De Novo Co-Assembly of Bacterial Genomes from Multiple Single Cells

Narjes S. Movahedi, Elmirasadat Forouzmand, Hamidreza Chitsaz
Department of Computer Science
Wayne State University
Detroit, MI 48202
{narges,elmira,chitsaz}@wayne.edu
http://compbio.cs.wayne.edu

***Abstract*—Recent progress in DNA amplification techniques, particularly multiple displacement amplification (MDA), has made it possible to sequence and assemble bacterial genomes from a single cell. However, the quality of single cell genome assembly has not yet reached the quality of normal multicell genome assembly due to the coverage bias and errors caused by MDA. Using a template of more than one cell for MDA or combining separate MDA products has been shown to improve the result of genome assembly from few single cells, but providing identical single cells, as a necessary step for these approaches, is a challenge. As a solution to this problem, we give an algorithm for *de novo* co-assembly of bacterial genomes from multiple single cells. Our novel method not only detects the outlier cells in a pool, it also identifies and eliminates their genomic sequences from the final assembly. Our proposed co-assembly algorithm is based on colored de Bruijn graph which has been recently proposed for *de novo* structural variation detection. Our results show that *de novo* co-assembly of bacterial genomes from multiple single cells outperforms single cell assembly of each individual one in all standard metrics. Moreover, co-assembly outperforms mixed assembly in which the input datasets are simply concatenated. We implemented our algorithm in a software tool called HyDA which is available from http://compbio.cs.wayne.edu/software/hyda.**

***Keywords*-colored de Bruijn graph, sequence assembly, single cell genomics**

I. Introduction

Today, a variety of studies in medical and environmental sciences benefit from genomic analysis of bacteria. A staggeringly large portion of the genomic content in environmental samples is comprised of sequences that have never been observed before. These uncharacterized sequences in diverse places ranging from oceans to human mucus are guessed to be predominantly of bacterial or viral origin. This demonstrates the extent of the complexity and ubiquity of bacterial genomes in our environment. Several projects including Human Microbiome Project (HMP) and Earth Microbiome Project (EMP) [1] aim at cataloguing microbial communities in various locations.

However, lack of enough DNA material for sequencing of one species at a time is often the main limiting factor in microbiome studies. To sequence the genome of a new bacterial species, we require a sample containing numerous identical cells to extract micrograms of DNA material. The conventional way consists in extracting a few bacterial cells and culturing them separately. If a cell cultures successfully, then the resulting colony provides the required amount of good quality DNA material. Unfortunately, the proportion of microbes that can successfully be cultured in the lab is less than 1% [2]. Many bacterial cells cannot be cultured in the lab since they often require complex symbiotic environments to grow. In the case of such uncultivable species, the only existing method to proceed involves whole genome amplification a billion fold from femtograms to micrograms.

Different amplification methods have been proposed that provide enough DNA material from very few cells or even a single cell: (1) PCR such as primer extension pre-amplification (PEP) and degenerate oligonucleotide primed PCR (DOP) whose products are usually short DNA fragments (less than 1 kbp), and (2) multiple displacement amplification (MDA) which generates long DNA products (up to 100 kbp with an average length of 12 kbp) [3], [4], [5]. MDA works using random primers and a special DNA polymerase called $\Phi 29$ with interesting characteristics. The enzyme $\Phi 29$ is able to open up double stranded DNA and continue its way without external thermal help if it encounters a double stranded region while it is synthesizing its complementary strand [2], [5]. This unique property of $\Phi 29$ makes MDA an isothermal reaction that does not suffer from the side effects of thermocycling such as GC-bias as opposed to PCR. MDA provides a better coverage of the genome in comparison to other methods, however, still some parts of the genome are lost or poorly covered while some others are orders of magnitude more abundant in the final product. Although MDA is not a perfect amplification method, it is currently the method of choice because of its efficacy and lower coverage bias in comparison to other methods [2]. Recently, a number of *de novo* assembly tools such as Velvet-SC [6], SPAdes [7], and IDBA-UD [8] have been developed for MDA datasets.

