Introduction:

Disease has been at the center of human mortality long before it was a comprehendible concept. To date there is still much to be discovered in terms of disease processes, inheritance and mediation. DNA sequencing has played a pivotal role in human genomics and disease research. Rapid advancement(s) in sequencing technologies have aided in cost effective, high throughput sequencing data which can allow for the identification of human genetic variants that may affect health, disease, and an individual’s response to medications (RefHap, Duitama et. al 2010, page 1, sec. intro, 3rd sentence). One crucial computational task associated with reconstructing an individual’s genome from a set of reads is haplotype assembly, the procedure of inferring two DNA sequences from the set of reads (FastHap, Mazrouee et. al 2014, page 1, sec.1 intro. last sentence).

To begin, we will concisely review concepts associated with haplotype phasing and assembly. Humans are referred to as diploid organisms, meaning there are two copies for each chromosome, inherited form the individual’s parents. Each copy, though almost identical, contains small amounts of variations (PEATH, Na et. al 2018, page 1, sec.1. intro, 2nd sentence). The smallest regions and most frequent form of genetic variants are single nucleotide polymorphisms (SNPs) (Lancia et. al 2001, page 2, section 1.2, 2nd sentence). SNPs refer to a difference in a single base or nucleotide, they occur in our genome having one of two possible alleles. The neighborhood of positions that surround SNPs all contain identical DNA content (Lancia et. al 2001, page 2, section 1.2, 3rd sentence). A haplotype is the sequence of these SNPs in a specific chromosome. Haplotypes contain more vital information regarding genetic variation than SNPs alone, and are therefore of high interest in studying gene function, evolutionary selection and inheritance analysis (AROHap, Olyaee et. al 2018, page 1, sec.1 intro., right side, 2nd sentence from the top). Previously, haplotypes were determined by haplotype inference algorithms based on SNP data. With the advancement(s) in high-throughput sequencing technologies, haplotypes can be identified more accurately (Rhee et. al 2016, page 2, right side, 3rd sentence from the top). Determining haplotypes of an individual using a set of sequence reads is called haplotype assembly.

There are challenges concerning haplotype assembly, mainly sequencing errors and short sequencing reads. Sequencing errors typically occur due to chemical or optical errors during the reading of SNPs, where a false base is placed in the position of the true base (AROHap, Olyaee et. al 2018, page 1, right side, 3rd to last sentence). Reads that contain high levels of sequencing errors can hinder the reconstruction of accurate haplotypes. Short sequencing reads produced by next generation sequencing (NGS), though cost effective, have the possibility of not linking distant variants into haplotypes (HapCUT2, Edge et. al 2017, page 1, right side, , 2nd sentence). Insertions of gaps can occur when overlapping or the redundancy of information regards SNPs is not available.

Key Features:

Linkage Disequilibrium:

Linkage disequilibrium (LD) is the dependence of alleles at nearby base positions in the genome to each other. Disregard for LD patterns and structures when phasing genotype data can reduce haplotype assembly accuracy (BEAGLE, page 2, right side, 2nd to last paragraph). Reduction in accuracy is due to the introduction of noise by sampling variation, which results in false correlations between distant markers (BEAGLE, page 2, right side, 2nd to last paragraph). Haplotype assembly software packages, like BEAGLE and HATS, account for LD by integrating a parameter into their algorithm to compensate for the dependent patterns. BEAGLE’s “approach to making use of the localized LD structure is a localized haplotype-cluster model,” modeling the haplotype frequencies on a restricted scale (BEAGLE, page2, last paragraph on right side). In HATS, HapSeq, and HapSeq2 LD information from phased haplotype sequences, found in the same population as the sample, are used to improve accurate haplotype construction (HATS, page 2, 1st paragraph on right side) (HapSeq, page 8, 1st paragraph on left) (HapSeq2, page 4, 1st paragraph on left). With differing approaches towards linkage disequilibrium, taking this feature into account aids in the accuracy of haplotype phasing.

Recombination:

An inverse relationship exists between recombination and linkage disequilibrium, where regions of high recombination along the genome have low linkage disequilibrium. Recombination occurs when combinations of alleles observed at different loci in two individuals becomes shuffled in their offspring. Software packages like IMPUTE2, MaCH, HapSeq, HapSeq2, SHAPEIT, SHAPEIT2, and HAPI-UR, use the transition probability of the hidden Markov model to reflect historical recombination events between observed alleles (IMPUTE2, page 4, right side 2nd paragraph) (MaCH, page15, 3rd paragraph) (SHAPEIT, page 4, paragraph 2 on left) (HapSeq, page 2, 2nd to last paragraph on right) (HAPI-UR, page 3, last paragraph on right). Yu Zhang’s dynamic Bayesian Markov model implements a recombination mechanism in which at each SNP the transition probability determines whether to select a new state probability or not (DBM, page 3, last paragraph on left side). BEAGLE utilizes its localized haplotype-cluster model to model the complex recombination patterns found in real data (BEAGLE, page 3, 2nd paragraph on right side). These methods account for the recombination of alleles which aids in the prediction of accurate haplotypes.

Mutation Rate:

Mutation rate refers to “the frequency of new mutations in a single gene or organism over time” (Crow 1997). Software packages like BEAGLE, HuRef, IMPUTE2, GAHap, HapAssembler, and MaCH incorporate mutation rate into their algorithm. BEAGLE, IMPUTE2, and Hap-SeqX account for the mutation rate by estimating for the emissions probability [[MR3]](#MR3) [[MR7]](#MR7) [[MR13]](#MR13). HuRef, MaCH, and HapSeq combine mutation rate along with other parameters like sequencing error and genotype error into one variable associated with their algorithms [[MR4]](#MR4) [[MR9]](#MR9) [[MR12](#MR12)]. Kang’s Genetic Algorithm along with both Wang’s algorithm and the GAHap algorithm, integrate single-point mutation, a mutation that occurs “when a single base pair is substituted with another within the DNA” (<https://www.genetargeting.com/mutation/single-point-mutation/>) [[MR1]](#MR1) [[MR5](#MR5)] [[MR6].](#MR6) Other algorithms like HapAssembler, SHAPEIT, PPHS, AROHap, and Zhao’s algorithm directly use the mutation rate in their algorithms [[MR2](#MR2)][[MR8](#MR8)][[MR10](#MR10)][[MR11](#MR11)][[MR14](#MR14)].

