Published online 17 August 2021

Nucleic Acids Research, 2021, Vol. 49, No. 19 ***e114***
https://doi.org/10.1093/nar/gkab709

SpecHap: a diploid phasing algorithm based on spectral graph theory

Yonghan YU †, Lingxi Chen †, Xinyao Miao† and Shuai Cheng Li *

Computer Science, City University of Hong Kong, Kowloon, Hong Kong 999077, China

Received December 16, 2020; Revised July 25, 2021; Editorial Decision July 28, 2021; Accepted August 02, 2021

ABSTRACT

Haplotype phasing plays an important role in understanding the genetic data of diploid eukaryotic organisms. Different sequencing technologies (such as next-generation sequencing or third-generation sequencing) produce various genetic data that require haplotype assembly. Although multiple diploid haplotype phasing algorithms exist, only a few will work equally well across all sequencing technologies. In this work, we propose SpecHap, a novel haplotype assembly tool that leverages spectral graph theory. On both *in silico* and whole-genome sequencing datasets, SpecHap consumed less memory and required less CPU time, yet achieved comparable accuracy with state-of-art methods across all the test instances, which comprises sequencing data from next-generation sequencing, linked-reads, high-throughput chromosome conformation capture, PacBio single-molecule real-time, and Oxford Nanopore long-reads. Furthermore, SpecHap successfully phased an individual *Ambystoma mexicanum*, a species with gigantic diploid genomes, within 6 CPU hours and 945MB peak memory usage, while other tools failed to yield results either due to memory overflow (40GB) or time limit exceeded (5 days). Our results demonstrated that SpecHap is scalable, efficient, and accurate for diploid phasing across many sequencing platforms.

INTRODUCTION

Humans and many other species possess diploid genomes with paternal and maternal sets of chromosomes (1). The majority of genetic variations between homologous chromosomes consists of single nucleotide variation (SNV), small insertion and deletion, and genome rearrangement through structure variation or copy number variation (2). Phasing, the reconstruction of specific allele sequences on individual chromosomes, is fundamental to our understanding of compound heterozygosity (3). Many studies have addressed the importance of haplotype phasing, including but not limited to allelic differential expressions, epigenomic regulations, and population development (4–7). Furthermore, several studies have claimed that haplotype analysis revealed disease pathogenicity that could not be inferred from unphased single nucleotide polymorphism (SNP) signals (8–11).

Advances in high-throughput sequencing have resulted in several sequencing protocols that have enabled credible identifications and linkages of genetic variants, significantly contributing towards the single individual haplotyping (SIH) problem, which refers to haplotype assembling or haplotype phasing problem for a single individual. Next-generation sequencing (NGS) technologies, including the Ion Torrent S5 system (Life Technologies), have been widely used to study the haplotypes leveraging paired-end reads (12). For high-throughput chromosome conformation capture (Hi-C), HaploSeq correctly phased ∼95% of heterozygous variants (13,14). However, *trans* interactions between homologous chromosomes complicate the process of phasing. Moreover, segmental duplication and simple repeats are likely related to incorrectly phased haplotype blocks based on Hi-C data (7,15,16). With higher NGS throughput and cost-effectiveness, 10× synthetic long reads (SLRs) protocol provides barcoded linked-reads (>100kb long-range information) that are suitable for assembly and phasing (1,17). Despite the high individual base error rate of ∼15%, the third-generation sequencing, including single-molecule real-time sequencing technology from Pacific Biosciences (PacBio SMRT) and Oxford Nanopore Technology (ONT), offers ultra-long reads with moderate coverage, significantly promoting haplotype completeness (18,19).

There are several algorithms for the SIH problem for diploid organisms from sequencing reads on various protocols. We refer to SIH for diploid organisms with diploid SIH in the rest of the article. These methods could be summarized into three categories: optimization by reads (fragments) partitioning, by minimum error correction (MEC) and by haplotype likelihood. Several routines are adopted

*To whom correspondence should be addressed. Tel: +852 34429412; Fax: +852 34420503; Email: shuaicli@cityu.edu.hk
†The authors wish it to be known that, in their opinion, the first three authors should be regarded as joint First Authors.

to solve the optimization including but not limited to dynamic programming, Markov chain Monte Carlo and heuristic graph cut. (i) FastHare, a fast algorithm based on fragments partitioning optimization (13,18). (ii) HapCUT, a method that assembles haplotype by minimizing the MEC respectively through a max-cut heuristic algorithm (20). (iii) DCHap, a divide-and-conquer algorithm that combines fragment partitioning and MEC optimization designed for third-generation sequencing data (21). (iv) ReFHap, a graph-cut heuristic formulation based on the graph max-cut algorithm for diploid SIH problem (22). (v) HapCUT2, a general algorithm for human haplotype assembly that adopts max-cut computations in the haplotype graph to find the haplotype with maximum likelihood (18). However, advances in sequencing technology continue to call for more computationally efficient, scalable, and accurate methods.

In this paper, we describe SpecHap, a novel fast and accurate scalable algorithm for diploid SIH designed for multiplex sequencing platforms, especially for the error-prone long-reads from third-generation sequencing. Spectral graph theory was adopted in the efficient identification of topological domain with Hi-C data by iteratively domain partitioning guided by Fieldler vector (23). Previous study also utilized spectral graph theory in population genetics for the identification of genetic ancestry (24). However, it has not been explored in the context of haplotype phasing. Instead of iteratively converging on the target haplotype by optimizing MEC or haplotype likelihood (18), SpecHap assembles haplotype efficiently by transforming diploid SIH into a linear algebra problem using spectral graph theory with divide-and-conquer strategy. We benchmarked SpecHap with four state-of-art phasing software packages and demonstrated its comparable accuracy on diverse sequencing protocols. Moreover, SpecHap phased an individual of amphibian species *Ambystoma mexicanum* (Axolotl), which possesses one of the largest sequenced genomes (32 billion base pairs), with ∼32× PacBio SMRT long-reads (25,26).

MATERIALS AND METHODS

Revisiting spectral graph theory

First, we introduce the graph terminology related to this work. Assume a connected undirected graph G consists of N vertices $V = \{v_1, v_2, \ldots, v_N\}$ and M edges denoted by $(v_i, v_j), i \neq j$. The similarity matrix $A^{N\times N}$ is constructed for graph G to store the linkage relationship between a pair of vertices v_i, v_j, such that $A_{i,j} = w_{i,j}$ where $w_{i,j}$ represents the weight of the edge (v_i, v_j). Denoted by d_i the weighted degree, $\sum_{1\le j\le N, j\neq i} w_{ij}$, of the vertex v_i in graph G, the degree matrix D is a diagonal matrix with the i-th diagonal element set to d_i. That is, $D_{i,j} = d_i$ if $i = j$, $D_{i,j} = 0$ otherwise. Then, by definition, the (unnormalized) Laplacian matrix L of graph G is constructed as $L = D - A$. Given a vertex subset $V_s \subset V$, the weight of a graph cut that bisects V into V_s and $\overline{V_s}$ is defined as $RatioCut(V_s, \overline{V_s}) = \frac{\sum_{i\in V_s, j\in \overline{V_s}} w_{ij}}{|V_s|} + \frac{\sum_{i\in V_s, j\in \overline{V_s}} w_{ij}}{|\overline{V_s}|}$ (27) where $|\cdot|$ denotes the cardinality.

