Title

Variation graph toolkit improves read mapping by representing genetic variation in the reference

Permalink

https://escholarship.org/uc/item/1rx606xj

Journal

Nature Biotechnology, 36(9)

ISSN

1087-0156

Authors

Garrison, Erik
Sirén, Jouni
Novak, Adam M
et al.

Publication Date

2018-10-01

DOI

10.1038/nbt.4227

Peer reviewed

Published in final edited form as:
Nat Biotechnol. 2018 October ; 36(9): 875–879. doi:10.1038/nbt.4227.

Variation graph toolkit improves read mapping by representing genetic variation in the reference

Erik Garrison[1,*], **Jouni Sirén**[1], **Adam M. Novak**[2], **Glenn Hickey**[2], **Jordan M. Eizenga**[2], **Eric T. Dawson**[1,3,4], **William Jones**[1], **Shilpa Garg**[5], **Charles Markello**[2], **Michael F. Lin**[6], **Benedict Paten**[2], and **Richard Durbin**[1,4,*]

[1]Wellcome Sanger Institute, Wellcome Genome Campus, Hinxton, Cambridge, UK

[2]UC Santa Cruz Genomics Institute, University of California, Santa Cruz, CA, USA

[3]National Cancer Institute, 9609 Medical Center Drive, Rockville, MD, USA

[4]Department of Genetics, University of Cambridge, Cambridge CB2 3EH, UK

[5]Max-Planck-Institut für Informatik, 66123 Saarbrücken, Germany

[6]DNAnexus, Mountain View, CA, USA

Abstract

Reference genomes guide our interpretation of DNA sequence data. However, conventional linear references represent only one version of each locus, ignoring variation in the population. Poor representation of an individual's genome sequence impacts read mapping and introduces bias. Variation graphs are bidirected DNA sequence graphs that compactly represent genetic variation across a population, including large scale structural variation such as inversions and duplications1. Previous graph genome software implementations2–4 have been limited by scalability or topological constraints. Here we present vg, a toolkit of computational methods for creating, manipulating, and utilizing these structures as references at the scale of the human genome. vg provides an efficient approach to mapping reads onto arbitrary variation graphs using generalised compressed suffix arrays5, with improved accuracy over alignment to a linear reference, and

[*]to whom correspondence should be addressed.

Author Contributions
EG conceived and led the development of vg, JS developed the GCSA2 index, AMN, GH, JME and ETD contributed to the software, EG, WJ, SG, CM, MFL and RD contributed results and data analysis, RD and BP oversaw the project, and all contributed to the manuscript.

Competing Financial Interests
ML is an employee of, and EG consults for, DNAnexus Inc. RD holds shares in and consults for Congenica Ltd and Dovetail Inc. The remaining authors declare no competing financial interests.

Data Availability
No new data were collected for this study. The human HG002 data used for figure 2b are available from ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/release/AshkenazimTrio/HG002_NA24385_son/latest (calls) and http://trace.ncbi.nlm.nih.gov/Traces/sra/?study=SRP047086 (reads). The yeast whole genome assemblies for Figures 1 and 3 are available from http://www.ebi.ac.uk/ena/data/view/PRJEB7245, the ChIP-seq data set from https://www.encodeproject.org/files/ENCFF000ATK/, and the viral metagenome data from https://www.ebi.ac.uk/ena/data/view/ERS396648.

Code Availability:
vg is available at https://github.com/vgteam/vg under the MIT open source software licence.

effectively removing reference bias. These capabilities make using variation graphs as references for DNA sequencing practical at gigabase scale, or at the topological complexity of *de novo* assemblies.

For small genomes, it is possible to study genetic variation by assembling whole genomes and then comparing them via whole-genome comparison6,7. For large genomes, such as the human genome, complete and accurate *de novo* genome assembly is impractical because of repeat complexity and scale. Therefore prior information is used to interpret new sequence data in its correct genomic context. The current practice is to align sequence reads to a single high-quality reference genome sequence that represents one haplotype at each location in the genome. Although much faster than *de novo* assembly, and simplifying discovery and reporting of genetic variants, this approach leads to mapping biases towards variants matching the reference sequence and away from alternative variants**.** There will even be some sequence in each new sample that is entirely absent in the reference8.

To avoid these biases, data would need to be aligned to a “personalized” reference sequence that already incorporates the individual’s variants9, but in general it is not known what variants are present in a sample before aligning data from it. However, most differences between any one genome and the reference are segregating in the population. Thus, a reference structure that represents known shared variation will contain most of the correct personalised sequence for any individual.

The natural computational structure for doing this is the sequence graph1. Sequence graphs or equivalent structures have been used previously to represent multiple sequences that contain shared differences or ambiguities in a single structure. For example, multiple sequence alignments have a natural representation as partially ordered sequence graphs10. The variant call format11 (VCF), which is a common data format for describing sets of genome sequences can be understood as defining a partially ordered graph similar to those implied by a multiple sequence alignment. Related structures frequently used in genome assembly include the De Bruijn graph12 and string graph13, which collapse long repeated sequences, so the same nodes are used for different regions of the genome. Graphs to represent genetic variation have previously been used for microbial genomes and localized regions of the human genome such as the Major Histocompatibility Complex2.