Sequencing Errors:

As previously mentioned, sequencing errors can occur when a false base is placed in the position of the true base. Multiple haplotype assembly algorithms integrate parameters and variables to compensate for this type of error. HuRef and MixSIH both have a variable in their algorithm that umbrellas sequencing errors along with other parameters like mutation rate and mapping errors[[SE2](#SE2)][[SE7](#SE7)]. Other algorithms like HARSH, DBM, and MixSIH use fixed constants for the parameter to describe sequencing errors. Both HARSH and DBM set their sequencing error parameter to 0.01 unless otherwise mentioned, while MixSIH sets a combination parameter for mapping errors and sequencing errors to 0.1 [[SE8](#SE8)][[SE9](#SE9)][[SE7](#SE7)]. PEATH uses the Phred Score to represent the probability of a sequencing error[[SE14].](#SE14) MaCH, PPHS,H-BOP, HapSeq, HapSeq2, HapCompass, HapTree, and ParticleHap all directly embed a single parameter into their algorithm’s to represent sequencing errors [[SE3](#SE3)][[SE4](#SE4)][[SE5](#SE5)][[SE6](#SE6)][[SE10](#SE10)][[SE11](#SE11)][[SE12](#SE12)][[SE13](#SE13)].

Sequencing Quality:

Sequencing quality refers to the accuracy of the read/sequence compared to the true DNA of the sample. A sequencing quality scores, “measure the probability a base is called incorrectly” (Ewing, 1998). HuRef designs its experiment around the usage of high-quality Sanger-based whole-genome sequencing to satisfy increased quality compared to earlier sequencing methods [[SQ1](#SQ1)]. HASH and FastHap both use a parameter to account for sequencing quality within their algorithm. HASH uses an error probability that estimates the occurrence of incorrectly calling a variant, which is based on the quality score of the sequence [[SQ2](#SQ2)]. FastHap uses a fixed constant associated with the sequencing quality parameter set at 0.2 for best performance [[SQ3](#SQ3)].

Phred Score:

A Phred quality score, is a measure of the quality of the identification of the bases generated by DNA sequencing (Ewing, 1998). Phred scores are widely accepted as being a viable way of categorizing the quality of sequenced DNA. HASH uses an error probability within their algorithm, this probability is based on the quality score of the sequence and thus can use the Phred quality score in the determination of the error probability [[PS2](#PS2)]. HapCol, WhatsHap, and HapCHAT have the advantage of easily adapting a weighted version of their MEC in “order to deal with phred-score error probabilities” [[PS3](#PS3)][[PS4](#PS4)][[PS5](#PS5)]. PEATH uses the Phred quality score as a direct parameter within its algorithm to represent the probability of a sequencing error [[PS6](#PS6)].

Sequencing Coverage:

An important aspect to consider when reconstructing haplotypes is the coverage available for a specific position. FastHare states that “the number of fragments that cover a given position”, if low, can affect the accuracy of the reconstructed strands [[SC1](#SC1)]. FastHare, P-WMLF/GS and MixSIH assumes/assigns coverage to be 10, 10 and 3.03, respectively [[SC1](#SC1)][[SC6](#SC6)][[SC11](#SC11)]. HuRef uses a Poisson distribution to estimate sequencing depth/coverage, the genome sequence coverage ranges from 4.9 for deletions, 5.5 for insertions and 7.5 for SNPs [[SC2](#SC2)]. HALLDORSSON estimates the coverage needed to assemble haplotypes based on the average length of the read, times the number of reads, then all divided by the total length of the genome [[SC8](#SC8)]. While other algorithms such as Deng’s algorithm, DBM, HGHap and HapCol incorporates a parameter into their algorithms that calculates the sequencing coverage of a fragment by SNP matrix [SC7/10] [SC12] [SC13] [SC15]. In HapCUT2, coverage is the average coverage per variant [SC16]. The ProbHap algorithm written by Volodymyr Kuleshov preforms optimally with coverages of 10-12X, when highly uneven coverages are present ProHap uses a heuristic to merged reads that are likely from the same haplotype until there are no reads that can confidently be merged [SC14]. WhatsHap uses a fixed parameter tractable approach where coverage is its only parameter. WhatsHap’s favorable coverage ranges from 10-15X but can perform proficiently with coverage as high as 20X [SC17]. Bansal states that, “for haplotype assembly to be feasible, one requires a high sequence coverage (sufficient overlaps between reads) and reads that are long enough to span multiple variant sites” [[SC4](#SC4)]. Next, we will look at the importance of read length and the effects it has on haplotype assembly.

Read Length:

Read length, measured in base pairs (bp), describes the size of DNA fragments used in haplotype assembly. Enhancements in sequencing technologies are leading to increasing read lengths and in turn positively affecting critical parameters like sequencing coverage. Chen Zhi’s algorithm incorporates the length of a sequencing read by taking the end position of the read, subtracting it from the starting position and then adding one to the overall remaining total. Read length is predominately a feature utilized to evaluate algorithms performance using specific read lengths generated by simulated data.

Genotype Errors:

A genotyping error occurs when the observed genotype of an individual does not correspond to the true genotype. Low sequencing coverage can affect genotyping errors, thus increasing their chances of occurring. Algorithms like MaCH and HapSeq account for genotyping errors with a general error variable. Mach combines gene conversion, mutation and genotyping error into a single variable in their algorithm (2010.Li.MaCH, page 15 second to last paragraph). HapSeq similarly combines multiple factors into one specific error variable, which includes gene conversion, mutation and genotyping errors (2012.Zhi.HapSeq, page 3, last paragraph on right). Though genotyping error is not a prevalent parameter incorporated into haplotype assembly algorithms, it is later seen to be a commonly used parameter for the evaluation of such software packages.

