UiO : Department of Informatics
University of Oslo

Alignment against a graph-based reference genome

Fuzzy searching in large and complex structures

Esten Høyland Leonardsen

Master's Thesis Spring 2016

Alignment against a graph-based reference genome

Esten Høyland Leonardsen

8th January 2016

Abstract

Contents

List of Figures

List of Tables

Preface

Part I

Introduction

Chapter 1

Background

1.1 DNA

Deoxyribonucleic acid (DNA) is a molecule which allows living creatures and viruses to store and pass on genetic information.How/What is stored

The DNA of an individual is made up by two complementary strands of nucleotides bound together in a double helix, where the nucleotides can contain the bases **A**denine, **C**ytosine, **G**uanine or **T**hymine. Complementary in this context means that instead of one singular sequence of bases DNA is made up by a sequence of paired bases, A's with T's and C's with G's, called base pairs. Due to the chemical structure of the molecules making up a single *red*strand of DNA each *strand* can be said to have a direction, upstream towards the 5' end or downstream towards the 3' end. Two complementary strands have opposing directions and are thus called the reverse complements of eachother.

The size of DNA varies across species, from a couple of thousand basepairs (kb) in some viruses to several hundred billion basepairs (gb) in larger, more complex organisms. The human genome comes in at the higher end of this range, with a length of roughly 3 gb. A continuous sequence of bases is called a contig, several contigs combined is a scaffold which again sits together into chromosomes, the building block of the genome of an organism.

1.1.1 Genomic variations

Over the span of time DNA is subject to change. Through random mutations and recombination a genetic sequence can be changed either within an individual or as a product of reproduction. The fact that these changes are able to survive and propagate through generations leads to a genepool where even though the DNA comes from a common ancestor, different individuals will have different variants of the original sequence. These variations form the basis for the division into species, but even within species alot of natural variation will occur.

The least complex of these variations are Single Nucleotide Polymorphisms (SNPs), where a single nucleotide has changed between two individuals, and insertions and deletions (Indels) where either one or a short sequence of bases have appeared or disappeared from the DNA of an individual. Longer and more complex structural variations can also occur when a larger part of a chromosome breaks free and disappears completely or inserts itself in a different place or the opposite direction.

1.1.2 Alignment

The fact that a genome is built by discrete entities, the bases A, C, G and T, means that any DNA sequence can be represented by a text string. The double-stranded nature of DNA could be encoded into the string, but as one side can easily be derived from the other representing one of the strands is usually expressive enough. The process of determining genetic variation between two separate individuals can then be seen as the problem of finding the similarities and differences in the two corresponding text strings. Motivation?

String comparison in computer science

There exists several ways of determining the difference, called the edit distance, between two strings mathematically. The main difference between the approaches are which operations are possible on the two strings and how the result is scored. Perhaps the most common technique is called Levenshtein distance, which allows the operations deletion, insertion, and substitution. All of the preceding operations works on a single character, and one operation by itself yields a distance of 1. Finding the optimal Levenshtein distance can be done by dynamic programming in linear time.

DNA sequence comparison

When comparing two DNA sequences the edit distance problem has to be modified in such a way that the most probable evolutionary changes are scored the highest. Most importantly there are two traits to capture: Some substitutions of bases are more common than others, and the fact that it's harder for a part of DNA to break off than it is to actually move around after it has detached. The first problem can be tackled with a substitution matrix: A matrix in which every possible pair of bases has a substitution score. The second trait can be captured either with a pure logarithmic gap penalty, or by an affine gap penalty: A different penalty for opening and extending the gap. In the latter the penalty for extending a gap is constant, but lower than it is for opening a gap (See figure 1.1). With small modifications the dynamic programming algorithm can encapsulate these ideas and be used to align DNA sequence comparisons.

