# ViPRA-Haplo: *de novo* reconstruction of viral populations using paired end sequencing data

Weiling Li, Raunaq Malhotra, Steven Wu, Manjari Jha, Allen Rodrigo, Mary Poss, and Raj Acharya, *Fellow, IEEE*

**Abstract**—Evolutionarily related viral strains subjected to high mutation and recombination rates during replication within host cells, characterize the genetic diversity. Thus, assembly of viral haplotypes provides important information for the development of clinical treatment and prevention. Next-generation sequencing (NGS) technologies bring more opportunities for sequencing viral populations. However, due to unknown number of haplotypes, high similarity between related viral strains, and sequencing errors in short reads during NGS, viral haplotype reconstruction is still a challenging task. We present ViPRA-Haplo, a *de novo* strain-specific assembly workflow for reconstructing viral haplotypes in a viral population from paired-end NGS data. We first proposed a scoring scheme for source-sink paths in a De Bruijn graph of reads using the pairing information of reads. The proposed Viral Path Reconstruction Algorithm (ViPRA) generates a subset of paths with high scores that serve as candidate paths for true contigs. The paths generated by ViPRA are an over-estimation of the possible contigs. We then proposed two methods to obtain an optimal set of contigs representing the viral haplotypes. The first method utlized a *de novo* clustering tool VSEARCH [1] to cluster the paths reconstructed by ViPRA based on the sequence similarity. The sequence with the highest score in each cluster represents a contig. Second, we proposed a method MLEHaplo that generates a maximum likelihood estimate of the viral populations using the candidate paths as a starting point. We evaluated our pipeline on both simulated and real quasispecies data. Experimental results show that ViPRA-Haplo outperforms the existing tools with higher genome covered fraction and retaining the diversity of the viral population. As severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) causes the recent COVID-19 pandemic, we extended experiments on a SARS-CoV-2 dataset sampled across the world. Our workflow can reconstruct contigs with the reference covered fraction larger than 95% and capture more sequencing variation.

**Index Terms**—*de novo* assembly, viral haplotype reconstruction, maximum likelihood estimation, heuristic path cover algorithm, COVID-19, SARS-CoV-2.

## 1 Introduction

VIRUSES replicating within a host exist as a collection of closely related genetic variants known as viral haplotypes. The diversity in a viral population, or quasispecies, is due to mutations (insertions, deletions or substitutions) or recombination events that occur during virus replication. These haplotypes differ in relative frequencies and together play an important role in the fitness and evolution of the viral population [2], [3], [4], [5]. The reconstruction of viral quasispecies from error-prone short reads is challenging for a number of reasons, including:

- The number of original haplotypes are usually unknown.
- It is hard to distinguish different strains with extremely small differences due to high similarity.
- The presence of sequencing errors in short reads in particular confounds the true mutational spectrum in the viral population because viral error rates and sequencing error rates are similar.

There are two groups of methods to reconstruct haplotypes in a population. The first group of methods use a high-quality alignment of short reads to a reference viral genome (See [6], [7] for review). Reference based reconstruction of viral haplotypes can be performed either locally or globally. Local haplotype estimation involves inferring haplotypes in short segments along a viral reference sequence to which the reads are aligned [8], [9]. Error correction is generally performed before [10], [11], [12], [13] or along with local haplotype estimation [8], [9], [14], [15]. Global estimation involves estimating viral haplotypes over segments larger than the length of reads. For global estimation, the reads aligned to a reference or to a consensus sequence are generally arranged in a read-graph, where vertices in the graph are reads and edges denote overlaps amongst the aligned reads. As the number of haplotypes in a population is unknown, an optimal set of viral haplotypes that explains the read-graph is obtained. There are a number of optimization frameworks for analyzing the read-graph [16], for example by modeling it as a network flow problem [17], [18], minimal cover formulation as well as a maximum bandwidth paths formulation [19], and maximum clique problem [20]. Probabilistic models have also been explored

- *W. Li is with The School of Informatics, Computing, and Engineering, Indiana University Bloomington, Bloomington, IN, 47405, USA. E-mail: wli6@iu.edu*
- *R. Malhotra is with the GNS Healthcare, Cambridge, MA, 02139, USA. E-mail: raunaq.123@gmail.com*
- *S. Wu is with the BioConsortia, Davis, CA, 95618, USA.*
- *M. Jha is with the Microsoft, Redmond, WA, 98052, USA.*
- *A. Rodrigo is with The Bioinformatics Institute, The University of Auckland, Auckland, 1010, New Zealand.*
- *M. Poss is with The Department of Biology, Pennsylvania State University, University Park, PA, 16802.*
- *R. Acharya is with The School of Informatics, Computing, and Engineering, Indiana University Bloomington, Bloomington, IN, 47405, USA.*

*In Progress*

for local haplotype estimation [9], [11], global haplotype estimation [12], [19], [21], and for analyzing recombination amongst viral haplotypes [14], [22].

The reference-based methods rely on alignment of reads to a reference sequence for error correction, orientation of reads, and for reconstruction of the viral population. A reference sequence can be helpful when it is representative of all haplotypes present in the viral population. However, due to the presence of insertions and deletions, recombination and high mutation rates in some viral populations (e.g. RNA viruses), a large percentage of the reads are unaligned to the reference genome, and are ignored while estimating viral diversity. In particular, lack of reference genomes during the breakout of emerging diseases, makes reference-based methods not plausible. Additionally, the reference based algorithms are known to have high false-positive rates in reconstruction of viral haplotypes and over-estimate the number of viral haplotypes in datasets with low sequencing diversity [23], [24], which is expected in a virus population obtained from an infected individual.

The second group of methods belongs to *de novo* assembly, providing an alternative to reference-based haplotype estimation. These methods have been widely used for assembly of reads from a single genome. Although there exist a number of *de novo* metagenomic assembly tools (en.wikipedia.org/wiki/De_novo_sequence_assemblers) that could be applied to haplotype assembly in viral population, these generic methods are not designed specifically for the viral quasispecies assembly problem. There are *de novo* assembly of viral haplotypes where the viral haplotypes are reconstructed directly from the sampled reads. For viral populations, overlap-layout-consensus based *de novo* assembly methods have been used to generate a consensus viral sequence [25], [26]. Recently, Shiver combined *de novo* and reference-based approaches to reconstruct the consensus sequence [27]. However, these methods assemble consensus sequences and is not suitable for investigating the diversity of the viral population. Currently, there exist two *de novo* strain-specific assembly methods to characterize new viral strains or novel haplotypes. SAVAGE [28] and PEHaplo [29] use paired-end reads in the overlap graph to compute a set of strings called contigs - putative portions of the complete genome. SAVAGE merges short reads into contigs using cliques. PEHaplo utilizes paired end information to guide path-finding and contig refinement.

Our work also belongs to *de novo* strain-specific assembler. The differences between the above two methods and our work are based on two different graph models to build a graph representing overlap relations between reads or substrings of reads: overlap graph and De Bruijn graph. Overlap graph represents overlaps between reads; while De Bruijn graph is based on the notions of k-mers - a length $k$ substring of the reads, and its edge connects two k-mers with a common prefix and suffix of length k-1. De Bruijn graphs usually require less computational resources; at the meantime, its simpler structure can allow some more sophisticated arguments to be applied [30].

With the availability of paired-end sequencing data produced by Illumina, methods that explicitly incorporate paired end read information in an aligned read graph for reconstructing viral haplotypes have been explored [31], [32], [33]. Reconstructing the viral haplotypes in a population from read graphs would involve simultaneously assembling multiple paths that collectively visit every vertex (a graph *cover*) and each path represents a single viral haplotype in the population. The problem of simultaneous assembly from short reads in a single individual is computationally intensive and challenging even for estimating the two paths corresponding to diploid haplotypes in the individual [34]. Evaluation of haplotype reconstruction tools has shown that it is a computationally challenging problem ( [6], [35]). The problem of reconstructing a minimal set of haplotypes under the constraints of paired-end reads from a read-graph or a De Bruijn graph is NP-hard [36], [37]. Thus, only heuristic algorithms for estimating viral haplotypes from paired-end sequencing reads are possible [36].

In this paper, we present a workflow using De Bruijn graph based *de novo* assembly for estimating viral haplotypes using paired-end sequencing data. The main advances made in this paper are (i) we utilize the pairing information of the paired reads to compute a score for paths in a De Bruijn graph that is generated from the overlaps in the reads, (ii) we develop a novel polynomial time heuristic algorithm, Viral Path Reconstruction Algorithm (ViPRA), that generates $M-$ paths corresponding to top $M$ scores through every vertex in the De Bruijn graph, and (iii) we utilize VSEARCH [1] to perform *de novo* clustering by similarity between paths from ViPRA, and propose an algorithm MLEHaplo that provides a maximum likelihood estimate of the viral population based on the proposed generative model.