Alignment/Mapping:

The alignment of reads to a reference genome is a common first step associated with high-throughput sequence data. Due to the commonality of this procedure many software packages have a generalized assumption that the reads needed to run their algorithms have previously been aligned. The assumption of alignment often dismisses the specifics on the reference genome and alignment software package used to align their reads. Software packages like HuRef, CSP and WhatsHap to the contrary provide such information. HuRef uses K-mer Analysis Toolkit (KAT) to create a one-to-one mapping of HuRef and NCBI human genome reference assembly [AI1]. CSP aligns both Kaper’s (2013.Kaper.dilution.based.method) and Duitama’s (2011.Duitama.Fosmid) datasets to the reference genome hg18 using bowtie (version 1.0.0) and bfast (version 0.7.0), respectively [AI5]. WhatsHap aligns all of their datasets using BWA MEM, however they did not specify which reference genome was used [AI6].

Minimum Error Correction (MEC):

A computational model that aims at solving the haplotype assembly problem in minimum error correction (MEC) (2005.Wang, page 1, last paragraph on right). MEC is the minimum number of changes, or corrections, need to make the read matrix a perfect bi-partition, such that corrected read maps to either haplotype exactly (2010.MaxSAT, page 3, 2nd paragraph on left). Thus, the haplotype assembly problem with MEC is a difficult objective function to optimize and is NP-hard, which correlates to the computational complexity of the problem in polynomial time (2010.MaxSAT, page 2, 2nd paragraph on left). Algorithms created by Wang, Xu, Chen.Zhi, Deng and software packages like MaxSAT, HapSAT and GAHap all have a MEC objective function.

Input:

The input needed to solve the haplotype assembly problem varies from algorithm to algorithm. At least 25 haplotype assembly software packages need a constructed matrix for their input. However, there are various types of matrices used ranging from SNP matrices, fragment matrices and customized matrices that no other software package uses. 18 software packages require input in the form of a SNP or SNP-fragment matrix, with rows corresponding to the number of fragments (aligned DNA short reads) and each column corresponds to a SNP site (2001.Lancia, page 9, 2nd paragraph), (2004.Panconesi.FastHare, page 5, 3rd paragraph from top), (2005.Wang, page 3, table 1.),(2007.Levy.HuRef, page 29, 2nd paragraph), (2008.Bansal.HASH, page 3, 2nd paragraph on right), (2008.Chen.HapRec, page 4, last paragraph/proof outline), (2010.Xu, page 2, 2nd paragraph on right), (2011.HALLDORSSON, page 2, last paragraph), (2012.Bayzid.HMEC, page 2, 2nd paragraph on right), (2012.Xie.BHOP, page 6, 2nd to last paragraph on left), (2012.Aguiar.HapCompass, page 3, 2nd paragraph), (2012.Wang.GAHap, page 2, 2nd paragraph on right), (2013.Matsumoto.MixSIH, page 4, first paragraph on left), (2014.Chen.HGHap, page 2 last sentence on left and first on right), (2014.Mazrouee.FastHap, page 2, 1st paragraph on right), (2014.Matsumoto.CSP, page 3, 2nd paragraph on right), (2016.HapCol, page 3, 2nd paragraph on left), (2017.WhatsHap, page 4, 2nd paragraph), (2018.Olyaee.AROHap,page 2, last paragraph on right), (2018.Tangherloni.GenHap, page 6, last paragraph). GAHap constructs a fragment matrix where the rows represent the number of fragments, similar to the SNP-fragment matrix, however the columns represent the length of the reference genome sequence (2012.Wang.GAHap, page 2, 2nd paragraph on right). There are 6 other algorithms that generate distinct matrices as their input. MaxSAT designs their input matrix with rows corresponding to the number of reads and columns associated to the number of heterozygous sites (2010.MaxSAT, page 2, last paragraph on right). HapSAT develops a read matrix with the rows representing the length of reads and the columns representing the number of reads (2011.Mousavi.HapSAT,page 2, 3rd paragraph from bottom left). Chen Zhi’s algorithm and ParticleHap builds their input matrix to focuses on the number of reads and the number of SNP sites (2013.Chen.Zhi, page 2, 2nd paragraph on right), .” (2015.Ahn.ParticleHap, Page 3, 2nd to last paragraph on left). ProbHap and PEATH both construct two matrices M of size m and n, m is the number of sequenced reads and n is the position of a heterozygous site, and Q a quality score matrix with the same size as M (2014.Kuleshov.ProbHap, page 4, 2nd paragraph on left), (2018.Na.PEATH, page 2-3, last paragraph on left and first paragraph on top right). There are over 20 algorithms that require other forms of input information. Kang’s algorithm, P-WMLF/GS and HapAssembler all require SNP fragment and weighted matrices, alon with other features like population size and mutation rate (2008.Kang, page 7, table 1), (2008.Xie.P\_WMLF/GS, page 6, figure 6), (2010.Kang.HapAssembler, page 2, 2nd paragraph on right). BEAGLE requires randomly phased data for its input for the first iteration of its algorithm (2007.BEAGLE, page 5, 3rd paragraph on left). Wu’s algorithm takes SNP values of fragments as its input (2009.Wu, page 5, 2n paragraph). MaCH, SHAPEIT, SHAPEIT2 require a set of genotypes and for the SHAPEIT’s a genetic map (2010.Li.MaCH, page 15, 2nd paragraph), (<http://mathgen.stats.ox.ac.uk/genetics_software/shapeit/shapeit.html#gettingstarted>). PPHS, HapSVT, HapNuc and HapOPT only requires the sequencing data as its input, while HARSH uses both the reference panel and sequencing data (2012.Efros.PPHS, page 2, 2nd paragraph on right), (2019.Majidian, page 5, algorithm 1) (2013.Yang.HARSH, page 3, 3rd paragraph on right). HapSeq and HapSeq2 require three specific files, the count file, the site file and the haplotype count file for their input (<https://github.com/ZhiGroup/HapSeq2/blob/master/HapSeq-Manual.pdf>), (<https://github.com/ZhiGroup/HapSeq2/blob/master/HapSeq2-Manual.pdf>). Deng’s algorithm uses a set of fragments sequenced from the two copies of each chromosome for it input (2013.Deng, page 2, paragraph 2 on right). DBM requires input of read counts for two alleles per SNP per individual (2013.Zhang.DBM, page 2, 2nd paragraph on right). HapCompass-Tumor uses three forms of input information, a list of sequences reads, variant calls and the number of distinct haplotypes (2014.Aguiar.HapCompass\_Tumor, page 17, algorithm 1 outline). WinHap2’s input consists of genotype vectors (2014.Pan.WinHap2, page 2, 2nd to last paragraph on right). The input to HapCUT2 consists of haplotype fragments and a list of heteroxenous variants (2017.Edge.HapCUT2, page 2, last paragraph on right). HapCHAT software packages takes inout in the form of BAM and VCF files (2018.HapCHAT, page 13, last paragraph on left).

