SpecHap: a fast haplotyping method based on spectral graph theory

Yonghan Yu[1], Lingxi Chen[1], Bowen Tan[1], Wenlong Jia[1], and Shuai Cheng Li[1,*]

[1]Department of Computer Science, City University of Hong Kong, Hong Kong, China
[*]Correspondence should be addressed to S.C.L. (shuaicli@cityu.edu.hk).

ABSTRACT

Haplotype phasing has fundamental implications in human genetics study. With rapid advancing of technology, novel sequencing protocols including linked-reads and third generation sequencing technology which enables whole genome haplotype phasing have been developed. Many phasing software, most of which bases on resources intensive heuristic algorithms, have been developed to handle various sequencing technologies. Here, we present a novel approach called SpecHap that adopts spectral graph theory for fast diploid haplotype construction from diverse sequencing protocols. On both simulated and real sequencing data set, SpecHap achieved higher sensitivity than existing tools, demonstrating high computational efficiency while preserving the length of the phased genomic regions.

Introduction

Human genomes are diploid with paternal and maternal sets of chromosomes. The two haplotypes, sequential presentation of heterogeneous single nucleotide polymorphism (SNP), small insertion and deletion (InDel), depict the most genetic variation of humankind. Whole-genome variants phasing, the reconstruction of individual haplotypes characterizes the DNA differences between homologous chromosomes, which has essential indications in several human genetic researches, including diseases susceptibility (F et al., 2006; Musone et al., 2008), allelic differential epigenomic regulations (Onuchic et al., 2018; Tan et al., 2018), and population development (Consortium et al., 2012, 2010).

Recent computational methods for haplotype phase generally fall into three categories(Glusman et al., 2014): 1) haplotype assembly for individual genome (HASH(Bansal et al., 2008), LongRanger(Zheng et al., 2016), HapCut2(Edge et al., 2017); 2) genetic analysis with inheritance related data (IBD analysis); 3) haplotype inference based on population data (SHAPEIT (Delaneau et al., 2008)). In this study, we only focus on the methodology of the haplotype assembly. With advances in high-throughput sequencing, multiple sequencing protocals enable credible identifications and linkages of genetic variants and greatly assist the construction of individual haplotype. Paired-end reads of whole-genome sequencing (WGS), and Hi-C data can be respectively used to study proximal and distal haplotypes (Selvaraj et al., 2013; Tan et al., 2018). Meanwhile, the 10x linked-reads technology leverages barcodes to construct long-range phased blocks (Zheng et al., 2016).

Existing tools utilize various approaches to mitigate sequencing noises in the context of linked-reads and long molecule reads. For instance, deprecated method HASH(Bansal et al., 2008) leverages Monte Carlo Sampling and considers the optimal probabilistic phasing configuration supported by heterogeneous SNP on pair-end reads and long-range barcode information. With HASH as the core, LongRanger(Zheng

et al., 2016) is the official tool to analysis and process the 10x Genomics linked reads. Instead of simply considering the optimal probabilistic phasing configuration supported by SNP on pair-end reads, LongRanger further incorporates the long molecule's barcode information so that false events like one barcode carries two molecules on opposite haplotypes of the same locus receive lower level of trust(Zheng et al., 2016). While the phasing process is encapsulated in LongRanger pipeline in a "black box" manner, it is not easy neither to catch the algorithm design nor to run the phasing part alone. More importantly, coded with Monte Carlo Sampling, both HASH and LongRanger are computational inefficient due to extra calculation. HapCUT2(Edge et al., 2017) is the state-of-art approach for haplotype assembly, which iteratively optimizes the maximum likelihood of the assembled haplotypes consistent with all supported read linkages associated heterogeneous SNP sites based on Max-cut search. HapCUT2 works with adequate speed and accuracy across diverse sequencing protocols like NGS short reads, 10x linked reads, PacBio/Oxford long reads, Hi-C reads (Edge et al., 2017). However, Max-cut algorithm may overextend phase block in region without high confident linkage. In order to preserve phasing accuracy, HapCUT2 use greedy splitting guided by posterior haplotype probability (Edge et al., 2017). Our experience with these tools shows that HapCUT2 frequently overextends with low-quality block.

In summary, high-throughput data have been widely used for haplotype assembly through computational means such as Monte Carlo Markov Chain (MCMC) model (Bansal et al., 2008; Zheng et al., 2016) and max-cut-based graph algorithms (Edge et al., 2017). However, these methods are computationally intensive, especially in the era of linked reads and long reads sequencing protocol, and their results are often affected by sequencing process fluctuations. Thus, we proposed a novel spectral graph theory-based phasing model with super-fast speed and high sensitivity compared with existing tools.

