第二代高通量测序组装综述

孙钊

January 9, 2013

Abstract

第二代高通量测序产生了大规模的短序列。大规模输入，指数级的序列重叠，高概率的序列重复使得早期基于“overlap-layout-consensus”的方法不再适用。新的基于de bruijn图的方法在短序列组装中取得了成功。本文以一个纯计算机专业的学生的角度阐述了第二代高通量测序方法，序列组装的挑战以及如何进行序列组装的问题。在文末描述了我自己序列组装领域做出的一些努力。

1 开始之前

首先我想解释一下我写这篇文章的动机。这篇文章的主题是关于基因工程的计算机辅助过程–序列组装。序列组装实际上是生物信息学的范畴。作为一个纯计算机背景的学生，我在2010年的夏天开始在这个领域进行研究。在读了大量的相关领域论文，与国内外的生物科研人员充分交流，尝试了各种各样的生物信息学中的工具之后，我没有取得任何创新性的成果，但是获得了非常宝贵的领域经验，并希望与那些对该领域一无所知又徘徊在领域门口的计算机的学生分享这些经验。本文我讲逐一回答“什么是基因测序”,“什么是序列组装以及它的挑战”, “如何组装序列”, “如何评价组装软件”这些问题，以及在最后列举我个人在这个领域做出的一些努力。本文中，我不会涉及太多算法的细节，我主要的目的是给读者一些关于基因序列组转中“做什么”和“如何做”的直觉印象，并且，我会提供大量的相关文章，材料和链接来帮助你构建一个完整的领域框架。

2 什么是基因测序

在基因工程和生物化学领域，测序意味着确定无支链生物聚合物的一级结构。

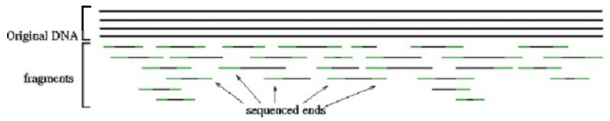

Figure 1:

DNA molecules are double-stranded helices, consisting of two long complement strands. According to base paring principle–‘A’ complements with ‘T’ and ‘G’ complements with ‘C’, the sequence of a strand can be inferred from its opposite one. Particular sequences denoted by 3’ and 5’ in the DNA strand is used to specify the orientation of the two strand, and for simplicity, if one strand is specified as 3’ to 5’, then the opposite one is 5’ to 3’. Example 1 shows a double strain DNA fragment with pair-end reads(red string). Note that the pair-end reads are positioned on opposite strand and will be sequenced all from 3’ to 5’. So the two pair-end reads string should be ‘AGCTAA’ and ‘GCCAA’.

3’→5’

AGCTAATGCTATCTTGGC
TCGATTACGATAGAACCG
5'→3'
Example 1

Sequencing methods and platform is developing as time goes. The typical genome analyser of first generation is Sanger[**?**] producing read lengths of approximately 800bp (typically 500-600bp with non-enriched DNA).Recently, new sequencing methods have emerged [**?**]. Commercially available technologies include 454 Sequencing [**?**], Illumina genome analyser [**?**] and SOLID sequencing(www.appliedbiosystems.com). Compared to traditional Sanger methods, these technologies function with significantly lower production costs and higher throughput. However, the reads produced by these next-generation sequencing technologies are much shorter than traditional Sanger reads, currently around 400-500 base pairs (bp) for 454, 50bp for Illumina and 35bp for SOLiD. Because of their length, they must be produced in large quantities and at greater coverage depths than earlier sequencing projects.
Reads are saved as a file by sequencing chip and the most popular format of read file is FASTA and FASTQ. A sequence in FASTA format begins with a single-line description, followed by lines of sequence data. The description line (def-line) is distinguished from the sequence data by a greater-than (">") symbol at the beginning. It is recommended that all lines of text be shorter than 80 characters in length. An example sequence in FASTA format is shown in Fig. 2.A FASTQ file normally uses four lines per sequence. Line 1 begins with a '@' character and is followed by a sequence identifier and an optional description (like a FASTA title line). Line 2 is the raw sequence letters. Line 3 begins with a '+' character and is optionally followed by the same sequence identifier (and any description) again. Line 4 encodes the quality values for the sequence in Line 2, and must contain the same number of symbols as letters in the sequence. A minimal FASTQ file might look like this Fig. 3. For more details about the quality line, please refer introduction on FASTQ Format.
The last thing to mention is RNA sequencing. As RNA is generated by transcription from DNA, the information is already present in the cell's DNA. The usual method to sequence RNA is first to reverse transcribe the sample to generate cDNA fragments. Then the cDNA are sequenced using DNA sequencing methods.
In a word, the next generation sequencing platform generates a FASTA/FASTQ format file consisting string of description, read and quality(only exists in FASTQ file) for DNA/RNA samples.

