MANUSCRIPT

Vol. 00 no. 00 2005
Pages 1–3

A comprehensive overview of genome assembly: Past, Present and Future

Fei Xia [1], Jiaqi Ma [1]

[1]Department of Automation, Tsinghua University

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Associate Editor: XXXXXXX

ABSTRACT

This paper reviews the history and recent development of genome assembly, giving representative examples such as de Bruijn Graph assembly and String Graph assembly. It also introduces several new technologies such as barcoding and contact-map and their potential for better genome assembly.

Contact: xf1280@gmail.com

1 INTRODUCTION

1.1 DNA Sequencing Technology

Early efforts at sequencing genes were painstaking, time consuming, and labor intensive, such as when Maxam and Gilbert (1979) reported the sequence of 24 base pairs using a method known as wandering-spot analysis. Thankfully, this situation began to change during the mid-1970s, when researcher Sanger *et al.* (1977) developed several faster, more efficient techniques to sequence DNA. Indeed, Sanger's work in this area was so groundbreaking that it led to his receipt of the Nobel Prize in Chemistry in 1980.

In 2005, with the Genome Analyzer, a single sequencing run could produce roughly one gigabase of data. By 2014, the rate climbed to a 1.8 terabases of data in a single sequencing runan astounding 1000 increase. It is remarkable to reflect on the fact that the first human genome, famously copublished in Science and Nature in 2001, required 15 years to sequence and cost nearly 3 billion dollars. In contrast, the HiSeq XTM Ten, released in 2014, can sequence over 45 human genomes in a single day for approximately $1000 each.

Beyond the massive increase in data output, the introduction of NGS technology has transformed the way scientists think about genetic information. The $1000 dollar genome enables population-scale sequencing and establishes the foundation for personalized genomic medicine as part of standard medical care. Researchers can now analyze thousands to tens of thousands of samples in a single year.

1.2 DNA Sequence Assembly

Sequence assembly is one of the overarching challenges in bioinformatics. To understand the assembly problem it helps to understand some basics of DNA sequencing. Consider a bacterium having a genome comprised of a single 5 megabase (5 million base pairs) chromosome. Ideally, sequencing machines would start at the beginning of the chromosome and read each of the 5 million base pairs until arriving at the end. Unfortunately, the current technology is limited to reading sequences between 30 and 10,000+ bases. The assembly problem is to take these short segments of DNA called reads and overlap them in such a way to recreate the original 5Mb chromosome.

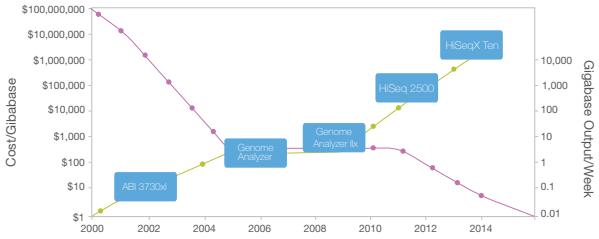

Fig. 1. Cost and Output of DNA sequencing over the from 2000 to 2014

There are two main classes of assembly algorithms used in de novo assembly: overlap-layout-consensus (OLC) from Myers (2005) and de bruijn graph (DBG) from Zerbino and Birney (2008) . Similar to the above example, OLC first finds reads with overlapping ends, builds a layout graph based on these overlaps, and lastly generates a consensus sequence as the graph is traversed. OLC was the first assembly method developed and works well with long-read, low-coverage sequencing technologies like Sanger (and possibly PacBio).

DBG based assemblers convert the set of reads into a set of k-mers (i.e. short DNA sequences of length k). These k-mers are then used to build a de bruijn graph from which the genomic sequence is inferred. DBG assemblers work well with high-coverage sequencing methods like Illumina and Ion Torrent.

2 PROBLEM FORMULATION

Throughout the history of the development of genome assembly, there are many formulations of genome assembly from shotgun sequencing data, some of them are more mathematical, while some of them can better represent the practical situations.

Shortest common superstring formulation developed by Kececioglu and Myers (1995) seeks to find the shortest common supersequence for a set of reads. Various maximum-likelihood formulations of the assembly problem proposed by Myers (1995), and Medvedev and Brudno (2009) seek to find the sequence that generate the reads

with the maximum likelihood. The most widely used formulations, however, are graph based formulations, including de Bruijn Graph by Pevzner *et al.* (2001), and String Graph by Myers (2005).

All the formulations share one common goal: perfect assembly, which means recover the genome perfectly, without any error, from the reads. This does not often happen in real life but is still of theoretical values when assessing different algorithms.

