Every organism has a *genome* in its cell nucleus which encodes the features of the organism. The genome is generally organized in *chromosomes*, and is a polymer molecule. Ignoring many details, it suffices to say that the genome is a sequence of so called *nucleotides*: Adenine, Guanine, Cytosine and Thymine (in DNA; while Uracil takes the place of Thymine in RNA). Writing down this sequence is an important problem in Biology: it is expected that it would lead to early detection of predisposition to diseases, development of new kinds of treatment of diseases, improving yield of plants, or even new forms of life.

The Human genome is about 3 billion nucleotides long. We may think of it as a string over the alphabet $\{A, G, C, T\}$, respectively representing the 4 DNA nucleotides. No direct techniques are available for reading off the sequence of nucleotides in molecules this long. Direct reading of the sequence is possible only for fragments which are around 700-1000 nucleotides long. So the following approach is used for reading long genomes:

1. Make many copies of the genome.
2. Break each copy (independently) randomly into small pieces, say about 700 nucleotides long.
3. Identify the sequence for each piece by direct techniques. Once the sequence is read, it is customary to call each piece a *read*.
4. If a long enough suffix of one read is identical to the prefix of another read, consider it to be evidence that the two pieces in fact overlap in the original genome. Thus they may be merged into a longer piece.
5. Repeat the above as possible.

There are many variations on this theme, but the basic idea is to detect and use overlaps between the pieces to assemble larger and larger fragments. There are many ways of doing this, we will discuss one such approach.

0.1 Overview

The approach has two main ideas. The first idea is to use the reads to construct a representation of all plausible genomes which could give rise to the given set of reads. The second idea is to determine which of these is most likely to have been the genome from which the reads resulted. In this we think of the process of chopping up the genome as a random process, and the reads as the outcome of the random process. The goal is to find the reconstruction which has the maximum probability of being the one from which the reads resulted. Usually, we cannot identify a single unique genome as the answer, instead we merely identify fragments (much larger than the reads themselves) that are likely to be parts of the genome.

1 Shortest Superstrings vs. Euler tours

Our algorithm starts with a set of n reads, $R = \{r_1, \ldots, r_n\}$. The goal is to reconstruct the genome G from which the reads arose.

Clearly, G should contain each read, possibly in an overlapped manner. Indeed, one possible but clearly inappropriate solution is simply the concatenation of the reads. This is deemed an unlikely solution, because it does not contain any overlaps, which would almost surely be present if the reads came from random fragmentation of identical copies of the genome. Early formulations of the problem therefore suggested that the reconstructed genome G should be be the shortest possible string that contains all the reads. Unfortunately, finding such a shortest superstring of R turns out to be an NP-complete problem.

There are actually several problems with this formulation. If a genome contains long repeated subsequences, the above method would never discover them, because this would go against the goal of shortening the reconstructed string.

It was also felt that not only should all reads appear in the assembled genome, but all the (significant) overlaps must also occur. The rationale was that if a long suffix (say longer than some parameter τ to be suitably fixed) of a read appears as a prefix of another read, then we should expect to find the reads in the overlapped manner in the genome. This gives rise to the Euler tour formulation of the assembly problem as follows.

We begin by fixing τ, a reasonable value is 100. If you assume (gross

approximation, of course) that the genome is a random string, then the probability that two reads have an overlap of 30 is $4^{-100} \approx 10^{-20}$. So a string of length 100 is unlikely to appear twice even in a billion long genome. Thus if two reads have a suffix-prefix overlap of 100, there is very good chance that they came from the same region of the genome.

The (τ) overlap graph for a read set R is a directed, weighted (called length in what follows), labelled graph defined as follows.

Vertices: For each read r_i that is not contained in any other read[1] we construct one vertex. We will refer to this vertex as vertex i.

Edges: If there exits an overlap of length at least τ between a suffix of r_i and a prefix of r_j, we construct a directed edge from vertex i to vertex j, i.e. edge (i, j).

Labels and lengths: Suppose (i, j) is an edge. Let r_j be a concatenation of string s followed by a string l, where s specifies the prefix overlapping with r_i. Then the edge (i, j) is labelled l, and is assigned length $|l|$, i.e. the length of the string l.

Consider an Euler tour which visits every vertex of the overlap graph at least once. The label of an edge (i, j) is simply the sequence that takes us from the end of read r_i to the end of read r_j. By concatenating the labels along the tour, we indeed get a string which contains all reads except the one corresponding to the starting vertex (See [1]), as well as a significant number of the overlaps between reads. Note that it is possible that the tour need not be simple, i.e. the tour may pass through certain vertices/edges more than once. This will correspond to genomes in which there are repeated substrings. Such repetition is common, say in plant genomes.