We define vector $f \in R^n$ with its entry f_i representing the membership of the ith element:

$$f_i = \begin{cases} \sqrt{\frac{|\overline{V_s}|}{|V_s|}} \text{if } v_i \in V_s \\ -\sqrt{\frac{|V_s|}{|\overline{V_s}|}} \text{if } v_i \in \overline{V_s} \end{cases},$$

which lead to the conclusion that $\|f\|_2 = \sqrt{|V|}$ and $1^T f = 0$.

$f^T L f$ is related to $RatioCut(V_s, \overline{V_s})$ as follows:

$$\begin{aligned} f^T L f &= \sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2 \\ &= 2\left(\frac{|\overline{V_s}|}{|V_s|} + \frac{|V_s|}{|\overline{V_s}|} + 2\right) \sum_{i\in V_s, j\in \overline{V_s}}^{n} w_{ij} \\ &= 2|V|\left(\frac{1}{|V_s|} + \frac{1}{|\overline{V_s}|}\right) \sum_{i\in V_s, j\in \overline{V_s}}^{n} w_{ij} \\ &= 2|V|\, RatioCut\left(V_s, \overline{V_s}\right) \end{aligned}$$

Thus, obtaining a minimum weighted graph cut is equivalent to minimize $f^T L f$. However, this problem is NP-hard (28) given the defined f. A relaxed optimization problem is achieved by letting $f_i \in R$:

$$\begin{aligned} &\text{Minimize} && f^T L f \\ &\text{Subject to} && 1^T f = 0,\ \|f\|_2 = \sqrt{|V|} \end{aligned}$$

The minimizer of this relaxed problem is given by the eigenvector which corresponds to the second smallest eigenvalue of L (Fiedler vector) according to the Rayleigh–Ritz theorem (28). The N vertices are bisected into two groups according to the element sign (+, –) of the Fiedler vector (29).

Haplotype phasing guided by spectral graph theory

We define linkage graph, an undirected graph of heterozygous variants loci with each pair of vertices representing two alleles at the corresponding locus. An edge between two vertices of different variants loci represents a potential pairwise haplotype; the edge's weight represents the logarithmic likelihood for the corresponding haplotype. Applying spectral analysis on this linkage graph allows the haplotype to be inferred from the Fiedler vector. The process is summarized in Figure 1A.

We also generalize the linkage graph where each pair of vertices denotes a haplotype block. The haplotype block refers to the group of variants loci with their haplotype phased. We defined the generalized linkage graph as an undirected graph of haplotype blocks with each pair of vertices representing the two self-complement haplotypes corresponding haplotype block. Edges are added by treating fragments into pairwise linkages between haplotype blocks. The Fiedler vector of generalized linkage graph might guide the connection of haplotype blocks: haplotype of different haplotype blocks with the same sign might originate from the same chromosome.

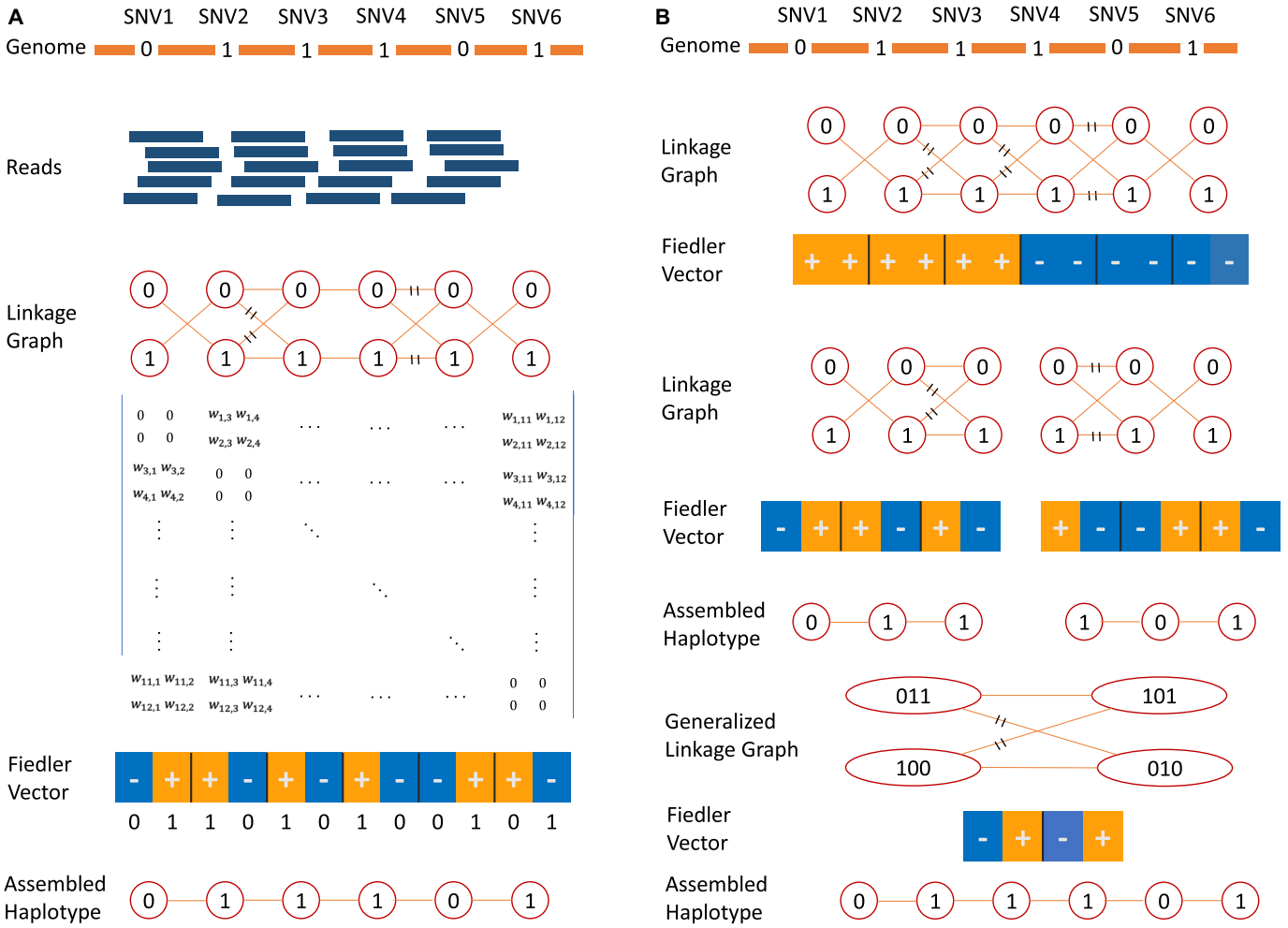

Figure 1. Illustration of SpecHap Algorithm. (**A**) An example with 6 variants loci demonstrating the haplotype phasing routine of SpecHap. The linkage graph is constructed from the sequenced reads and noisy edges are marked. The Fiedler vector is calculated from the adjacency matrix of the linkage graph. Haplotype can then be deduced by the sign of the Fiedler vector. (**B**) An example with 6 variants loci demonstrating how SpecHap constructs haplotype when Fieldler vector categorizes variants into two groups. SpecHap cut the graph into two sub-graphs accordingly and calculate the Fiedler vector respectively. The haplotype is deduced for the two sub-graphs. Then, SpecHap constructs a generalized linkage graph for the two haplotype blocks. Finally, SpecHap, connects the two haplotype blocks according to the Fielder vector.

