SyntenyFinder: A Synteny Blocks Generation and Genome Comparison Tool

Intern: Ilya Minkin
Advisor: Son Pham

We present an algorithm for finding synteny blocks in genomes represented as nucleotide sequences. The algorithm is based on colored de Bruijn graphs and graph simplifications. Our method is suitable for finding synteny blocks in genomes that contain regions of highly conserved DNA, i.e. genomes that are evolutionary close to each other. With some modifications the algorithm can be applied in more complicated cases.

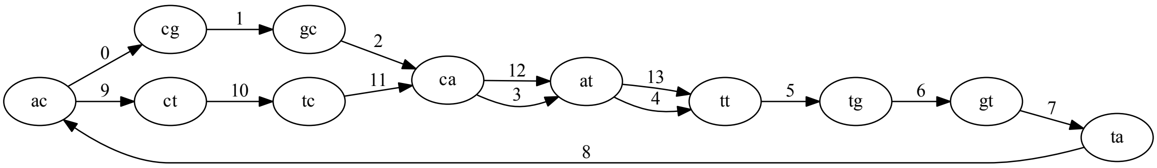
1. Introduction

Recent growth of number of sequenced genomes arises question about their evolution relationship. In order to perform rearrangement analysis, the genomes must be decomposed into conservative segments, called synteny blocks. Currently existing tools for solving this problem, like DRIMM-Synteny [1], require the genomes to be presented as sequences of enumerated local alignments, or *anchors*. Usually, anchors represent homologous genes. At this moment, there are no general purpose tools that can find synteny blocks from the genomes represented as unannotated nucleotide sequences.

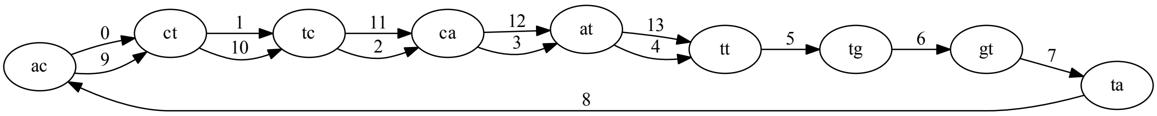
De Bruijn graphs are extensively used in bioinformatics for genome assembly [2, 3]. In this work we address problem of finding synteny blocks from nucleotide sequences. We propose new algorithm for this task based on colored de Bruijn graph.

2. Problem definition

Suppose that we are given a set $S = \{s_1, s_2, \dots, s_n\}$ of chromosomes, and each chromosome is represented as a string over alphabet $\{A, C, G, T\}$. The task of finding synteny blocks is to find a set of so called conserved regions $C = \{C_1, C_2, \dots, C_n\}$, where each conserved region C_i is a set of substrings of chromosomes from S. Such regions are



(a) De Bruijn graph built from string "*acgcattgtactcatt*" and $k = 2$. Non-branching paths correspond to multiple copies of the same substrings.



(b) Same de Bruijn graph after simplification. Replacing "*acGca*" by "*acTca*" we obtain long non-branching path that corresponds to the synteny block.

Figure 1: Illustration of de Bruijn graphs and graph simplification

supposed to cover most of the genome for closely related species. All substrings forming a conserved region C_i must be similar to each other according to some criterion of similarity. Note that problem of finding synteny blocks in a set of chromosomes is equivalent to a problem of finding synteny blocks in one superchromosome obtained from concatenating all chromosomes from the set. We can just separate chromosomes by special characters.

At this moment there is no generally accepted formal criterion of similarity exist, so the problem of finding synteny blocks is ill-defined. In our work we introduce new criterion of similarity based on de Bruijn graphs and graph simplifications.