Output:

The output generated by haplotype assembly software packages typically is a pair of haplotypes, though some algorithms produce additional supportive information. 30 software packages solely produce haplotype pairs, often denoted h1 and h2 (2005.Zhao, page 3, table 1), (2007.BEAGLE, page 11, 2nd paragraph on right), (2008.Kang, page 7, table1), (2008.Chen.HapRec, page 4, Proof outline/ last paragraph), (2008.Xie.P\_WMLF/GS, page 6, figure 6), (2008.Genovese.SpeedHap, page 4, last paragraph on right), (2009.Wu, page 5, 3rd paragraph), (2010.Kang.HapAssembler, page 2, 2nd paragraph on right), .” (2010.MaxSAT, page 3 first paragraph on left), (2010.Xu, page 2, 2nd paragraph on right), .”(2010.Duitama.ReFHap, page 4, 2nd paragraph on right), (2011.HALLDORSSON, page 7, 3rd paragraph from bottom), (2011.Mousavi.HapSAT, page 2, 3rd paragraph from bottom left), (<http://mathgen.stats.ox.ac.uk/genetics_software/shapeit/shapeit.html#output>), (2012.Bayzid.HMEC, page 4, Algorithm 1: HMEC Algorithm), (2013.Yang.HARSH, page 4, algorithm 1), (2013.Chen.Zhi, page 2, 3rd paragraph on right), (2013.Deng, page 4, 2nd to last paragraph on right), (2014.Aquiar.HapCompass\_Tumor, page 17, algorithm 1 outline), (2014.Chen.HGHap, page 5, 1st paragraph on right), (2014.Pan.WinHap2, Page 7, last paragraph on right), (2015.Ahn.ParticleHap, page 8, 1st paragraph on right), (2016.HapCol, Page 3, paragraph 4 on left), (2017.WhatsHap, page 7, 2nd to last paragraph), (2018.Olyaee.AROHap, page 3, 3rd paragraph on left), (2018.Na.PEATH, page 2, right side of page) (2019.Majidian, page 5, algorithm 1).. 7 additional algorithms construct haplotype pairs along with various supporting data. FastHare outputs three objects, a pair of haplotypes, a SNP-fragment matrix and a partition of rows of fragments into two groups, each group corresponding to one of the two haplotypes (2004.Panconesi.FastHare, page 5, 4th paragraph from top). HapSeq and HapSeq2 outputs both the imputed genotypes at each site and the inferred pair of haplotypes, while HapSeq2 also generates a file on genotypes obtained from each outer iteration of their algorithm (<https://github.com/ZhiGroup/HapSeq2/blob/master/HapSeq-Manual.pdf>). (2013.Zhang.HapSeq2, page 2, 2nd to last paragraph on right). The output of ProbHap is a set of haplotype blocks along with three confidence scores, at each position, that can be used to identify locations where the phasing results are less accurate (2014.Kuleshov.ProbHap, page 2, 4th paragraph on left). DBM includes the haplotypes pairs inferred, the recombination probabilities at each SNP and the underlying haplotype structures (2013.Zhang.DBM, page 4, 2nd paragraph on left). CSP generates two output files upon completion, the first contains the haplotypes and their probabilities for each SNP fragment and the second contains the CSP values for each SNP fragment (<https://sites.google.com/site/hmatsu1226/software/csp>). At least 8 software packages generate output that is not in the traditional format of haplotype pairs. PPHS generates a tree that is set of haplotypes (2012.Efros.PPHS, page 5, 1st paragraph on right). HapCompass outputs the phasing corresponding to any spanning tree of GC, while MixSIH outputs a hidden phase vector (2012.Aguiar.HapCompass, page 6, Algorithm 1 outline), (2013.Matsumoto.MixSIH, page 4, first paragraph on left). HAPIUR produces output in either Eiganstrat or IMPUTE2 format(<https://code.google.com/archive/p/hapi-ur/downloads>). Common output file formats are FASTQ, CVF and SAM files (2018.Tangherloni.GenHap, page 11, last paragraph), (<https://github.com/vibansal/HapCUT2/blob/master/README.md>).

Assumptions:

Assumptions are generally made to create a more controlled or stable environment for a algorithm to generate haplotype pairs. An extremely common assumption made by the vast majority of haplotype assembly software packages is that the sequencing data has properly been aligned (32 algorithms/54). There are 12 algorithms that assume that the alleles of each SNP are bi-allelic in nature (2007.Levy.HuRef, page 28, 3rd paragraph on left), (2012.Wang.GAHap, page 7, 2nd paragraoh on right), (2012.Zhi.HapSeq, page 2, 1st paragraph on right), (2012.Williams.HAPIUR, page 5, 2nd paragraph on right), (2012.Efros.PPHS, page 2, last paragraph on right), (2013.Zhang.HapSeq2, page 2, last paragraph on right), (2014.Berger.HapTree, page 2, 2nd paragraph on right), (2015.Ahn.ParticleHap, page 2, 2nd paragraph on left) (2019.Majidian, page 1, 1st paragraph). A few software packages assume Hardy-Weinberg equilibrium, that is allele and genotype frequencies in a population will remain constant from generation to generation in the absence of other evolutionary influences (2007.BEAGLE, page 4, 2nd to last paragraph on left), (2009.Howie.IMPUTE2, page 4 , 2nd to last paragraph on right). Another assumption made is that all fragments come from one organism, this is made in both Wang, Kang and Zhao’s algorithms (2005.Wang, page 1, last paragraph on right), (2005.Zhao, page 2, 2nd paragraph on left) (2008.Kang, page 2, 1st paragraph). Less frequent assumptions are assuming that sequencing coverage or read length are fixed (2004.Panconesi.FastHare, page 2, last paragraph), (2011.Mousavi.HapSAT, page 2, 2nd paragraph on left).