Fragment extraction. We first extract the fragment information from the alignment and variant file. This step is completed with a refined version of ExtractHAIRs (Extract HAplotype Informative Reads), which is a program from the HapCUT2 software package that generates haplotype fragment information from aligned sequence and variants (18). The ExtractHAIRs was adopted since it supports accurate genotype information extraction for diverse sequencing protocols.

Logarithmic likelihood as edge weight. To calculate the edge weight of the linkage graph, we introduce a likelihood heuristic based on the Phred probability of nucleotide at variants loci in fragments. Consider $q[j]$ as the likelihood that nucleotide at variant locus j is incorrect on fragment R_i. Given haplotype h, the likelihood of observing fragment R_i with j variants is deduced as:

$$p(h) = \prod_{R_i[j]=h[j]} (1 - q_i[j]) \prod_{R_i[j]\neq h[j]} q_i[j] \quad (1)$$

Since we consider heterozygous variants, the other haplotype can be deduced as $\bar{h}$, the complementary of haplotype h. Given a self-complement haplotype pair $H = (h, \bar{h})$, the likelihood $p(H)$ that a fragment R_i have been observed is generalized as $MAX(p(h), p(\bar{h}))$. The likelihood that fragment set R have been observed is:

$$p(H) = \prod_i p(H) \quad (2)$$

Our linkage graph incorporates edges that represent conflicting haplotypes. To decrease the noise and computational load, we define the edge weight given conflicting haplotype H_1 and H_2 as:

$$E_{H_1} = max\left(log\left(\frac{p(H_1)}{p(H_2)}\right), 0\right) \quad (3)$$

$$E_{H_2} = max\left(log\left(\frac{p(H_2)}{p(H_1)}\right), 0\right) \quad (4)$$

Since edge weight is calculated as the logarithmic likelihood, the RatioCut is expressed as

$$RatioCut\left(V_s, \overline{V_s}\right) = \frac{|V_s| + \left|\overline{V_s}\right|}{|V_s|\left|\overline{V_s}\right|} \prod_{i \in V_s, j \in \overline{V_s}} p_{ij}$$

Thus, the finding the minimum graph cut guided by the Fiedler vector is equivalent to minimizing the product of likelihood of pairwise haplotype linkages to be removed.

Linkage graph construction. The extracted fragments might be treated as 'hyperedges' of covered variants loci. Fragments are treated as pairwise edges between covered loci. The weights of edges are calculated as logarithmic likelihood given all fragments as described in the previous section. The average genomic span between heterozygous variants loci in the linkage graph varies among different sequencing protocols. Therefore, linkage graphs with same cardinality might cover genomic regions of different lengths. Besides, with the introduction of third-generation sequencing, the average number of variants covered by fragments also increases.

Interpretation of fiedler vector. After eigendecomposition of the Laplacian of the constructed linkage-graph, we infer the haplotype according to the Fiedler vector. An expected Fiedler vector resembles the vector demonstrated in Figure 1A, in which elements corresponding to parallel alleles of given variant loci are assigned numbers with opposite signs. The haplotype is constructed based on the sign: alleles of different variant loci with the same sign belong to the same haplotype. However, our experiments demonstrated that this method might not work. There are two exceptions where haplotype cannot be directly inferred from the Fiedler vector. The first exception happens when the linkage graph suggests that the two conflicting haplotypes share equal likelihood at a specific variant locus. The resulting Fiedler vector possesses close-to-zero entries of the corresponding variant locus, which is pruned by our algorithm.

The second exception leads to a Fiedler vector that categorizes variants into two sub-blocks, as demonstrated in Figure 1B. Such Fiedler vectors are commonly seen with error-prone third-generation sequencing and linked-reads sequencing where reads with identical barcodes might originate from different DNA fragments (see Supplementary Material for detailed description). To infer haplotype with such a Fiedler vector, we partition the adjacency matrix accordingly and apply spectral graph analysis to each sub-block respectively. Since the Fiedler vector of sub-blocks might categorize variants further, SpecHap partitions variants recursively until haplotype can be deduced from the Fiedler vector. SpecHap will prune variants when we fail to resolve haplotype for sub-blocks with two variants. The sub-blocks are then merged by treating each as a vertex. A generalized linkage graph is constructed, and the merging of sub-blocks is guided by spectral analysis (Figure 1B). The SpecHap algorithm might be summarized as below:

Algorithm 1: SpecHap haplotype assembling routine

```
1  Construct linkage graph with adjacency matrix;
2  if No conflicting haplotype in graph then
3  |  Return haplotype with depth-first search;
4  else
5  |  Calculate Fielder vector;
6  |  |  if Fiiedler vector guides variant partitioning then
7  |  |  |  Partition the linkage graph accordingly;
8  |  |  |  Jump to Line 2;
9  |  |  |  Connect the phased haplotype blocks;
10 |  |  else
11 |  |  |  Return Haplotype with Fielder vector;
12 |  |  end
13 end
```

Haplotype assembling with linked-reads

In our experiments of phasing with 10× linked-reads, we identified reads with the same barcode providing conflicting haplotype linkage when comparing with the phase3 released trio-phased haplotype from 1000 genome project (30). It might be introduced by the rare but significant situations where reads with the same barcode are from two different DNA molecules. Thus, heuristics were taken to increase the accuracy of the assembled haplotype. First, variants are filtered by their allele depth and quality before the phasing algorithm. We also disallow a haplotype block from striding over 30 continuously filtered variants. Then, the covering range of each barcode is inferred based on the alignment results. In our implementation, a barcode can neither start nor end on an aligned read with mapping quality <30, and the overall barcode spanning length cannot be longer than 60 kb. A detailed description of adopted parameters can be found in Supplementary Table S1.

Haplotype assembling with Hi-C

For Hi-C data that introduce *trans* interactions between homologous chromosomes, SpecHap treats possible *trans* interactions as general errors and does not model them specifically. SpecHap filters read pairs with insertion larger than 40M base-pair to avoid linkage with higher *trans* interaction error rate.

Chromosome level haplotype construction

Since many mammalian species possess large genomes, the number of heterozygous variants on a single chromosome might be enormous. The construction of linkage graph and matrix operation will consume a massive amount of CPU and memory if all the heterozygous variants are phased at the same time. Thus, a divide-and-conquer strategy is applied. SpecHap divides the chromosome into multiple intervals with user-defined length. A depth-first search (DFS) will then be applied to find the connected variants within the interval. Then, connected variants in each interval are phased by applying spectral graph theory and the phased result of each interval is introduced when phasing its successor. The detailed description can be found in the Supplementary Material.

Algorithm complexity

The computation performed by SpecHap comprises four aspects: DFS, graph construction, calculation of graph Laplacian and analysis of the Fiedler vector. Let k be the number of eigenvectors to be calculated. Assume that a chromosome with N heterozygous variants is divided into overlapping intervals with each interval containing n variants. Let m be the number of fragments. Then, for each segmented interval, the time complexity for constructing a graph is $T(n) = O(n^2)$ and the time to run DFS is $T(n) = O(n + m)$. The construction of the unnormalized graph Laplacians involves matrix addition of complexity $T(n) = O(n^2)$. The eigen-calculation requires $T(n) = O(kn^2)$ time.