We define a variation graph as a sequence graph together with a set of paths representing possible sequences from a population (Figure 1). Recently, software packages have been introduced that support a subset of variation graphs that reflect local variation away from a linear reference2,3, formalising approaches introduced in FreeBayes and the GATK HaplotypeCaller for the 1000 Genomes Project analysis14–16. Our model goes beyond these in that it does not require the graph to be based on an initial linear reference, or indeed directionally ordered, and thus supports cycles and inversions. vg is the first openly available tool with these properties to scale practically to the multi-gigabase scale required for whole vertebrate genomes.

The core data model, data structures and algorithms, and implementation of vg are described in the Online Methods. Indicative memory and compute run time requirements are given in

Supplementary Table 1. Below we present results demonstrating the functionality of vg. Variant calling using vg against a variety of different human genome variation graphs is described elsewhere17.

For a species such as human, with only 0.1% nucleotide divergence on average between individual genome sequences, over 90% of 100bp reads will derive from sequence exactly matching the reference. Therefore new mappers should perform at least as well for linear reference mapping as the current standard, which we take to be bwa mem18 with default parameters. We show that vg does this, and then that around divergent sites vg maps more informatively.

The final phase of the 1000 Genomes Project (1000GP) produced a data set of approximately 80 million variants in 2504 humans16. We made a series of vg graphs containing all variants or those with minor allele frequency thresholds at 0.1%, 1% or 10%, as well as a graph corresponding to the standard GRCh37 linear reference sequence without any variation. The full vg graph uses 3.92 GB when serialized to disk, and contains 3.181 Gbp of sequence, which is exactly equivalent to the length of the input reference plus the length of the novel alleles in the VCF file. Complete file sizes including indices vary from 25GB to 63GB, with details including build and mapping times given in Supplementary Table 1.

We next aligned ten million 150 bp paired end reads simulated with errors from the parentally phased haplotypes of an Ashkenazi Jewish male NA24385, sequenced by the Genome in a Bottle (GIAB) Consortium19 and not included in the 1000GP sample set, to each of these graphs as well as to the linear reference using bwa mem. Figure 2a shows the accuracy of these alignments compared with bwa mem for the 1% allele frequency threshold graph, in terms of Receiver Operating Characteristic (ROC) curves. Comparable plots for other data are given in Supplementary Figure 1.

Reads that come from parts of the sequence without differences from the reference (middle panel of Fig. 2a) map slightly better to the reference sequence (green) than to the 1000GP graph (red), which we attribute to a combination of the increase in options for alternative places to map reads provided by the variation graph, and the fact that we needed to prune some search index k-mers in the most complex regions of the graph. As expected this difference increases as the allele frequency threshold is lowered and more variants are included in the graph (Supplementary Figure 1).

For reads that were simulated from segments containing non-reference alleles (approximately 10% of reads), which are the reads relevant to variant calling, vg mapping to the 1000GP graph (red) gives better performance than either vg (green) or bwa mem (blue) mapping to the linear reference (Figure 2b), because many variants present in NA24385 are already represented in the 1000GP graph. This is particularly clear for single end mapping, since many paired end reads are rescued by the mate read mapping. Overall vg performs at least as well as bwa mem even on reference-derived reads, and substantially better on reads containing non-reference variants.

We also mapped a real human genome read set with approximately 50X coverage of Illumina 150bp paired end reads from the NA24385 sample to the 1000GP graph. vg produced mappings for 98.7% of the reads, 88.7% with reported mapping quality score 30 on the Phred scale, and 76.8% with perfect, full-length sequence identity to the reported path on the graph. For comparison, we also used vg to map these reads to the linear reference. Similar proportions of reads mapped (98.7%) and with reported quality 30 (88.8%), but considerably fewer with perfect identity (67.6%). Markedly different mappings were found for 1.0% of reads (0.9% mapping to widely separated positions on the two graphs, and 0.1% mapping to one graph but not the other). The reads mapping to widely separated positions were strongly enriched for repetitive DNA. For example, the linear reference mappings for 27.5% of these read pairs overlap various types of satellite DNA identified by RepeatMasker, compared to 3.0% of all read pairs.

To illustrate the consequences of mapping to a reference graph rather than a linear reference, we stratified the sites independently called as heterozygous in NA24385 by deletion or insertion length (0 for single nucleotide variants) and by whether the site was present in 1000GP, and measured the fraction of reads mapped to the alternate allele for each category. The results show that mapping with vg to the population graph when the variant is present in 1000GP (95.4% of sites) gives nearly balanced coverage of alternate and reference alleles independent of variant size, whereas mapping to the linear reference either with vg or bwa mem leads to a progressively increasing bias with increasing deletion and (especially) insertion length (Figure 2b), so that for insertions around 30bp a majority of insertion containing reads are missing (there are over twice as many reference reads as alternate reads).

