Graph Alignment for Human DNA

UNIVERSITY OF ICELAND
SCHOOL OF ENGINEERING AND NATURAL SCIENCE
FACULTY OF INDUSTRIAL ENGINEERING,
MECHANICAL ENGINEERING AND COMPUTER SCIENCE

Tom Willy Schiller
Supervisor: Páll Melsted

May 2015

Abstract
Being able to read out a person's individual genetic makeup helps with medical applications, the establishment of heritage lines etc. Often a large amount of short parts (“reads”) of the individual DNA is read out and afterwards assembled based on a reference string representing the average human genome.
In this thesis, a new approach to this DNA assembly is implemented, in which the reads are not aligned to a reference string, but instead to a reference graph. This can lead to better alignment results and a more accurate picture of the DNA that was read out.

Introduction
This Master's thesis aims to improve current approaches from the field of human genome sequencing used to sequence whole individual genomes. This can be used to find previously unknown links between genetic variants and diseases throughout large population groups, or to diagnose rare diseases. Such a genome analysis is often done via a pipeline of separate programs working on a single genomic reference sequence string.
However, we should think of the Human Genome as a collection of individual genomes rather than a single rigid reference genome string. This corresponds to viewing the reference as a graph rather than a linear genome. On this graph, each variation can be represented by a branch and each personal genome by a path. This not only simplifies finding common variants, but also makes it possible to pick up combinations of variants which would currently be lost in the alignment step.
The particular aim of this Master's thesis is the creation of a software tool that can align short read sequences to a variation graph such as seen in Figure 1 and can be used in the analysis pipeline.

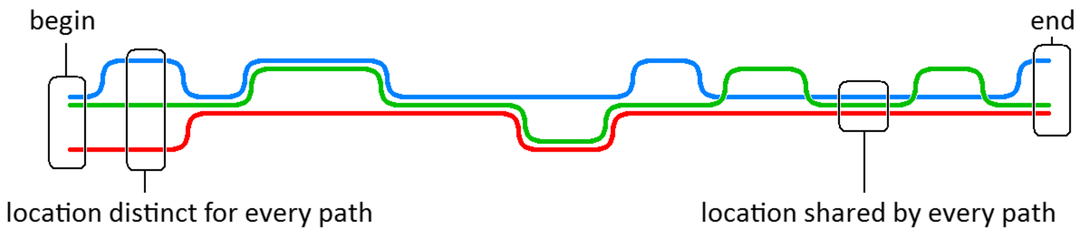

Figure 1: Genomic variation graph based on three individuals

Methodology
Two different ways for implementing the alignment to a reference graph are pursued, with the first one being based on the Burrows-Wheeler Transform (in short BWT) and the second one on hash based indices that keep track of reference graph sequences of a given length.
Working with the BWT, an inclusion in one of the most commonly used alignment tools can be achieved. This tool is called BWA and due to it being an open source software, new ideas like the reference graph approach can be included directly into the existing program. This enables benchmarking the results of using a reference graph rather than a reference string with real world data and a real world tool.
The hashing approach on the other hand addresses one of the key problems with the reference graph, which is that we lose the straightforward indexing of the reference string that could otherwise be used. By working with hashes for reads of a specific length, we can circumvent this central problem.

