Tsinghua Univerisity

Computational Biology

Lecture 9: April 21, 2014

Sequence Assembly

Scribe:
Weiyi Chen

Lecturer:
Dr. Michael Zeng

April 28, 2014

Contents

1 Introduction

In bioinformatics, sequence assembly refers to aligning and merging fragments of a much longer DNA sequence in order to reconstruct the original sequence. This is needed as DNA sequencing technology cannot read whole genomes in one go, but rather reads small pieces of between 20 and 30000 bases, depending on the technology used. Typically the short fragments, called reads, result from shotgun sequencing genomic DNA, or gene transcript (ESTs). [Myers, E. W., 2000]

1.1 The Sequencing Problem

Sequencing machines cannot read the entire genomic sequence, but can read fragments. As stated, DNA sequencing technology cannot read whole genomes in one go, but rather reads small pieces of between 20 and 30000 bases, depending on the technology used. Therefore we need to make use of dividing and conquering approach.

- Shear whole DNA sequence into millions of small fragments
- Read 500 - 700 nucleotides each time from small fragments
- Assemble the sequenced fragments

1.2 Different Sequencing Methods

The complexity of sequence assembly is driven by two major factors: the number of fragments and their lengths. While more and longer fragments allow better identification of sequence overlaps, they also pose problems as the underlying algorithms show quadratic or even exponential complexity behavior to both number of fragments and their length. And while shorter sequences are faster to align, they also complicate the layout phase of an assembly as shorter reads are more difficult to use with repeats or near identical repeats. [Batzoglou, S., 2002]

1.2.1 Sanger sequencing: 800bp-1000 bp

In 1975, the Sanger sequencing was invented and until shortly after 2000, the technology was improved up to a point where fully automated machines could

churn out sequences in a highly paralleled mode 24 hours a day. Large genome centers around the world housed complete farms of these sequencing machines, which in turn led to the necessity of assemblers to be optimized for sequences from whole-genome shotgun sequencing projects where the reads are about 800-900 bases long.

1.2.2 454 sequencing: 300-400 bp

By 2004 / 2005, pyrosequencing had been brought to commercial viability by 454 Life Sciences. This new sequencing method generated reads much shorter than those of Sanger sequencing: initially about 100 bases, now 400-500 bases. Its much higher throughput and lower cost (compared to Sanger sequencing) pushed the adoption of this technology by genome centers, which in turn pushed development of sequence assemblers that could efficiently handle the read sets. [Google group]

1.2.3 Illumina Genome Analyzer: 35-150 bp

From 2006, the Illumina (previously Solexa) technology has been available and can generate about 100 million reads per run on a single sequencing machine. Compare this to the 35 million reads of the human genome project which needed several years to be produced on hundreds of sequencing machines. Illumina was initially limited to a length of only 36 bases, making it less suitable for de novo assembly (such as de novo transcriptome assembly), but newer iterations of the technology achieve read lengths above 100 bases from both ends of a 3-400bp clone.

1.2.4 Helicos: 30bp

Announced at the end of 2007, the Helicos assembler by Dohm et al. was the first published assembler that was used for an assembly with Solexa reads. It was quickly followed by a number of others. [Dohm, J. C.; 2007]

2 Sequence Assembly Algorithms

The problem of sequence assembly can be compared to taking many copies of a book, passing each of them through a shredder with a different cutter, and piecing

the text of the book back together just by looking at the shredded pieces. [Myers, E. W., 2000]

2.1 Shortest Superstring Problem (SSP)

The problem is defined as finding a shortest fragment that contains all of the reads, given a set of strings. Specifically,

- Parameter: $(s_1, s_2, ..., s_n)$ as strings
- Return: a fragment s, such that contains all strings $(s_1, s_2, ..., s_n)$ as substrings and the length of s is minimized

2.1.1 Assumptions

There are some assumptions of the problem,

- We do not consider errors with parameters, which states all reads are 100% accurate
- No repeats: we do not consider identical reads in different genome region, which is to say identical reads must be from the same genome region
- Given demands, the optimized return or solution is defined as the shortest string.

2.1.2 Example

- Parameter: {ACG, CGA, CGC, CGT, GAC, GCG, GTA, TCG}
- Return: TCGACGCGTA (length 10)

This is an NP-hard problem. Note that sequencing errors are ignored here.