3 PAST

In this section, I will mainly describe the two mainstreams of genome assembly, DBG and OLC. They are very classic algorithms and is still widely used today.

3.1 de Bruijn Graph assembly

De Bruijn Graph assembly uses a notion called kmer. kmer stands for length-k subsequent string in the reads. The de Bruijn graph assembly pipeline is the following:

1. Get all k-mers from all the reads, those are nodes on the graph.
2. Get all $(k+1)$-mers from all the reads, create edge from $[0 : k-1]$ to $[1 : k]$.
3. Merge unambiguous paths, clip dead ends, resolve self loops, etc.
4. Output contigs.

Above is the basic idea of de Bruijn Graph assembly, first covered by Pevzner *et al.* (2001). Later, there are many variants of such algorithm, mainly on finding different heuristics to deal with different artifacts, resolve repeats, etc. Zerbino and Birney (2008) developed Velvet, which is still a widely used short read assembler. Peng *et al.* (2010) developed IDBA, using a iterative process of construction de Bruijn Graph. Peng *et al.* (2012) developed IDBA-UD based on IDBA, and is good at dealing with uneven coverage. Bankevich *et al.* (2012) developed spades, it used paired-end information to resolve repeats, which is a highlight.

3.2 String Graph assembly

String graph assembly is first proposed by Myers (2005). The basic process is the following:

1. Overlap: create pairwise overlap of all reads
2. Layout: filter the reads, set a minimum overlap θ, preserve all overlaps exceeding θ, create an edge between these two reads.
3. Transitive Reduction: reduce transitive edges
4. Merge nodes, clip dead ends, etc.
5. Consensus, produce contigs

The overlap step in the String Graph assembly pipeline is the most time consuming, to make this feasible, several improvements are made. Simpson and Durbin (2010) proposed FM-index based method for overlapping. Myers (2014) developed Daligner for overlapping.

An available software using string graph assembly is SGA.

4 PRESENT

4.1 Development from theoretical aspects

Over the years, advances for theory aspects of assembly problem emerge. On core questions is this field is to determine the fundamental limits for genome assembly. The question is what is the coverage required for perfect assembly. However, here we have to make some assumptions: whether the genome is random (for pure theoretical results) or fixed; whether the reads have errors.

- Error-free, Random: Fundamental limits for this problem is solved in Motahari *et al.* (2013).
- Error-free, Determined: Solved in Bresler *et al.* (2012).
- Error, Random: Solved in arXiv:1304.2798.
- Error, Determined: Solved in arXiv:1501.06194.

5 FUTURE

With the emerging technologies at a increasingly rate, we see hope for better genome assembly. Here we show three relatively new technologies and how they became game changers for genome assembly.

5.1 SMRT sequencing

As is shown by Bresler *et al.* (2013), the bottleneck of genome assembly is the interleaved repeats and triple repeats, if the read length does not exceed l_{crit}, perfect assembly is never possible. The widely used Illumina sequencing produces short reads, much shorter than l_{crit}, thus perfect assembly from Illumina reads is not quite possible. However, with SMRT reads, which produces long reads, perfect assembly can be addressed. Additionally, Illumina sequencing has severe GC bias, the coverage is uneven, thus creating gaps in the final assembly. SMRT sequencing also could help on this regard. Researchers have already achieved some good assemblies from SMRT data. Chin *et al.* (2013) use a non-hybrid hierarchical approach using Pacbio long reads to generate finished assembly for several bacterial genomes. Loman *et al.* (2015) use nanopore data only to generate finished assembly for several bacterial genomes. Heng Li propose an error-correction free pipeline for SMRT reads, achieving a very short running time.

5.2 Barcoding/synthetic long reads

GemCode barcoding

The 10X technology consists of some novel approaches to biological sample preparation, molecular barcoding of the DNA stretches to keep them in tidy little buckets, and software to assemble the DNA in each of those buckets into linked-reads. If scientists can consistently get such long, linked reads, they should be able to speed up the sequencing process considerably because they will know more about which parental chromosome the DNA within the long-read stretch belongs to.

Illumina Truseq

Truseq technology is another technology to use barcodes to maintain long range information while using short read sequencers, it is shown to be useful in work by McCoy *et al.* (2014).

5.3 Contact-map technologies

Contact-map methods, including Hi-C and 3C, are ways to resolve chromatin structure. Naturally such kind of information could be used for assembly. It is used in scaffolding stage of the assembly, determining orders of contigs. Burton *et al.* (2013) achieve chromosome scale scaffolding for human genome. It is a way to upgrade existing genomes with Hi-C data.