3 What is sequence assembly

What the sequencing stage produces is a collection of DNA fragments's nucleotide sequence, whereas, what we need is the nucleotide sequence of the original one. Sequence assembly tries to align and merge the DNA fragments in order to reconstruct the original sequence. The intuition is to take advantage of

```
>AB000263 |acc=AB000263|descr=Homo sapiens mRNA for prepro cortistatin like peptide, complete
cds.||len=368
ACAAGATGCCATTGTCCCCCGGCCTCCTGCTGCTGCTGCTCTCCGGGGCCACGGCCACCGCTGCCCTGCC
CCTGGAGGGTGGCCCCACCGGCCGAGACAGCGAGCATATGCAGGAAGCGGCAGGAATAAGGAAAAGCAGC
CTCCTGACTTTCCTCGCTTGGTGGTTTGAGTGGACCTCCCAGGCCAGTGCCGGGCCCCTCATAGGAGAGG
AAGCTCGGGAGGTGGCCAGGCGGCAGGAAGGCGCACCCCCCCAGCAATCCGCGCGCCGGGACAGAATGCC
CTGCAGGAACTTCTTCTGGAAGACCTTCTCCTCCTGCAAATAAAACCTCACCCATGAATGCTCACGCAAG
TTTAATTACAGACCTGAA
```

Figure 2:

```
@SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
+
!''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65
```

Figure 3:

overlap information between reads to tie them up and piece into a longer DNA sequence. Challenges listed blow makes sequence assembly a very complicated task.

- One cell contains multiple chromosomes as well as DNA chains which are sequenced together. DNA fragments from different DNA chain may overlap.
- DNA is a double-stranded structure, one strand may overlap with its opposite one.
- Sheared fragments along the genome cannot be modeled as a perfect Poisson process so that there may be some region not covered by reads.
- DNA sequence may contains repeat regions, and these regions may be incorrectly merged by overlap.
- Sequencing errors causes nucleotide deleting, replacing or inserting inside reads.
- Second generation sequencing platform produces much shorter reads so that the overlap possibility is exponentially increased.

Among those above challenges, the repeat problem is the most challenge thing for short read assembly. A simple example is shown in Fig. 4A, where the assembler incorrectly collapses the two copies of repeat A leading to the creation of two contigs instead of one Fig. 4B.

The basic strategies for assembling sequences are categorized by two groups. One is reference based assembly, and the other one is de novo assembly. Reference based methods using a mapping tool to align reads against an existing

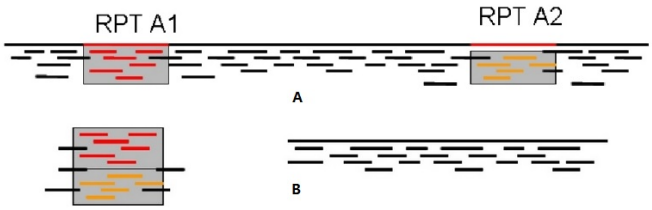

Figure 4:

reference sequence and build a sequence that is similar but not necessarily identical to the reference. The de novo assembly strategy does not use a reference sequence and is much useful for species whose reference sequence are unknown. This article only covers the de novo assembly and will illustrate the details in the subsequent sections.
Ideally, an assembly program should produce one contig for every chromosome of the genome being sequenced. However, because of challenge3 and challenge4, the output of the assembler is interrupted into a set of contigs. In summary, Inputting a FASTA/FASTQ read file, the assembler outputs a set of contigs.