Result

Overview of SpecHap Algorithm

We developed an alternative algorithm, SpecHap, that leverages spectral graph analysis to realize genome-wide haplotyping in diploid. In our graph model, each heterozygous locus is represented by two nodes, each representing one of the two parallel alleles (**Fig3:a-c**). Two types of linkage (direct and indirect) of heterozygous alleles are integrated into edges (**Fig3:c**). The edge weight is calculated from the mapping quality of the sequencing reads supporting the linkage (see **Methods**, **Fig3:d**). SpecHap partitions chromosomes into sliding windows, and constructs their linkage graphs separately. The unnormalized graph Laplacian is then calculated on the adjacency matrix of the ladder-shape graph, and a cut leading to two haplotype strings guided by the Fielder Vector is performed (Chung and Graham, 1997) (**Fig3:e-f**). Instead of calculating posterior probability, SpecHap locates conflicting region when the Fielder Vector fails to provide two haplotypes, and cuts the phase block accordingly (**Fig3:g**). The results are further merged to the chromosome-level haplotype. Finally, SpecHap outputs the VCF file that records phased variants with block identifiers.

Evaluation of SpecHap Performance

We evaluated SpecHap's performance to both HapCut2 and longranger, on general paired-end, Hi-C, and 10X sequencing protocol, by comparing their phase set with template haplotypes from 1000 Genomes Institution. Simulations with sample HG00403 were conducted first to evaluate the feasibility in using the spectral graph method for haplotype phasing. For paired-end and 10X sequencing protocol, we adopted the sequenced reads of sample NA12878 and NA19240 from 1000 Genomes Institution and 10X Genomics

respectively. While for Hi-C, we combined data from seven cells of sample NA12878 from previous study. To justify the comparison, GATK was used in calling SNPs as it was in the longranger pipeline. Since different cells were used for sequencing in different protocols, only the intersections of the called SNPs and SNPs from template haplotype were compared. All the software and pipeline were executed with their default argument. Details are in the Method section.

In order to assess the credibility of the haplotypes generated by SpecHap, we first evaluated their flipped accuracy, that is, the percentage of correct flips in a haplotype string of specific phase block. This gives us a measure of the local consistency of the phased block with the template haplotype from 1000 Genome Institute. We examined each generated phase block for both simulated and real sequencing data. We observed that the results generated by SpecHap are as good as HAPCUT2 with regard to paired-end and Hi-C sequencing protocol **(Fig1:a, Supplementary Fig S3, S7)**. For 10x linked reads, HAPCUT2 fared better on the simulated sample while there is no significant difference among SpecHap, HAPCUT2 and longranger on either NA12878 and NA19240 **(Fig1:a, Supplementary Fig S5, S8)**.

Next, we evaluated whether SpecHap could maintain the global consistency of phase block by examining the phasing accuracy, that is, the percentage of SNPs that are correctly phased in corresponding block; a single flip error might lead to inaccurate haplotype especially in large phase block. For each dataset with general paired-end and Hi-C protocol, SpecHap generated similar haplotype compared to HAPCUT2 **(Fig1:a, Supplementary Fig S2, S6)**. For 10x linked reads, we first evaluated the phasing accuracy on the simulated data. We notice that SpecHap and longranger produced similar results while HAPCUT2 generated less inaccurate block **(Fig1:a, Supplementary Fig S4, S9)**. However, HAPCUT2 frequently overextends haplotype in highly fluctuated loci with real 10x dataset on both NA12878 and NA19240, leading to many extremely long blocks with low accuracy **(Supplementary FigS9)**. Instead, SpecHap and longranger accurately clipped haplotype blocks with low confidence, thus potentially limited the number of inaccurate large block. We also observed that on 10x sample NA12878, where there are large data fluctuation, SpecHap produced smaller block compared to longranger, thus avoiding unreliable long haplotypes.

The distribution of block size is also measured on different phasing protocol. It was observed that the results of SpecHap and HAPCUT2 share high similarity with respect to block size on both paired-end and Hi-C protocol **(Fig2:b,c,e,g,h)**. With 10x linked reads, phase block by SpecHap shows the same pattern with those of HAPCUT2 and longranger **Fig2:f**. On sample NA12878, SpecHap generated shorter block. While on sample NA19240, SpecHap clips frequently in fluctuated loci, leading to more blocks with less than 100 variants (**Fig2:d**).

Finally, we assessed the CPU time and memory usage for all three software. When phasing with paired-end and Hi-C protocal, SpecHap is more than three times faster than HAPCUT2 **(Fig1:b)**. SpecHAP finished phasing with 80X coverage data set NA12878, NA19239 and NA19240 within 5 minutes with around 150MB memory consumption while HAPCUT2 took hours to complete with gigabytes-level memory consumption. When phasing 10x linked reads with around 100X coverage, SpecHap is more than 100 times faster compared to longranger, while producing similar results. SpecHap is also much faster than HAPCUT2 on 10x linked reads with more accurate results and lower memory consumption. Details of the benchmark result is in **Supplementary Table 2S**.