This removal of bias is important when mapping functional genomics data such as ChIP-seq data, where allele specific expression analysis can reveal genetic variation that affects function but is confounded by reference mapping bias20, especially given that read lengths are typically shorter for these experiments. We compared mapping with bwa or vg for data set ENCFF000ATK from the ENCODE project21, which contains 14.9 million 51bp ChIP-seq reads for the H3K4me1 histone methylation mark from the NA12878 cell line. When mapping with bwa the ratio of reference to alternate allele matches at heterozygous sites was 1.20, whereas with vg to the 1000GP graph the ratio was 1.01, effectively eliminating reference bias.

We also explored integration of vg with the recently published GraphTyper12 method, which calls genotypes by remapping reads to a local partially ordered variation graph built from a VCF file, relying on initial global assignment to a region of the genome by mapping with bwa to a linear reference. Therefore, although GraphTyper also scales to the whole human genome because it is essentially a local method, its functionality is complementary to that of vg, which maps to a global variation graph and does not directly call genotypes. In experiments where we used vg rather than bwa as the primary mapper for GraphTyper, true positives increased marginally (0.02% for SNPs and 0.06% for indels) while false positives increased for SNPs by 0.15% and decreased for indels by 0.03%. We note, however that GraphTyper was developed by its authors for bwa mapping.

The graphs that we have used so far were constructed from variation data obtained from mapping to a linear reference, and so are directed acyclic graphs. We next demonstrate the ability of vg to work with arbitrary graphs that include duplications, inversions, and translocations, by showing its use with multiple yeast strains independently assembled de novo using long read data22. These assemblies manifest large scale structural variation and novel sequence not detected in reference-based sequencing, including extensive rearrangement and reordering in subtelomeric regions22 as illustrated in Figure 1.

We compared four vg graphs: a linear reference graph from the standard S288c strain, a linear reference from the SK1 strain, a pangenome graph of all seven strains, and a “drop SK1” variation graph in which all sequence private to the strain SK1 was removed from the pangenome graph. The multiple genome graphs were constructed with the Cactus progressive aligner6, which generates graphs that typically contain cycles and are not partially ordered.

Similarly to the human experiments, we simulated 100,000 150bp paired reads from the SK1 reference, modelling sequencing errors, and mapped them to the four references. The resulting ROC curves are shown in Figure 3a. Not surprisingly, the best performance is obtained by mapping to a linear reference of the SK1 strain from which the data were simulated, with substantially higher sensitivity and specificity compared to mapping to the standard linear reference from the strain S288c with either vg or bwa mem. Mapping to the variation graphs gives intermediate performance, with over one percent more sensitivity and lower false positive rates than to the standard reference. There is surprisingly little difference between mapping to graphs with and without the SK1 private variation, probably because much of what is novel in SK1 compared to the reference is also seen in other strains. We see lower sensitivity compared to mapping just to the SK1 sequence, likely because of suppression of GCSA2 index kmers in complex or duplicated regions. In Figure 3b we show the benefit of aligning long reads to a pangenome graph compared to the S288c reference, using a set of 43,337 Pacific Biosciences SK1 reads (mean length 4.7kb) from Ref. 22.

Finally, to further demonstrate the ability of vg to map to arbitrary sequence graphs, we constructed a vg graph from a metagenomic assembly of a polar freshwater viral DNA community23 that was constructed with the minia324 assembler. We then aligned a held-out subset of 100,000 reads to this assembly graph using vg, and to the linear contigs using bwa. Although both methods map approximately 96% of the reads, vg has an average identity score of 95% compared to 87% for bwa, reflecting that the bwa alignments in many cases are not full length (Supplementary Figure 2).

In conclusion, vg implements a suite of tools for genomic sequence data analysis using general variation graph references. Using the vg toolkit, we can construct or import a graph, modify it, visualize it, and use it as a reference. vg can accurately map new sequence reads to the reference using succinct indexes of the graph and its sequence space, and can describe variation between a new sample and an arbitrary reference embedded as a path in the graph. Elsewhere17 we discuss the use of vg to map read sets and call variants against a number of alternative human reference graphs built from multiple regions of the human genome with different properties.

There are many areas for potential future development and application of vg. These include further improvements in the mapping and variant calling algorithms, potentially using long range statistical haplotype structure information, stored in a graph extension of the PBWT haplotype compression and search data structure[25], as proposed by Novak[26]. Beyond variant calling, the ability to map in an unbiased way to both reference and alternate alleles is potentially important when quantitating allele-specific protein binding as shown with ChIP-seq data above or allele-specific expression[27]. We note that graphs can also naturally represent the relationships between transcribed, spliced and edited RNA sequences and the genome from which they are transcribed, so the vg software can potentially be used for splice-aware RNAseq mapping[28].

We believe that genome variation graphs will underpin a new paradigm for genome sequence data analysis[1]. They support the representation of structural variation using the same components (edges, nodes and paths) that are used to represent single base changes. For human, they allow more accurate and complete read mapping (Figure 2). The benefits will only be greater for other organisms with higher levels of genetic variation, or for which uncertainties remain in the reference assembly. For the biological research community to exploit these advantages, it needs software for variation graphs that scales to the genomes of humans and other complex organisms. vg is a robust and openly available platform to fulfil this need.

A life sciences reporting summary is available.