4 How to assembly sequence

Overlap-consensus-layout [**?**] is the traditional approach for first generation sequence. It denotes each read as a separate node, where two reads presenting a clean overlap are connected by a bidirected edge. Although overlap-consensus-layout approach is both intuitive and robust, especially in the case of long reads, it is very costly for pair-wise overlap computing when assembling high quantity short reads produced by second generation sequencing platform like SOLID and Illumina. Only one microread assembler, EDENA [**?**] was developed using this approach. Later, Idury and Waterman [**?**] introduced the use of a sequence graph to represent an assembly, and this idea has been developed into the most popular framework for short read assembly–De Bruijn graph. Sequence De Bruijn graph is constructed from a set of kmers(k is a parameter) as nodes, and two kmers are connected if one kmer's last k-1 nucleotides is identical to the first k-1 nucleotides. kmers are produced by moving a fixed length(k) sliding window on the original read. Examples of kmers and de bruijn graph are show in Fig. 5 and Fig. 6. The sequence de bruijn graph has following properties:

- Read with length of n generates n-k+1 kmers.
- Read is mapping to a unique path in the de bruijn graph.
- One node has at most 4 successors and 4 predecessors by adding 'A', 'G', 'C', 'T' at the first or last of the k-1 nucleotides.

According the property2 and property3, the node scale of the de bruijn graph is bounded by O(n-k+1) and the edge scale of it is bounded by O(8(n-k+1)). By

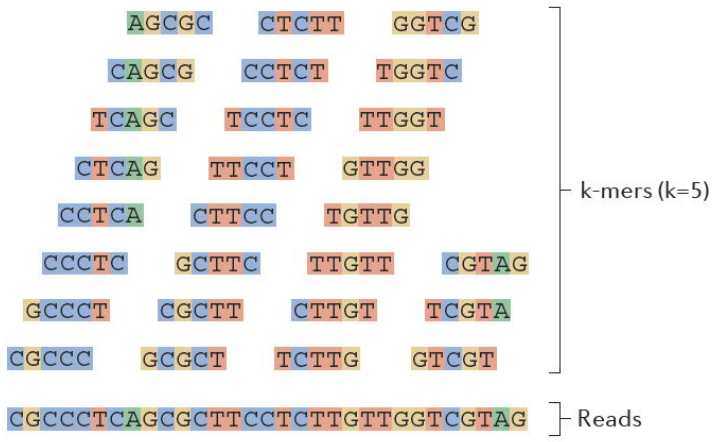

Figure 5:

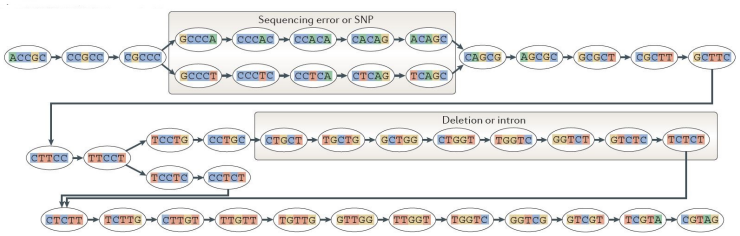

Figure 6:

the way, the length of human is about 3G base pairs whose de bruijn graph can be stored in a single server machine or cluster configured with 128GB. genome Property3 release the heavy cost of searching for all pair-wise overlap, thus making de bruijn graph approach feasible. Softwares include Velvet [**?**], ABYSS [**?**], ALLPATHS [**?**], SOAPDenovo [**?**] follows the de bruijn graph method. ABYSS and ALLPATHS claimed that they successfully assembled human's genome. A brief comparison of these softwares is show in Table 1. Software links can be found in Table 2.