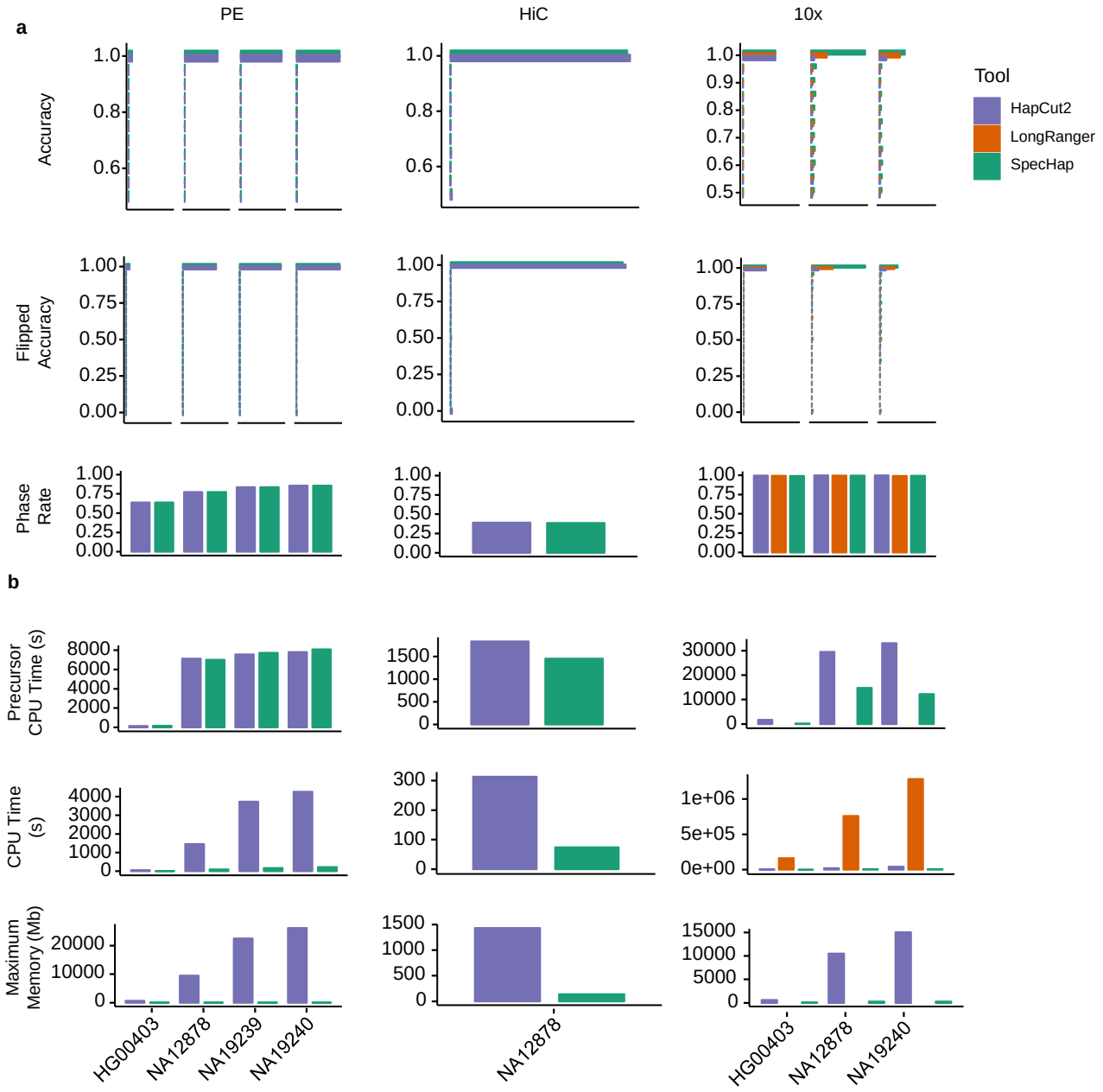

Figure 1. Performance benchmarking among SpecHap, HAPCUT2 and longranger. Measurements of SpecHap of different tools on different samples respect to different protocols. **a** Count of phased blocks by accuracy, Count of phased block by flipped accuracy and phase rate by SpecHap, HAPCUT2 and longranger. Note that the majority of the phased block are of both high flipped accuracy and accuracy. The count of block is not normalized and may vary by different sample, phasing protocol and software. **b** Precursor CPU time, CPU time, and maximum memory. The precursor CPU time and maximum memory of longranger is not applicable. See supplementary table 2S for detailed benchmark data.