**Features Used for Evaluation/Comparison:**

Evaluation Using LD:

HuRef, HASH, MaCH, MixSIH, DBM, and ParticleHap use LD to evaluate the performance of their algorithms. HuRef looks at variants in strong LD regions to compare their inferred haplotype with the haplotypes provided from the dataset HapMap to test for accuracy (HuRef, page 17, 1st paragraph on right) (HASH, page 8, 2nd paragraph on right). HASH similarly accounts for LD strength, but extends the usage of LD to “infer a switch error with some probability” (HASH, page 8, 2nd paragraph on right), based on LD strength between the pairs of variants. In MaCH, LD is an accuracy measure that looks at the effectiveness of using imputed genotypes (MaCH, page 5, paragraph 1). MixSIH notes a correlation with increased minimum connectivity (MC) scores and the usage of LD information, stating that “the accuracy of predictions might be improved by using both pieces of information (MC score and LD)” (MixSIH, page 10, last paragraph on left). DBM and ParticleHap mention how LD information can “greatly help improving the accuracy of genotype calling” (DBM, page 4, 2nd to last paragraph on right) (ParticleHap, page 8, last paragraph on left). 2014 CSP authors emphasized that LD information can be used for statistical phasing that is helpful for the haplotype assembly indirectly (Page 10 of CSP left panel comparing MixSIN with PHASE). LD may not solve the problems of estimating haplotypes for regions with less or no SNP fragment due to sequencing and mapping/alignment issues.

Evaluation Using Sequencing Errors:

Inconsistencies between the reconstructed haplotypes and the available fragments are a result of sequencing error [[SE16](#SE16)]. Sequencing error/rates are used as one of many features to evaluate an algorithm’s ability to accurately reconstruct haplotypes. Multiple algorithms assimilate sequencing error/rates into their evaluations quite differently. HapAssembler results show that a sequencing error rate that is smaller than or equal to 0.2 can affect the performance enough to have the genetic algorithm slightly out preform the dynamic clustering algorithm reviewed in their evaluation [[SE18](#SE18)]. While, PPHS measures the accuracy of the compared algorithms as a function of the coverage and sequencing errors with a set of five individuals [[SE19](#SE19)]. H-BOP, on the other hand, creates simulated data with a sequencing error rate that is then every allele is flipped with a probability equal to the sequencing error rate to introduce sequence error [[SE20](#SE20)]. Similarly, HapSeq generates simulated data with sequencing error rates, though they set at the rate at either 0.2 or 0.5% [[SE21](#SE21)]. Finally, WhatsHap evaluates “future-generation” reads, long reads that have notably higher sequencing error rates, by setting the sequencing error rate of the hypothetical sequencer at 0.01 [[SE22](#SE22)].

Evaluation Using Sequencing Quality:

Sequencing quality is not a commonly used parameter for evaluating an algorithms performance, however HASH untraditionally uses sequencing quality within the evaluation of their algorithm. In which, HASH computed the log-likelihood from the sequencing quality values of the haplotype assemblies for the greedy algorithm and HASH, ultimately comparing each log-likelihood value [[SQ4](#SQ4)].

Evaluation Using Sequencing Coverage:

Sequencing coverage can be used as a evaluation feature to compare the performance of multiple algorithms amongst one another, where coverage of a data set can be set to a fixed constant or generated at various values creating a range of coverage for comparisons. Algorithms that fixed their simulated datasets coverage are Hap-SeqX, AroHap and HapSeq (HapSeq2) increasing from 1X, 3X to 4X, respectively [SC24] [SC35] [SC21] ([SC27]). Many algorithms use a range of sequencing coverages to evaluate the proficiency of their algorithm alongside others. Xu, HARSH, HapTree, PartliceHap, HapCol, WinHap, Chen Zhi’s algorithm, FastHap and HGHap all generate multiple datasets with different coverages for comparison. The most common ranges are sets with {3,5,8,10} (Chen Zhi and HGHap), {4,6,8,10} (Harsh and ParticleHap) and {5,10,15,20,100} (HapTree, HapCol, and WinHap) [SC20] [SC26] [SC28] [SC32] [SC33] [SC34] [SC25] [SC31] [SC29]. For Majidian’s HapSVT, HapNuc and HapOPT, the simulated data coverage value was set to 3, 5 and 10 based on the read lengths of N = 561, 936, and, while the Fosmid dataset had a coverage of 3 throughout the experiment (2019.Majidian, page 7, 2nd paragraph). H-Bop and CSP both used calculations to determine their coverage and evaluated the performance of their algorithm when the coverage increased or decreased [SC23] [SC30].

Evaluation Using Genotyping Errors:

As previously mentioned, a genotyping error occurs when the observed genotype of an individual does not correspond to the true genotype. Algorithms that consider genotyping errors use it to further asses and compare their algorithm with others. Harsh uses a combined measurement of genotyping error and switch error rates to evaluate their algorithm [GE3]. HATS and PPHS both make mention of upon comparing the performance of the simulated data with various coverages, the genotyping error rate is inversely related [GE1] [GE2]. Genotyping error rates tend to be higher as sequencing coverage is decreased. DBM uses the percentage of genotyping errors along with sample size to see the quality of their algorithm’s assembled haplotypes compared with others [GE4].