I've read code and run programs of Velvet and ABYSS and provide some details about them. Velvet consists of two modules named velveth and velvetg where 'h' means hash and 'g' means graph. Velvet designed a novel graph construction algorithm and claim that it is efficient and memory saved approach on a single machine. velveth sequentially reads the FASTA/FASTQ read file and for each k-mer observed in the set of reads, a hash table records the ID of the first read encountered containing that k-mer and the position of its occurrence within that

read. Each k-mer is recorded simultaneously to its reverse complement sequence. To ensure that each k-mer cannot be its own reverse complement, k must be odd. This first scan allows each read to be re-written as a set of new k-mers combined to overlaps with previously hashed reads. This new representation of the reads sequence is called a roadmap which is written into a file named ROADMAP. velvetg reads the ROADMAP file and creats a second database with the opposite information. It records, for each read, which of its original k-mers are overlapped by subsequent reads. The ordered set of original k-mers of that read is cut each time an overlap with another read begins or ends. For each uninterrupted sequence of original k-mers, a node is created. Finally, reads are traced through the graph using the roadmaps. Knowing the correspondence between original k-mers and the newly created nodes, it is possible to proceed from one node to the next, creating a new directed arc or incrementing the multiplicity of an existing one as appropriate at each step. The final output of velvet is produced by velvetg and saved as config.fa. ABYSS constructs de bruijn graph directly following the definition. The MPI version uses a hash function to determine the host machine for a node. The structure of ABYSS program is organized by a make file driver named "abyss-pe". abyss-pe automatically finds the dependency of abyss modules and perform a minimum number of running modules. The final output is saved as *-contigs.fa, *-scanffolds.fa, *-unitigs.fa and a statistic file named *-stats.

After building the de bruijn graph, a graph correction process is applied. The

Table 1:

Software	Parallelism	Large Scale Assembly	Memory Cost
Velvet	OpenMP	No	High
ABYSS	OpenMP+MPI	Yes	Low
ALLPATH	?	Yes	?
SOAPDenovo	?	Yes	Low

Table 2:

Software	Software Available
Velvet	http://www.ebi.ac.uk/ zerbino/velvet/
ABYSS	http://www.bcgsc.ca/platform/bioinfo/software/abyss/
ALLPATH	http://www.broadinstitute.org/science/programs/genome-biology/crd
SOAPDenovo	http://soap.genomics.org.cn/soapdenovo.html

idea of the graph correction process is to simplify the original graph into a long path by removing branches. There are two types of branches–tips and bubbles. A tip is a chain of nodes which is disconnected on one end and a bubble is a circle with two common end points. Whenever a node A has only one outgoing

arc that points to another node B that has only one incoming arc, the two nodes can be merged into a long node. Tips and bubbles are shown in Fig .7. Note that repeat region also creates circles in the graph, and we can not determine how many times should the repeat sequence occur. In the last step, if pair-end reads are provided, we can estimate the distance of seed nodes and find a feasible path between them.
The last thing need to mention is that RNA assembly also named transcript assembly. RNA assembly is more complex than DNA assembly, observing the reasons:

- some transcripts have low coverage, whereas others are highly expressed.
- reads with sequencing errors derived from a highly expressed transcript may be more abundant than correct reads from a transcript that is not highly expressed.
- transcripts encoded by adjacent loci can overlap and thus can be erroneously fused to form a chimeric transcript.

Directly using DNA assemblers to assemble RNA sequences only outputs rather bad results. Velvet and ABYSS have extend themselves into Velvet_Oases [**?**] and Trans-ABYSS [**?**] separately to address the additional challenge presented by RNA assembly. Recently, Trinity [**?**] was specially designed for transcript assembly. Zhao et. al [**?**] made a early comparison of transcript assembly and give a full view of performance, result, resource usage evaluation. It shows Trinity has a relative better assembling ability than Velvet_Oases and Trans-ABYSS but much slower and memory cost while assembling large scale sequences. A group of HPC developers [**?**] made efforts to accelerating trinity and reducing its memory usage which make trinity a first choice when assembling transcript sequences.