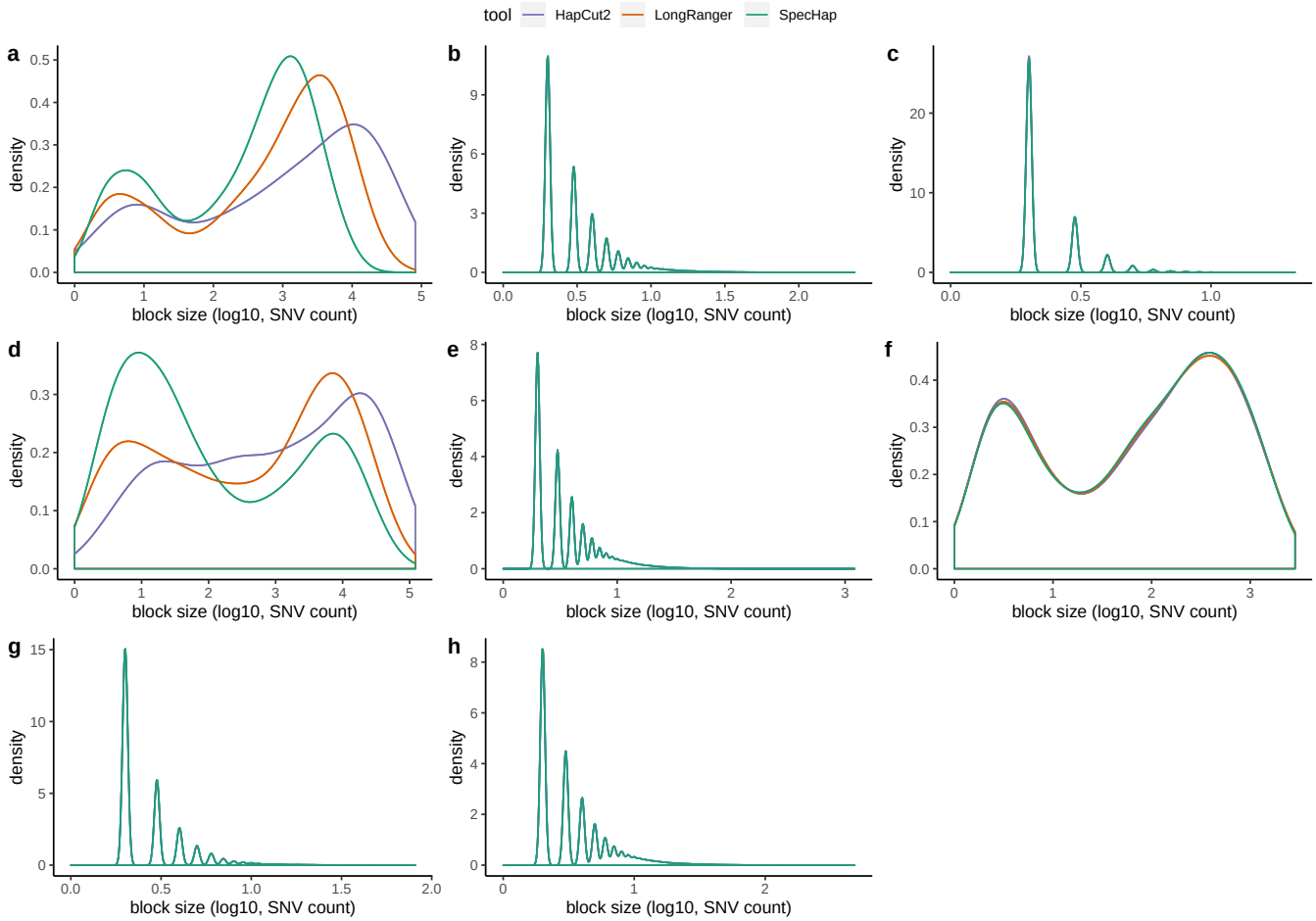

Figure 2. Phased block size distribution among SpecHap, HAPCUT2 and longranger. Measurements of phased block size between SpecHap and other tools on different samples respect to different protocols. **a**: Phased block size distribution of sample NA12878 with 10X linked reads. **b**: Phased block size distribution of sample NA12878 with paired-end reads. **c**: Phased block size distribution of sample NA12878 with Hi-C reads. **d**: Phased block size distribution of sample NA19240 with 10X linked reads. **e**: Phased block size distribution of sample 19240 with paired-end reads. **f**: Phased block size distribution of simulated sample HG00403 with 10X linked reads. **g**: Phased block size distribution of simulated sample HG00403 with paired-end reads. **h**: Phased block size distribution of sample NA19239 with paired-end reads.

Discussion

We have shown that SpecHap is able to accurately phase chromosome-level haplotype with lower resource consumption for the three sequencing protocols it currently supports. The benchmark results demonstrate that Fiedler Vector can provide high confidence phasing result and locate conflicting region without any further measurement even with high fluctuation. We believe this accuracy in conflicting region can be further improved with noise filtering. Spectral graph theory is known to be suitable for many binary classification tasks; the robustness of SpecHap confirms its suitability in the task of haplotype phasing in diploid genome.

We intend to extend SpecHap to support third generation sequencing technology including PacBio and

Oxford Nanopore with the full functionality to phase all type of variant. Phasing with RNA sequences is also proposed in the future. At the cohort level, SpecHap has the potential to perform phasing by involving a reference panel, which can be established by SpecHap iteratively or obtained from other known panels, to help improve the phasing rate and accuracy for haplotypes in a particular population. Additionally, it is possible for SpecHap to impute genotyptes at missing loci with modifications to the algorithm and reference panels.

Finally, the analysis and bench-marking of haplotypes were conducted with SNVs only. Short insertions and deletions (INDEL) and structural variations (SV), though plays an important role in restoring individual haplotype, need special concerns compared with phasing with SNVs. In theory, SpecHap is able to phase with all kind of genetic variation when required information is provided. Supporting phasing with different sequencing protocol and different form of genetic variation will be a progressive work for SpecHap.

METHODS

Data preparation

Synthetic data

Sequencing data were simulated to assess SpecHap with other haplotyping software. First, we extracted the haplotype information of chromosome 1, 21, and 22 of sample HG00403 from 1000 Genomes Project (McCarthy et al., 2016) as simulation templates. Then, 30x WGS reads were simulated by WGSIM on hg19 for chr1, chr21 and chr22 based on the generated haplotype template, respectively. For running configuration, the read length was set to 150bp and the insertion size to 350bp. Furthermore, with simulated paternal and maternal haplotype templates as input fasta, 30x linked reads were produced utilizing LRSIM (Luo et al., 2017) based on hg19 for chr1, chr21, and chr22, respectively. For the LRSIM configuration, the read length was set as 150bp, insertion size as 350bp, and long molecule DNA length as 50k. The simulation was performed with a sequencing error of 0.0001 to imitate the major sequencing error. Then, BWA and longranger were utilized separately for alignment of paired-ends and 10x reads with reference genome set hg19. The template genotype is utilized to provide variant information for haplotyping with SpecHap and HapCUT2.