2.2 Reducing SSP to Traveling Salesman Problem (TSP)

The traveling salesman problem (TSP) asks the following question: Given a list of cities and the distances between each pair of cities, what is the shortest possible

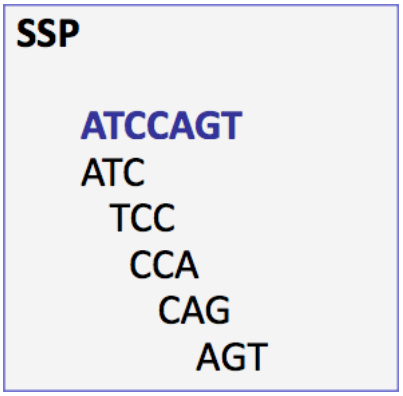
route that visits each city exactly once and returns to the origin city? It is an NP-hard problem in combinatorial optimization, important in operations research and theoretical computer science. Since this is a problem we have already analyzed, we can reduce SSP to the maximum Traveling salesman problem (TSP), and then analyze TSP.

2.2.1 Implementation of Reduction

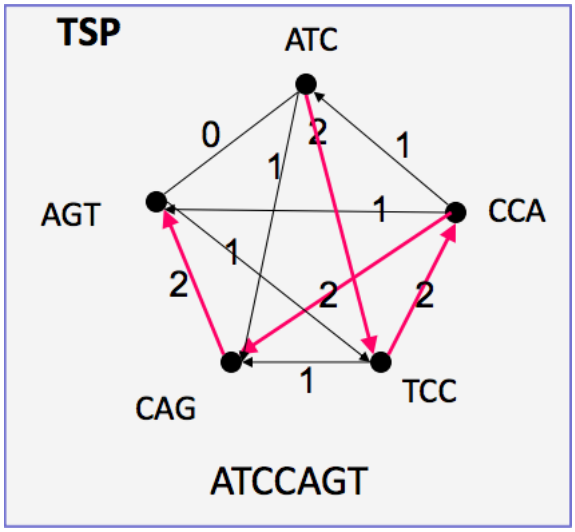
- Define overlap (s_i, s_j) as the length of the longest prefix of s_j that matches a suffix of s_i.
- Make graph with n vertices representing n reads $s_1, s_2, ..., s_n$.
- Insert negative edges of length overlap (s_i, s_j) between vertices s_i and s_j.
- Find the shortest path which visits every vertex exactly once.

2.2.2 Example

Suppose the parameters are $S = \{ATC, CCA, CAG, TCC, AGT\}$, then SSP will be like



And TSP will be like



Unfortunately we have millions of reads and the TSP is NP-complete. What can we do? It's "Eulerian Path". We will talk about that later.

2.3 Greedy Algorithm

Given a set of sequence fragments the object is to find the shortest common super-sequence. The greedy algorithm to solve TSP is

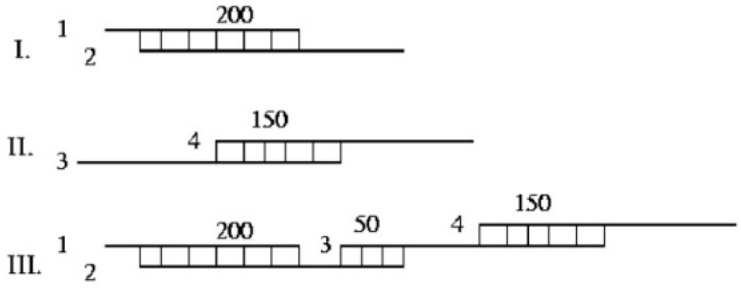
1. calculate pairwise alignments of all fragments
2. choose two fragments with the largest overlap
3. merge chosen fragments
4. repeat step 2. and 3. until only one fragment is left

The result is a suboptimal solution to the problem. The conjecture says that the Greedy Algorithm has approximation factor 2 for the shortest superstring problem. Currently the best bound of the approximation factor for the greedy algorithm is 3.5. It's still an open problem.

2.3.1 Example

An example is shown in graph below, where the assembler joins, in order, reads 1 and 2 (overlap = 200 bp), then reads 3 and 4 (overlap = 150 bp), then reads 2 and 3 (overlap = 50 bp) thereby creating a single contig from the four reads

provided in the input. One disadvantage of the simple greedy approach is that because local information is considered at each step, the assembler can be easily confused by complex repeats, leading to mis-assemblies.