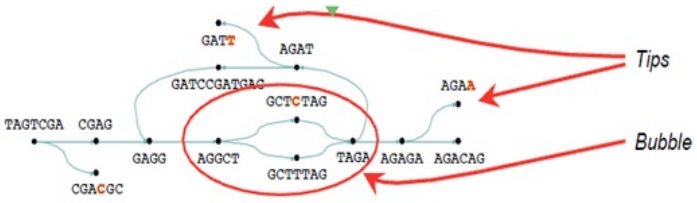

Figure 7:

5 How to Evaluate Assembler

To evaluate present assemblers or assemblers designed by ourselves, the first thing to do is to prepare input data. Usually, two kinds of data are used to

evaluate assemblers, synthetic data and real data. dwgsim [1] is whole genome simulator based on a reference genome. Reference genome can be found on ftp://ftp.ncbi.nlm.nih.gov/refseq/, ftp://ftp.ncbi.nlm.nih.gov/genomes/ or NCBI's nucleotide database (http://www.ncbi.nlm.nih.gov/nucleotide). Details about refseq is explained on http://www.ncbi.nlm.nih.gov/books/NBK50679/. NCBI's SRA database [2] provides a large collection of read read sequences. Users search this database using an access code begin with SRP(studies), SRS(samples), SRX(experients) or SRR(runs). Multiple SRRs of same samples can be used together as input. By the way, files downloaded from SRA database are in sra format and users should dump FASTA/FASTQ format using fastq-dump from NCBI SRA Toolkit. If pair-end data are expected to be dumped, pass '–split-3' or '–split-files' to fastq-dump. Metrics for evaluate assembler can be N50 length, running time, memory cost, 95% map ratio and so on. Given a set of contigs(output of assemblers) of varying lengths, the N50 length is defined as the length N for which half of all bases in the sequences are in a sequence of length L $<$ N. Obviously, the larger the N50 length, the better the assembled contigs. On linux like systems, users can use '/usr/bin/time -v' command to record the running time and peak memory usage of a user process. The 95% mapping ratio is defined as the number of contigs with at least 95% identity mapping to the reference dividing by the total number of contigs. BLAST [**?**] and BLAT [**?**] are two popular mapping tools for aligning contigs to references. I often choose BLAT for reasons of efficiency. BLAT outputs a very big table(PSL format) to store the mapping info for each query sequence. Users who are strange to PSL format and the meaning of each column should refer to PSL format. Each query may have multiple candidate alignment positions, and BLAT program doesn't provide a percent identity value column or score column like the web BLAT [3]. Users can replicate the two columns referring to http://www.genoscope.cns.fr/blat-server/cgi-bin/vitis/webBlat and use the best hit to calculate 95% mapping ratio.

6 My Own Contribution

My own contribution to the sequence assembly is shown in Table 3.

7 Summary

In a programmer's perspective, sequence assembly is piecing string fragments into a set of long strings using overlap among them. However, things are much complicated in practice due to biological challenges. Most of assembly softwares follow the de bruijn graph approach and reach a satisfying point. However, the running time and memory cost for large scale sequence is still not very good

[1] http://sourceforge.net/apps/mediawiki/dnaa/index.php?title=Whole_Genome_Simulation
[2] http://www.ncbi.nlm.nih.gov/sra
[3] http://www.genoscope.cns.fr/blat-server/cgi-bin/vitis/webBlat

Table 3:

Contribution	Links
Velvet and Oases for windows	https://github.com/zixiaojindao/velvet.git
ABYSS for windows	https://github.com/zixiaojindao/ABYSS.git
MemoryUsageMonitor for windows	https://github.com/zixiaojindao/MemoryUsageMonitor.git
blat-statistics for windows	https://github.com/zixiaojindao/blat-statistics.git
Sequence Assembly Information	https://github.com/zixiaojindao/RNA-sequence-info.git

as well as assembled contigs. Hope new computer algorithms and biological methods could help address this problem much better in the future.

Reference