Paired-ends real data

The paired-end alignment files together with the phase 3 variants information for sample NA12878, NA19239 and NA19240 was downloaded directly from the website of 1000 Genomes Project to benchmark the performance of haplotyping paired-end reads.

10x real data

The 10x linked-read WGS sequencing data for sample NA12878 and NA19240 was gathered from 10x Genomics official site. We ran the data through longranger_v2.2.1 to acquire alignment and variant information together with its CPU and memory consumption. A separate VCF is called with GATK3.5 based on the longranger aligned reads to benchmark SpecHap and HapCUT2.

Hi-C real data

We downloaded the Hi-C sequencing data from NCBI PRJNA473369. Sequenced reads from seven selected cells (SRR7226668, SRR7226671, SRR7226678, SRR7226679, SRR7226681, SRR7226682 and SRR7226685) were merged. Further processing was based on the merged Fastq. We used BWA to align the combined sample and to locate the mate-pair information. GATK 3.5 HaplotypeCaller was then applied to extract variant information.

SpecHap

Linkage graph construction

We first extract the linkage information from the BAM and VCF files using a modified version of extractHAIRS (Edge et al., 2017), which provides pairwise linkages among heterozygous SNVs (het-SNVs) loci along the same chromosome. The average distance between linked het-SNVs varies among different sequencing technologies. For WGS with short insert size, paired-end reads are scanned for het-SNVs in a local region with hundreds of base pairs length; for 10x linked-reads, barcodes are also examined with its range inferred from the alignment result and hence the linked fragment can cover het-SNVs loci separated by thousands of base pairs; for Hi-C data, linkages among het-SNVs apart from millions of base pairs can be extracted.

With the fragment file, a *linkage graph* is constructed in order to apply spectral graph theory methods. In the linkage graph, we define a pair of nodes for each het-SNVs locus to represent its two parallel alleles (**Fige3:a,b,c**). For one given het-SNV locus S_i, we denote its two parallel alleles as S_{iu} and $S_{i\bar{u}}$, respectively. Similarly, for the other het-SNV locus S_j, we have S_{jv} and $S_{j\bar{v}}$.

From the fragment file, we extract all linkages between alleles of these two het-SNVs (S_i and S_j). There are two types of linkages: direct and indirect. Taking alleles S_{iu} and S_{jv} as an instance, the direct linkage is provided by the fragment covering S_{iu} and S_{jv}, while the indirect linkage is provided by the fragment covering parallel alleles $S_{i\bar{u}}$ and $S_{j\bar{v}}$. We integrate both direct and indirect linkages into an edge connecting S_{iu} and S_{jv} nodes. The edge weight, which is calculated based on the mapping quality scores of reads, denotes the probability that S_{iu} and S_{jv} lie on the same chromosome. Assume that the direct linkage is provided by n fragments $r_1, r_2, \ldots, r_n$ covering alleles S_{iu} and S_{jv}, and the indirect linkage is provided by m fragments $r_{n+1}, r_{n+2}, \ldots, r_{n+m}$ covering the parallel alleles $S_{i\bar{u}}$ and $S_{j\bar{v}}$. The edge weight $p_{iu,jv}$ is calculated by the equation below.

$$p_{iu,jv} = Pr(S_{iu}, S_{jv} | r_1, r_2, \ldots, r_{n+m}) = 1 - \prod_{k=1}^{n+m} (1 - g_k) \tag{1}$$

where g_k is calculated from the mapping quality score c_k of the k-th fragment as follows. (Note that we take the minimum mapping quality score when fragment is inferred from reads pair.)

$$g_k = 1 - 10^{\frac{-c_k}{10}} \tag{2}$$

Since the linkage graph is constructed based on the mapping quality of reads and there are cases when a mate-pair of two paired-end reads are supposed to provide single linkage, weights of linkage will be calculated based on the reads with lower quality. Thus, we sort the extracted linkage information according to the variants' order prior to linkage graph construction.