Fortunately, larger amounts of DNA as the initial template can decrease this kind of amplification bias. This is due to the fact that amplification bias in multiple MDA reactions of the same genome happen randomly in different regions [9]. This suggests that lost or severely underrepresented genomic regions in one MDA reaction may be present in another

978-1-4673-2560-8/12/$31.00 ©2012 IEEE

MDA reaction of the same genome. To benefit from this fact, the product of different MDA processes on several cells with the same genome can be combined and used together. In this way, some MDA products represent those sequences that are lost in the other processes. As a related but different strategy, single cells can be pooled prior to amplification, and MDA reaction can be done on the pool which again helps to generate more uniform coverage of the genome. In fact, it has been shown that starting with 5 to 10 initial cells as opposed to a single cell significantly improves the result.

These pooling strategies are applicable only if we are able to isolate a few single cells with identical genomes. However, this turns into a chicken-and-egg problem: how can we know that a few cells are identical before sequencing their genomes while we require to know that they are identical for a complete sequencing of their genomes? If the cells are guessed to be identical but are not actually identical, then mixing their DNA before or after amplification results in a chimeric assembly which has to be strongly avoided.

We propose an elegant solution to this problem, which is based on colored de Bruijn graph [10]. Iqbal *et al.* recently proposed the colored de Bruijn graph in a different context for *de novo* structural variation detection. We use it here for a different application. Our algorithm accommodates multiple input datasets and assigns a unique color to each dataset. In our colored de Bruijn graph, a subset of colors is associated with each k-mer vertex based on its occurrence in the input datasets. Figure 1 depicts a window of *E. coli* genome that has high coverage in one MDA reaction (Lane 8 of [6]) and low coverage in another (Lane 4 of [6]). Both datasets have $\sim 600\times$ average coverage. Our algorithm rescues this region in Lane 4 through co-assembling it with its high coverage counterpart in Lane 8. Our colored de Bruijn graph encodes enough information to detect non-identical input datasets. Therefore, our solution improves the assembly significantly without the risk of mixing non-identical genomes into a chimeric assembly.

II. Methods

A. Indexing Colored k-mers

A k-mer is stored in a 2-bit compressed form, for k up to a compile time constant $MAXK$. In- and out-edges of a k-mer are $(k+1)$-mers that are represented by their extremal left or right nucleotides. Therefore, in- and out-edges can be stored in a bit array of size 8 (one byte). Our de Bruijn graph of the input reads is stored in a hashed collection of splay trees whose vertices are k-mers with their associated data: colored counts, in- and out-edges, and internal flags. To save space which is the bottleneck for de Bruijn graph algorithms, we implemented splay tree in an array with relative rather than absolute pointers. Each k-mer is stored exactly once, therefore, this data structure defines a map from k-mers to data. In the case of double stranded sequences, which is the default mode, a k-mer and its dual (reverse complement) are treated as a single entry in the map. In that case, the minimum of the k-mer and its dual is stored in the map.

B. Condensation of the De Bruijn Graph

A 1-in 1-out chain of k-mers can be condensed into an equivalent long vertex which we call a *unitig*. A maximal unitig that cannot be extended further due to a branch in the graph is called a *contig* [11]. In the first step (`assemble-unitig`), our algorithm randomly picks 1-in 1-out k-mers in the de Bruijn graph, extends them into unitigs, and outputs those unitigs into a file. The remaining k-mers, which are not 1-in 1-out, are output as single k-mer unitigs. In the second step (`assemble-finish`), 1-in 1-out unitigs are condensed in the same manner into contigs.

Note that our condensation is solely based on the topology of the graph without any attention to the colorings. Ignoring colorings is particularly helpful for single cell MDA datasets; see Figure 1 for a case in which this strategy rescues a contig that is poorly covered in one dataset. Note that condensation of only similarly colored 1-in 1-out vertices would break some contigs into multiple pieces due to lack of coverage in some regions.