3. General idea

As previously mentioned, our method is based on de Bruijn graph. Given a fixed value k and a string S we can build de Bruijn graph G from the string as follows. Let's denote by k-prefix of a string S first k characters of S, and by k-suffix last k characters of S. For each unique substring of length k (called k-mer) found in S, we add a vertex to G and mark it with corresponding k-mer. For each $(k+1)$-mer w found in S we add edge that connects vertex corresponding to k prefix of w with vertex corresponding to k suffix of w and label the edge with position of first character of w (multiedges with different labels are allowed).

In this graph we consider only paths that have consecutive labels on edges. It is easy to see that with such restriction every path in G corresponds to a substring in S. Example of such de Bruijn graph built from the string $S =$ "*acgcattgtactcatt*" and $k = 2$

is depicted on Figure 1a.

Note that two copies of substring "*catt*" form a non-branching path consisting of edges with multiplicity 2 in this graph. Single mismatch in substrings "*acGca*" and "*acTca*" form so-called "bulge", unoriented cycle generated by two valid paths with coinciding ends. If we replace one branch of the bulge by another (replace "*acGca*" by "*acTca*" for example), we will obtain a long non-branching path (Figure 1b).

This heuristic forms basis of our method – conserved regions in different parts of the genome contain conserved basepairs, but such regions are disrupted by indels and mismatches. These differences form bulges in the graph that make it different to infer structure of the synteny blocks. We remove bulges having size less than some predefined constant and thus obtain non-branching paths corresponding to the conserved regions. The process of removing bulges from the graph is called *simplification*.

Conserved regions can be located on opposite strands of DNA. To handle this, we use *colored* de Bruijn graphs [3]. Given string S, for each $(k+1)$-mer found in S we add corresponding edge to the graph and color it *blue*, for each $(k+1)$-mer found in reverse-complementary counterpart of S we add corresponding edge to the graph and color it *red*. In this graph, non-branching paths with different colors represent synteny blocks located on opposite strands of DNA.

Complete pipeline is following:

1) Concatenate input chromosomes into one superchromosome
2) Build de Bruijn graph from the superchromosome
3) Simplify the graph
4) Output synteny blocks as non-branching path in the graph

Our algorithm depends on two parameters: k and δ (minimum allowed size of a bulge). It is reasonable to use as high k as possible ($k > 50$) to keep graph structure simple and avoid connecting regions that are actually not homologous. So, our method requires that conserved regions in input genomes contain exact shared k-mers. This is not a problem in genomes that are very close to each other (like different strains of a bacteria), but it can create difficulties in genomes that are separated by many evolutionary events.

This issue can be solved, for example, by finding a set of all local alignments in the genomes and substituting one subsequence in each found alignment by another. In results section we will demonstrate on a practical half-synthetic example that with such modifications our method is able to handle complicated cases. So at this point our method is directly applicable to only evolutionary close genomes and in near future we plan to extend it to wider range of use.

4. Detailed description

In this section we formally describe our algorithm for finding synteny blocks. Suppose that we are given a string $S = (s_0, s_1, \ldots, s_{n-1})$ over alphabet $\Sigma = \{A, C, G, T\}$, two numbers k and δ. We denote by $S(i,j)$ the (i,j) substring of S, $S(i,j) = (s_i, s_{i+1}, \ldots, s_j)$. Let's denote by $\bar{S}$ string that is the reverse complementary of S.

Colored De Bruijn graph is graph $G_k = (V, E)$ where $V = \Sigma^k$. We define three

functions:
1) $Pos: E \to \mathbb{N}$
2) $Spell: E \to \Sigma^{k+1}$
3) $Color: E \to \{Blue, Red\}$

For each $i \in \{0, 1, \dots, n-k-1\}$ we add two oriented edges to the graph:
1) $e = (S(i, i+k-1), S(i+1, i+k))$, $Color(e) = Blue$, $Pos(e) = i$, $Spell(e) = S(i, i+k)$
2) $\bar{e} = (\bar{S}(i, i+k-1), \bar{S}(i+1, i+k))$, $Color(\bar{e}) = Red$, $Pos(\bar{e}) = i$, $Spell(\bar{e}) = \bar{S}(i, i+k)$

A path in G_k is a sequence of edges $P = (e_1, e_2, \dots, e_n)$ iff $Pos(e_{i+1}) = Pos(e_i) + 1$ and $Color(e_{i+1}) = Color(e_i)$. Let's denote by $Start(P)$ first vertex of the path P and by $End(P)$ the last vertex of P.