We evaluate our algorithm on simulated datasets with sequencing errors, and discuss about the parameters. Our goal is to recover the viral population accurately with a high identity to the reference. Our results demonstrate that ViPRA-Haplo is able to retain the diversity of the viral population, and reconstruct near full length haplotypes. The presence of sequencing errors affects the recovery of viral haplotypes and error correction methods for viral population are essential for improving the performance of ViPRA-Haplo. ViPRA-Haplo does not have a bias towards the reference sequence which occurs in the reference based methods and we compare our results to existing *de novo* assembly based methods in the real datasets (HIV-1 quasispecies data and SARS-CoV-2 data), and observe that viral haplotype estimation in ViPRA-Haplo can reconstruct more contigs with higher genome fraction on the references and capture more variation observed in the reads.

## 2 Methods

### 2.1 Definitions

A viral population $\mathbf{H}$ is a collection of viral haplotypes $\{H_1, H_2, \ldots, H_P\}$, where each haplotype $H_i$ is a string of length $GS_i$ defined over the alphabet $\Sigma = \{A, G, C, T\}$. The sequence similarity between any two haplotypes $H_i, H_j \in \mathbf{H}$ is assumed greater than 90% as they are all generated via mutations and recombinations on a recent common ancestral sequence. A paired read $(R_f, R_r)$ is a pair of two strings over the alphabet $\Sigma$ that are sequenced using NGS technologies from an $IS$ length segment of haplotype $H_i \in \mathbf{H}, i \in \{1, 2, \ldots, P\}$. The collection of paired reads sampled from the viral population $\mathbf{H}$ is denoted as $\{(R_{1f}, R_{1r}), (R_{2f}, R_{2r}), \ldots, (R_{nf}, R_{nr})\}$. The statistics of insert length $IS$ for the NGS reads are known, although the insert length for a particular paired read is unknown.

A substring $u$ of length $k$ from $H_i, R_f$, or $R_r$ is known as a $k$-mer and the reverse complement of the $k$-mer $u$ is denoted as $\bar{u}$. The number of times a $k$-mer $u$ is observed in the sampled reads is known as count of the $k$-mer $u$. The two $k$-mers spanning a $(k+1)$ length string in a read are known as consecutive $k$-mers. A De Bruijn graph is a representation of the sampled reads and consists of $k$-mers as its vertices. Directed edges are drawn in the graph between consecutive $k$-mers from the first to the second $k$-mer. The vertices representing consecutive $k$-mers are known as consecutive vertices, and a sequence of consecutive vertices in the graph is known as a *path*. A path starting at a vertex with no edges into it (*source* vertex) to a vertex with no edges going out of it (*sink* vertex) is known as a *source-sink* path in the graph.

## 2.2 The whole pipeline of ViPRA-Haplo

De Bruijn graph based methods are amenable to address the problem of recovering viral haplotype diversity from shotgun or amplicon sequence data if the challenges discussed above can be solved. The five main components of ViPRA-Haplo are shown in Fig. 1 (c-g). In the first pre-processing stage, for the real data sets, reads with low-quality or ambiguous base calls are filtered or trimmed using Trimmomatic [38]. As sequencing errors increase the complexity in a De Bruijn graph, errors or indels are corrected using alignment-based error correction, and store the $k$-mers from the error corrected reads in the De Bruijn graph $G$, where the vertices are $k$-mers and consecutive $k$-mers in a read form an edge. The paths in the graph starting at a *source* vertex and ending in a *sink* vertex are candidates for viral haplotypes. In order to prune through the millions of *source-sink* paths in the graph, we store in a paired set $PS$, the pairing information of $k$-mers observed in the error corrected reads along with their numbers of occurrence. The detailed pre-processing description can be found in Appendix A.

Our proposed Viral Path Reconstruction Algorithm (ViPRA), a heuristic polynomial time algorithm, uses a parameter $M$ and the evidence from the paired set to retain a small number of *source-sink* paths as candidate haplotypes. We limit the over-estimation of number of haplotypes in the viral population by proposing two methods. The first method applies a *de novo* clustering tool VSEARCH [1] on the paths reconstructed by ViPRA and reports the representative(s) with the highest path score for each cluster as a contig. We also proposed a maximum likelihood estimator MLEHaplo for estimating the likelihood of a set of sequences in the viral population using the paths reconstructed by ViPRA as putative candidates for these sequences.

## 2.3 Scoring *source-sink* paths in the graph using the set $PS$

Consider a path $P = (u_1, u_2, u_3, \ldots, u_r)$ in the graph $G$, where $u_1$ is the *source* vertex, $u_r$ the *sink* vertex. The vertices $(u_1, u_2, u_3, \ldots, u_r)$ correspond to the $k$-mers in the sampled

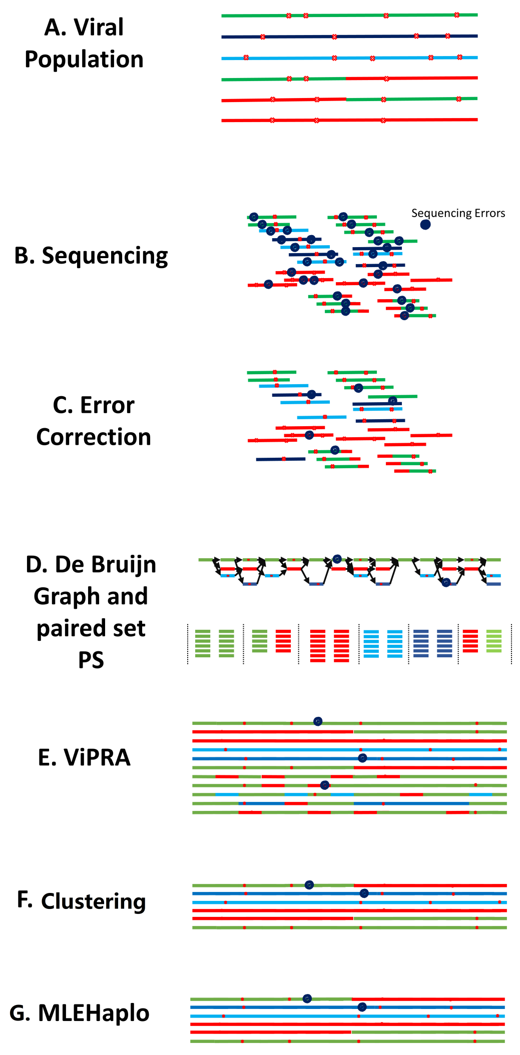

Fig. 1. **Reconstruction of viral haplotypes using paired-end data:** (A) A viral population example consisting of four viral haplotypes (red, green, dark blue, blue lines) with mutations (red dots) and a recombinant haplotype (red-green line) depicted. (B) Sequencing technology generates paired-reads with sequencing errors (black dots on short segments from reads). (C) Error correction is implemented using karect [39], which uses alignments between reads, without a reference genome. (D) The error-corrected reads are stored in a De Bruijn graph. The pairing information of a paired read constitutes all $k$-mer pairs observed in it, and it is stored in the set $PS$. Colors denote $k$-mers obtained from the corresponding viral haplotypes in (A). (E) The proposed heuristic algorithm ViPRA reconstructs only a fraction of the total paths in the De Bruijn graph. Those paths have a high score that measures the support of $k$-mer pairs found in the set $PS$. (F) A *de novo* clustering, VSEARCH [1] is applied on the paths that ViPRA over estimates. The sequence with the highest score in each cluster represents one contig. (G) MLEHaplo reconstructs a maximum likelihood estimate of the viral population, where the sampled reads are modeled to be sampled from a viral population consisting of paths from either Step E (ViPRA) or Step F (Clustering) as a starting point.

reads. Assuming that all paired $k$-mers from the viral population separated by insert size distance $IS$ are sampled in the observed reads, we define a score $S(P)$ for the path $P$ using the elements of the paired set $PS$ as follows:

$$S(P) = \frac{1}{E(P)} \cdot \sum_{(r,s) \in P \cap [d(s)-d(r)<IS]} \{\mathbb{1}[(r,s) \in PS] - pen \cdot \mathbb{1}[(r,s) \notin PS]\} \tag{1}$$

The score for the path $P$ is defined over vertex-pairs $\{(r,s); r \in P \;\&\; s \in P\}$ that are within distance $IS$ on the path $P$. It is proportional to the number of vertex-pairs present in $PS$. The score is also penalized by a penalty term $pen$ for every vertex-pair in $P$ within $IS$ that is not present in $PS$, as under assumption of high coverage, all pairs of $k$-mers from true viral haplotypes within distance $IS$ would have been observed in the paired reads. The score is normalized by $E(P)$, the expected number of vertex-pairs in a path $P$ that are within the insert size $IS$.