C. Memory Footprint Reduction

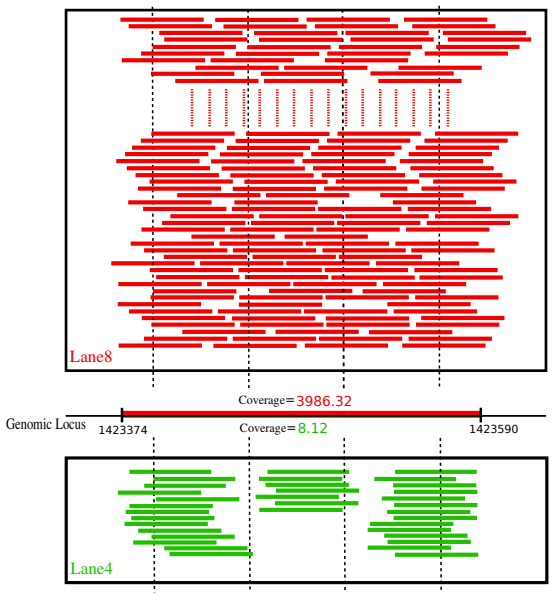
We now describe our novel technique to partition the de Bruijn graph into multiple slices to reduce the peak memory requirement for each slice. The key idea is a fast hash function that assigns a slice to each k-mer. The simplest case is a naïve hash function that does not consider the adjacency of k-mers. On the other hand, we would ideally like a hash function that assigns all k-mers of a contig to the same slice.

Let $w < k$ be an integer; define the *minimizer* of a vertex (k-mer) to be the alphabetically minimum w-mer in the k-mer. The minimizers were previously used to reduce the number of required pairwise alignments in overlap-layout-consensus assembly approaches [12]. We use that idea for a different purpose. Let ω be the minimizer w-mer of k-mer κ. Let $s_{\max}$ be the number of slices into which the graph is partitioned. In that case, κ is assigned to the slice number

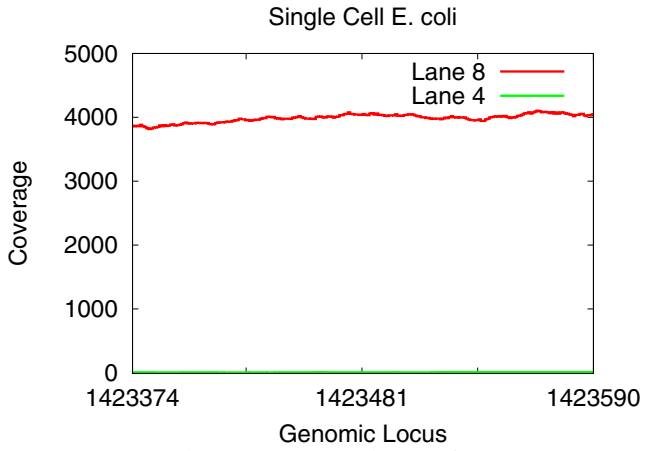
$$s(\kappa) = h(\omega) \bmod s_{\max}, \tag{1}$$

in which h is an arbitrary uniform hash function. Since adjacent k-mers have identical minimizers with high probability, traversing the graph under this scheme imposes less slice-change overhead in comparison to the naïve approach. In `HyDA`, this slicing is performed in `assemble-unitig` and the output unitigs of all slices are simply merged to build the input unitigs to `assemble-finish`.

As long as w is large enough, this slicing scheme provides a roughly balanced partition. Note that a contig may still be broken into multiple subsequences in different slices unless w is very small. We observed that $w = 8$ gives a suitable tradeoff between partition balance and slice-contiguity both for bacterial and mammalian genome assembly; `HyDA` uses $w = 8$ by default.



(a) Aligned reads.



(b) Coverage per base pair.

Figure 1: An *E. coli* contig (genomic locus 1,423,374-1,423,590) in Lanes 4 and 8 of single cell datasets [6]. It has low coverage ($8.12\times$) in one dataset and high coverage ($3986.32\times$) in the other. Our co-assembly algorithm rescues this low coverage contig.

D. Iterative Error Removal

The variable coverage cutoff, starting from 1 and gradually increasing, is used to remove contigs with low coverage. The low coverage contig removal process is iterated with increasing the cutoff in each round. In each iteration, the contigs whose maximum coverage over all colors is less than the cutoff are eliminated, and the remaining graph is recondensed. This causes some contigs to merge into larger ones with recomputed average coverages. This process is similar to `Velvet-SC`'s low coverage contig removal, but instead of considering one average coverage, `HyDA` considers the *maximum* average coverage over all colors [6]. We chose maximum instead of mean since the information of each color is important. For instance, mean would dilute the effect of one distinct genome among hundreds, and would cause the contigs of the distinct genome to be eliminated, whereas maximum preserves those contigs. Our results in Table I demonstrate that maximum performs well in eliminating erroneous contigs while it preserves well-covered outlier contigs, which could arise from distinct genomes, for further downstream analysis.