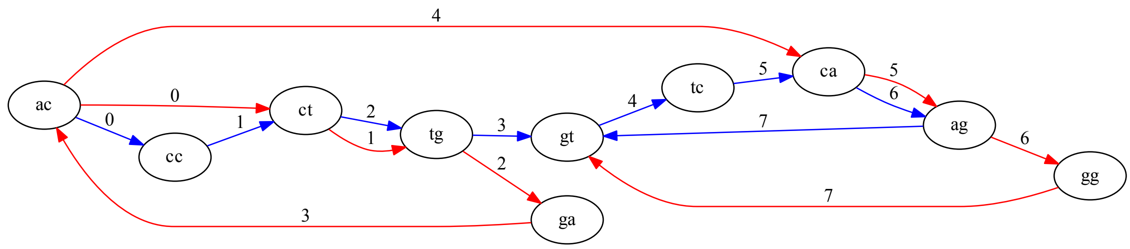


Figure 2: Colored De Bruijn graph for string $S = "acctgtcagt"$

A pair of paths $B = \{b_1, b_2\}$ is called a *bulge*, iff following holds:
1) $|b_1| < \delta \;\wedge |b_2| < \delta$
2) $Start(b_1) = Start(b_2) \wedge End(b_1) = End(b_2)$
3) b_1 and b_2 have no common vertices except $Start(b_1)$ and $End(b_1)$
4) There are no edges $e_1 \in b_1, e_2 \in b_2$ such that $Spell(e_1) = Spell(e_2)$

A vertex v is called *bifurcation* iff there are at least two outgoing (ingoing) edges e_1, e_2 incident v such that $Spell(e_1) \neq Spell(e_2)$. A set of paths $P_{nb} = \{P_1, P_2, \ldots, P_n\}$ is said to form a *non-branching path* iff $|P_1| = |P_2| = \ldots |P_n|$ and $Spell(e_{i,k}) = Spell(e_{j,k})$, where $e_{i,j}$ denote j-th edge in the i-th path, i.e. all paths spell the same substring.

Let's illustrate above definitions on a simple example. Colored De Bruijn graph built from string $S = "acctgtcagt"$ is depicted on Figure 2. Here $\bar{S} = "actgacaggt"$. Vertices $"ac", "ct", "tg"$ are bifurcations, while $"cc", "tc", "ga"$ are not. Two paths $("ac", "ct")$ and $(("ac", "cc"), ("cc", "ct"))$ form a bulge. Two multiedges $("ct", "tg")$ form a non-branching path.

5. Experimental results

Results

6. Conclusion

Conclusion

References

[1] Son K. Pham, Pavel A. Pevzner. DRIMM-Synteny: decomposing genomes into evolutionary conserved segments. Bioinformatics (2010) 26 (20): 2509-2516.

[2] Pavel A. Pevzner, Haixu Tang, Michael S. Waterman. An Eulerian path approach to DNA fragment assembly. Proc. Natl. Acad. Sci. USA. 2001 Aug 14; 98(17): 9748-53.

[3] Zamin Iqbal, Mario Caccamo, Isaac Turner, Paul Flicek, McVean. De novo assembly and genotyping of variants using colored de Bruijn graphs. Nat Genet. 2012 Jan 8;44(2):226-32. doi: 10.1038/ng.1028.

[4] Manolis Kellis, Bruce W. Birren, Eric S. Lander. Proof and evolutionary analysis of ancient genome duplication in the yeast Saccharomyces cerevisiae. Nature 2004 Apr 8;428 (6983): 617-24.