Spectral graph theory

The standard spectral graph theory can be described as follows. Assume an undirected graph G consists of N vertices $V = \{v_1, v_2, ..., v_N\}$, and M edges $E = \{e_1, e_2, ..., e_M\}$ where each edge $e_m = (v_i, v_j), i \neq j$. We construct an adjacency matrix $A^{N \times N}$ for graph G to store the linkage relationship between a pair of vertices v_i, v_j, such that $A_{i,j} = 1$ if $(v_i, v_j) \in E$ and $A_{i,j} = 0$ otherwise. Assume d_i is the degree of a given vertex v_i in graph G. The degree matrix D can be defined as a diagonal matrix with i-th diagonal element assigned to d_i: $D_{i,j} = d_i$ if $i = j$, $D_{i,j} = 0$ otherwise. Then, by definition the Laplacian matrix L of graph

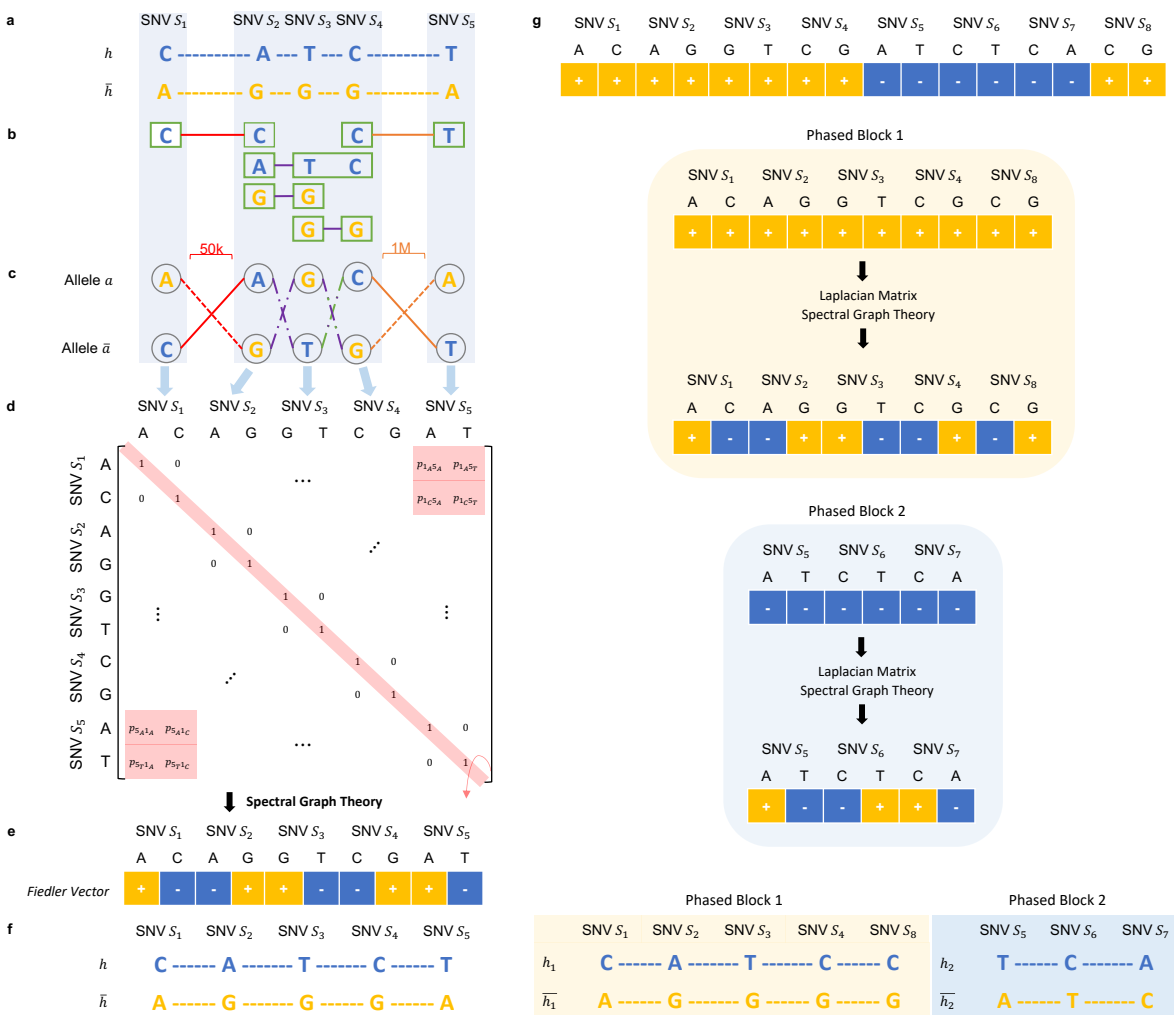

Figure 3. SpecHap Algorithm.
(a) Real haplotypes to be resolved. Blue and yellow alleles are allied to haplotypes h and $\overline{h}$ respectively. **(b)** Observed sequencing reads. Green frame refers to single-end read; red line implies linkage supported by 10x barcode; purple line represents the paired-end linkage; orange line stands for the Hi-C paired-end linkage. **(c)** The linkage graph. The solid edge refers to the direct linkage from sequencing reads, the dashed edge is the indirect linkage from parallel alleles pair, and the "solid-dashed" alternation represents edge composed from direct and indirect linkage together. **(d)** The weighted adjacent matrix for graph. e.g. $p_{5_A 1_C}$ and $p_{1_C 5_A}$ are the same, they denote the linkage probability between SNV S_5 allele "A" and SNV S_1 allele "C". **(e)** Fielder vector procured by spectral graph theory. **(f)** Resolved haplotypes h and $\overline{h}$ determined by Fielder vector. **(g)** Conflicting region where Fielder Vector failed to separate two haplotypes. Based on the Fielder Vector, the graph is divided into subgraphs from which haplotypes are calculated with the same routine above.