Paths in the graph that have high scores are candidates for possible viral haplotypes (See Supplementary Text Section 2 for rationale). Moreover, a path *cover* of the graph consisting of such paths is a candidate for the viral population $\mathbf{H}$. Equation (1) can be modified in the presence of sequencing errors, wherein, instead of membership in $PS$ within a distance $IS$, the contribution of a vertex pair $(u,v) \in PS$ is weighted by its relative frequency of occurrence.

### 2.4 ViPRA: A heuristic algorithm for estimating top $M$ paths per vertex

We propose Viral Paths Reconstruction Algorithm (ViPRA), a heuristic algorithm that computes a path *cover* of the graph using high scoring $M$ paths per *source* vertex in the graph $G$. ViPRA computes $M$ high scoring paths through a vertex using precomputed $M$ high scoring paths through the vertex's neighbors. It starts with the *sink* vertices in the graph and iteratively builds on them using memorization of $M$ paths through each vertex to generate high scoring *source-sink* paths in the graph. This is possible as the score for a path $P$ can be constructed using the scores of its sub-paths starting at a particular vertex to the end (See Algorithms 1, 2 and time complexity analysis in Supplementary Text Section 3).

The goal of the scoring mechanism $S(P)$ is to ensure that the paths corresponding to true haplotypes have a high score and thus they are retained in the top paths for a vertex as the algorithm propagates from the *sink* vertex to the *source* vertex. Thus, the choice of $M$ is important as it would affect the number of paths generated per *source* vertex. For vertices that are not spanned by paths from the *source* vertices, additional $M$ paths are generated through these vertices.

### 2.5 *De novo* clustering by VESEARCH

The path *cover* generated by ViPRA is an over-estimation of the possible paths that represent the viral population. Uncorrected sequencing errors inflate the number of the ViPRA paths. The first method of building the viral haplotype is employing VSEARCH [1] to perform *de novo* clustering on the candidate paths reconstructed by ViPRA. VSEARCH is a centroid-based algorithm with a sequence similarity threshold specified with the *id* option [1]. It can perform in parallel. In each cluster, the sequences with the highest score $S(P)$ represent contigs.

### 2.6 Maximum Likelihood Estimate of the viral population

We proposed the other method for constructing the viral haplotypes by generating a maximum likelihood estimate of the population using the ViPRA paths as a starting point.

The generative model for sampling paired reads from the viral population $\mathbf{H}$ is as follows: For a paired-read $(R_f, R_r)$ of insert length $IS$, it can be sampled from only a single location in the viral haplotype under the assumption that there are no long repeats in viral haplotypes. Thus, the probability of observing the paired read $(R_f, R_r)$ from the viral population $\mathbf{H}$ can be expressed as the ratio of number of haplotypes that share such a read segment to the total number of locations from which any paired read of length $IS$ can be sampled:

$$P((R_f, R_r)|\mathbf{H}) = \frac{q}{P \cdot (GS - IS + 1)} = z_R \tag{2}$$

In this equation, $q$ is the number of haplotypes in $\mathbf{H}$ that have $(R_f, R_r)$ as their sub-string, $P$ is the number of haplotypes in $\mathbf{H}$, and $GS$ is the average length of the viral haplotypes. It should be noted that Equation 2 also holds if $(R_f, R_r)$ denotes a $k$-mer pair.

Thus, for a collection of paired reads $\{(R_{1f}, R_{1r}), (R_{2f}, R_{2r}), \ldots, (R_{nf}, R_{nr})\}$ where $(R_{if}, R_{ir})$ is sampled $c_i$ times, assuming independent sampling, the joint probability of observing the paired reads given the viral population $\mathbf{H}$ can be expressed as a multinomial expression:

$$P(\{c(R_{1f}, R_{1r}) = c_1, \ldots, c(R_{nf}, R_{nr}) = c_n\}|\mathbf{H}) = \frac{M!}{c_1! \cdot c_2! \ldots c_n!} \prod_{i=1}^{n} z_{R_i}^{c_i} \tag{3}$$

where, $M = \sum_{i=1}^{n} c_i$.

Given the above formulation, a maximum likelihood set of haplotypes $\mathbf{H_{ml}}$ can be estimated using equation 3 as follows:

$$\mathbf{H_{ml}} = \max_{\forall \mathbf{H}} P(\{c(R_{1f}, R_{1r}) = c_1, \ldots, c(R_{nf}, R_{nr}) = c_n\}|\mathbf{H}) \tag{4}$$

where the maxima is computed over all possible subsets of $\mathbf{H}$ generated by ViPRA.

### 2.7 Backward Elimination for estimating $\mathbf{H_{ml}}$

The top $M-$paths through all the *source* vertices are used as candidates for possible haplotypes for computing the maximum likelihood set of haplotypes using the backward elimination algorithm. The likelihood computation begins with all the top $M-$paths through the *source* vertices and iteratively remove one path from the set of all paths until the likelihood of the remaining paths in the set starts to decrease. The remaining set of haplotypes constitute a maximum likelihood estimate of the viral population.

'

## 3 Results

### 3.1 Simulation studies for genome reconstruction of viral populations

We use a simulated HIV-1 sequence data set adopted by Chen et al [29] for performance evaluation. The simulated dataset has 250 bp error-containing MiSeq reads from five HIV-1 reference strains with the average insert size of 600 bp and standard deviation of 150 bp [29]. The coverage for each strain in the simulated data set is: 89.6 - 2190x, HXB2 - 1095x, JRCSF - 730x, NL43 - 547x, and YU2 - 438x [29].

All the experiments were conducted on Indiana University's Carbonate high-performance computer cluster. Carbonate compute nodes run Red Hat Enterprise 7.x, each with 256 GB of RAM and two 12-core Intel Xeon E5-2680 v3 CPUs. ViPRA-Haplo used only one thread. We benchmark our work against PEHaplo because it outperforms SAVAGE and other benchmarked de novo viral assembly tools [29]. We use a tool MetaQUAST [40] to compare our method with PEHaplo. The tool partitions all contigs into groups aligned to each reference genome using only the very best set. Note that a contig may belong to several groups simultaneously if it aligns to several references [40].

#### *3.1.1 Evaluation of ViPRA with varying M*

Our algorithm ViPRA prunes through millions of paths in the graph to generate a set of paths having a high paired-end support score, as explained below, that form candidates for viral haplotypes. It takes as input a parameter $M$, the De Bruijn graph $G$, and the paired set $PS$ for a given sequencing data. The output of ViPRA is a path *cover* of the graph containing $M$ *source-sink* paths for each *source* vertex in the graph $G$. The $M$ paths per *source* vertex are ranked based on their support present in the paired set $PS$. A negative penalty is applied to a path when there is no support found for its $k$-mer pairs within insert size $IS$ in the set $PS$ and we only consider paths with positive scores for evaluation. Additional $M$ paths are also generated through vertices not visited by any of the top $M$ paths generated from the *source* vertices to obtain the path *cover* of the graph.

We assess the number of paths reconstructed by ViPRA with the variation in parameter $M$ = (5, 10, 15). The number of paths recovered is independent of the total number of source-sink paths in G for a given dataset but is directly proportional to the parameter $M$ and overall complexity of the graph $G$ (Supplementary file S1). Partial length haplotypes are obtained for each value of $M$ indicating that not all vertices were visited by the *source-sink* paths. Nevertheless, the largest alignment is near full-length. We also evaluate the genome covered fraction, that is, the percentage of the reference genome covered by the aligned contigs. As $M$ increases, the coverage of the reference sequence increases. Supplementary file S1 suggests $M$ = 5 is sufficient for recovering the true haplotypes because it can recover near full-length haplotypes with the high genome covered rate (>98%) in each strain. The paths reconstructed at $M$ = 5 are used for the further reduction.

#### *3.1.2 Evaluation of VSEARCH clustering*

VSEARCH [1] is a centroid-based *de novo* clustering with a prespecified sequence similarity threshold $id$. We applied the VSEARCH tool to reduce the number of paths reconstructed from ViPRA. The sequence with the highest score $S(P)$ in each cluster represents a contig. We evaluated VSEARCH -id = (0.99, 0.999, 0.995) after VIPRA on the simulated HIV-1 data. $id$ =0.995 is sufficient to reduce the name of ViPRA paths on the simulated dataset while maintaining a high genome covered fraction (Supplementary file S1).