2.4 Overlap Layout Consensus

The relationships between the reads provided to an assembler can be represented as a graph, where the nodes represent each of the reads and an edge connects two nodes if the corresponding reads overlap.

2.4.1 Overlap Graph

A graph (G) consists of vertices (V) and edges (E)

$$G = (V, E)$$

Edges can either be directed (directed graphs) or undirected (undirected graphs). The order of a graph is $|V|$ (the number of vertices). A graph's size is $|E|$, the number of edges. The degree of a vertex is the number of edges that connect to it, where an edge that connects to the vertex at both ends (a loop) is counted twice.

Overlap graph: For a set of sequence reads S, construct a directed weighted complete graph

$$G = (V, E, w)$$

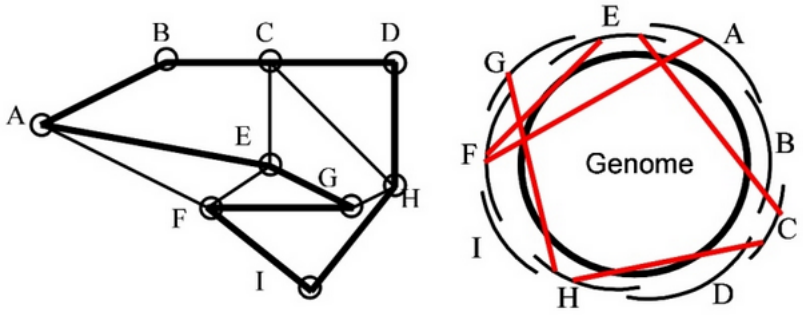
where one vertex per read in the graph, and

$$w(v_i, v_j) = overlap(s_i, s_j)$$

$overlap()$ is the length of longest suffix of s_i that is a prefix of s_j.

2.4.2 Example

The assembly problem thus becomes the problem of identifying a path through the graph that contains all the nodes - a Hamiltonian path in the below graph. This formulation allows to use techniques developed in the field of graph theory in order to solve the assembly problem. An assembler following this paradigm starts with an overlap stage during which all overlaps between the reads are computed and the graph structure is computed. In a layout stage, the graph is simplified by removing redundant information. Graph algorithms are then used to determine a layout (relative placement) of the reads along the genome. In a final consensus stage, the assembler builds an alignment of all the reads covering the genome and infers, as a consensus of the aligned reads, the original sequence of the genome being assembled.



The thick edges in the picture on the left (a Hamiltonian cycle) correspond to the correct layout of the reads along the genome (figure on the right). The remaining edges represent false overlaps induced by repeats (exemplified by the red lines in the figure on the right).

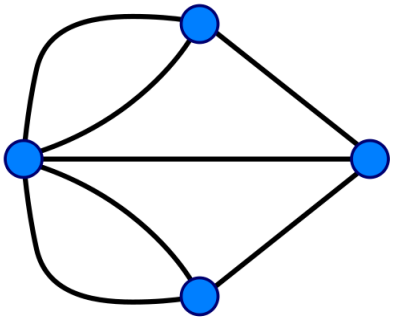
Unfortunately similar to the Traveling salesman problem, finding Hamiltonian path is an NP-complete problem, Checking whether a graph has a Hamiltonian path or not is very difficult.

2.5 Eulerian path

In graph theory, an Eulerian trail (or Eulerian path) is a trail in a graph which visits every edge exactly once. Similarly, an Eulerian circuit or Eulerian cycle is an Eulerian trail which starts and ends on the same vertex. They were first discussed by Leonhard Euler while solving the famous Seven Bridges of Knigsberg problem

in 1736. Mathematically the problem can be stated like this:

Given the graph below, is it possible to construct a path (or a cycle, i.e. a path starting and ending on the same vertex) which visits each edge exactly once?



2.5.1 Properties

- An undirected graph has an Eulerian cycle if and only if every vertex has even degree, and all of its vertices with nonzero degree belong to a single connected component.

- An undirected graph can be decomposed into edge-disjoint cycles if and only if all of its vertices have even degree. So, a graph has an Eulerian cycle if and only if it can be decomposed into edge-disjoint cycles and its nonzero-degree vertices belong to a single connected component.