Evaluation Using Read Length:

The length of sequenced reads is typically measured in base pairs(bp) and referred to as the read length. Enhancements in technology have enabled the increase in read length of sequenced data, increasing from 100 bps to over 400 bps (2008.Bansal.HASH, page 10 1st paragraph on left). Read length is used as an evaluation parameter to compare the performance of haplotype assembly software packages as lengths vary in size. HapRec, HALLDORSSON, HapSeq and HapSeq2 create simulated data with read lengths between 36 bps to 200 bp. Other algorithms like 2009.Wu, HARSH, HapSeq2, WinHap2, HapCol and WhatsHap generate sequenced reads ranging by the thousands from 1,000 to 50,000 bp. For Majidian’s HapSVT, HapNuc and HapOPT, the simulated data read lengths were set to 561, 936, while the Fosmid dataset had an average read length of 40 kb throughout the experiment (2019.Majidian, page 8, 1st paragraph). Real datasets look at categorizing the performance based on average read length of all the reads processed. The smallest average of fragment length being H-BOP’s real dataset of 18.03 bp, 2010.Xu’s real dataset averaged 500 bp. Larger average read lengths used to evaluate algorithms like HapCHAT ranged from 4746 bp to 40 kilo-bases.

Evaluation Using Reconstruction Rate (RR):

Reconstruction rate quantifies the ability for an algorithm to reconstruct the correct haplotype. A common definition of the reconstruction rate is the ratio of the number of SNP sites that are correctly inferred to the total number of the SNP sites of the haplotypes (2008.Xie.P\_WMLF\_GS, page 6, 2nd paragraph on right), however other measurements are based on the number of bases or SNP fragments correctly inferred. Algorithms that use reconstruction rate as a tool for evaluation are HapRec, P-WMLF/GS, SpeedHap, GAHap, HMEC, HGHap, ParticleHap, AROHap, HapSVT, HapNuc, HapOPT and the algorithms created by Wang, Wu, Chen Zhi, and Deng (RR1-RR12) ((2019.Majidian, page 7, last paragraph).

Evaluation Using Switch Error rate:

Switch error is calculated as the number of positions where the two chromosomes of a proposed phase must be switched in order to agree with the true phase (2014.Berger.HapTree, page 6, last paragraph on left). The switch error metric accurately captures the close-range relationship between adjacent SNP phase (2012.Aguiar.HapCompass, page 3, section 3.1, 2nd paragraph). Algorithms that utilize switch error to compare and evaluate their algorithm amongst other leading haplotype assembly algorithms are HASH ReFHap, MaCH, H-BOP, HapCompass, Hap-SeqX, HapTree, WinHap2, ProbHap, CSP, HapCol, WhatsHap, HapCUT2, HapCHAT, PEATH, HapSVT, HapNuc and HapOPT (2019.Majidian, page 8, 2nd to last paragraph). Algorithms like PPHS, HapSeq and HARSH combine switch error with other error rates like mismatch and genotyping errors to create a total error rate.

Evaluation Using MEC:

As previously mentioned, the minimum error correction (MEC) is the number of conflicts between the sequence reads and the constructed haplotypes (2010.MaxSAT, page 2, 2nd paragraph on left). Thus, a MEC score is the minimum number of variant calls in the fragment matrix that need to be modified for every fragment to perfectly match one of the two haplotypes. (2008.Bansal.HASH, page 7, last paragraph on right). Software packages like HASH, RefHap, MaxSAT, Chen.Zhi’s, HapTree, FastHap, ParticleHap and PEATH use MEC to evaluate their performance in comparison with other leading software packages (MEC1-MEC8). ReFHap extends the use of the MEC score by dividing it by the number of total of allele calls in the input matrix to determine an allele calling rate (2010.Duitama.ReFHap, page5, 1st paragraph on right).

Evaluation Using Running Time:

The running time for an algorithm to generate haplotypes is a critical feature used in evaluating the algorithm’s performance. Some software package’s literature focuses solely on the amount of time it takes to produce a pair of haplotypes, while other algorithms look at an array of evaluation features and how running time correlates among these features.

There are 20 software packages that report running time as a key evaluation feature, of these packages only 2 fail to evaluate their running time against other software packages for comparison. FastHare sees no change in running time even with an increase in the number of fragments from 100 to 800 (2004.Panconesi.FastHare, page 9, 1st paragraph). Zhao’s algorithm shows an average running time of 1 minute for various scenarios (2005.Zhao, page 4, 1st paragraph on left). BEAGLE was 5 to 172 times faster than HaploRec-S, with respect to the dataset (2007.BEAGLE, page 8, 2nd to last paragraph on right). IMPUTE v2's running time increased by a factor of 1.1, whereas BEAGLE's run took 3.3 times longer when comparing their performances of the full dataset versus the restricted dataset, respectfully (2009.Howie.IMPUTE2, page 12, 2nd paragraph on right). ReFHap compares itself in various scenarious with HapCUT, one example compares the running time of each for a real instance with 32347 SNPs and 13905 fragments in chromosome 22, where ReFHap takes 73.04 sec and HapCUT requres almost an hour of running time(2010.Duitama.ReFHap, page 7, table 2). MaCH has a running time of 2 minutes (20 iterations) to 3.9 hours (3000 iterations), while GAHap has an average running time of less than 20 minutes for various scenarios of length, error rate and coverage (2010.Li.MaCH, page 25, Table I) (2012.Wang.GAHap, page 7, last paragraph on right). HMEC has a running time of a fraction of a second to reconstruct a haplotype with 936 sites (2012.Bayzid.HMEC, page 2, 2nd paragraph on left). The running time of Hap-seqX is 5 hours and the running time of Hap-seq is 9 hours (2013.He.Hap\_seqX, page 4, 1st paragraph on right). DBM ran up to 8 times faster than THUNDER (2013.Zhang.DBM, page 5, 1st paragraph on left), while the running time for HapSeq2 is 2 to 3 times that of THUNDER (2013.Zhang.HapSeq2, page 7, 1st paragraph on left). MixSIH admittedly states that though its running time is comparative to HapCUT, it is slower than all other algorithms compared to, ReFHap, HapCUT2, FastHare and DGS (2013.Matsumoto.MixSIH, page 12, last paragraph on right). HapTree outperforms HapCompass, an example being the median running time for 106 coverage and block length 10 was 0.00702 and 0.633 seconds for HapTree and HapCompass, respectively (2014.Berger.HapTree, page 7, 2nd to last paragraph on right). WinHap2 running time is around 60 times faster than that of SHAPEIT2 (2014.Pan.WinHap2, page 6, 2nd paragraph on left). ProbHap underperforms in running time compares against ReFHap, FastHare, DGS and MixSIH (2014.Kuleshov.ProbHap, page 3, last paragraph on right). WhatsHap and Deng et. al.’s algorithms have low running times for low coverage, however as coverage goes up Deng et al. increasingly exceeds WhatsHap (2017.WhatsHap, page 8, last paragraph) and HapCHAT has a running time comparable to WhatsHap (2018.HapCHAT, page 14, last paragraph on right).. In practice HapCUT2 significantly surpasses the performance of HapCUT in both memory usage and running time (2017.Edge.HapCUT2, page 4, 1st paragraph). AROHap can solve various scenarios for reconstructing a haplotype in less than 5 minutes (2018.Olyaee.AROHap, page 9, 1st paragraph on left). GenHap outperformed HapCol with a running time 4 times fasted in reconstructing the haplotypes (2018.Tangherloni.GenHap, page 15, 1st paragraph).