G is constructed as $L = D - A$. According to spectral graph theory, these N vertices can be grouped into two clusters by the element sign $(+, -)$ of the Fiedler vector, that is, the eigenvector which correspond to the second smallest eigenvalue of L (Chung and Graham, 1997).

SpecHap adopts the spectral graph theory and utilizes Fielder vector as a guidance to generate two

haplotype string (**Fig3:e,f**). the The implementation of SpecHap deviates from the standard spectral graph theory at three places:

First, SpecHap applies the weighted adjacent matrix as input. Instead of only considering binary linkages, our linkage graph assigns the reads linkage probability to each edge such that $W_{i,j} = p_{i,j}$ (textbf-Fig3:d), as mentioned in the previous section.

Spectral graph theory states that the multiplicity of zero eigenvalues of the graph Laplacian signifies the number of connected components in the graph, and the eigenvector contains the union of spectral signal for all the connected subgraphs. It is also highly possible that batch of blocks are not fully connected, resulting in a graph with multiple connected components, which may lead to inaccurate result as we may not be able to decompose the eigenvector to acquire spectral information for each fully-connected sub-graph. Therefore, a depth-first search is applied to extract connected component before any eigen calculation. The Laplacian matrix is then prepared separately for further computation.

The third change is as follows. Our experiments show that the choice of the eigenvector for clustering could influence the phasing result. While in most cases the Fiedler vector will provide the perfect solution, we found two exceptions to this rule. The first case happens when the linkage graph contains pairs of SNVs that hold equal possibilities of two complement haplotype, which will increase the multiplicity of the second smallest eigenvalue by one. Thus, SpecHap filters out those SNVs that cannot be phased. The second case is where the Fiedler vector does provide useful information: categorizing SNVs instead of haplotypes. Such a situation may be the result of a few false positive linkages with low quality or highly conflicting regions which may lead to overextending of the block. SpecHap will cut the block accordingly and perform spectral analysis for both block correspondingly (**Fig3:g**).

Finally, SpecHap utilizes spectral graph theory to determine two haplotypes h and $\overline{h}$. The degree matrix D and clustering process are defined and performed identically as the standard procedure. All the variant alleles with positive eigenvector entry are considered to be in the first phased variant set h, while all other variant alleles associated with negative eigenvector element are classified as the second haplotype $\overline{h}$.

Phasing entire chromosome

The extremely long Homo Sapiens chromosome makes it impossible to phase all the SNVs at the same time. Therefore, a divide and conquer strategy is applied. For simplicity in implementation, a definition of variant block and phasing batch is introduced. First, SNVs that are supposed to be phased together may lie in one variant block, w which may also contain single unphased SNVs. By default, the SpecHap will assign multiple blocks based on the input variant information, thus we transform the problem of phasing SNVs into the problem of phasing variant blocks. Second, SpecHap will phase a batch, which maintains user-designated length, of the variant block at one time. Each pair of adjacent batch maintains overlap of variant blocks, and the phased result of the previous batch will be introduced when phasing its neighbor.

For Hi-C sequenced reads which may provide linkage information across the entire chromosome, this information may be dropped as it may not fit inside a single phasing batch of variant. In order to preserve this long-linkage information, our phasing algorithm consists of two steps. First, the Hi-C sequenced reads are treated as regular WGS data to conduct phasing. Then, based on the phased result, a special phasing batch containing the covered SNVs will be constructed, and phasing will be performed with the long-spanning linkage information. In such a way, SpecHap guarantees that all the linkage information regardless of its spanning will be utilized during phasing.

10X Specific Phasing

In our experiments of phasing with 10X sequenced reads, we identified major inconsistency between the phased haplotype and the actual haplotype. Thus, special cares were taken in order to obtain maximum phasing accuracy while preserving the size of the phased set. First, variants are filtered by their allele depth and quality before the initiation of the phasing algorithm. We also disallow a phasing block from striding over 30 continuously filtered variants. Then, the covering range of each barcode is inferred based on the alignment result. In our implementation, a barcode can neither start nor end on an aligned read with mapping quality less than 30, and two reads with the same barcode cannot be more than 60kbp apart. This procedure filtered out the rare but significant situations where reads with the same barcode are from two different molecules.

Evaluation

As the genetic information of cell used for sequencing with 10X and Hi-C protocol may vary from the origin germline, we only considered the shared variants from different pipelines for evaluation. We did not take into account any heterozygous SNVs with called genotype that are opposed to the corresponding template genotype. These SNVs consist of five categories: SNVs showing only REF allele, SNVs showing only ALT allele, SNVs located in N zone, SNVs with low reads coverage, and SNVs with novel ALT genotype. Note that the software-specific filtered variants still count when evaluating the result. GATK CombineVariants tool was adopted in this process.

For each phased block, the accuracy, flipped accuracy, and phasing rate were compared among tools. Meanwhile, CPU-time and maximum consumed memory were measured as well.