Online Methods

Model

We define a variation graph to be a graph with embedded paths $G = (N, E, P)$ comprised of a set of nodes $N = n_1 \ldots n_M$, a set of edges $E = e_1 \ldots e_L$, and a set of paths $P = p_1 \ldots p_Q$, each of which describes the embedding of a sequence into the graph.

Each node n_i represents a sequence seq(n_i) which is built from an alphabet $A = \{A,C,G,T\}$. Nodes may be traversed in either the forward or reverse direction, with the sequence being reverse-complemented in the reverse direction. We write n^*_i for the reverse-complement of node n_i, so that seq(n_i) = revcomp(seq(n^*_i)); note that $n_i = n^{**}_i$. For convenience, we refer to both n_i and n^*_i as "nodes". Edges represent adjacencies between the sequences of the nodes they connect. Thus, the graph implicitly encodes longer sequences as the concatenation of node sequences along walks through the graph. Edges can be identified with the ordered pairs of oriented nodes that they link, so we can write $e_{ij} = (n_i,n_j)$. Edges also can be traversed in either the forward or the reverse direction, with the reverse traversal defined as $e^*_{ij} = (n^*_j,n^*_i)$. Note that graphs in vg can contain ordinary cycles (in which n_i is reachable from n_i), reversing cycles (in which n_i is reachable from n^*_i), and non-cyclic instances of reversal (in which both n_i and n^*_i are reachable from n_j). We implement paths as an edit string with respect to the concatenation of node sequences along a directed walk through the graph. We do not require the alignment described by the edit string to start at the beginning of the sequence of the initial node, nor to terminate at the end of the sequence of the terminal node.

Implementation

The vg implementation is multithreaded and written in C++11, and is available from https://github.com/vgteam/vg (version v1.6.0 for code used in the mapping experiments) under the MIT open source software license. It provides both a primary application to support the operations we describe here, and a library libvg which applications can use to access the data structures, indexes and low level operations.

Our core representation of the graph uses Google's open source protobuf system, which directly supports serialisation onto disk for storage. We also provide a protobuf alignment format, GAM (for “Graph Alignment Map”), with analogous functionality to BAM29, but can also export mappings with respect to embedded references in BAM format. To enable read mapping and other random access operations against large sequence graphs we have implemented a succinct representation of a vg variation graph (xg) that is static but very memory and time efficient, using rank/select dictionaries and other data structures from the Succinct Data Structures Library (SDSL)30. Graphs can be imported from and exported to a variety of formats, including the assembly format GFA and the W3C graph exchange format RDF. Further details about the implementation and features are available in the supplement and at the Github website.

Alignment

A key requirement for a reference genome is the ability to efficiently and accurately find an optimal alignment for a new DNA sequence such as a sequencing read. Analogous to the way that read mappers to linear references work, our approach to this problem is to find seed matches by an indexed search process, cluster them if there are multiple seeds close together, and then perform a local constrained dynamic programming alignment of the read against a region of the graph around each cluster. A brief description of the key steps in this process is given here, with further details in the supplement.

The GCSA2 library4 that vg uses for seeding can perform linear time exact match queries independent of the graph size to find super-maximal exact match (SMEM) seeds, subject to a maximum query length, in time comparable to the corresponding operations in bwa mem. SMEMs are exact matches between a query substring and a reference substring that cannot be extended in either direction, and for which there is no extension of the query substring that matches elsewhere in the graph.

After obtaining SMEMs for a query sequence using GCSA2, we cluster them using a global approximate distance metric and distance estimates provided by any nearby paths. For paired reads, we cluster all the SMEMs for both reads in the pair to preferentially support mappings where the SMEMs match a fragment model that we establish online during the alignment of the read set.

We next chain the SMEMs within each cluster by selecting the maximum likelihood path through a Markov model that rewards long SMEMs and short colinear gaps between SMEMs. In many cases there is just one SMEM in the sequence, but there are complex cases where the best SMEM sequence is not correct, and to catch these we recursively mask out the SMEMs in paths found so far and re-run the algorithm to obtain additional disjoint SMEM sequences if available.

For each consistent sequence of SMEMs, we then obtain the subgraph containing the cluster. To avoid the complications introduced by cycles and inversions31, we transform the local graph region into a directed acyclic graph (DAG) while maintaining an embedding in the original, potentially cyclic bidirected graph (Supplementary figures S3, S4). We can then perform partial order alignment to the DAG9, using banded dynamic programming and an extension of Farrar's SIMD-accelerated striped Smith Waterman algorithm32.

When mapping long sequences, we split them into overlapping “chunks” (default 256bp with 32bp overlap), map those as above, then chain them using the same colinear Markov model method as described for SMEMs within a chunk. This scales effectively linearly in sequence length up to multiple megabases.

The vg alignment tool also uses base qualities in alignment scores and calculates adjusted mapping quality scores. Base qualities are probabilistic estimates of the condence of each base call in a read provided by the sequencing technology. vg combines these with a probabilistic interpretation of alignment33 to adjust the scoring function for alignments, which has previously been shown to improve variant calling accuracy34. Mapping qualities35 are a probability-based measure of the confidence in the localisation of a read on the reference that is important for variant calling and other downstream analyses. vg computes mapping qualities by comparing the scores of optimal and suboptimal alignments under the probabilistic alignment model, in a similar fashion to bwa mem.