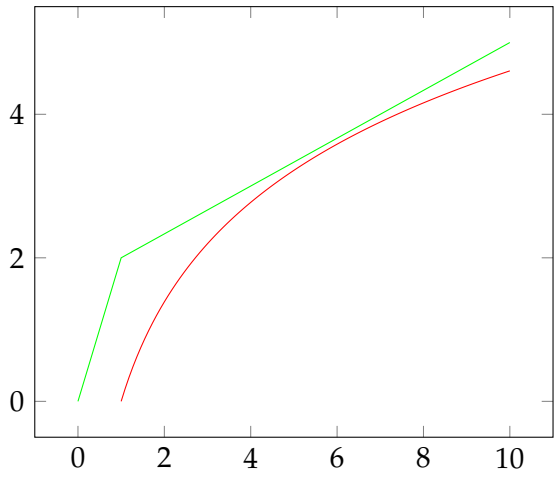

Figure 1.1: A logarithmic (red) and an affine gap (green) penalty function

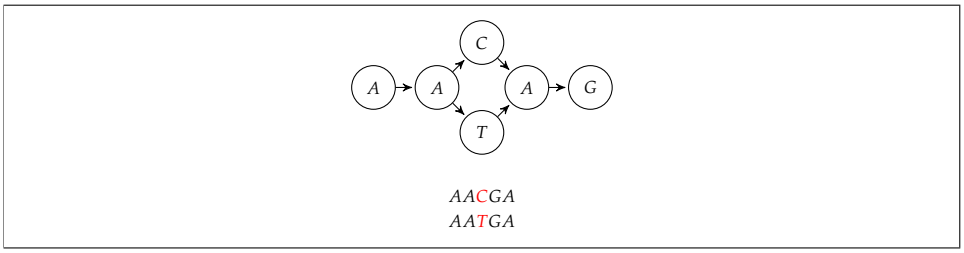

Figure 1.2: Two sequences with an SNP, and the corresponding graph

1.1.3 Reference genomes

To say that an individual of a species deviates from the normal in any way there has to be a standard to compare it to, a reference genome. For humans the current reference genome is the GRCh38 developed and maintained by the Genome Reference Consortium. Something about the represented variation

1.2 Graph-based genome representation

Motivation

1.2.1 De Bruijn graphs

1.2.2 Mapping/Coordinate system

1.2.3 Alignment

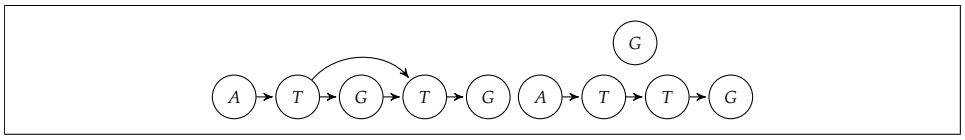

Figure 1.3: Two sequences with a single nucleotide indel, and its two corresponding (equivalent) graph representations

Chapter 2

Method

In this chapter I will present an algorithm for aligning input sequences against graph-based reference genomes. At first I define some of the terms

2.1 Definitions

Definition 1. The *input sequence* I is a string over the alphabet A, C, G, T. Smaller parts of the sequence are called *subsequences*, the subsequence spanning from x to y is denoted $I_{x:y}$. An individual character in the sequence on position x is denoted I_x.

Definition 2. *Reference genome graph*, *graph*

Definition 3. *node*

Definition 4. An *edge* is an ordered pair or nodes, $e = \{v_x, v_y\}$. The nodes are accessible by the notation $(e)_s = v_x$, $(e)_e = v_y$

Definition 5. A *path* is a series of consecutive edges. Two edges e_x, e_y are consecutive if $(e_x)_s = (e_y)_e$

Definition 6. A path which starts at S and ends in E *complete path*. All sequences represented in the graph has a corresponding complete path. All complete paths are possible sequences.

Definition 7. A node n_x is a *critical node* if and only if every complete path contains n_x

Definition 8. A *region* R_x is a subset of nodes from a graph G containing a start-node S_{R_x} and an end-node E_{R_x} and every node on every path between them. Both S_{R_x} and E_{R_x} are critical nodes. S_{R_x} and E_{R_x} can be the same node, yielding a one-node region. Two consecutive regions share exactly one node, such that $E_{R_x} = S_{Rx+1}$

Definition 9. A region R_x is a *critical region* if every node $v \in R_x$ is a critical node. Every region that is not critical is called a *non-critical region*

Definition 10. A critical region R_x is a *maximal critical region* if S_{R_x} has either none or several incoming edges and E_{R_x} has either none or several outgoing edges

Definition 11. A region R_x is a *minimal non-critical* region if every node $v \in (R_x \backslash \{S_{R_x}, E_{R_x}\})$ is not critical

Corollary 1. Every graph is a series of consecutive alternating maximal critical and minimal non-critical regions

2.2 The problem

2.3 The algorithm

For any sequence/pair graph there are 3 possible scenarios:

1. The entire sequence maps to an already existing complete path in the graph
2. The sequence maps partially to one or several existing paths in the graph
3. The sequence does not map to any existing structures in the graph

This is also true for all subsequences of length >= 2. The first main idea behind the algorithm is that for a number of subsequences one can decide whether they are instances of the first or third scenario alot more efficiently than doing a complete search. This is based on the assumption that every input sequence that should be mapped to an existing part of the graph have large subsequences isomorphic to the corresponding region. The notions of both isomorphic and large in this context will be more clearly defined in the following more technical specification of the steps in algorithm. Another important assumption is that whenever the algorithm are unable to find any such isomorphic regions it is correct to represent the sequence by an entirely new path in the graph. The second main idea is that when we have identified these easily mappable parts of the sequence and mapped them to parts of the graph, the remaining, currently unmapped, subsequences of the sequence can be mapped against largely pruned areas of the graph. This comes from the assumptions that we are not interested in mapping an individual character to an individual node, but larger subsequences to larger regions. This is an assumptions which is always true when we are dealing with graphs that have more nodes than legal node values.

At this point the algorithm can be mapped as out follows:

1. Locate easily mappable regions
2. Assign unmapped subsequences to regions based on current mappings
3. Do fine grained mapping of these subsequences against their corresponding regions

Bibliography

[1] Deanna M. Church et al. 'Extending reference assembly models'. In: *Genome Biology* (2015).