Phasing rate

Phasing rate measures the percentage of phased SNV among all observed SNV sites.

$$phase_rate = \frac{\text{\# of phased SNVs}}{\text{\# of total SNVs}} \tag{3}$$

Accuracy

Accuracy measures the proportion of correctly inferred SNV among all phased SNV sites for a phased block. Suppose $h \in [0,1]^N$ and $\hat{h} \in [0,1]^N$ are the ground-truth and estimated haplotype for a phased block with N number of SNVs, respectively.

$$accuracy(h,\hat{h}) = \frac{\sum_{i=1}^{N} h_i = \hat{h}_i}{N} \tag{4}$$

Flipped accuracy

$is_flipped(g_i, g_{i+1})$ measures whether the genotype is flipped between two adjacent SNVs or not. This allows us to define the flipped accuracy, which measures the proportion of correctly inferred adjacent linkages among all adjacent linkages for a phased block.

$$\begin{aligned} is_flipped(g_i, g_{i+1}) &= \begin{cases} 0, g_i = g_{i+1} \\ 1, g_i \neq g_{i+1} \end{cases} \\ flipped_accuracy(h,\hat{h}) &= \frac{\sum_{i=1}^{N-1} is_flipped(h_i, h_{i+1}) = is_flipped(\hat{h}_i, \hat{h}_{i+1})}{N-1} \end{aligned} \tag{5}$$

Outputs specification

SpecHap outputs the phased genotype in standard VCF format for the convenience of further analysis. Due to the possible discreteness of phased result, a tag named phased set is added. SNVs belonging to the same phased set will maintain an identical value, which is the start position of the set. Several filters are also defined by SpecHap, which can be found in the documentation of the package.

Acknowledgement

All the data used in this paper can be retrieved from public database. All the experiments can be reproduced with dedicated version of software with default argument.

Disclosure declaration

There is no competing interests for authors of this article.

Author Information

Affiliations

Department of Computer Science, The School of Engineering, City University of Hong Kong,
Hong Kong, China
Yonghan Yu, Lingxi Chen, Bowen Tan, Wenlong Jia and Shuaicheng Li

Contributions

S.L. conceived and led this work. Y.Y. and S.L. designed the algorithm. Y.Y. implemented the SpecHap software and led the experiment. L.C. and B.W. simulated data and benchmarked the result. Y.Y., L.C., B.W. and W.J. wrote the manuscript with feedback from W.J. and S.L..

Corresponding author

Correspondence to Shuaicheng Li.

References

Bansal V, Halpern AL, Axelrod N, and Bafna V. 2008. An mcmc algorithm for haplotype assembly from whole-genome sequence data. Genome research **18**: 1336–1346.

Chung FR and Graham FC. 1997. Spectral graph theory. 92. American Mathematical Soc.

Consortium GP et al.. 2012. An integrated map of genetic variation from 1,092 human genomes. Nature **491**: 56.

Consortium IH et al.. 2010. Integrating common and rare genetic variation in diverse human populations. Nature **467**: 52.

Delaneau O, Coulonges C, and Zagury JF. 2008. Shape-it: new rapid and accurate algorithm for haplotype inference. BMC bioinformatics **9**: 540.

Edge P, Bafna V, and Bansal V. 2017. Hapcut2: robust and accurate haplotype assembly for diverse sequencing technologies. Genome research **27**: 801–812.

F CD, Mattias J, Graham C, Xiaoquan W, D WJ, A RN, and K PJ. 2006. A worldwide survey of haplotype variation and linkage disequilibrium in the human genome. Nature genetics **38**: 1251.

Glusman G, Cox HC, and Roach JC. 2014. Whole-genome haplotyping approaches and genomic medicine. Genome medicine **6**: 73.

Luo R, Sedlazeck FJ, Darby CA, Kelly SM, and Schatz MC. 2017. Lrsim: a linked-reads simulator generating insights for better genome partitioning. Computational and structural biotechnology journal **15**: 478–484.

McCarthy S, Das S, Kretzschmar W, Delaneau O, Wood AR, Teumer A, Kang HM, Fuchsberger C, Danecek P, Sharp K, et al.. 2016. A reference panel of 64,976 haplotypes for genotype imputation. Nat Genet **48**: 1279–83.

Musone SL, Taylor KE, Lu TT, Nititham J, Ferreira RC, Ortmann W, Shifrin N, Petri MA, Kamboh MI, Manzi S, et al.. 2008. Multiple polymorphisms in the tnfaip3 region are independently associated with systemic lupus erythematosus. Nature genetics **40**: 1062.

Onuchic V, Lurie E, Carrero I, Pawliczek P, Patel RY, Rozowsky J, Galeev T, Huang Z, Altshuler RC, Zhang Z, et al.. 2018. Allele-specific epigenome maps reveal sequence-dependent stochastic switching at regulatory loci. Science **361**: eaar3146.

Selvaraj S, Dixon JR, Bansal V, and Ren B. 2013. Whole-genome haplotype reconstruction using proximity-ligation and shotgun sequencing. Nature biotechnology **31**: 1111.

Tan L, Xing D, Chang CH, Li H, and Xie XS. 2018. Three-dimensional genome structures of single diploid human cells. Science **361**: 924–928.

Zheng GX, Lau BT, Schnall-Levin M, Jarosz M, Bell JM, Hindson CM, Kyriazopoulou-Panagiotopoulou S, Masquelier DA, Merrill L, Terry JM, et al.. 2016. Haplotyping germline and cancer genomes with high-throughput linked-read sequencing. Nature biotechnology **34**: 303.