The DNA string used by the learning algorithm *L-VEDA* was a one million bp sample of *C. Trachomatis*; the system of fragments was generated at random with the average length of 500 bp, 1% fragment error rate, and the average cover ratio ten. The generated fragments cover 99% of the DNA; the frequencies of the k -mers ($k = 20$) range from 1 to 44. The distribution of the k -mers by their frequencies is given in Figure 1; the probability of a k -mer with a given frequency to be unique in the DNA is given in Figure 2. In particular, the k -mers with frequencies in the interval $[5, 13]$ are, with high probability, unique.

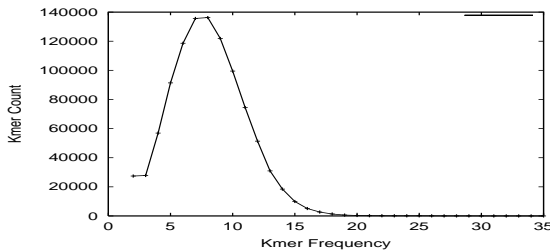


Figure 1

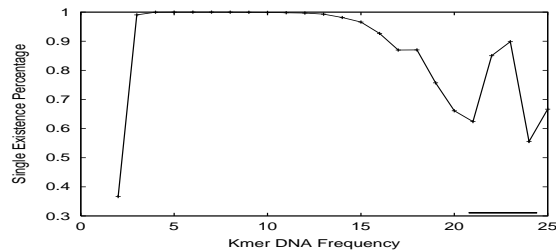


Figure 2

VEDA, with the parameters determined by *L-VEDA*, was tested on prokaryotic and eukaryotic DNAs, in particular, on two genomes used for testing EULER ([5]). In all but two cases cited in [5], the testing was done on randomly selected intervals of length from one to ten million of the corresponding genomes. All genomes used for testing were taken from GenBank. The table below shows the genome, size of the initial target DNA strand, percentage of the target covered by the resulting contigs, and the number of contigs.

Name	Size	Percent Covered	Name	Size	Percent Covered
B. Burgdoferi	1000000	96.6%	T. Maritima	1000000	96.7%
A. Fulgidus	1000000	97.0%	Human 22 (#1)	2000000	96.8%
C. Jejuni	1641481	97.1%	Human 22 (#2)	2000000	97.1%
N. Meningitis	2184496	97.4%	Human 22 (#3)	10000000	97.7%

References

- [1] J. K. Bonfield, K. F. Smith, and R. Staden, *A new DNA sequence assembly program*, *Nucleic Acids Research*, 23; 4992-4999, 1995.
- [2] X. Huang and A. Madan, *CAP3: A DNA sequence assembly program*, *Genome Research*, 9:869-877, 1999.

- [3] R. Idury, and M.S. Waterman, *A new algorithm for DNA sequence assembly*, Journal of Computational Biology 2(2), 291-306, 1995.
- [4] J. Kececioglu and J. Yu, *Separating repeats in DNA sequence assembly*, Proceedings of RECOMB 2001, pp. 176-183.
- [5] P. Pevzner, H. Tang, and M. Waterman, *A New Approach to Fragment Assembly in DNA Sequencing*, Proceedings of RECOMB 2001, 256-265.