Graph editing and construction

We can build a graph either by direct construction from external graphs such as from de novo assemblies, or by a series of editing operations applied to simple starting graphs such as standard linear reference genomes. To support editing of existing graphs, vg supports operations that can split a node where sequences diverge and insert additional edges and nodes. While doing this it keeps track of the relationship to the previous graph in a translation object, which supports projection of coordinates from one version of the graph to another.

We make use of the editing operations to construct graphs from Variant Call Format (VCF) files10 as produced by population sequencing projects such as the 1000 Genomes Project16, inserting a cluster of nodes and edges into a linear reference for each overlapping subset of VCF records. Edit operations also allow progressive construction of a vg graph from a set of sequences by repeated alignment and editing, so that all the initial sequences are embedded in the graph as paths. Last but not least, edit operations allow new variants to be added to an existing vg reference graph to support use cases such as incorporating novel variants from new individuals mapped and called against the graph, while retaining a coordinate mapping to the existing reference. These actions are also invertible, in that vg can generate VCF to describe the graph as a set of variants, using an arbitrarily chosen embedded path as a reference.

Experiments

Experiments were carried out on a dedicated compute node with 256 gigabytes of RAM and two 2.4GHz AMD Opteron 6378 processors with a total of 32 CPU cores. Mapping comparisons were to bwa version 0.7.15-r1142.

GraphTyper comparisons

To test how vg map complements genotyping in Graphtyper, we mapped reads from the Genome In A Bottle (GIAB) Ashkenazi Jewish Trio benchmark sample HG002 readset, and analyzed variant calling performance on chromosome 21 against the HG002_GRCh37_GIAB_highconf_CG-IllFB-IllGATKHC-Ion-10X-SOLID_CHROM1-22_v.3.3.2_highconf_triophased.vcf.gz calls using Illumina's Haplotype Comparison Toolset available from https://github.com/Illumina/hap.py. Bwa mem mappings were against the GRCh37d5 reference, and vg mappings against the 1000GP graph then projected onto the GRCh37d5 reference, which is embedded in the 1000GP graph. GraphTyper version 1.3 was run using the dbSNP "common variant" chromosome 21 VCF from NCBI (ftp://ftp.ncbi.nlm.nih.gov/snp/organisms/human_9606_b150_GRCh37p13/VCF/common_all_20170710.vcf.gz). Code used for the analysis is available on request.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgements

EG, JS and RD were funded by the Wellcome Trust (grants 206194 and 207492). ETD was funded by an NIH Cambridge Trust studentship, and WJ by a Wellcome Trust MGM studentship (109083/Z/15/Z). AMN, GH, JME and BP were supported by the National Institutes of Health (5U41HG007234), the W. M. Keck Foundation (DT06172015) and the Simons Foundation (SFLIFE# 35190). We thank members of the GA4GH Reference Variation Working Group for support, ideas and comments, and Hannes Eggertsson for assistance in the integration with GraphTyper.

References

1. Paten B, Novak AM, Eizenga JM, Garrison E. Genome graphs and the evolution of genome inference. Genome Res. 2017; 27:665–676. [PubMed: 28360232]
2. Dilthey A, Cox C, Iqbal Z, Nelson MR, McVean G. Improved genome inference in the MHC using a population reference graph. Nat Genet. 2015; 47:682–688. [PubMed: 25915597]
3. Eggertsson HP, et al. Graphtyper enables population-scale genotyping using pangenome graphs. Nat Genet. 2017; 49:1654–1660. [PubMed: 28945251]
4. Rakocevic G, et al. Fast and accurate genomic analyses using genome graphs. bioRxiv preprint. 2017; doi: 10.1101/194530
5. Siren J. Indexing variation graphs. Proc 19th Workshop on Algorithm Engineering and Experiments (ALENEX); 2017.
6. Delcher L, et al. Alignment of whole genomes. Nucleic Acids Res. 1999; 27:2369–2376. [PubMed: 10325427]
7. Paten B, et al. Cactus: Algorithms for genome multiple sequence alignment. Genome Res. 2011; 21:1512–1528. [PubMed: 21665927]
8. Li R, et al. Building the sequence map of the human pan-genome. Nat Biotech. 2010; 28:57–63.

9. Yuan S, Qin Z. Read-mapping using personalized diploid reference genome for RNA sequencing data reduced bias for detecting allele specific expression. IEEE International Conference on Bioinformatics and Biomedicine Workshops; 2012.

10. Lee C, Grasso C, Sharlow MF. Multiple sequence alignment using partial order graphs. Bioinformatics. 2002; 18:452–464. [PubMed: 11934745]

11. Danecek P, et al. The variant call format and VCFtools. Bioinformatics. 2011; 27:2156–2158. [PubMed: 21653522]

12. Pevzner PA, Tang H, Waterman MS. An eulerian path approach to DNA fragment assembly. Proc Nat Acad, Sci USA. 2001; 98:9748–9753. [PubMed: 11504945]

13. Myers EW. The fragment assembly string graph. Bioinformatics. 2005; 21(Suppl 2):79–85.

14. Garrison E, Marth G. Haplotype-based variant detection from short-read sequencing. arXiv preprint. 2012 arXiv:1207.3907.