When calculating the Fielder vector recursively, SpecHap iteratively bisects the graph and there exists at most $\log(n)$ bisections. Hence the worst-case time complexity for iteratively calculating the Fielder vector is given by $T(n) = O(kn^2 \log n)$. SpecHap calculates the first two eigenvectors to obtain the Fiedler vector, which implies $k = 2$. The procedure is applied for $\frac{N}{n}$ intervals. Thus, the time complexity of the algorithm can be summarized as:

$$T(n) = O(Nn \log n)) \tag{5}$$

In most situations, SpecHap does not trigger a recursive procedure. The detailed time complexity analysis of different algorithms is in Supplementary Table S2.

Human genome dataset processing

NGS data. The high-coverage NGS alignment data for NA12878 and NA19240 were downloaded from the 1000 Genome Project with phase 3 release (30).

10× linked-reads data. 10x linked-reads for NA12878 were gathered from 10× genomics officials at https://support.10xgenomics.com/genome-exome/datasets/2.0.0/NA12878_WGS and NA19240 from https://support.10xgenomics.com/genome-exome/datasets/2.2.1/NA19240_WGS_v2. The alignment was performed with LongRanger2.2.1.

Hi-C data. Hi-C sequencing data were downloaded from NCBI PRJNA473369 for sample NA12878. Sequenced reads from seven selected cells (SRR7226668, SRR7226671, SRR7226678, SRR7226679, SRR7226681, SRR7226682 and SRR7226685) (7) were combined for further analysis. An additional set of WGS sequenced ∼36× Hi-C data with multiple enzyme protocol by Arima Genomics were acquired for sample NA12878 with accession SRR6675327 (31). For sample NA19240, ∼31× data were acquired from the 1000 genome SV project with accession NCBI PRJEB11418 (32). The alignment and insertion size of each reads-pair were determined with BWA mem with option -5SP (33).

PacBio SMRT data. PacBio SMRT ∼44× WGS data were acquired with NCBI accession number SRX1607993 for NA12878 from Genome in a Bottle Consortium and ∼120× SRR11363956 for sample NA19240 (34). Alignment was performed with minimap2 (35) with pre-set parameters.

Nanopore data. Nanopore reads were downloaded from ENA with accession PRJEB30620 for ∼40× NA12878 (36) and PRJEB26791 for ∼80× NA19240 (37). Alignment was performed with minimap2 (35) with pre-set parameters.

Ambystoma mexicanum genome dataset processing

We also acquired the PacBio SMRT sequencing reads of *Ambystoma mexicanum* with NCBI accession code PRJNA378970 (25). Alignment was performed by minimap2 to its chromosome-scale assembly GCA 002915635.2 (38). The called variant is accessible from EBI with accession number ERZ1668256.

Sequencing data simulation

The trio-phased sample HG00403 was taken as the haplotype for simulation on chromosomes 1, 21 and 22. Reads of length 150 bp and insert size 350 bp were simulated for NGS and linked-reads with wgsim and LRSIM (39) respectively with 30× coverage. 50× PacBio SMRT and ONT reads were also simulated based on the same haplotype by PBSIM (40) and DeepSimulator (41) with protocol-specific sequencing error rates.

Evaluation metrics

To benchmark the result of SpecHap, we adopted the following criteria which have been widely used metrics to access the completeness and accuracy of haplotype assembly. They are N50 of adjusted genomic span (AN50), the number of phased heterozygous sites, short-range switch error rate (mismatch error rate), and long-range switch error rate (switch error rate) (18,42–44). The metric AN50 stands for the N50 of adjusted genomic span, which is the span in reference base pairs from first to last phased variant multiplied by the fraction of phased variants over total variants spanned by the haplotype block. The number of phased variants was also summarized to assess the continuity of the haplotype. For haplotypes where the 'ground truth' was provided, the accuracy of the haplotype can be evaluated by the mismatch error rate and switch error rate. The mismatch error rate is calculated by the number of mismatch errors, that is, the number of errors that can be fixed by flipping the haplotype assignment at a single variant site, over the number of all possible mismatch errors. Similarly, the switch errors refer to the flipping of the haplotype assignments at each of two or more consecutive variants to attain the 'ground truth'.

To access the efficiency of programs, we collected the CPU time and peak memory consumption with Oracle Grid Engine on CentOS 7 with Intel Xeon CPU E7-4850 v2. To ensure a fair comparison, all the methods adopted the same set of fragments as input and default parameters are used for all sequencing protocols and methods. For SpecHap, we set the interval size to be 200 with interval overlap to be 60 for all sequencing protocols.

RESULTS

We assessed the performance of SpecHap with RefHap, HapCUT2, FastHare and DCHap for NGS, 10x Genomics

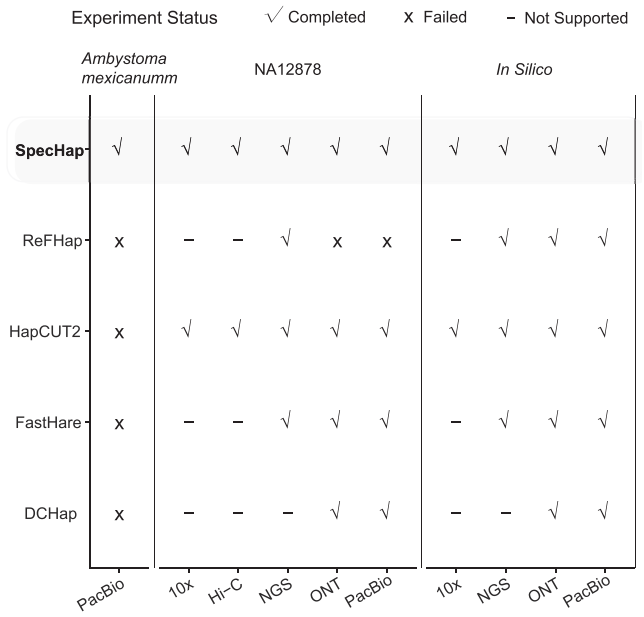

Figure 2. Overview of conducted experiment setting and status for samples with coverage less than 50×. Software that does not support a specific data type is marked with '–'. For *Homo sapiens*, methods that exceed the 24 CPU hours limit are marked as 'Failed'. For *Ambystoma mexicanum*, methods that exceed 5 CPU days and 40GB peak memory usage are marked as 'Failed'.

linked-read, Hi-C, PacBio SMRT and ONT sequencing. *In silico* data for HG00403 and WGS data for NA12878 and NA19240 were adopted for benchmarking. The trio-phased haplotype from the 1000 Genomes Project was taken as the 'ground truth' and variants set to assemble haplotype for each data type. The sample NA19240 contains ∼1.5-fold more heterozygous variants. We also conducted experiments phasing ∼32× PacBio SMRT for *Ambystoma mexicanum.* Figure 2 illustrates the overview of the conducted experiment setting and status for datasets under 50× coverage. SpecHap managed to pass all experiment settings, while the other four software packages either did not support a specific data type or failed to complete the task within the usage boundary (24 CPU hours for *Homo sapiens*, 5 CPU days and 40GB peak memory usage for *Ambystoma mexicanum*). The overall and per-haplotype-block benchmark statistics were demonstrated in Figures 3 and 4 correspondingly.