There are 8 software packages that utilize running time with other features to show how each feature is correlated to the others. Wang’s algorithm evaluates running time alongside the error rates of fragments to show that as the error rate increases so does the running time (2005.Wang, page 4, 1st paragraph on right). Xie's P-WMLF/GS algorithm has the following relationship, when the sample size increase, the reconstruction rate decreases, and the running time increases (2008.Xie.P\_WMLF\_GS, page 7, 2nd paragraph on right). HapRec compares the how as the error rate increases, running time increases and reconstruction rate decrease (2008.Chen.HapRec, page 8, 3rd paragraph from bottom). Chen Zhi’s algorithm shows that as coverage increases, the number of errors increase, along with the MEC score increasing and as a result the running time increases (2013.Chen.Zhi, page 6, last paragraph on right). Deng et al.’s Algorithm examines how as the size of the boundOfCoverage increase, the reconstruction rate gets higher and the running time increases accordingly (2013.Deng, page 6, last paragraph on left). FastHap runs faster than HapCUT when the haplotype length increases (2014.Mazrouee.FastHap, page 6, 1st paragraph on right). ParticleHap demonstrates that as the haplotype length and sequencing coverage increase, so too does the running time (2015.Ahn.ParticleHap, page 8, last paragraph on right). HapCol is slower if not comparable to WhatsHap relative to the sequencing coverage. (2016.HapCol, page 7, 2nd to last paragraph on right). HapOPT is compared along with AltHap, SDhaP, and HapCUT2, thought HapOPT has the best performance with regard to reconstruction rate and switch error rate its runtime is 71 times longer than SDhaP, with 355 minutes (2019.Majidian, page 10, table 6).

Evaluation using Error Rate:

Error rate is a parameter used to compare the performance of haplotype assembly software packages on simulated data. An error rate typically accounts for incorrect flipping of variant calls to achieve the most accurate haplotype (2014.Mazrouee.FastHap, page 6, 2nd paragraph on right). Simulated datasets are generated with different parameter values for comparison, error rate along with coverage and haplotype length are commonly used (2005.Wang, page 3, 2nd to last paragraph on right), (2012.GAHap, page 5, 1st paragraph on left). Wang and Wu’s algorithms, along with GAHap, FastHap, HapCompass-Tumor, AROHap and GenHap all utilize differing error rate percentages for comparing their performance against leading haplotype assembly packages (2005.Wang, page 4, 1st paragraph on left), (2009.Wu, page 8, 2nd to last paragraph), (2012.GAHap, page 5, 1st paragraph on left), (2014.Mazrouee.FastHap, page 2, last paragraph on right), (2014.Aguiar.HapCompass\_Tumor, page 8, 1st paragraph), (2018.Olyaee.AROHap, page 6, 2nd paragraph on left), (2018.Tangherloni.GenHap, page 14, 2nd to last paragraph)

Single Cell Haplotype Assembly:

We review 3 single cell haplotype assembly packages. Single cell sequencing technologies can provide reads that have an average length of 10K bps and cover around 90% of positions over chromosomes (SC.2018.Guo.SMS\_Hap\_Assembly, page 1, last paragraph on right). Haplotype assembly from single cell sequencing data can enable the construction of more complete haplotype sequences (SC.2018.Guo.SMS\_Hap\_Assembly, page 2, 1st paragraph on left). Similar approaches to the haplotype assembly problem are used after amplification of the single cell sequencing data has been achieved. Guo’s algorithm uses a Bayesian statistical model along with accounting for the phred score, sequencing coverage and read length within it’s algorithm 2018.Gou, page 3, 1st paragraph on left), (2018.Guo, page 2, 2nd paragraph on right), (2018.Guo, page 1, right paragraph). Additionally, Satas’ and Chu’s single cell haplotype assembly algorithms account for sequencing errors like any NGS sequencing data haplotype assembly packages (2018.Satas, page 4, 3rd paragraph from bottom left), (SC.2017.Chu, page 4, 2nd paragraph on right). Both Chu and Guo’s algorithms use BWA-MEM to align their single sell sequencing data (2018.Guo, page 4, last paragraph on left), (SC.2017.Chu, page 2, 1st paragraph on right). All three single cell haplotype assembly software packages use switch error rate for evaluating the performance of their algorithms (2018.Satas, page 4, 2nd paragraph on right), (2018.Guo, page 2, 2nd paragraph on right), (SC.2017.Chu, page 3, 3rd paragraph on right). Chu’s and Satas’ algorithms use the N50 length as a evaluation parameter (2018.Satas, page 5, 1st paragraph on left), (SC.2017.Chu, page 3, 3rd paragraph on right). The N50 length is the size of a haplotype block where half of all phased variants in the block are at least as long 2018.Satas, page 4, 2nd paragraph on right). Guo’s algorthim also evaluates on the basis of MEC score as well as running time, which as previously mentioned are to prevalent evaluation features of other haplotype assembly software packages (2018.Guo, page 6, 2nd paragraph on left), (2018.Guo, page 6, 2nd paragraph on left).