15. DePristo MA, et al. A framework for variation discovery and genotyping using next-generation DNA sequencing data. Nat Genet. 2011; 43:491–498. [PubMed: 21478889]

16. 1000 Genomes Project Consortium. et al. A global reference for human genetic variation. Nature. 2015; 526:68–74. [PubMed: 26432245]

17. Novak AM, et al. Genome graphs. bioRxiv preprint. 2017; doi: 10.1101/101378

18. Li H. Aligning sequence reads, clone sequences and assembly contigs with bwa-mem. arXiv preprint. 2013 arXiv:1303.3997.

19. Zook JM, et al. Extensive sequencing of seven human genomes to characterize benchmark reference materials. Sci Data. 2016; 3 160025.

20. McDaniell R, et al. Heritable individual-specific and allele-specific chromatin signatures in humans. Science. 2010; 328:235–239. [PubMed: 20299549]

21. ENCODE Project Consortium. An integrated encyclopedia of DNA elements in the human genome. Nature. 2012; 489:57–74. [PubMed: 22955616]

22. Yue J-X, et al. Contrasting evolutionary genome dynamics between domesticated and wild yeasts. Nat Genet. 2017; 49:913–924. [PubMed: 28416820]

23. de Cárcer DA, López-Bueno A, Pearce DA, Alcamí A. Biodiversity and distribution of polar freshwater DNA viruses. Science Advances. 2015; 1:e1400127. [PubMed: 26601189]

24. Chikhi R, Rizk G. Space-efficient and exact de Bruijn graph representation based on a Bloom filter. Algorithms Mol Biol. 2013; 8:22. [PubMed: 24040893]

25. Durbin R. Efficient haplotype matching and storage using the positional Burrows-Wheeler transform (PBWT). Bioinformatics. 2014; 30:1266–1272. [PubMed: 24413527]

26. Novak AM, Garrison E, Paten B. A graph extension of the positional Burrows-Wheeler transform and its applicationsAlgorithms in bioinformatics. Firth M, Pedersen CN, editorsSpringer; Heidelberg: 2016. 246–256.

27. Ge B, et al. Global patterns of cis variation in human cells revealed by high-density allelic expression analysis. Nat Genet. 2009; 41:1216–1222. [PubMed: 19838192]

28. Beretta S, , et al. Mapping RNAseq Data to a Transcript Graph via Approximate Pattern Matching to a HypertextAlgorithms for Computational Biology (AlCoB). Lecture Notes in Computer Science. Figueiredo D, Martn-Vide C, Pratas D, Vega-Rodrguez M, editorsVol. 10252. Springer; Champaign-Urbana: 2017. 49–61.

29. Li H, et al. The Sequence Alignment/Map format and SAMtools. Bioinformatics. 2009; 25:2078–2079. [PubMed: 19505943]

30. Gog S, Beller T, Moat A, Petri M. From theory to practice: Plug and play with succinct data structures. International Symposium on Experimental Algorithms; Springer; 2014. 326–337.

31. Myers EW, Miller W. Approximate matching of regular expressions. Bull Math Biol. 1989; 51:5–37. [PubMed: 2706401]

32. Farrar M. Striped Smith-Waterman speeds database searches six times over other SIMD implementations. Bioinformatics. 2007; 23:156–161. [PubMed: 17110365]

33. Durbin R, Eddy SR, Krogh A, Mitchison G. Biological sequence analysis: probabilistic models of proteins and nucleic acids. Cambridge University Press; 1998.

34. Hamada M, Wijaya E, Frith MC, Asai K. Probabilistic alignments with quality scores: an application to short-read mapping toward accurate snp/indel detection. Bioinformatics. 2011; 27:3085–3092. [PubMed: 21976422]