The overlap graph is thus a representation of all plausible genomes which can give rise to R. Two points must be noted, however. First, we do not use the overlap graph directly, some simplifying transformations can be applied to it[1]. Second, typically, from all the set of plausible reconstructions as defined by the overlap graph, we will not be able uniquely identify a single Euler tour as the *correct* one. The best we can do is to decide how many times each edge of the overlap graph is traversed. The traversal counts will further enable us to simplify the graph. In the end it is the simplified graph that we output. Any tour in it which confirms to the traversal counts we predict is a possible assembly. Alternately, we can say that every edge in our

[1]However, the contained vertices are not completely ignored; they will figure in the subsequent processing, see paper.

final overlap graph is a fragment of the genome from which R was obtained.

The traversal counts are predicted using a maximum likelihood formulation as follows.

2 Maximum Likelihood

We first describe the maximum likelihood idea in general, and only later apply it to the problem of predicting traversal counts. The basic idea is to simply consider all possible genomes which contain all the reads that have been observed, and pick the one which has the greatest probability of generating the observed read set.

Here is a probabilistic model of the generative process. First, a genome G is picked "from the wild" with probability $P[G]$. It is randomly broken into the read set R. We wish to find that G such that $Pr[G|R]$ is maximum.

$$Pr[G|R] = Pr[GR]/Pr[R] = Pr[R|G]P[G]/P[R]$$

This quantity is to be maximized over different G. For this maximization $P[R]$ does not change with G. It is customary to assume that all genomes G are likely to be chosen in the first step, thus $P[G]$ is the same for all G, and so does not matter for the purpose of maximization. So we pick that G which maximizes $Pr[R|G]$. This is the so called maximum likelihood method. Notice that we wanted to find a G that maximizes $P[G|R]$ – we are instead finding a G which maximizes $P[R|G]$ which is a much simpler problem.

For calculating $Pr[R|G]$, remember that you actually have each r_i and you have fixed a G. To estimate the probability, we need to define our generative model further. After selecting G we generate the ith read as follows. L_G denotes the length of G.

1. Select a length l_i from a fixed distribution P (independent of the genome G).

2. Pick a starting point for the read. There are $L_G - l_i \approx L_G$ choices for this. The sequence of nucleotides from the chosen position of the chosen length then becomes a read.

So now we can estimate the probability that the given read set R got generated in the given order. Suppose a read r_i appears δ_i times in a candidate

genome G. Then since the starting point is picked at random, the probability of r_i of length L_i being generated is simply $P(L_i)\delta_i/(L_G - L_i)$, where $P(L_i)$ denotes the probability of selecting a length L_i for the read from the length distribution. The probability of generating the set R in the sequence $r_1, \ldots, r_n$ is simply $\prod_i P(L_i)\delta_i/(L_G - L_i)$. However, the same read set R can be generated through other permutations, e.g. if all reads are distinct, then the probability of observing R is $n!$ times the probability above. In general the exact probability is $\alpha \prod_i P(L_i)\delta_i/(L_G - L_i)$, where α depends upon the read set R but not on the genome G. Assuming $L_G \gg L_i$ we may write $L_G - L_i \approx L_G$, and observing that $\prod_i P(L_i)$ also does not depend upon G, we may write the probability P_G of generating R from G as:

$$P_G = \alpha' \prod_{i=1}^{n} \frac{\delta_i}{L_G}$$

where $\alpha' = \alpha \prod_i P(L_i)$ is a constant depending on R (and the length distribution) but not on G. Our goal is to find G that maximizes P_G, or equivalently $\prod_i \delta_i/L_G$. The maximization must satisfy certain constraints, which we describe next.

For the rest of the story, please refer to [1].

Exercises

1. Consider the read set $R = \{ACTG, CTGG, GGAC\}$, and assume that tau=2. (a) Draw the overlap graph, showing all the labels and weights. Include the source and sink vertices, but do not perform transitive reduction or any other optimizations. (b) Calculate the probability that these reads result from genomes ACTGGAC, CTGGACTG, ACTGCTGGGGAC? Do not use the approximations $L_G - l_i \approx L_G$ but use the exact values.

2. Use some public domain solver to find the optimal value for the traversal counts. On the basis of these state what you think is the most likely genome from which R as above results.

References

[1] A. Varma, A. Ranade, and S. Aluru. An improved maximum likelihood formulation for accurate genome assembly. In *IEEE International Conference on Computational Advances in Bio and medical Sciences (ICCABS)*, 2011.

`www.cse.iitb.ac.in/~ranade/GraphAssembly.pdf`

.