SpecHap demonstrated runtime and memory efficiency on *in silico* data

SpecHap assembles the haplotype by graph bisection based on the min-cut heuristic guided by spectral graph theory. Since many algorithms model diploid SIH with graph bisection, some other heuristics for graph cut are adopted (18,20,22). To evaluate the efficiency of the min-cut heuristic implementation based on spectral graph theory, we compared the graph cut module of SpecHap with the greedy maximum likelihood cut heuristic of HapCUT2. We performed simulations on graphs with 50–500 heterozygous SNVs for 10×, 30× and 50× read coverage with conflicting haplotypes exhibited on a single variant locus. The CPU time of the two graph-cut routines was plotted in Supplementary Figure S1. Comparing with HapCUT2, the CPU time of the SpecHap graph-cut routine demonstrates no statistical differences for data with different coverage. When the graph contains 50 heterozygous SNVs, both graph-cut routines finish within 60 CPU microseconds. When the number of SNVs in the graph increases to 500, SpecHap finishes within 500 CPU microseconds for all coverage profiles. HapCUT2, however, spends more than 1000 CPU microseconds for data with 10× coverage and more than 4000 CPU microseconds for data with 50X coverage.

We then used *in silico* data to benchmark SpecHap with the existing method focusing on the third-generation sequencing protocol. When phasing with PacBio SMRT sequence, SpecHap outperformed all four existing methods considering both CPU time and peak memory usage based on the simulation. SpecHap was around 40 times faster than HapCUT2, 100 times faster than ReFHAP and as fast as FastHare. DCHap did not function efficiently and accurately enough on simulations with PacBio SMRT reads. While assembling haplotype with ONT reads, SpecHap achieved 20 times faster than HapCUT2, 30 times faster than ReFHAP, 2 times faster than DCHap and as fast as FastHare. FastHare, however, demonstrated significantly higher error rates among methods.

We also benchmarked SpecHap's performance with multiple interval sizes on both simulated PacBio SMRT and ONT data as demonstrated in Supplementary Table S3. The CPU time of SpecHap increases as the interval size increases. When the interval size increased to 1000, the CPU time of SpecHap tripled. It is also noticeable that with different interval sizes, SpecHap assembled haplotypes might be different. The small variation in statistics was considered acceptable comparing with the results of other software packages.

SpecHap persisted runtime and memory efficiency on WGS sequenced data

SpecHap persisted its efficiency on WGS data. For high coverage NGS data, SpecHap completed the haplotype assembling with the least CPU usage (2 CPU minutes) and peak memory (109 MB) for NA12878. As for NA19240, SpecHap finished with only 329 CPU seconds and 157MB memory. ReFHap did not scale well while FastHare and HapCUT2 consumed excessive memory (18GB and 25GB respectively) on NA19240. For Hi-C data, SpecHap outperformed HapCUT2 on computational load by completing the haplotype assembly within minutes of CPU time, with minimum memory consumption (∼200 MB) on both ∼36× NA12878 sample ∼30× NA19240 sample. With 10× linked-reads, SpecHap and HapCUT2 achieved comparable speed. However, HapCUT2 requires additional computation on fragment linking (32 CPU hours on the linkage of fragments with the script provided by HapCUT2 for sample NA12878).

As for the PacBio SMRT data, SpecHap similarly demonstrated its efficiency by completing the assembly with around 13 CPU minutes and 120MB peak memory for the ∼44× NA12878 sample. When the sequencing depth

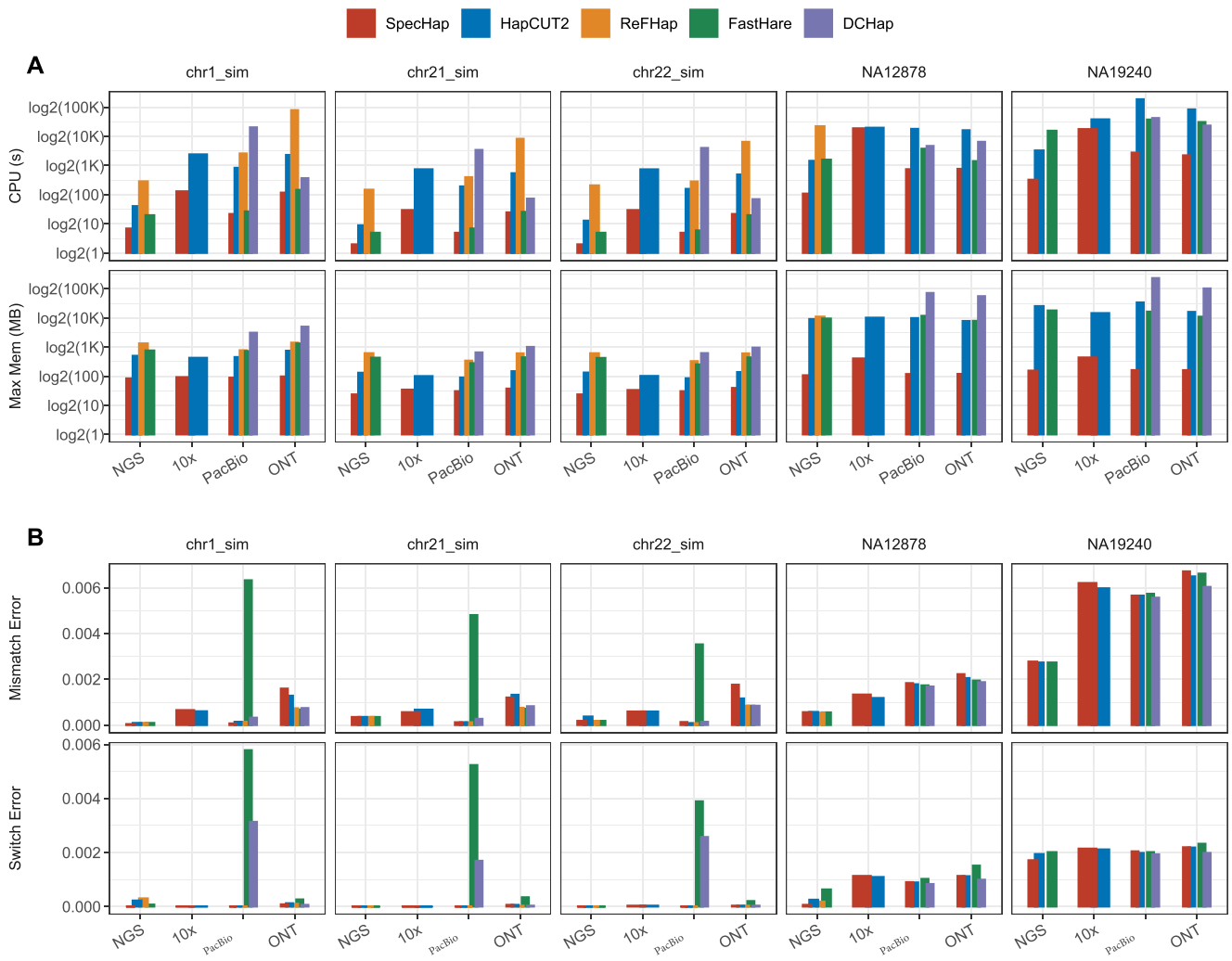

Figure 3. Overall benchmarked data on both simulation and WGS sample NA12878 for diverse sequencing protocol. (**A**) Log_2-scaled CPU and peak memory usage, in seconds and Megabytes respectively. Value exceeding 30 000 CPU seconds and 15GB peak memory usage is capped. (**B**) Overall switch error rate and mismatch error rate are calculated as the number of errors divided by the number of possible errors.