E. Maximal Contig Sets

The output of iterative error removal is a collection of contigs. Let $C = \{c_1, c_2, \cdots, c_n\}$ be the set of remaining contigs after error removal. Let $A_j(c_i)$ denote the average coverage of contig c_i in color j, for $1 \leq i \leq n$ and $1 \leq j \leq m$. Pick $\epsilon \geq 0$ and let $C_j = \{c_i \in C \mid A_j(c_i) > \epsilon\} \subset C$ be the set of contigs for each color j. The parameter ϵ determines the tradeoff between specificity and sensitivity. We chose $\epsilon = 0$ in this study, but a non-zero ϵ might be needed if there are erroneous or contaminant k-mers in one color which also occur in the true genomic sequence of another color.

Equip $\mathcal{F} = \{C_1, C_2, \cdots, C_m\}$ with the subset partial order. Maximal elements in $\mathcal{F}$ are the assemblies of candidate species. They are candidate and not precisely different species, because it is possible to have two maximal contig sets for the same species due to lack of coverage in some regions caused by MDA. Such cases may be discovered by looking at the graph topology and how those exclusive contigs complete other contig sets. We leave this idea for future work as it requires larger real or simulated datasets.

In this work, we computed the maximal elements in $\mathcal{F}$ as output assemblies. Pritchard has given a sub-quadratic algorithm to compute the maximal elements in a partial order induced by subset relation [13]. It is practically more efficient than the naïve all-pairs comparison if m, the number of colors, is large. For this study, we implemented the naïve algorithm to compute the maximal elements of $\mathcal{F}$. We plan to implement the Pritchard's algorithm in the future.

III. Results

A. Datasets

We used the datasets published by Chitsaz *et al.* [6] in which two *E. coli* single cells and one *S. aureus* single cell were isolated and their genomes were independently amplified with MDA. The resulting amplified DNA of one *E. coli* cell was sequenced in 4 lanes of Illumina GAIIx (Lanes 1-4), and the resulting amplified DNA of the other *E. coli* cell was sequenced in 3 lanes of Illumina GAIIx (Lanes 6-8). Therefore, we have 7 lanes of *E. coli* genomic MDA sequence from two biological replicates, in 100 bp long reads with an average coverage of $\sim 600\times$ per lane.

The resulting amplified DNA of *S. aureus* was sequenced in 2 lanes of Illumina GAIIx with an average coverage of $\sim 1800\times$ per lane. We have two lanes of *S. aureus* each with three times the coverage of one *E. coli* lane.

B. Experiments

We assembled the following datasets with $k = 55$:

- Mixed unicolored assembly of 7 *E. coli* lanes and 2 *S. aureus* lanes: we concatenated the reads in all those lanes and assembled the resulting aggregated dataset with `HyDA`. We used a coverage cutoff of 400.
- Colored assembly of 7 *E. coli* lanes and 2 *S. aureus* lanes: we assigned a unique color to each of the 9 datasets and assembled them with `HyDA`. We used a coverage cutoff of 100. We then extracted *E. coli* contigs from the final assembly based on their coverage in the 7 *E. coli* colors.
- Mixed assembly of 7 *E. coli* lanes: we concatenated the reads in the 7 *E. coli* lanes and assembled the resulting aggregated dataset with `HyDA` using cutoff 400.
- Colored assembly of 7 *E. coli* lanes: we assigned a unique color to each of the 7 datasets and assembled them with `HyDA` using cutoff 100.

C. Evaluation of Assemblies

We used GAGE [14] which is a standard assembly evaluation tool to compare the 4 assemblies described above in Section III-B and the `E+V-SC` single cell assembly of *E. coli* Lane 1 in [6]. The results are presented in Table I. It is clear from Table I that when we assemble a few identical cells the result is more accurate than the assembly of just one cell. The first column of Table I shows the evaluation results of an *E. coli* assembly using just one uncultivated cell. That assembly has 281,060 missing reference bases (about 6.06%), while another *E. coli* assembly using multiple identical cells has just 1,289 missing reference bases (about 0.03%) shown in the second column. Also, when some identical cells are used in assembly, other important factors such as SNPs, indels, and other variations improve significantly. Our results show that the single cell assembly has 100 SNPs and 34 indels < 5 bps, whereas multicell assembly has only 5 SNPs and 6 indels < 5 bps.