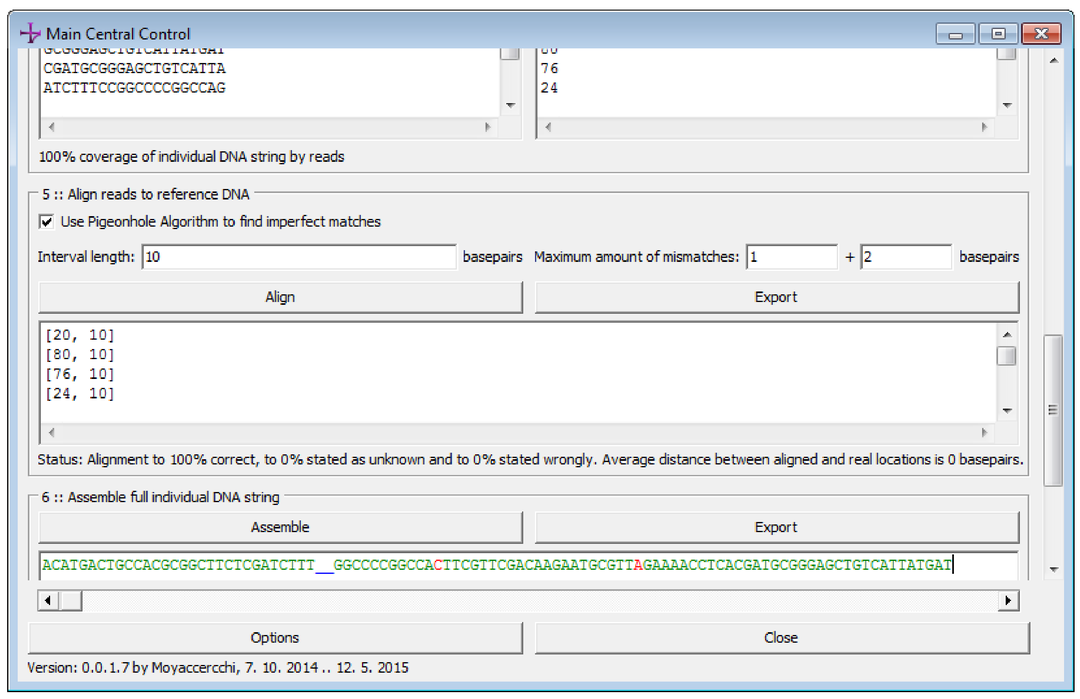

Figure 2: Main control program to run the test pipeline

Implementation
Considering different approaches to using a reference graph in the alignment pipeline in theory is very helpful. However, to really make any informed choices about which approaches work the best, it is necessary to actually implement and test them.
To do so, I first develop my own regular DNA alignment pipeline as test environment. As the speed of this pipeline is not as important as the ease of implementation, I write the background scripts that work on different steps in python and link them all to a main control program written in Delphi, which can be seen in Figure 2. The main program takes care of creating artificial test data, running the pipeline and analyzing how well it performed.
After implementing the regular pipeline based on string graphs I amend this test pipeline to work with population reference graphs. One challenge here is the lack of a standard file format for genomic graphs, with the main options being FASTG and GFA. For testing purposes I also create my own file format called STPU, which can keep track of any non-cyclic bubbles.

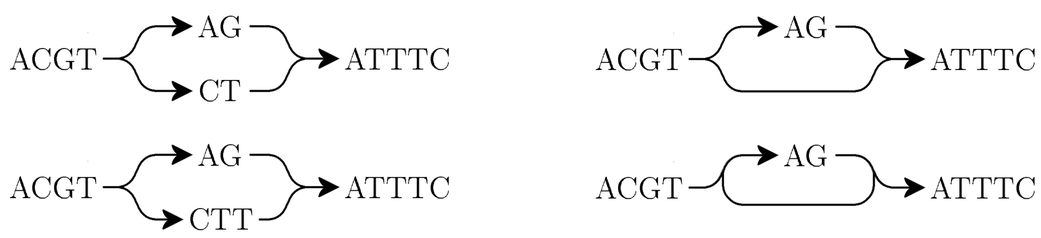

Figure 3: Kinds of bubbles: same length alternatives (top left), insertion (top right), different length alternatives (bottom left), cycle (bottom right)

The main challenge however is the lack of a straightforward indexing mechanism for the reference, as the various steps of the pipeline need to communicate between each other about locations within the reference and need a common language to do so. To simplest solution to this problem is by using the regular string indexing regardless of the reference now being a graph, by first reducing all bubbles and then working with that flattened graph in later steps. This makes it possible to use the full reference graph in the pre-processing step, while not having to deal with the indexing problem in the alignment step.
Finally, there is also the challenge of how to assess the performance of the pipeline. For test data that has been artificially generated, the alignment quality can be checked as it is known where the reads originated from.
If however data is supplied for which the original location of the reads is not known, such as real world data, then the analysis can only show how well the aligned reads fit together during assembly. To do so, for each position in the individual string that is given out the information from all overlapping reads at that position is considered, as can be seen in Figure 4.

Position i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Read 1	G	G	C	G	A	T	C	G	T	A	A					
Read 2				G	A	T	C	G	T	A	A	T	G	G	A	C
Read 3			C	G	A	C	C	G	T	A	A	T	G	G	A	C
Read 4	G	G	C	G	A	T	C	G	G	A	A	T	G			
Read 5	G	G	C	A	T	C	C	G	T	A	A	T	G	G	A	C
Assembled DNA	G	G	C	G	A	T	C	G	T	A	A	T	G	G	A	C
Quality q_i	1.0	1.0	1.0	0.8	0.8	0.6	1.0	1.0	0.8	1.0	1.0	1.0	1.0	1.0	1.0	1.0

Figure 4: Assembly quality assessment with five reads spanning parts of the assembled DNA

Results and Next Steps
Using a reference graph rather than a reference string does improve the alignment quality, although several problems arise that limit the usability of this method for now. Most of all, the actual alignment step is still based on an artificially flattened reference string constructed from the graph and an explicit suffix-array-like structure, which makes the implementation very straightforward but undoes some of the advantages originally introduced by the method.
The next steps therefore include using the reference graph in the alignment step and creating a BWT-based implementation that works on more general problems in a fashion more closely resembling the problem in the real world.

A small preview of this thesis and some of the source codes created for it can be found at http://tomschiller.de/graphalign