increase to ∼120× for sample NA19240, most software started to consume excessive memory (16GB for FastHare, 230GB for DCHap and 30GB for HapCUT2), while HapCUT2 took the most time to complete (52 CPU hours). SpecHap, however, remained efficient (∼3000 CPU seconds) with minimum memory usage (within 200MB). For the ONT data, SpecHap was able to assemble the haplotype with around 10 CPU minutes on the ∼40× NA12878 dataset. When the sequencing depth increased to 80X on sample NA19240, SpecHap persisted with its efficiency and completed haplotype assembly with 2257 CPU seconds and 163MB peak memory consumption. While the second most efficient method, FastHare, took ∼11 CPU hours and 16GB peak memory to complete. ReFHap did not scale well due to time limit exceeded on both PacBio SMRT and ONT data. The CPU time and peak memory usage for all the methods were summarized in Figure 3:A and Supplementary Table S4.

SpecHap accurately phased individual NA12878 and NA19240 on diverse sequencing protocols

NGS data. To access SpecHap on NGS data, we first took a high coverage sequenced data for individual NA12878 from the 1000 Genomes Project. As displayed in Figure 3B, the switch error rate for SpecHap was less than ReFHap, the second-most accurate algorithm among all other software. HapCUT2 shared a virtually similar switch error of ReFHap while keeping more variants unpruned. FastHare, however, demonstrated a higher switch error rate with more pruned variants compared with SpecHap and HapCUT2. As for the mismatch error rate, all four methods shared comparable results with no statistically significant difference. As for the measurement of the dataset for sample NA19240, SpecHap persisted its accuracy with the least switch error. The AN50 metric for haplotypes generated by different methods demonstrated no statistically significant difference for both individuals.

Figure 4. Log2 scaled violin plot of benchmarked data on a per-haplotype-block scale. (**A**) Switch error rate. Methods with zero switch error on the simulation dataset are not showed (10× chr21 sim, NGS chr21 sim, chr22 sim, PacBio chr1 sim, chr21 sim and chr22 sim). (**B**) Mismatch error rate. (**C**) Adjusted span, defined as haplotype block span times ratio of the number of phased SNPs over the number of total SNPs, in base pair. (**D**) Number of phased SNPs.

Hi-C data. Since most methods are not specialized for haplotype assembling with Hi-C data, we only compared the quality of haplotype assembly generated by SpecHap and HapCUT2. Three sets of sequencing data were chosen to benchmark the completeness and accuracy of the assembled haplotype: NA12878, ∼36× NA12878_Arima and ∼31× NA19240. HapCUT2 models the translocation error by iteratively estimating the probability of reads originating from different homologous chromosomes (18). SpecHap achieves a comparable switch error rate and AN50 among three datasets (Supplementary Figures S3 and S4).

10× linked-read data. We benchmarked SpecHap on 10× genomics linked-reads. The 10x linked-reads label short reads that originated from a single long DNA fragment with the same barcode. Although DNA fragments generally span around 60k base-pair range, reads might be sparsely distributed across the original fragment. It is also possible that two reads that shared the same barcode originated from different DNA molecules. The dataset we acquired has around 50X coverage for both individuals NA12878 and NA19240; reads were filtered so that only fragments with white-listed barcodes were kept. The same sets of variants were used to extract fragment information. SpecHap achieved a compa-

rable accuracy and AN50 on both datasets with HapCUT2 (Supplementary Figure S2).

PacBio SMRT data and ONT Data. PacBio SMRT sequencing and ONT sequencing are known for their error-prone (∼10%) long reads. For individual NA12878, on PacBio SMRT data, SpecHap achieved a virtually identical switch error rate compared to HapCUT2. FastHare, with second-best efficiency, demonstrated a higher switch error rate with most variants pruned. DCHap, with slightly more accurate results, assembled the haplotype with the lowest AN50 and pruned the second most variants. All methods demonstrated no statistically significant differences concerning the mismatch error rate. On ONT data, SpecHap similarly demonstrated its accuracy with a comparable switch error rate. DCHap, with the least switch errors, pruned >2000 SNVs than SpecHap and HAPCUT2. ReFHap failed to finish the assembly process with both PacBio SMRT reads and ONT reads due to excessive time consumption. The switch and mismatch error rate, AN50 and the number of phased SNVs for all methods were summarized in Supplementary Tables S5 and S6.

SpecHap demonstrated scalability by phasing with giant diploid genome

SpecHap also demonstrated high scalability by assembling the ∼32× PacBio SMRT reads with N50 read length around 14.2 kb of an *Ambystoma mexicanum* individual, which possesses 32 billion base-pair-long genome (25,38). As illustrated in Supplementary Figure S5, SpecHap was able to finish the assembly within 6 CPU hours with only 945MB of peak memory consumption, while all other methods failed to finish within the time limit (5 CPU days) and memory limit (40GB). The assembled haplotype blocks have the AN50 of 206Kbp within a total of 20375931 number of phased SNPs, which is 99.8% of total heterozygous SNPs of *Ambystoma mexicanum*). The most-heterozygous variants-phased haplotype block has an adjusted span of more than 2 Mb. For per-haplotype-block adjusted span and the number of phased variants, see Supplementary Figure S6.

DISCUSSION

SpecHap, a novel diploid SIH algorithm, supports sequencing data with different coverage from diverse platforms. SpecHap transforms diploid SIH into a linear algebra problem by applying spectral graph theory. The allele partitioning guided by Fiedler Vector might be interpreted as a min-cut heuristic on our linkage graph (28). SpecHap adopts divide-and-conquer to accelerate the computation by dividing chromosomes into intervals of user-defined size. A larger interval size might affect SpecHap's efficiency. Although there is no guarantee that SpecHap provides optimal solutions and the interval size might affects the phasing result, we demonstrated that our model succeeded in efficiently and accurately assembling haplotypes with diverse sequencing reads for individual NA12878 and NA19240. SpecHap is also scalable to phase one of the largest sequenced genomes of *Ambystoma mexicanum* with around ∼32× PacBio SMRT data, showing it to be a promising tool for future research on the evolutionary history of amphibians and other organisms with immense genome scale.

Long-read sequencing introduces significant advances considering the completeness and continuity of assembled haplotypes. However, PacBio SMRT and ONT long-reads maintain a higher per-base error rate and may fail to accurately identify SNVs, particularly heterozygous ones (45). In our experiments, we adopted the trio-phased high-quality variants set for NA12878 and NA19240 from the 1000 Genome Project as input for all methods. Since most diploid SIH methods, including SpecHap, require a high-quality set of variants to conduct phase, 30X Illumina short reads sequencing might be performed to obtain reliable calls for SNVs, short insertions and deletions. Some recent approaches based on deep learning were also able to identify variants accurately from long-read sequencing (46).