Mixed and colored assemblies have roughly similar characteristics. The second column of Table I represents the characteristics of mixed assembly when `HyDA` assembled a pool of 7 *E. coli* lanes coming from 2 biological replicates with cutoff 400. The third column is the colored assembly of the same dataset with cutoff 100. Both assemblies are missing less than 0.05% of reference genome bases and containing about 12% extra bases that are shown to be contaminants [6]. The only noticeable difference between the two assemblies is their N50, NG50, and corrected N50 [14], [15]. The N50 of the mixed assembly is 34,040 and its corrected N50 is 37,682, while the N50 and corrected N50 of the colored assembly are about 6 kbps less, at 27,562 and 31,445 respectively.

The only noticeable weakness of colored assembly is the size of contigs. The reason is that low multiplicity k-mers are iteratively eliminated. The multiplicity of a k-mer in the mixed dataset is the number of its occurrences in all reads from all cells, but the colored multiplicity of a k-mer is an array that keeps the multiplicity of each color separately. The maximum over all colors is computed for determining low coverage contigs in the colored case, whereas the mixed coverage is the sum of the colored coverages. That is why the cutoffs are different for the colored and mixed assemblies. This slight difference causes a mild difference in the N50 and NG50 of mixed and colored assemblies.

The colored assembly using 7 *E. coli* lanes and 2 *S. aureus* lanes coming from 2 uncultivated *E. coli* and one uncultivated *S. aureus* cells contained 3,784 contigs. The coverage in both *S. aureus* colors of 2,130 contigs was zero while most of their *E. coli* colors had positive coverage. On the other hand, the coverage in *E. coli* colors of 1,862 contigs was zero while both *S. aureus* colors had positive coverage. The final assembly of 7 *E. coli* and 2 *S. aureus* lanes with colored de Bruijn graph, when *S. aureus* contigs were eliminated, was similar to the colored assembly of 7 *E. coli* lanes; see Table I. Provided that the input species are sufficiently different and MDA reactions cover a high portion of the genomes, we can easily find identical cells by clustering contigs. Performing the same experiment on more than 2 species is left as future work as it requires more data.

As mentioned above, the advantage of colored de Bruijn graph becomes clear when a combination of non-identical cells is assembled. The last two columns of Table I represent the results of the mixed and the *E. coli* portion of the colored assembly of 7 *E. coli* and 2 *S. aureus* lanes. The mixed assembly covers the reference genome very well and just misses 1,289 bps of 4,639,675 bases (about 0.03%) of *E. coli* genome, but it contains 3,662,149 extra bases (about 44%) corresponding to *S. aureus* genome and contaminants. On the other hand, the *E. coli* portion of the colored assembly misses only 2,114 bps (about 0.05%) while it contains nearly one sixth extra bases (526,119 bps) which are only contaminant [6].

IV. Conclusion

We gave an algorithm and its implementation `HyDA` for *de novo* co-assembly of bacterial genomes from multiple single cells. Our proposed co-assembly algorithm is based on colored de Bruijn graph [10] in which each input dataset is annotated by a unique color. Using `HyDA`, we showed that co-assembly of bacterial genomes from multiple single cells outperforms single cell assembly of each individual one in all metrics. Moreover, co-assembly outperforms mixed assembly in all metrics except contiguity. To mitigate this mild contiguity loss, we suggest a mixed re-assembly of

	E+V-SC	HyDA			
	Single Cell Lane 1	Mixed Assembly of Identical Cells	Colored Assembly of Identical Cells	Mixed Assembly of Non-identical Cells	Colored Assembly of Non-identical Cells
Assembly Size	4,570,583	5,272,627	5,240,693	8,274,855	5,281,487
Missing Ref Bases (%)	281,060 (6.06%)	**1,289 (0.03%)**	2,114 (0.05%)	**1,289 (0.03%)**	2,114 (0.05%)
Extra Bases (%)	279,721 (6.12%)	659,982 (12.59%)	615,808 (11.75%)	3,662,149 (44.26%)	656,771 (12.44%)
SNPs	100	**5**	**5**	**5**	**5**
Indels $<$ 5 bp	34	8	**6**	8	**6**
Indels $\geq$ 5 bp	**4**	**4**	**4**	**4**	**4**
Inversions	**0**	**0**	**0**	**0**	**0**
Relocations	4	**2**	3	**2**	3
N50	32,485	34,040	27,562	29,823	26,903
NG50	32,051	39,340	32,051	54,505	32,051
Corrected N50	30,094	**37,682**	31,445	**37,682**	31,445