6 CONCLUSIONS

The paper reviews the history and recent development of genome assembly, and points out some key advances in this field. It also gives a discussion about the future of genome assembly.

REFERENCES

Bankevich,A., Nurk,S., Antipov,D., Gurevich,A.A., Dvorkin,M., Kulikov,A.S., Lesin,V.M., Nikolenko,S.I., Pham,S., Prjibelski,A.D. *et al.* (2012) Spades: a new genome assembly algorithm and its applications to single-cell sequencing. *Journal of Computational Biology,* **19** (5), 455–477.

Bresler,G., Bresler,M. and Tse,D. (2013) Optimal assembly for high throughput shotgun sequencing. *BMC bioinformatics,* **14** (Suppl 5), S18.

Bresler,M., Sheehan,S., Chan,A.H. and Song,Y.S. (2012) Telescoper: de novo assembly of highly repetitive regions. *Bioinformatics,* **28** (18), i311–i317.

Burton,J.N., Adey,A., Patwardhan,R.P., Qiu,R., Kitzman,J.O. and Shendure,J. (2013) Chromosome-scale scaffolding of de novo genome assemblies based on chromatin interactions. *Nature biotechnology,* **31** (12), 1119–1125.

Chin,C.S., Alexander,D.H., Marks,P., Klammer,A.A., Drake,J., Heiner,C., Clum,A., Copeland,A., Huddleston,J., Eichler,E.E. *et al.* (2013) Nonhybrid, finished microbial genome assemblies from long-read smrt sequencing data. *Nature methods,* **10** (6), 563–569.

Kececioglu,J.D. and Myers,E.W. (1995) Combinatorial algorithms for dna sequence assembly. *Algorithmica,* **13** (1-2), 7–51.

Loman,N.J., Quick,J. and Simpson,J.T. (2015) A complete bacterial genome assembled de novo using only nanopore sequencing data. *Nature Methods,* **12** (8), 733–735.

Maxam,A.M. and Gilbert,W. (1979) Sequencing end-labeled dna with base-specific chemical cleavages. *Methods in enzymology,* **65** (1), 499–560.

McCoy,R.C., Taylor,R.W., Blauwkamp,T.A., Kelley,J.L., Kertesz,M., Pushkarev,D., Petrov,D.A. and Fiston-Lavier,A.S. (2014) Illumina TruSeq synthetic long-reads empower de novo assembly and resolve complex, highly-repetitive transposable elements. *PLoS ONE,* **9** (9), e106689.

Medvedev,P. and Brudno,M. (2009) Maximum likelihood genome assembly. *Journal of computational Biology,* **16** (8), 1101–1116.

Motahari,A.S., Bresler,G. and Tse,D.N. (2013) Information theory of dna shotgun sequencing. *Information Theory, IEEE Transactions on,* **59** (10), 6273–6289.

Myers,E.W. (1995) Toward simplifying and accurately formulating fragment assembly. *Journal of Computational Biology,* **2** (2), 275–290.

Myers,E.W. (2005) The fragment assembly string graph. *Bioinformatics,* **21** (suppl 2), ii79–ii85.

Myers,G. (2014) Efficient local alignment discovery amongst noisy long reads. In *Algorithms in Bioinformatics*. Springer pp. 52–67.

Peng,Y., Leung,H.C., Yiu,S.M. and Chin,F.Y. (2010) Idba–a practical iterative de bruijn graph de novo assembler. In *Research in Computational Molecular Biology* pp. 426–440 Springer.

Peng,Y., Leung,H.C., Yiu,S.M. and Chin,F.Y. (2012) Idba-ud: a de novo assembler for single-cell and metagenomic sequencing data with highly uneven depth. *Bioinformatics,* **28** (11), 1420–1428.

Pevzner,P.A., Tang,H. and Waterman,M.S. (2001) An eulerian path approach to dna fragment assembly. *Proceedings of the National Academy of Sciences,* **98** (17), 9748–9753.

Sanger,F., Nicklen,S. and Coulson,A.R. (1977) Dna sequencing with chain-terminating inhibitors. *Proceedings of the National Academy of Sciences,* **74** (12), 5463–5467.

Simpson,J.T. and Durbin,R. (2010) Efficient construction of an assembly string graph using the fm-index. *Bioinformatics,* **26** (12), i367–i373.

Zerbino,D.R. and Birney,E. (2008) Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome research,* **18** (5), 821–829.