#### *3.1.3 Evaluation of MLEHaplo*

Our algorithm MLEHaplo generates a maximum likelihood estimate $\mathbf{H_{ml}}$ of the viral population using the paths reconstructed by ViPRA. It reduces the number of paths from ViPRA by iteratively removing one haplotype by backward elimination. Because the number of paths generated from ViPRA is large, we here use VSEARCH clustering to generate the input for MLEHaplo. Table 1 shows ViPRA-Haplo can achieve high genome covered fraction (>97%) of each HIV-1 strain.

TABLE 1
**Assembly results per method on a simulated HIV data set for each HIV-1 strain and the combined reference (concatenation of all the 5 strains)**

| Tools | 89.6 (9712 bp) | HXB2 (9719 bp) | JRCSF (9535 bp) | NL43 (9709 bp) | YU2 (9706 bp) | Combined |
|---|---|---|---|---|---|---|
| PEHaplo | | | | | | |
| Contigs num | 1 | 3 | 1 | 6 | 1 | 12 |
| Genomes covered (%) | 99.897 | 100 | 95.438 | 99.485 | 93.509 | 97.675 |
| Largest alignment (bp) | 9,662 | 8,711 | 9,292 | 4,541 | 9,274 | 9,662 |
| ViPRA-Haplo | | | | | | |
| Contigs num | 100 | 81 | 53 | 41 | 46 | 315 |
| Genomes covered (%) | 98.219 | 100 | 97.997 | 97.94 | 100 | 97.741 |
| Largest alignment (bp) | 9,466 | 9,703 | 9,234 | 8,657 | 10,967 | 10,374 |

#### *3.1.4 Comparison to PEHaplo*

We reconstructed the simulated reads with ViPRA-Haplo and PEHaplo, and the results are summarized in Table 1. In total, PEHaplo generated 12 contigs, and the genome covered fraction is 97.675%; while ViPRA-Haplo generates 315 contigs, and the genome covered rate is 97.741% (Table 1). Both methods reconstruct near full length haplotypes. ViPRA-Haplo generated some contigs longer than the reference, which reflects the repeat sequence flanking the HIV-1 genome. The most notable difference between the two methods is on the reconstruction of NL43. These comparisons show ViPRA-Haplo has higher genome coverage and longer contigs compared to PEHaplo. PEHaplo took minutes to finish the experiment, while ViPRA-Haplo took hours. In ViPRA-Haplo, the paired set generation is the most time-consuming, which took 5 1/2 hours while taking 64 GB of memory at its peak.

### 3.2 Evaluation on HIV MiSeq data

We evaluate the performance of ViPRA-Haplo on the real sequencing data from NCBI with the accession number SRR961514, where there are additional sources of errors during sample preparation. A laboratory mixture of five known Human Immunodeficiency Virus-1 (HIV-1) strains was sequenced using Illumina paired end sequencing. The dataset contains around 715K pair reads with 250 bp read lengths that cover each strain to 4,000x on average. To minimize the number of contigs and sequencing errors while maintaining a high genome fraction, we report $M$ = 5, VSEARCH -id=0.99 and contigs are at least 500 bp from ViPRA-Haplo reconstruction. ViPRA-Haplo generates more contigs with higher genome covered fraction and longer contigs than PEHaplo (Supplementary file S3, and Table 2). The K-mer counts reference viral strains 89.6 and HXB2 are lower compared to the other strains (see Supplementary Fig. 1). However, there are regions where the k-mers across all the strains are identical (indicated by the peaks in the K-mer counts). This could explain why Vipra-Haplo is unable to reconstruct a single long strain for 89.6 and HXB2 as the common k-mers led to selection of an alternate path in the De Bruijn graph. In addition, both PEHaplo and ViPRA-Haplo assembly results can preserve almost all the unique k-mers (60-mers) with kmer counts $> 1000$ from the sequencing data. PEHaplo assembly results only preserve 9.27% of k-mers whose counts $<= 1000$, while ViPRA-Haplo can preserve 28.38% of them. It implies that PEHaplo loses the diversity in lower count. The total running time for ViPRA-Haplo is around 15 hours with 64 GB of memory.

TABLE 2
**Comparisons of results to PEHaplo [29] on a real HIV data set**

| Tools | 89.6 (9712 bp) | HXB2 (9719 bp) | JRCSF (9535 bp) | NL43 (9709 bp) | YU2 (9706 bp) | Combined |
|---|---|---|---|---|---|---|
| PEHaplo | | | | | | |
| Contigs num | 8 | 5 | 11 | 16 | 5 | 45 |
| Genomes covered (%) | 94.357 | 74.586 | 95.438 | 94.356 | 91.222 | 89.969 |
| Largest alignment (bp) | 7,632 | 5,013 | 7,401 | 6,150 | 3,949 | 7,632 |
| ViPRA-Haplo | | | | | | |
| Contigs num | 99 | 53 | 108 | 88 | 70 | 401 |
| Genomes covered (%) | 95.923 | 98.477 | 100 | 100 | 97.455 | 97.954 |
| Largest alignment (bp) | 5,814 | 4,994 | 8,893 | 8,429 | 8,513 | 8,893 |

## 3.3 Evaluation on SARS-CoV-2 samples across the world

Coronavirus disease 2019 (COVID-19) is a newly emerging and devastating respiratory disease currently spreading across the world. It is caused by a novel coronavirus, severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). Although the substitution rate of SARS-CoV-2 appears to be lower than that of HIV-1, the emergence of viral variants is complicating public health measures to slow transmission and vaccine development efforts. We assess the ViPRA-Haplo pipeline on SARS-CoV-2 samples sequenced using the Illumina Amplicon Technology (See Supplementary file S4 for SRA numbers). Our dataset includes 40 samples obtained from across the world sequenced between March 2020 to February 2021. We mapped contigs reconstructed with ViPRA-Haplo (settings -M=5 and VSEARCH -id=0.9) on the NCBI Reference Sequence, NC_045512.2 (Wuhan reference, 2019, 29,903 bp) [41]. The genome covered fractions for those samples range from 95.576% to 99.873% (Supplementary file S4).

The coronavirus SARS-CoV-2 entry into host cells is mediated by the spike glycoprotein (S-glycoprotein), interacting with human angiotensin-converting enzyme 2 (ACE2) receptors [42]. Thus, the spike (S) protein plays key roles in neutralizing antibodies and became the main target in vaccine and therapeutic design [43], [44], [45] in response to the COVID-19 pandemic. Mutations in the spike protein are identified for each sample (Supplementary File S4). 31 out of 40 samples have a mutation at the site 23,403 (A to G) of the NCBI reference. This mutation has been identified as D614G (A23403G), which has widely circulated and become the most prevalent variant in the global pandemic [46]. With the spread of coronavirus, more variants of concern were detected in the samples sequenced in 2021. Four mutations (G21600T, G22018T, T22917G, A23403G) found in the spike protein in SRR13744684 (February 2021, Chan Zuckerberg Biohub) are linked to the b.1.429 (Epsilon) lineage, which has been emerged in California since September 2020 [56]. The prevalence of variant detected among sequenced samples over time and between geographical locations is consistent with the SARS-CoV-2 virology and epidemiology. Moreover, this study shows the merit of our method is preserving the sequencing variation (Fig. 2).

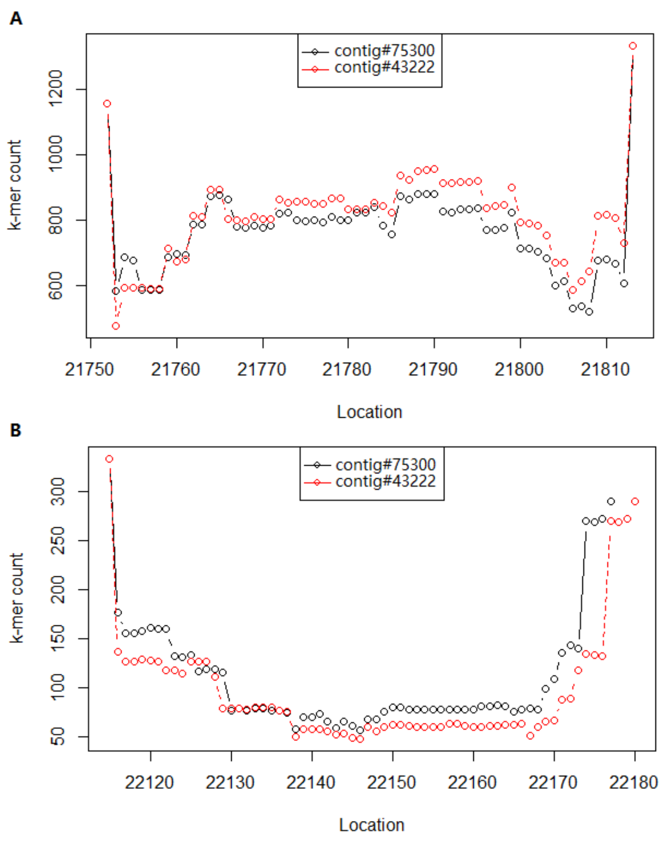

Fig. 2. **Plot of k-mer (k=60) counts for reconstructed contigs #75300 and #43222 from ERR4440368 in two regions in the spike protein.** Contigs #75300 and #43222 have the same sequences in the spike protein except the above two regions (A) & (B). These two variations have similar k-mer counts. It implies that our ViPRA-Haplo workflow can capture the sequence variation. The sequences of Contig #43222 in the spike protein are identical to the NCBI reference and Contig #75300 includes two mutations in the spike protein: C21812T, and deletions in 22,175-22,177.