- An undirected graph has an Eulerian trail if and only if at most two vertices have odd degree, and if all of its vertices with nonzero degree belong to a single connected component.

- A directed graph has an Eulerian cycle if and only if every vertex has equal in degree and out degree, and all of its vertices with nonzero degree belong to a single strongly connected component. Equivalently, a directed graph has an Eulerian cycle if and only if it can be decomposed into edge-disjoint directed cycles and all of its vertices with nonzero degree belong to a single strongly connected component.

- A directed graph has an Eulerian trail if and only if at most one vertex has (out-degree) - (in-degree) = 1, at most one vertex has (in-degree) - (out-degree) = 1, every other vertex has equal in-degree and out-degree, and all

of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.

2.5.2 Eulerian Cycle Algorithm

- Start at any vertex v, traverse unused edges until returning to v
- While the cycle is not Eulerian:
  - pick a vertex w along the cycle for which there are untraversed outgoing edges
  - traverse unused edges until ending up back at w
  - join two cycles into one cycle

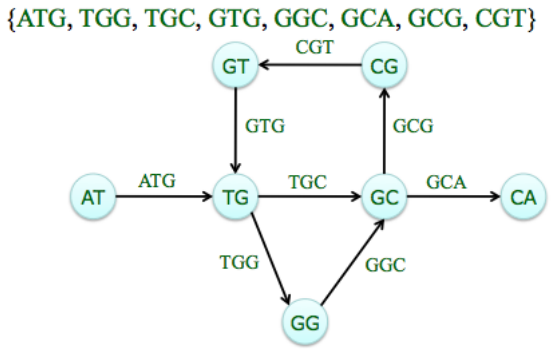
2.5.3 Assembly algorithms

Eulerian path approaches are based on early attempts to sequence genomes through a technique called sequencing by hybridization. In this technique, instead of generating a set of reads, scientists identified all strings of length k (k-mers) contained in the original genome. While this experimental method did not produce a viable alternative to Sanger sequencing, it led to the development of an elegant approach to sequence assembly. This approach, also based on a graph-theoretic model, breaks up each read into a collection of overlapping k-mers. Each k-mer is represented in a graph as an edge connecting two nodes corresponding to its k-1 bp prefix and suffix respectively. It is easy to see that, in the graph containing the information obtain from all the reads, a solution to the assembly problem corresponds to a path in the graph that uses all the edges - an Eulerian path. One advantage of the Eulerian approach is that repeats are immediately recognizable while in an overlap graph they are more difficult to identify.

According to the statement above, we can construct a deBruijn graph:

- Edges represent k-mers that occur in s
- Vertices correspond to $(k-1)$-mers

- Directionality goes from the $k-1$ prefix to the $k-1$ suffix



- Find a DNA sequence containing all k-mers, which corresponds to find a path that visits every edge of the graph exactly once.

3 Conclusion

Assembly is a Complicated Problem. Besides the obvious difficulty of this task, there are some extra practical issues: the original may have many repeated paragraphs, and some shreds may be modified during shredding to have typos. Excerpts from another book may also be added in, and some shreds may be completely unrecognizable.

For the repeats, it is hard to know where repeats begin or end, whether reads are not long enough, possibility that human genome has many different repeats and gene duplications. For the sequencing errors, it is hard to distinguish true from error-based overlaps too.

References

[Myers, E. W., 2000] Myers, E. W.; Sutton, GG; Delcher, AL; Dew, IM; Fasulo, DP; Flanigan, MJ; Kravitz, SA; Mobarry, CM et al. (March 2000). A whole-genome assembly of Drosophila *Science*, 287 (5461): 2196-204.

[Batzoglou, S., 2002] Batzoglou, S.; Jaffe, DB; Stanley, K; Butler, J; Gnerre, S; Mauceli, E; Berger, B; Mesirov, JP; Lander, ES (January 2002). ARACHNE: a whole-genome shotgun assembler *Genome Research*, 12 (1): 177-89.

[Google group] Copy in Google groups post announcing MIRA 2.9.8 hybrid version *Usenet group*

[Dohm, J. C., 2007] Dohm, J. C.; Lottaz, C.; Borodina, T.; Himmelbauer, H. (November 2007). SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing *Genome Research*, 17 (11): 1697?706.