35. Li H. A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. Bioinformatics. 2011; 27:2987–2993. [PubMed: 21903627]

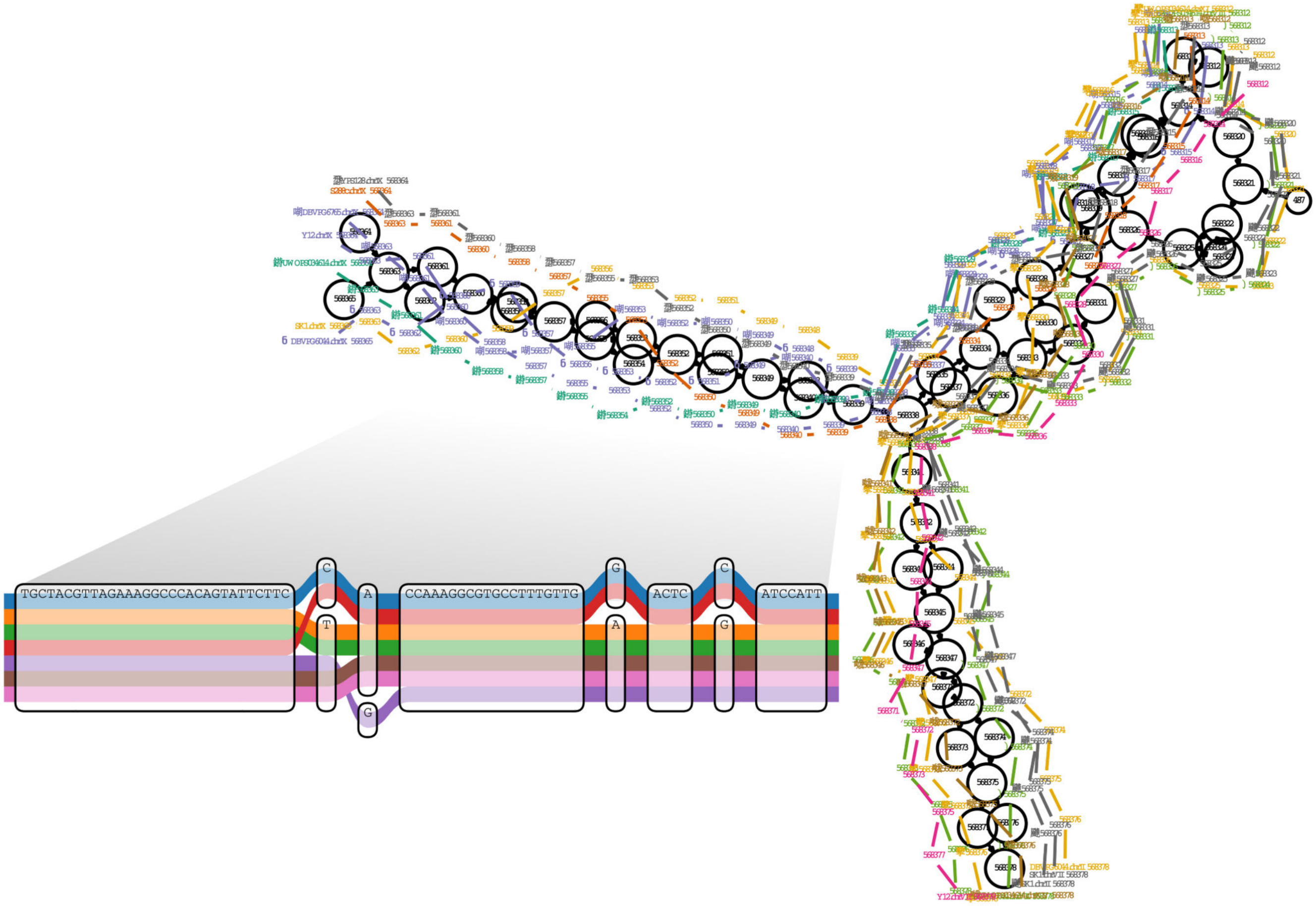

Figure 1.

A region of a yeast genome variation graph. This displays the start of the subtelomeric region on the left arm of chromosome 9 in a multiple alignment of the strains sequenced by Yue et al.22, built using vg from a full genome multiple alignment generated with the Cactus alignment package6. The inset shows a subregion of the alignment at single base level. The colored paths correspond to separate contiguous chromosomal segments of these strains. This illustrates the ability of vg to represent paths corresponding to both colinear (inset) and structurally rearranged (main figure) regions of genomic variation.

