SUBPATH CONSTRAINTS

for Minimum Flow Decomposition in strain-aware full-genome viral haplotype assembly

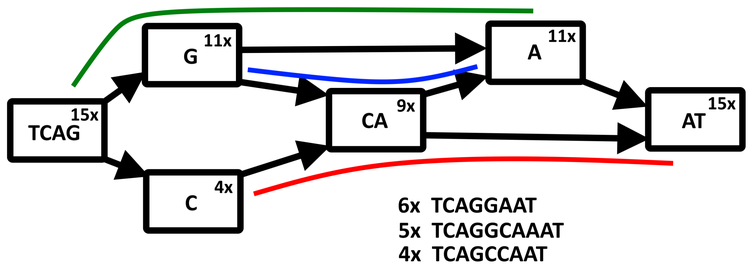
Jona Bedaux
CSE3000 RP
28/01 2026
Supervisor: Jasmijn Baaijens

TU Delft

Introduction

Viruses mutate so often that many different viral particles with different RNA are present in an infection. Tracking these **haplotypes** is useful to improve treatment or study viral evolution. But sequencing technology can only read small parts of genomes, without knowing which haplotype a part is from. We focus on one method to recover the full haplotype genomes from these parts.

First, the parts are processed into larger sequences called **contigs**. Puzzling these contigs together will yield full genomes. We can build a **contig variation graph** (CVG), a directed graph where each node is an RNA sequence and nodes are connected if they appear sequentially in a contig. We also calculate the abundance for each node in the CVG. An s-t-path in the CVG gives a possible full genome. We want to find the smallest set of s-t-paths that covers the whole CVG to find the haplotypes, with weights adding up to the abundances on the nodes (Minimum Flow Decomposition). This is very hard, but we can help solvers along by restricting the solution paths to go through the contigs, since we already know those are correct. This is done by enforcing **subpath constraints** on MFD. The exact way we enforce those constraints is not yet researched in literature, and is the subject of my thesis.



Research question

What is the effect of different **subpath constraint formulations** on the performance of the ILP to solve MFD for viral genome assembly under **varying levels of noise**?

Subpath constraints formulations

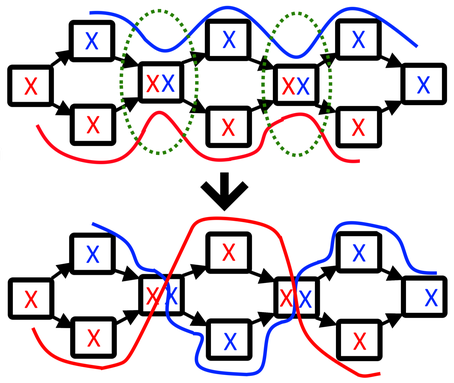
We formalize and implement (in *flowpaths*) the following subpath constraints formulations:

all-used (baseline)	All subpaths are fully part of at least one solution path.
p-used	A portion p of all subpaths are fully part of at least one solution path.
p-used-by-length	Same as p-used, but the portion is weighted by subpath length.
covered-p	All subpaths are for a portion p part of at least one solution path.
only-subpaths-p	All solution paths are built up entirely of portions of subpaths of at least p.

Generating noisy CVGs

We want to create CVGs with contig misassemblies in the subpaths, to test the effects of the subpath constraints formulations under noise. First, mutations are simulated from a random parent genome to get haplotypes. From these haplotypes we slice contigs and build a CVG.

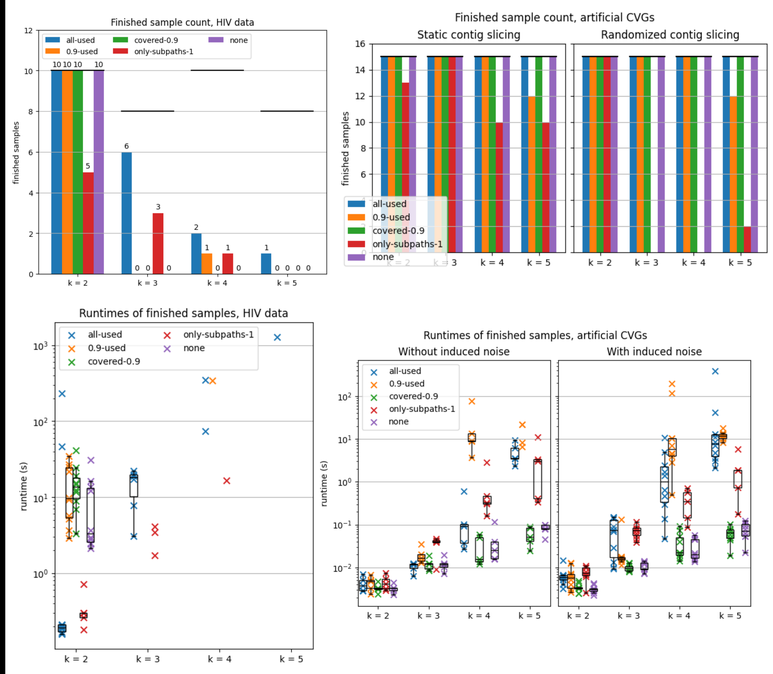
Then, we randomly select overlapping subpaths and identify their "overlap regions". The regions in between the overlap can be swapped to create two **chimeric** contigs, i.e. with pieces from different haplotypes.



Experiments

One dataset of realistic CVGs and one dataset of noisy CVGs are used to evaluate runtimes of the formulations on DelftBlue.

Results



Conclusion

From our test results we conclude that at the present, all-used is still the best formulation.
0.9-used performs worse than all-used in almost all scenarios, covered-0.9 and none have very similar performance to 0.9-used on HIV but underestimate k on the noisy CVGs. These are less suitable than the baseline.

The last formulation, only-subpaths-1, often leads to infeasibility (impossible to solve for any k). This is because some CVG nodes become unreachable for some subpaths. However, when it does converge the runtimes are better than for all-used.

We recommend more research into solutions for this problem, then only-subpaths-p may help improve MFD performance.