## 4 Discussion

We have proposed ViPRA-Haplo, a De Bruijn graph based *de novo* assembly pipeline for viral haplotypes that uses paired-end NGS data to reconstruct the viral population. Reads are represented in a De Bruijn graph and their pairing information is stored as pairs of $k$-mers in the set $PS$. The support found in the set $PS$ for pairs of vertices is used to score *source-sink* paths, and the scoring mechanism ensures that the paths represent possible viral haplotypes. As reconstructing a minimal cover of the graph under paired constraints is NP-hard, we have proposed a polynomial time heuristic algorithm, ViPRA, that recovers a small fraction of the total number of paths in the graph. Then, we proposed two methods to obtain the optimal set of contigs representing different viral haplotypes. The first method is to apply VSEARCH [1] for clustering the paths generated by ViPRA based on the sequence similarity and the sequence with the high path score in each cluster represents a contig. We also propose a second method, MLEHaplo for reconstructing a maximum likelihood estimate of the viral population using the paths recovered by ViPRA based on a generative model for sampling paired reads from the viral population.

The usage of De Bruijn graphs for viral haplotype reconstruction has a number of advantages: The $k$-mers as vertices avoids the costly computations of reads overlaps. Also, because the De Bruijn graph construction is *de novo*, it contains all the variation observed in the viral population which can be lost by aligning reads to a reference genome. The current method relies on generating an acyclic De Bruijn graph for estimating the viral haplotypes. Choosing $k$ (i.e., $k$=60) greater than $D$, the size of the largest repeat in the viral genome, as repeats in RNA viruses are generally short, acyclic De Bruijn graphs can be readily obtained for viral populations. For larger repeats, we can further use the reads only in the forward orientation, ensures that the De Bruijn graph obtained is acyclic.

However, the De Bruijn graph is sensitive to the presence of errors, and an efficient error correction algorithm is essential. Error correction using the error correction software Karect [39] in addition to removal of *source-sink* paths that had low coverage at their ends aided in the reconstruction of paths from the algorithm ViPRA. The concept of removing low coverage paths from the graph has been used earlier in whole genome assembler SPADES [47] where tips and bubbles are collapsed using similar techniques. Moreover, the availability of error detection methods for viral population that reduce the false positive number of $k$-mers while retaining the true $k$-mers will be essential for accurate estimation of viral haplotypes in the viral population [48].

The set $PS$ stores the $k$-mer pairs observed in paired reads and it can also combine pairing information from multiple overlapping paired reads that are within the insert size $IS$. Efficient storage and computation are possible using binary representation for k-mers and parsing the reads in multiple iterations [49], [50]. Additionally, this computation needs to be only performed once for a given sequencing dataset. The usage of pairing information for reconstruction of paths from De Bruijn graphs has been shown to improve performance for single genome assembly [47], [51]. We have used a simplistic version where the number of occurrences of a pair of $k$-mers is the only information required for the path scoring. The presence of variable insert size mate-pair data or longer length reads can provide additional support for $k$-mer pairs in the paired set and further improve the performance of ViPRA. As the paired set computation is quadratic in the length of the reads, it may not be practically feasible to generate the paired set for longer length reads.

The proposed scoring mechanism for paths in the De Bruijn graph ensures that paths corresponding to the true viral haplotypes have a high score. Under the assumption of *sufficient* coverage across a viral haplotype, a path in the De Bruijn graph corresponding to a true viral haplotype should have high paired $k$-mer support within the insert size $IS$ in set $PS$ and thus will have a high score. Penalizing the score for $k$-mer pairs that are missing only within the insert size length $IS$ is essential in discarding false positive paths, as the absence of such $k$-mer pairs has a low probability under *sufficient* coverage. On the other hand, we do not expect to see $k$-mer pairs that are at distances greater than insert size due to our sequencing process, and thus such $k$-mer pairs are not penalized. This ensures that when ViPRA retains the $M$ top scoring sub-paths at a vertex these sub-paths correspond to segments of the true haplotypes and as ViPRA propagates to the *source* vertices of the graph, ViPRA reconstructs paths corresponding to the true haplotypes.

The number of paths generated by ViPRA is dependent on the choice of the parameter $M$. The choice of $M$ is dependent on the inherent diversity of the viral population, which can be inferred by the number of vertices in the De Bruijn graph. The relative ratio of the counts of occurrences of most frequent $k$-mers to the average sequencing coverage can be used as a lower bound for determining $M$. As the most frequent $k$-mers are likely to be shared amongst the viral haplotypes in the population, such an $M$ is the minimum that is required for ViPRA to reconstruct all viral haplotypes that share this $k$-mer's vertex. We experimentally determined that M = 5-10 was sufficient for reconstructing all the true viral haplotypes in the simulated datasets and real datasets, even in the presence of sequencing errors.

To reduce the path generated from the ViPRA step, VSEARCH [1] is applied to do *de novo* clustering to quickly identify similar sequences. In each cluster, the sequence with the highest path score represents a contig. VSEARCH -id $\geq$ 0.9 is sufficient in the experiments. The higher the $id$ value is, the more ViPRA paths are reserved. To further reduce the reconstructed genomes with the low scores, MLEHaplo generates a maximum likelihood set of paths using the set of the ViPRA paths or its subset according to representatives in VSEARCH clustering. As the number of viral haplotypes in the population is unknown, MLEHaplo uses a backward elimination optimization that iteratively removes a path from the set of paths reconstructed by ViPRA such that the likelihood of the sampled paired reads under the remaining set of paths increases. Thus, MLEHaplo further reduces the number of paths reconstructed from the graph thereby reducing the false positive paths at convergence. The advantage of MLEHaplo is that all the sampled paired reads are explained by one of the reconstructed paths at convergence. However, it is sensitive to the paths reconstructed by ViPRA and can only improve on this set of paths. Additionally, as a path removed by backward elimination once is never considered again, it leads to the retention of haplotypes that are close to the true sequences but do not affect the likelihood. Removing one path iteratively makes MLEHaplo a time-consuming step, and parallelization of the code can improve the speed of computation. The partial length paths are generated due to a drop in sequencing depth of reads. This leads to termination of paths in the De Bruijn graph and an inflation in the number of haplotypes estimated by MLEHaplo.

We have tested the ViPRA-Haplo pipeline on a simulated HIV-1 dataset with sequencing errors, ViPRA-Haplo reconstructs near full-length contigs with the high genome covered fraction for the true haplotypes, as well as retains the diversity of the viral population. We also evaluated on a real dataset of HIV-1. The decrease in the genome covered fraction in the case of real HIV-1 strains is understandable as the reads are generated from replicating viruses where one would observe additional sequence variation with respect to the reference. This has been documented as single nucleotide polymorphisms with respect to the reference in the original study [52] and as an increase in sequence alignment score in a previous method [53]. Other than the high genome covered fraction, the other advantage of ViPRA-Haplo over the existing methods is that it retains the variation observed in the viral population, even when the reconstructed paths are not an exact match to the true viral haplotypes; while PE-Haplo only reconstructs representatives. When ViPRA overestimates the number of haplotypes, they can be clustered together by VSEARCH, and further reduced their number by MLEHaplo. Thus, the viral diversity can be correctly inferred from the reconstructed paths. Nevertheless, the paths chosen in the De Bruijn graph are consistent with the k-mer count plot; and the contigs reconstructed by ViPRA-Haplo better explained the sequenced reads. For short contigs, there are tools to reconstruct viral haplotype from preassembled contigs [54], [55]. When testing on the real SARS-CoV-2 data, ViPRA-Haplo can retain the sequencing variation, which is crucial for biomedical scientist to track how the coronavirus changes as it spreads.