In this study, we introduced a novel diploid SIH algorithm SpecHap and demonstrated its robustness and efficiency. By transforming diploid SIH into a linear algebra problem, SpecHap assembled haplotype with ultrafast speed while preserving comparable accuracy. Moreover, a comprehensive analysis on the influence of technological specific error over phasing quality may be conducted for Hi-C and 10× linked-reads. Although our algorithm works on the diploid genome, generalization towards high ploidy is expected. For multiploidy genomes, it is possible to encounter multi-allelic variants. Since we are expecting multiple haplotypes, the current graph bisection model might not fit when the ploidy increases. Besides, the determination of haplotype-resolved structural variation might also be an important feature to be introduced in the future. Haplotype-resolved structural variations are often determined with haplotype-resolved de novo assembly (47) and most SIH software packages including SpecHap support SNPs, short insertions and deletions only. Phasing with structure variations is challenging since they introduce complex genome rearrangement events. To assemble haplotype-resolved structural variation, a refined linkage graph that incorporates genome rearrangements is expected in the future.

DATA AVAILABILITY

All the data used in this paper can be retrieved from public databases. All the experiments are reproducible with the dedicated version of the software with default arguments. SpecHap source code is deployed at https://github.com/deepomicslab/SpecHap and a copy is attached in the supplemental material. To allow reproduction of the results in this manuscript, the commands to run programs during simulation, alignment, fragment extraction, and benchmarking were also attached. The fragment information was extracted with a refined version of ExtractHAIRs (18) which is packed with SpecHap. We adopted the implementation of FastHare from Duitama et al. (44) for benchmarking.

SUPPLEMENTARY DATA

Supplementary Data are available at NAR Online.

ACKNOWLEDGEMENTS

We would like to express sincere gratitude to Dr Zijun Xiong of the Chinese Academy of Sciences for suggestions on data collection. We appreciate Miss Santu for her creation of an image for *Ambystoma mexicanum*. We would also like to thank Dr Wenlong Jia and Mr. Bowen Tan for their valuable assistance and advice. We would like to thank Dr Lu Zhang for his suggestion on pre-processing 10X linked reads. Finally, we would like to express our gratitude towards the editor and anonymous reviewers whose constructive suggestions greatly contributed to this manuscript.

FUNDING

Innovation and Technology Fund [PRP/052/19FX]. Funding for open access charge: Innovation and Technology Fund.

Conflict of interest statement. None declared.

REFERENCES

1. Zheng,G.X.Y., Lau,B.T., Schnall-Levin,M., Jarosz,M., Bell,J.M., Hindson,C.M., Kyriazopoulou-Panagiotopoulou,S., Masquelier,D.A., Merrill,L., Terry,J.M. *et al.* (2016) Haplotyping germline and cancer genomes with high-throughput linked-read sequencing. *Nat. Biotechnol.*, **34**, 303–311.
2. Glusman,G., Cox,H.C. and Roach,J.C. (2014) Whole-genome haplotyping approaches and genomic medicine. *Genome medicine*, **6**, 73.
3. Tewhey,R., Bansal,V., Torkamani,A., Topol,E.J. and Schork,N.J. (2011) The importance of phase information for human genomics. *Nat. Rev. Genet.*, **12**, 215–223.
4. The,1000 Genomes Project Consortium (2012) An integrated map of genetic variation from 1,092 human genomes. *Nature*, **491**, 56.
5. International,HapMap 3 Consortium, Altshuler,D.M., Gibbs,R.A., Peltonen,L., Altshuler,D.M., Gibbs,R.A., Peltonen,L., Dermitzakis,E., Schaffner,S.F., Yu,F. *et al.* (2010) Integrating common and rare genetic variation in diverse human populations. *Nature*, **467**, 52.
6. Onuchic,V., Lurie,E., Carrero,I., Pawliczek,P., Patel,R.Y., Rozowsky,J., Galeev,T., Huang,Z., Altshuler,R.C., Zhang,Z. *et al.* (2018) Allele-specific epigenome maps reveal sequence-dependent stochastic switching at regulatory loci. *Science*, **361**, eaar3146.
7. Tan,L., Xing,D., Chang,C.-H., Li,H. and Xie,X.S. (2018) Three-dimensional genome structures of single diploid human cells. *Science*, **361**, 924–928.
8. Begnini,A., Tessari,G., Turco,A., Malerba,G., Naldi,L., Gotti,E., Boschiero,L., Forni,A., Rugiu,C., Piaserico,S. *et al.* (2010) PTCH1 gene haplotype association with basal cell carcinoma after transplantation. *Br. J. Dermatol.*, **163**, 364–370.
9. Conrad,D.F., Jakobsson,M., Coop,G., Wen,X., Wall,J.D., Rosenberg,N.A. and Pritchard,J.K. (2006) A worldwide survey of haplotype variation and linkage disequilibrium in the human genome. *Nat. Genet.*, **38**, 1251.
10. Musone,S.L., Taylor,K.E., Lu,T.T., Nititham,J., Ferreira,R.C., Ortmann,W., Shifrin,N., Petri,M.A., Kamboh,M.I., Manzi,S. *et al.* (2008) Multiple polymorphisms in the TNFAIP3 region are independently associated with systemic lupus erythematosus. *Nat. Genet.*, **40**, 1062.
11. Trégouët,D.-A., König,I.R., Erdmann,J., Munteanu,A., Braund,P.S., Hall,A.S., Großhennig,A., Linsel-Nitschke,P., Perret,C., DeSuremain,M. *et al.* (2009) Genome-wide haplotype association study identifies the SLC22A3-LPAL2-LPA gene cluster as a risk locus for coronary artery disease. *Nat. Genet.*, **41**, 283.
12. Qi,J., Wang,T.-J., Chen,L.-P., Wang,X.-F., Wang,M.-N. and Wu,J.-H. (2018) Utility of next-generation sequencing methods to identify the novel HLA alleles in potential stem cell donors from Chinese Marrow Donor Program. *Int. J. Immunogenet.*, **45**, 225–229.
13. Panconesi,A. and Sozio,M. (2004) Fast Hare: A Fast Heuristic for Single Individual SNP Haplotype Reconstruction. In: Jonassen,I. and Kim,J. (eds). *Algorithms in Bioinformatics. WABI 2004. Lecture Notes in Computer Science*. Vol. **3240**. Springer, Berlin Heidelberg, pp. 266–277.
14. Selvaraj,S., Dixon,J.R., Bansal,V. and Ren,B. (2013) Whole-genome haplotype reconstruction using proximity-ligation and shotgun sequencing. *Nat. Biotechnol.*, **31**, 1111–1118.
15. Burton,J.N., Adey,A., Patwardhan,R.P., Qiu,R., Kitzman,J.O. and Shendure,J. (2013) Chromosome-scale scaffolding of de novo genome assemblies based on chromatin interactions. *Nat. Biotechnol.*, **31**, 1119–1125.
16. Kaplan,N. and Dekker,J. (2013) High-throughput genome scaffolding from in vivo DNA interaction frequency. *Nat. Biotechnol.*, **31**, 1143–1147.
17. Zhang,L., Zhou,X., Weng,Z. and Sidow,A. (2019) Assessment of human diploid genome assembly with 10x Linked-Reads data. *GigaScience*, **8**, giz141.
18. Edge,P., Bafna,V. and Bansal,V. (2017) HapCUT2: robust and accurate haplotype assembly for diverse sequencing technologies. *Genome Res.*, **27**, 801–812.
19. Pollard,M.O., Gurdasani,D., Mentzer,A.J., Porter,T. and Sandhu,M.S. (2018) Long reads: their purpose and place. *Hum. Mol. Genet.*, **27**, R234–R241.
20. Bansal,V. and Bafna,V. (2008) HapCUT: an efficient and accurate algorithm for the haplotype assembly problem. *Bioinformatics*, **24**, i153–i159.
21. Li,Y. and Lin,Y. (2020) DCHap: a divide-and-conquer haplotype phasing algorithm for third-generation sequences. *IEEE/ACM Trans. Comput. Biol. Bioinf.*, https://doi.org/10.1109/TCBB.2020.3005673.
22. Duitama,J., Huebsch,T., McEwen,G., Suk,E.-K. and Hoehe,M.R. (2010) ReFHap: a reliable and fast algorithm for single individual haplotyping. In: *Proceedings of the First ACM International Conference on Bioinformatics and Computational Biology*. Association for Computing Machinery. pp. 160–169.
23. Chen,J., Hero,A.O. III and Rajapakse,I. (2016) Spectral identification of topological domains. *Bioinformatics*, **32**, 2151–2158.
24. Lee,A.B., Luca,D. and Roeder,K. (2010) A spectral graph approach to discovering genetic ancestry. *Ann. Appl. Stat.*, **4**, 179–202.
25. Nowoshilow,S., Schloissnig,S., Fei,J.-F., Dahl,A., Pang,A.W.C., Pippel,M., Winkler,S., Hastie,A.R., Young,G., Roscito,J.G. *et al.* (2018) The axolotl genome and the evolution of key tissue formation regulators. *Nature*, **554**, 50–55.
26. Weisrock,D.W., Hime,P.M., Nunziata,S.O., Jones,K.S., Murphy,M.O., Hotaling,S. and Kratovil,J.D. (2018) Surmounting the Large-Genome "Problem" for Genomic Data Generation in Salamanders. In: Hohenlohe,P.A. and Rajora,O.P. (eds). *Population Genomics: Wildlife. Population Genomics*. Springer International Publishing, Cham, pp. 1–28.
27. Hagen,L. and Kahng,A.B. (1992) New spectral methods for ratio cut partitioning and clustering. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, **11**, 1074–1085.
28. Von Luxburg,U. (2007) A tutorial on spectral clustering. *Stat. Comput.*, **17**, 395–416.
29. Chung,F.R.K. and Graham,F.C. (1997) In: *Spectral graph Theory*. American Mathematical Society.
30. Sudmant,P.H., Rausch,T., Gardner,E.J., Handsaker,R.E., Abyzov,A., Huddleston,J., Zhang,Y., Ye,K., Jun,G., Hsi-Yang Fritz,M. *et al.* (2015) An integrated map of structural variation in 2,504 human genomes. *Nature*, **526**, 75–81.
31. Ghurye,J., Rhie,A., Walenz,B.P., Schmitt,A., Selvaraj,S., Pop,M., Phillippy,A.M. and Koren,S. (2019) Integrating Hi-C links with assembly graphs for chromosome-scale assembly. *PLoS Comput. Biol.*, **15**, e1007273.
32. Clarke,L., Fairley,S., Zheng-Bradley,X., Streeter,I., Perry,E., Lowy,E., Tassé,A.-M. and Flicek,P. (2017) The international Genome sample resource (IGSR): A worldwide collection of genome variation incorporating the 1000 Genomes Project data. *Nucleic Acids Res*, **45**, D854–D859.
33. Li,H. and Durbin,R. (2009) Fast and accurate short read alignment with Burrows–Wheeler transform. *Bioinformatics*, **25**, 1754–1760.
34. Vollger,M.R., Dishuck,P.C., Sorensen,M., Welch,A.E., Dang,V., Dougherty,M.L., Graves-Lindsay,T.A., Wilson,R.K., Chaisson,M.J.P. and Eichler,E.E. (2019) Long-read sequence and assembly of segmental duplications. *Nat. Methods*, **16**, 88–94.