Table I: Comparison of GAGE [14] evaluation results for *E. coli* assemblies obtained from a single cell with E+V-SC (Lane 1 of [6]) and multiple identical (Lanes 1-4 and 6-8 of [6]) and non-identical cells (Lanes 1-4 and 6-8 plus two lanes of *S. aureus* in [6]) with HyDA in mixed and colored mode. GAGE's corrected N50 is the N50 of maximal contiguous pieces of contigs that align to *E. coli* K-12 reference genome without $\geq$ 5 bp indels [14]. *E. coli* contigs are extracted from the colored assembly of non-identical cells in a post-processing step based on their coverage in *E. coli* colors. Extra bases are due to contaminants, and that is why the number of extra bases in multicell assemblies is higher than that in single cell one [6]. The best results are shown in bold face. NG50 is the size of the contig the contigs larger than which cover half of the genome size [15]. All contigs are considered in all assemblies, particularly ultrashort single k-mer ones.

those datasets that fall into the same cluster, i.e., identified as the same species, after clustering the contigs based on their colored coverages.

V. Acknowledgement

We gratefully acknowledge our helpful discussion with Glenn P. Tesler and Pavel A. Pevzner on the algorithmic and software design of HyDA.

References

[1] J. G. Caporaso *et al.*, "Ultra-high-throughput microbial community analysis on the Illumina HiSeq and MiSeq platforms," *ISME J*, Mar 2012.

[2] R. S. Lasken, "Single-cell genomic sequencing using Multiple Displacement Amplification," *Curr. Opin. Microbiol.*, vol. 10, pp. 510–516, Oct 2007.

[3] F. B. Dean *et al.*, "Comprehensive human genome amplification using multiple displacement amplification," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 99, pp. 5261–5266, Apr 2002.

[4] ——, "Rapid amplification of plasmid and phage DNA using Phi 29 DNA polymerase and multiply-primed rolling circle amplification," *Genome Res.*, vol. 11, pp. 1095–1099, Jun 2001.

[5] R. S. Lasken and M. Egholm, "Whole genome amplification: abundant supplies of DNA from precious samples or clinical specimens," *Trends in Biotechnology*, vol. 21, pp. 531–535, Dec 2003.

[6] H. Chitsaz *et al.*, "Efficient de novo assembly of single-cell bacterial genomes from short-read data sets," *Nature Biotech*, vol. 29, no. 10, pp. 915–921, Oct 2011.

[7] A. Bankevich *et al.*, "SPAdes: A New Genome Assembly Algorithm and Its Applications to Single-Cell Sequencing," *J. Comput. Biol.*, vol. 19, no. 5, pp. 455–477, May 2012.

[8] Y. Peng *et al.*, "IDBA-UD: a de novo assembler for single-cell and metagenomic sequencing data with highly uneven depth," *Bioinformatics*, vol. 28, no. 11, pp. 1420–1428, Jun 2012.

[9] A. Raghunathan *et al.*, "Genomic DNA amplification from a single bacterium," *Appl. Environ. Microbiol.*, vol. 71, pp. 3342–3347, Jun 2005.

[10] Z. Iqbal *et al.*, "De novo assembly and genotyping of variants using colored de bruijn graphs," *Nat Genetics*, vol. 44, pp. 226 – 232, 2012.

[11] D. R. Zerbino and E. Birney, "Velvet: algorithms for de novo short read assembly using de Bruijn graphs," *Genome Res.*, vol. 18, pp. 821–829, May 2008.

[12] M. Roberts *et al.*, "Reducing storage requirements for biological sequence comparison," *Bioinformatics*, vol. 20, pp. 3363–3369, Dec 2004.

[13] P. Pritchard, "On computing the subset graph of a collection of sets," *J. Algorithms*, vol. 33, no. 2, pp. 187–203, 1999.

[14] S. L. Salzberg *et al.*, "GAGE: A critical evaluation of genome assemblies and assembly algorithms," *Genome Res.*, Dec 2011.

[15] D. Earl *et al.*, "Assemblathon 1: A competitive assessment of de novo short read assembly methods," *Genome Res.*, vol. 21, pp. 2224–2241, Dec 2011.