The software implementation for ViPRA-Haplo is available at https://github.com/lwl1112/ViPRA-Haplo.

## Appendix A
## Pre-processing of reads: Error Correction, De Bruijn Graph Construction and Paired Constraints Set

### A.1 Error Correction

Error correction software Karect [39] was used for error correction, which is based on the high-coverage information, typically improves *de novo* assembly. We perform additional error-correction when reconstructing paths in the De Bruijn graph. A path that either begins or ends in vertices corresponding to $k$-mers that have $k$-mer counts less than the threshold (also known as tips in the De Bruijn graph) are removed. These vertices typically correspond to sequencing errors, and this technique has parallels in existing softwares for assembly of single genomes [47]. Other error correction methods, specifically developed for viral population reconstruction [48], can also be used for pre-processing the reads and removing sequencing errors.

### A.2 De Bruijn Graph

The $k$-mers from the error corrected paired reads are represented in a De Bruijn graph. As the orientation of the sequenced reads are unknown, $k$-mers from the paired reads and their reverse complements are stored in the graph. In order to reduce storage space, the De Bruijn graph $G$ is converted into a compacted graph $G_c$ in which linear chains of vertices are condensed into a single vertex, while preserving the edge relationships in the graph $G$. This technique has also been used in the assembler SPADES for single genomes [47].

For viral populations, especially for RNA viruses, it is possible to obtain an acyclic De Bruijn graph for reasonable values of $k$ (around 50-60) as the length of repeats in the viral genomes are typically small (except for terminal repeats in some viral genomes).

Two properties of a directed acyclic De Bruijn graph $G$ or the compacted graph $G_c$ are particularly useful for reconstructing sequence of viral haplotypes. First, the insert size $IS$ of a particular paired read can be computed using the distance $d(u, v)$ between two vertices $u$ and $v$ corresponding to the beginning and end of the paired read. This distance $d(u, v)$ is defined as the length of strings spelled by the shortest $u - v$ path in the graph, which can be computed uniquely for a directed acyclic graph. Second, a haplotype of the viral population is spelled by a *source-sink* path in the graph $G$ or $G_c$. A collection of such *source-sink* paths that spans all vertices (a path *cover*) represents the observed viral population.

### A.3 Paired constraints set (PS)

A Paired Constraints Set $PS$ is generated from the paired reads and contains a list all pairs of $k$-mers observed in the paired reads along with the number of times a $k$-mer pair is observed. The set $PS$ for graph $G_c$ can be computed once in time linear in the number of reads and quadratic in the length of the reads. As there are millions of *source-sink* paths in a typical graph for viral populations, the pairing information of the paired reads is used to guide their selection. The elements of the paired set $PS$ indicate whether a pair of $k$-mers appear together in one of the viral haplotypes in the population.

## Appendix B
## Rationale that scoring mechanism selects paths that correspond to viral haplotypes in the population

The $k$-mer pairs in the set $PS$ is a summarization of the observed paired reads and as the scoring of paths is based on the set $PS$, it is useful in selecting *source-sink* paths that represent the viral population. As defined in the text, the score $S(P)$ of a path $P$ is defined as :

$$\begin{aligned} S(P) = \frac{1}{E(P)} \cdot \\ \sum_{(r,s)\in P\cap[d(s)-d(r)<IS]} \mathbb{1}[(r,s) \in PS] \\ \sum_{(r,s)\in P\cap[d(s)-d(r)<IS]} -pen \cdot \mathbb{1}[(r,s) \notin PS] \end{aligned} \quad (5)$$

We provide a rationale for the paths corresponding to true viral haplotypes to have a high score $S(P)$.

**Definition 1** (Sufficient Coverage)**.** *The sampled paired reads are defined to have sufficient coverage of the viral population* $\mathbf{H}$ *if there exists paired reads* $(R_f, R_r)$ *that sample every haplotype*

*$H \in \mathbf{H}$ and there exists a paired read $(R'_f, R'_r)$ that samples a pair of k-mers $(u_i, u_j) \in H$ with $d(u_i, u_j) < IS$.*

Given *sufficient* coverage and the definition of the paired set $PS$, if two vertices $u_i$ and $u_j$ are present in a paired read, then a path $P$ that contains both vertices $u_i$ and $u_j$ can be a possible viral haplotype. Using *sufficient* coverage, we can show that paths corresponding to the true viral haplotypes in $\mathbf{H}$ have a path score $S(P) = 1$. Thus in order to extract paths from the graph $G$, it is sufficient to focus on the paths with high scores.

**Theorem 1.** *Given sufficient coverage of the viral population $\mathbf{H}$, for a path $P$ in the graph $G$ that corresponds to a viral haplotype in $\mathbf{H}$, the score $S(P) = 1$.*

*Proof.* The proof can be broken into two parts:

1. For every viral haplotype $H_i \in \mathbf{H}$, there exists a path $P$ in the graph $G$, and
2. The score $S(P)$ for such paths is 1.

As *sufficient* coverage implies that all the viral haplotypes $H_i \in \mathbf{H}$ are sampled by the paired reads, for a given viral haplotype $H$, it follows that there exists vertices $u_j$ in the graph $G$ that are sampled from the viral haplotype $H$, which implies that a path $P = (u_1, u_2, \dots, u_m)$ , where $u_i \in H$, exists in the graph.

The score $S(P)$ for such a path, by definition in equation 5, is the summation over all pairs of vertices $(u_i, u_j) \in P$ that are within the distance $IS$. Again, by the definition of *sufficient* coverage, all pairs of vertices from a viral haplotype $H \in \mathbf{H}$ within distance $IS$ are sampled by some paired read, which implies that $(u_i, u_j) \in PS$. Thus the score $S(P)$ is :

$$S(P) = \frac{1}{E(P)} \sum_{(r,s) \in P \cap d(s,r) < IS} \mathbb{1}[(r,s) \in PS] = \frac{1}{E(P)} \cdot E(P) = 1 \quad (6)$$

□

## Appendix C
## Viral Path Reconstruction Algorithm (ViPRA)

Algorithm 1 describes the pseudo-code of ViPRA that reconstructs the top $M$ paths through every vertex in the graph. ViPRA starts by initializing the paths from all vertices $u$ to the *sink* vertex $u_{sink}$ to empty sets. The scores for all paths from each vertex are also initialized to empty sets (Lines 1-4). Next it iterates over all vertices $u \in V_c$ in increasing order of their distance to the sink vertex ($d(u, u_{sink})$) to compute the top $M-$ paths through each vertex $u$ using the function TOP-M-PATHS-FOR-VERTEX($u, E_c, PS$) (Lines 5-10). When a graph has multiple *sink* vertices, a universal *sink* vertex is defined that has an edge from each of the *sink* vertex to it.

Algorithm 2 describes the memoized algorithm that computes the top $M-$ paths from a vertex $u$ to the *sink* vertex $u_{sink}$ ($TOP-M-PATHS-FOR-VERTEX(u, E, PS)$). It first recovers the top $M-$ paths

**Algorithm 1** ViPRA() : Top $M$ paths per vertex based on set PS and the graph $G$

**Input:** Directed De Bruijn graph $G(V, E)$, Set $PS$, d(.) the distance between vertex pairs in $G$.
**Output:** $Paths(u) = \{P_{u1}, P_{u2}, \dots, P_{uM}\}\ \forall u \in V$
$Score(u) = \{S_{u1}, S_{u2}, \dots, S_{uM}\}\ \forall u \in V$

```
1:  for each vertex $u \in V$ do
2:      $Paths(u) = [\emptyset]$ //Initialize the paths for each vertex
3:      $Score(u) = [\emptyset]$
4:  end for
5:  $Score(u_{sink}) = \{0\}$
6:  $Paths(u_{sink}) = \{``\_"\}$ // Place holder between vertices

7:  $N$ = SORT-INCREASING($V, D(.)$) // Sort vertices in
    increasing order of distance to the sink vertex and store
    it in $N$
8:  for $i = 1, \cdots, |N|$ do
9:      $[Paths(N[i], Score(N[i]))]$ =TOP-M-PATHS-FOR-
        VERTEX($N[i], E, PS$)
10: end for
```

from all the neighbors of $u$ having an incoming edge from $u$ into the array $TP$ and their scores in $SP$ (Lines 1-6). As the distance $d(u, u_{sink})$ is greater than the distance of its neighboring vertex $s$, ($d(u, u_{sink}) > d(s, u_{sink})$), the arrays $Paths(s)$ and $Score(s)$ have already been computed (Algorithm 1 in line 9). The score for each of the path in $TP$ is updated when adding the vertex $u$ to the path. The path scores are updated taking into account memberships in the set $PS$ (Lines 7-20). The penalty term ($pen$) is proportional to the length of the path $T$ (Line 16). The paths stored in the array $TP$ are sorted based on their updated scores (Line 21). The first $M-$paths in the array $TP$ and their scores $SP$ are stored as the paths through vertex $u$ to the *sink* vertex $u_{sink}$ (Lines 22-26).