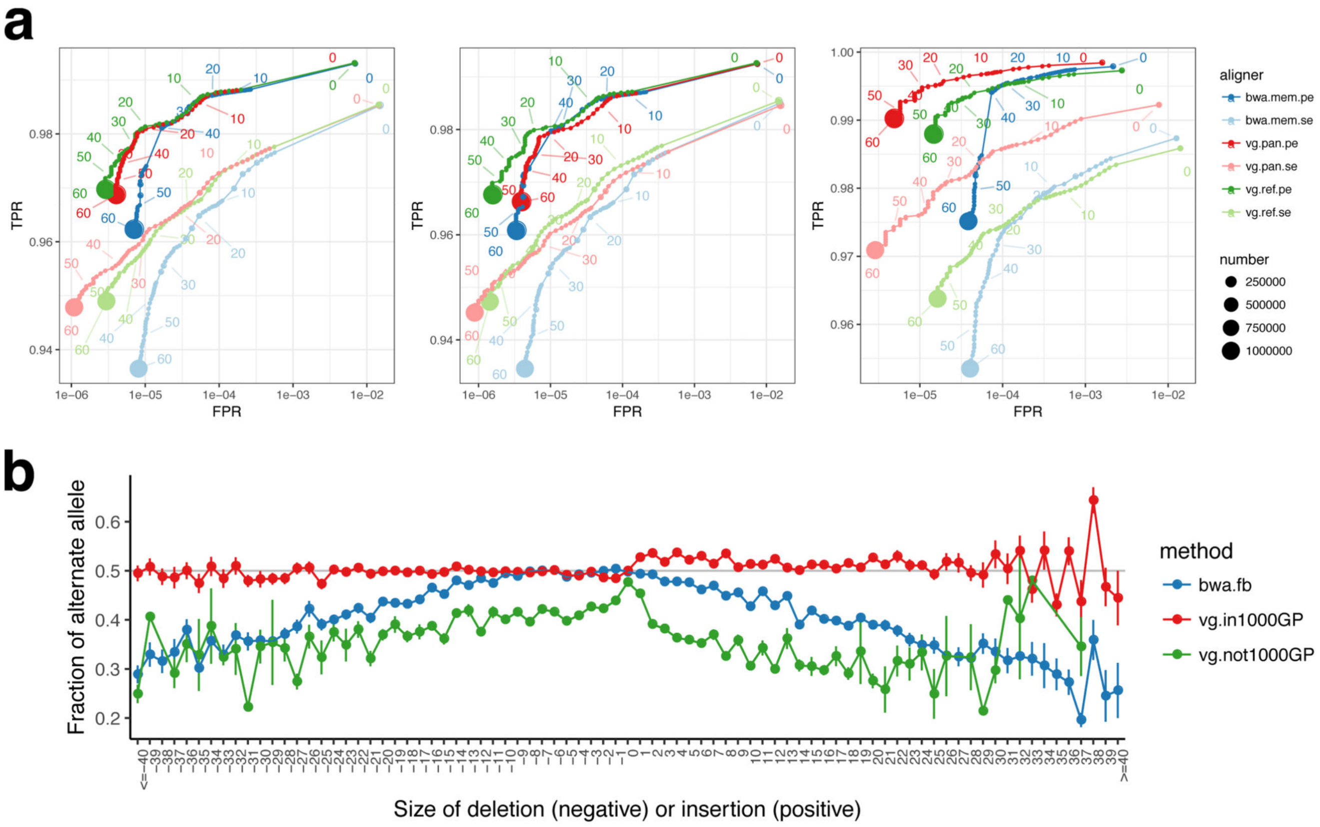

Figure 2.

Mapping accuracy for vg against the human genome. (a) ROC curves parameterised by mapping quality for 10M read pairs simulated from NA24385 as mapped by bwa mem, vg with the 1000GP 1% allele frequency threshold pangenome reference, and vg with a linear reference, using single end (se) or paired end (pe) mapping. Left: all reads, middle: reads simulated from segments matching the linear reference, Right: reads simulated from segments different from the linear reference. (b) the mean alternate allele fraction at heterozygous variants previously called19 in NA24385 as a function of deletion or insertion size (SNPs at 0). Error bars are +/- one standard error.

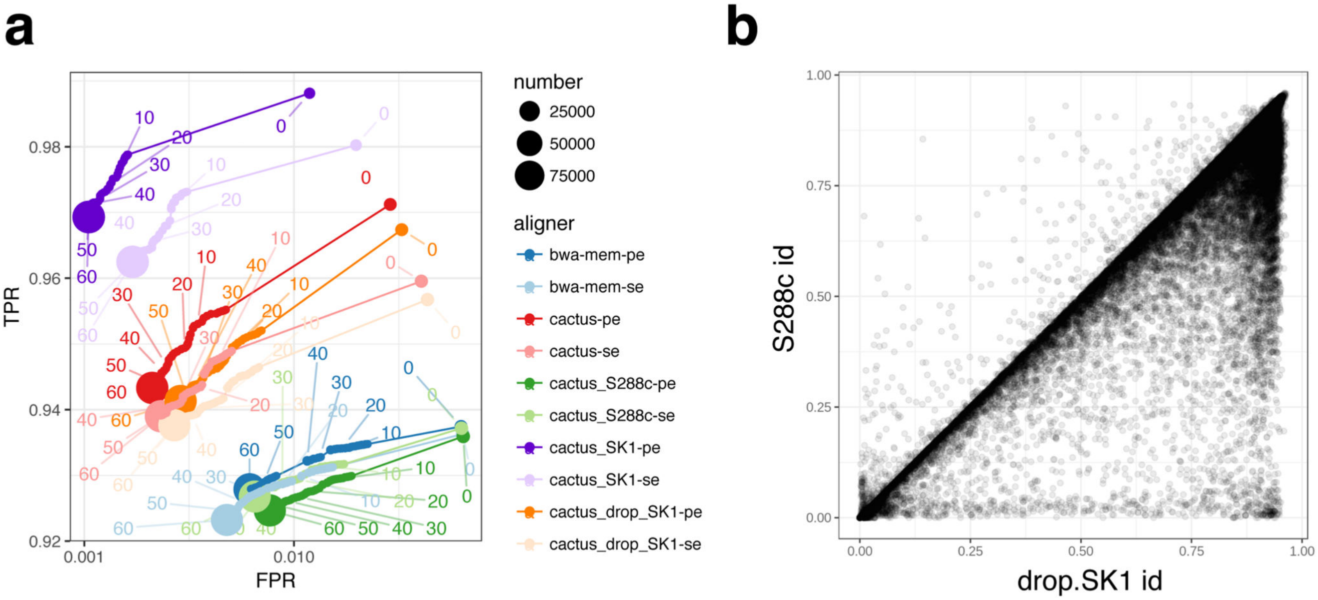

Figure 3.

Mapping short and long reads with vg to yeast genome references. (a) ROC curves obtained by mapping 100,000 simulated SK1 yeast strain 150bp paired reads against a variety of references described in the text; (b) a density plot of identity fraction when mapping 43,337 Pacific Biosciences long reads from the SK1 strain to the drop.SK1 reference or the S288c reference.