35. Li,H. (2018) Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, **34**, 3094–3100.
36. Bowden,R., Davies,R.W., Heger,A., Pagnamenta,A.T., de Cesare,M., Oikkonen,L.E., Parkes,D., Freeman,C., Dhalla,F., Patel,S.Y. *et al.* (2019) Sequencing of human genomes with nanopore technology. *Nat. Commun.*, **10**, 1869.
37. De Coster,W., De Rijk,P., De Roeck,A., De Pooter,T., D'Hert,S., Strazisar,M., Sleegers,K. and Van Broeckhoven,C. (2019) Structural variants identified by Oxford Nanopore PromethION sequencing of the human genome. *Genome Res.*, **29**, 1178–1187.
38. Smith,J.J., Timoshevskaya,N., Timoshevskiy,V.A., Keinath,M.C., Hardy,D. and Voss,S.R. (2019) A chromosome-scale assembly of the axolotl genome. *Genome Res*, **29**, 317–324.
39. Luo,R., Sedlazeck,F.J., Darby,C.A., Kelly,S.M. and Schatz,M.C. (2017) LRSim: a linked-reads simulator generating insights for better genome partitioning. *Comput. Struct. Biotechnol. J.*, **15**, 478–484.
40. Ono,Y., Asai,K. and Hamada,M. (2012) PBSIM: PacBio reads simulator—toward accurate genome assembly. *Bioinformatics*, **29**, 119–121.
41. Li,Y., Han,R., Bi,C., Li,M., Wang,S. and Gao,X. (2018) DeepSimulator: a deep simulator for Nanopore sequencing. *Bioinformatics*, **34**, 2899–2908.
42. Snyder,M.W., Adey,A., Kitzman,J.O. and Shendure,J. (2015) Haplotype-resolved genome sequencing: experimental methods and applications. *Nat. Rev. Genet.*, **16**, 344–358.
43. Choi,Y., Chan,A.P., Kirkness,E., Telenti,A. and Schork,N.J. (2018) Comparison of phasing strategies for whole human genomes. *PLoS Genet*, **14**, e1007308.
44. Duitama,J., McEwen,G.K., Huebsch,T., Palczewski,S., Schulz,S., Verstrepen,K., Suk,E.-K. and Hoehe,M.R. (2012) Fosmid-based whole genome haplotyping of a HapMap trio child: evaluation of single individual haplotyping techniques. *Nucleic Acids Res*, **40**, 2041–2053.
45. Edge,P. and Bansal,V. (2019) Longshot enables accurate variant calling in diploid genomes from single-molecule long read sequencing. *Nat. Commun.*, **10**, 4660.
46. Luo,R., Sedlazeck,F.J., Lam,T.-W. and Schatz,M.C. (2019) A multi-task convolutional deep neural network for variant calling in single molecule sequencing. *Nat. Commun.*, **10**, 998.
47. Chaisson,M.J.P., Sanders,A.D., Zhao,X., Malhotra,A., Porubsky,D., Rausch,T., Gardner,E.J., Rodriguez,O.L., Guo,L., Collins,R.L. *et al.* (2019) Multi-platform discovery of haplotype-resolved structural variation in human genomes. *Nat. Commun.*, **10**, 1784.