### C.1 Time complexity analysis of ViPRA

ViPRA runs Algorithm 2 as its sub-routine and the time complexity of the two algorithms is evaluated together. Lines 1-7 of ViPRA (Algorithm 1) takes $O(|V_c| + |V_c| \log |V_c|)$ time to initialize and then sort the vertices. The $|V_c|$ calls to the sub-routine TOP-M-PATHS-FOR-VERTEX($N[i], E_c, PS$) take $O(M \cdot |E_c|)$ time (Lines 8-10 of ViPRA, Lines 1-6 of Algorithm 2). Each edge $(u, v) \in E_c$ is encountered at most $M$ times to store the top $M-$paths for the vertex $u$. Lines 7-20 in Algorithm 2 look at each path $T$ in array $TP$ and query for vertex-pairs $(u, w) \in PS$. The number of queries are proportional to the length of the path $T$. Thus, the total number of queries is bounded by $O(|TP| \cdot |T|)$. As the length of path increases linearly to the maximum depth in the graph $G$ ($O(|V_c|)$, and the size of $|TP|$ is bounded by $O(M)$, the total time complexity for lines 7-20 is $O(M \cdot |V_c|^2)$. As lines 21-26 perform a sorting operation for $|TP|$ elements at a time, its time complexity is bounded by $O(M \cdot |E_c| \log (L \cdot |E_c|))$. Thus overall running time of ViPRA is $O(M \cdot |V_c|^2 + M \cdot |E_c| \log (M \cdot |E_c|))$. Notice that the running time of ViPRA increases log linearly with

**Algorithm 2** TOP-M-PATHS-FOR-VERTEX $(u, E, PS)$ : Top $M-$paths for a vertex $u$ in the graph $G$

```
**Input:** Vertex $u$, Edge set $E$, Set $PS$, Insert size $IS$
**Output:** $Paths(u), Score(u)$, Set of top $M-$paths for the vertex $u$, and their respective scores
 1: $TP = [\emptyset]$
 2: $SP = [\emptyset]$
 3: for each $\{v, (u,v) \in E\}$ do
 4:     $TP$ = JOIN $(TP, Paths(v))$ //Obtain top $M$ -paths of the neighbor
 5:     $SP$ = JOIN $(SPScore(v))$
 6: end for
 7: for each path $T \in TP$ do
 8:     $l$ = length($T$)
 9:     $SP(T) = SP(T) \cdot \frac{l \cdot (l-1)}{2}$
10:     for each vertex $w \in T$ do
11:         if $(u, w) \in PS$ then
12:             $SP(T) = SP(T) + 1$
13:         else if $D(w) - D(u) > IS^*$ then
14:             $SP(T) = SP(T) + 1$
15:         else
16:             $SP(T) = SP(T) - l$
17:         end if
18:     end for
19:     $SP(T) = SP(T) \cdot \frac{2}{(l+1) \cdot l}$
20: end for
21: $(TP, SP)$ = SORT-DECREASING$(TP, SP(.))$ // Sort $TP$ paths based on the corresponding scores $SP$ of the paths
22: for $i = 1 \ldots M$ do
23:     $Paths(u)$ = JOIN $(\{u``\_"TP[i]\}, Paths(u))$ // Add $u$ to the path
24:     $Score(u)$ = JOIN $(\{SP[i]\}, Score(u))$
25: end for
26: Return $Paths(u), Score(u)$
```

the parameter $M$ of the algorithm, and is quadratic in the input graph parameters, namely $G_c(V_c, E_c)$.

## Acknowledgments

This research was supported in part by the National Science Foundation award numbers 1724008 and 1720635.

## References

[1] Deorowicz S, Kokot M, Grabowski S, Debudaj-Grabysz A. KMC 2: fast and resource-frugal k-mer counting. Bioinformatics. 2015 May 15;31(10):1569-76.

[2] Boerlijst MC, Bonhoeffer S, Nowak MA. Viral quasi-species and recombination. Proceedings of the Royal Society of London Series B: Biological Sciences. 1996;263(1376):1577–1584.

[3] Bruen TC, Poss M. Recombination in feline immunodeficiency virus genomes from naturally infected cougars. Virology. 2007;364(2):362–370.

[4] Domingo E, Holland J. RNA virus mutations and fitness for survival. Annual Reviews in Microbiology. 1997;51(1):151–178.

[5] Domingo E, Sheldon J, Perales C. Viral quasispecies evolution. Microbiology and Molecular Biology Reviews. 2012;76(2):159–216.

[6] Eliseev A, Gibson KM, Avdeyev P, Novik D, Bendall ML, Pérez-Losada M, Alexeev N, Crandall KA. Evaluation of haplotype callers for next-generation sequencing of viruses. Infection, Genetics and Evolution. 2020 Aug 1;82:104277.

[7] Knyazev S, Hughes L, Skums P, Zelikovsky A. Epidemiological data analysis of viral quasispecies in the next-generation sequencing era. Briefings in bioinformatics. 2020 Jun 22.

[8] Zagordi O, Bhattacharya A, Eriksson N, Beerenwinkel N. ShoRAH: estimating the genetic diversity of a mixed sample from next-generation sequencing data. BMC bioinformatics. 2011;12(1):119.

[9] Prabhakaran S, Rey M, Zagordi O, Beerenwinkel N, Roth V. HIV-haplotype inference using a constraint-based Dirichlet process mixture model. In: Machine Learning in Computational Biology (MLCB) NIPS Workshop; 2010. p. 1–4.

[10] Wang C, Mitsuya Y, Gharizadeh B, Ronaghi M, Shafer RW. Characterization of mutation spectra with ultra-deep pyrosequencing: application to HIV-1 drug resistance. Genome Research. 2007 Aug;17(8):1195–1201.

[11] Zagordi O, Geyrhofer L, Roth V, Beerenwinkel N. Deep sequencing of a genetically heterogeneous sample: local haplotype reconstruction and read error correction. Journal of computational biology. 2010 Mar;17(3):417–428.

[12] Zagordi O, Klein R, Däumer M, Beerenwinkel N. Error correction of next-generation sequencing data and reliable estimation of HIV quasispecies. Nucleic Acids Research. 2010;38:7400–7409.

[13] Prosperi M, Prosperi L, Bruselles A, Abbate I, Rozera G, Vincenti D, et al. Combinatorial analysis and algorithms for quasispecies reconstruction using next-generation sequencing. BMC Bioinformatics. 2011;12:5.

[14] Töpfer A, Zagordi O, Prabhakaran S, Roth V, Halperin E, Beerenwinkel N. Probabilistic inference of viral quasispecies subject to recombination. Journal of Computational Biology. 2013;20(2):113–123.

[15] Prosperi MC, Salemi M. QuRe: software for viral quasispecies reconstruction from next-generation sequencing data. Bioinformatics. 2012;28(1):132–133.

[16] Astrovskaya I, Tork B, Mangul S, Westbrooks K, Măndoiu I, Balfe P, et al. Inferring viral quasispecies spectra from 454 pyrosequencing reads. BMC Bioinformatics. 2011;12 (6).

[17] Westbrooks K, Astrovskaya I, Campo D, Khudyakov Y, Berman P, Zelikovsky A. HCV quasispecies assembly using network flows. Bioinformatics Research and Applications. 2008;p. 159–170.

[18] Skums P, Mancuso N, Artyomenko A, Tork B, Mandoiu I, Khudyakov Y, et al. Reconstruction of Viral Population Structure from Next-Generation Sequencing Data Using Multicommodity Flows. BMC Bioinformatics. 2013;14(Suppl 9):S2.

[19] Eriksson N, Pachter L, Mitsuya Y, Rhee SY, Wang C, Gharizadeh B, et al. Viral Population Estimation Using Pyrosequencing. PLoS Comput Biol. 2008 05;4(5):e1000074.

[20] Knyazev S, Tsyvina V, Melnyk A, Artyomenko A, Malygina T, Porozov YB, Campbell E, Switzer WM, Skums P, Zelikovsky A. Cliquesnv: Scalable reconstruction of intra-host viral populations from ngs reads. BioRxiv. 2018 Jan 1:264242.

[21] Ahn S, Vikalo H. aBayesQR: A bayesian method for reconstruction of viral populations characterized by low diversity. InInternational Conference on Research in Computational Molecular Biology 2017 May 3 (pp. 353-369). Springer, Cham.

[22] Leviyang S, Griva I, Ita S, Johnson WE. A penalized regression approach to haplotype reconstruction of viral populations arising in early HIV/SIV infection. Bioinformatics. 2017 Aug 15;33(16):2455-63.

[23] Schirmer M, Sloan WT, Quince C. Benchmarking of viral haplotype reconstruction programmes: an overview of the capacities and limitations of currently available programmes. Briefings in Bioinformatics. 2012;

[24] Prosperi MC, Yin L, Nolan DJ, Lowe AD, Goodenow MM, Salemi M. Empirical validation of viral quasispecies assembly algorithms: state-of-the-art and challenges. Scientific reports. 2013;3.

[25] Yang X, Charlebois P, Gnerre S, Coole MG, Lennon NJ, Levin JZ, et al. De novo assembly of highly diverse viral populations. BMC genomics. 2012;13(1):475.

[26] Hunt M, Gall A, Ong SH, Brener J, Ferns B, Goulder P, Nastouli E, Keane JA, Kellam P, Otto TD. IVA: accurate de novo assembly of RNA virus genomes. Bioinformatics. 2015 Jul 15;31(14):2374-6.

[27] Wymant C, Blanquart F, Golubchik T, Gall A, Bakker M, Bezemer D, Croucher NJ, Hall M, Hillebregt M, Ong SH, Ratmann O. Easy and accurate reconstruction of whole HIV genomes from short-read sequence data with shiver. Virus evolution. 2018 Jan;4(1):vey007.

[28] Baaijens JA, El Aabidine AZ, Rivals E, Schönhuth A. De novo assembly of viral quasispecies using overlap graphs. Genome research. 2017 May 1;27(5):835-48.

[29] Chen J, Zhao Y, Sun Y. De novo haplotype reconstruction in viral quasispecies using paired-end read guided path finding. Bioinformatics. 2018 Sep 1;34(17):2927-35.

[30] Rizzi R, Beretta S, Patterson M, Pirola Y, Previtali M, Della Vedova G, Bonizzoni P. Overlap graphs and de Bruijn graphs: data structures for de novo genome assembly in the big data era. Quantitative Biology. 2019 Dec;7(4):278-92.

[31] Töpfer A, Marschall T, Bull RA, Luciani F, Schönhuth A, Beerenwinkel N. Viral Quasispecies Assembly via Maximal Clique Enumeration. PLoS Comput Biol. 2014 03;10(3):e1003515.

[32] Mangul S, Wu NC, Mancuso N, Zelikovsky A, Sun R, Eskin E. Accurate viral population assembly from ultra-deep sequencing data. Bioinformatics. 2014;30(12):i329–i337.

[33] Jayasundara D, Saeed I, Maheswararajah S, Chang BC, Tang SL, Halgamuge SK. ViQuaS: an improved reconstruction pipeline for viral quasispecies spectra generated by next-generation sequencing. Bioinformatics. 2015 Mar 15;31(6):886-96.

[34] Bansal V, Bafna V. HapCUT: an efficient and accurate algorithm for the haplotype assembly problem. Bioinformatics. 2008;24(16):i153–i159.

[35] Deng ZL, Dhingra A, Fritz A, Götting J, Münch PC, Steinbrück L, Schulz TF, Ganzenmüller T, McHardy AC. Evaluating assembly and variant calling software for strain-resolved analysis of large DNA viruses. Briefings in Bioinformatics. 2020 Jan 1.

[36] Rizzi R, Tomescu A, Makinen V. On the complexity of Minimum Path Cover with Subpath Constraints for multi-assembly. BMC Bioinformatics. 2014;15(Suppl 9):S5.

[37] Beerenwinkel N, Beretta S, Bonizzoni P, Dondi R, Pirola Y. Covering Pairs in Directed Acyclic Graphs. In: Dediu AH, Martín-Vide C, Sierra-Rodríguez JL, Truthe B, editors. Language and Automata Theory and Applications. vol. 8370 of Lecture Notes in Computer Science. Springer International Publishing; 2014. p. 126–137.

[38] Bolger AM, Lohse M, Usadel B. Trimmomatic: a flexible trimmer for Illumina sequence data. Bioinformatics. 2014 Aug 1;30(15):2114-20.

[39] Allam A, Kalnis P, Solovyev V. Karect: accurate correction of substitution, insertion and deletion errors for next-generation sequencing data. Bioinformatics. 2015 Nov 1;31(21):3421-8.

[40] Mikheenko A, Saveliev V, Gurevich A. MetaQUAST: evaluation of metagenome assemblies. Bioinformatics. 2016 Apr 1;32(7):1088-90.

[41] Zhu N, Zhang D, Wang W, Li X, Yang B, Song J, Zhao X, Huang B, Shi W, Lu R, Niu P. A novel coronavirus from patients with pneumonia in China, 2019. New England Journal of Medicine. 2020 Jan 24.

[42] Tai W, He L, Zhang X, Pu J, Voronin D, Jiang S, Zhou Y, Du L. Characterization of the receptor-binding domain (RBD) of 2019 novel coronavirus: implication for development of RBD protein as a viral attachment inhibitor and vaccine. Cellular & molecular immunology. 2020 Jun;17(6):613-20.

[43] Sternberg A, Naujokat C. Structural features of coronavirus SARS-CoV-2 spike protein: Targets for vaccination. Life sciences. 2020 Jul 6:118056.

[44] Bangaru S, Ozorowski G, Turner HL, Antanasijevic A, Huang D, Wang X, Torres JL, Diedrich JK, Tian JH, Portnoff AD, Patel N. Structural analysis of full-length SARS-CoV-2 spike protein from an advanced vaccine candidate. Science. 2020 Nov 27;370(6520):1089-94.

[45] Salvatori G, Luberto L, Maffei M, Aurisicchio L, Roscilli G, Palombo F, Marra E. SARS-CoV-2 SPIKE PROTEIN: an optimal immunological target for vaccines. Journal of translational medicine. 2020 Dec;18:1-3.

[46] Korber B, Fischer WM, Gnanakaran S, Yoon H, Theiler J, Abfalterer W, Hengartner N, Giorgi EE, Bhattacharya T, Foley B, Hastie KM. Tracking changes in SARS-CoV-2 Spike: evidence that D614G increases infectivity of the COVID-19 virus. Cell. 2020 Aug 20;182(4):812-27.

[47] Bankevich A, Nurk S, Antipov D, Gurevich AA, Dvorkin M, Kulikov AS, et al. SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. Journal of Computational Biology. 2012;19(5):455–477.

[48] Malhotra R, Jha M, Poss M, Acharya R. A random forest classifier for detecting rare variants in NGS data from viral populations. Computational and Structural Biotechnology Journal. 2017;15:388–395.

[49] Rizk G, Lavenier D, Chikhi R. DSK: k-mer counting with very low memory usage. Bioinformatics. 2013;29(5):652–653.

[50] Deorowicz S, Kokot M, Grabowski S, Debudaj-Grabysz A. KMC 2: Fast and resource-frugal k-mer counting. Bioinformatics. 2015;31(10):1569–1576.

[51] Medvedev P, Pham S, Chaisson M, Tesler G, Pevzner P. Paired de bruijn graphs: a novel approach for incorporating mate pair information into genome assemblers. Journal of Computational Biology. 2011;18(11):1625–1634.

[52] Giallonardo FD, Töpfer A, Rey M, Prabhakaran S, Duport Y, Leemann C, Schmutz S, Campbell NK, Joos B, Lecca MR, Patrignani A. Full-length haplotype reconstruction to infer the structure of heterogeneous virus populations. Nucleic acids research. 2014 Aug 18;42(14):e115-.

[53] Jayasundara D, Saeed I, Chang B, Tang SL, Halgamuge SK. Accurate reconstruction of viral quasispecies spectra through improved estimation of strain richness. BMC bioinformatics. 2015;16(Suppl 18):S3.

[54] Baaijens JA, Van der Roest B, Köster J, Stougie L, Schönhuth A. Full-length de novo viral quasispecies assembly through variation graph construction. Bioinformatics. 2019 Dec 15;35(24):5086-94.

[55] Chen J, Shang J, Wang J, Sun Y. A binning tool to reconstruct viral haplotypes from assembled contigs. BMC bioinformatics. 2019 Dec 1;20(1):544.

[56] Zhang W, Davis BD, Chen SS, Martinez JM, Plummer JT, Vail E. Emergence of a novel SARS-CoV-2 variant in Southern California. Jama. 2021 Apr 6;325(13):1324-6.