Polyploid Haplotype Assembly:

We review 2 software packages that can reconstruct haplotypes for both diploid and polyploid organisms. Polyploid organisms are organisms like potatoes and wheat, having more than two homologous sets of chromosomes. Haplotype assembly for these organisms is more challenging than for diploid species (Poly.2015.Das.SDhaP, page 2, 2nd paragraph on right). AltHap requires a SNP fragment matrix for its algorithm’s input and assumes that SNP sites are biallelic in nature (Poly.2018.Hashemi.AltHap, page 3, 2nd paragraph on right), (Poly.Hashemi.AltHap, page 3, last paragraph on left). SDhaP, similarly to other haplotype assembly algorithms, accounts for sequencing errors within its software package and uses sequencing error rate as a parameter for evaluation(Poly.2015.Das.SDhaP, page 4, 2nd paragraph on left) (Poly.2015.Das.SDhaP, page 9, 1st paragraph on left). AltHap evaluates its performance and accuracy based on reconstruction rate and read length (Poly.2018.Hashemi.AltHap, page 9, last paragraph on right), (Poly.2018.Hashemi.AltHap, page 12, 2nd paragraph on right). SDHap also uses switch error rate and MEC score to evaluate the accuracy and performance of its algorithm (Poly.2015.Das.SDhaP, page 9, 1st paragraph on right), (Poly.2015.Das.SDhaP, page 8, 2nd paragraph on right). Both SDhaP and AltHap uses sequencing coverage and running time as parameters for evaluating their software packages (Poly.2015.Das.SDhaP, page 8, 2nd paragraph on right), (Poly.2018.Hashemi.AltHap, page 11, last paragraph on left), (Poly.2015.Das.SDhaP, page 9, 1st paragraph on right), (Poly.2018.Hashemi.AltHap, page 12, last paragraph on right).

**Annex**

**Linkage Disequilibrium (LD):**

1. 2007.BEAGLE: “Correlation between markers is a localized phenomenon, since LD decays with distance.” (2007.BEAGLE, page 2, right side, 2nd to last paragraph)
2. 2007.BEAGLE: “Our approach to making use of the localized LD structure is a localized haplotype-cluster model,1,12 which empirically models haplotype frequencies on a local scale.” (2007.BEAGLE, page 2, right side, 2nd to last paragraph)
3. 2010.Li.MaCH: “Remarkably, we observed that imputed genotypes could also be used to obtain very accurate estimates of LD between pairs of untyped markers, or of LD between a genotyped marker and an untyped marker. (2010.Li.MaCH, page 5, paragraph 1)
4. “estimates of LD between two SNPs obtained using imputed data are much closer to the results obtained by actually genotyping the two SNPs than estimates obtained by looking up the two markers in the HapMap CEU database (Supplementary Figure 2 shows a similar comparison for D’ estimates).” (2010.Li.MaCH, page 5, paragraph 1)
5. “Even with some imprecision in estimates of individual genotypes, the increased sample size compensates to reduce variation in the estimated LD measures.” (2010.Li.MaCH, page 5, paragraph 1)
6. 2012.Dewal.HATS: “We, therefore, collate information from multiple heterozygous sites by leveraging the known structure of linkage disequilibrium (LD) between these variants within the population being interrogated.” (2012.Dewal.HATS, page 2, lastparagraph on left).
7. 2012.Dewal.HATS : “This training data provides LD information across sites, allowing for more accurate haplotype construction versus examining each site in isolation.” (2012.Dewal.HATS, page 2, 1st paragraph on right).
8. 2012.Zhang.HapSeq: “We have developed an LD-based method and implemented an efficient algorithm for genotype calling that can incorporate the haplotype information from reads that cover two adjacent PPSs.” (2012.Zhi.HapSeq, page 8, 1st paragraph on left)
9. 2013.Zhang.HapSeq2: “It can be seen that we use both the information of haplotypes from the sequencing reads (R3) and the LD information (T) from this set of samples or the reference samples to update the haplotypes of that individual” (2013.Zhang.HapSeq2, page 4, 1st paragraph on left)

**Evaluation Using LD:**

1. 2007.Levy.HuRef: “For adjacent pairs of such variants that were in strong LD, fewer than 1 in 40 of the HuRef-inferred haplotypes conflicted with the preferred HapMap phasing.” (2007.Levy.HuRef, page 17, 2nd paragraph on right).
2. 2008.Bansal.HASH:“One of the benefits of inferring haplotypes from sequence data is that the local accuracy of the haplotypes is unlikely to be affected by the level of LD in a region. This also presents the opportunity of using LD in population data to detect switch errors in the HuRef haplotypes. For a pair of variants that are in strong LD in population data, the correct HuRef phasing is expected to match the more likely population based phasing.” (2008.Bansal.HASH, page 8, 2nd paragraph on right).
3. 2008.Bansal.HASH: “If the inferred HuRef phasing does not match the preferred population phasing, one can infer a switch error with some probability (the probability value depends upon the strength of LD between the pair of variants).” (HASH, page 8, 2nd paragraph on right).
4. 2013.Matsumoto.MixSIH : “SIH can accurately infer the haplotypes in many regions with low linkage disequilibrium, but there are also regions with reduced precision and high |D’| values. This suggests that the accuracy of predictions might be improved by using both pieces of information.” (2013.Matsumoto.MixSIH, page 10, last paragraph on left).
5. 2013.Zhang.DBM: “greatly help improving the accuracy of genotype calling” (2013.Zhang.DBM, page 4, 2nd to last paragraph on right).
6. 2014.Matsumoto.CSP: “Statistical phasing is weak in determining haplotype regions where linkage disequilibrium values are high and there are multiple haplotypes in population. To investigate these differences, we compared the reliabilities of MixSIH and PHASE” (2014.Matsumoto.CSP, page 10, 3rd paragraph on left).
7. 2015.Ahn.ParticleHap: “the genotype call accuracy for high call rates was up to 96 % with the use of LD information, from 78 % and 87 % with the use of single sample and multiple individuals, respectively, for the 62 CEU individuals).”(2015.Ahn.ParticleHap, page 8, last paragraph on left)

**Recombination (R):**

1. 2009.Howie.IMPUTE2: “we use a hidden Markov model that is based on an approximation to the coalescent-with recombination process [16].” (2009.Howie.IMPUTE2, page 4, right side 2nd paragraph).
2. 2010.Li.MaCH: “ denotes the prior probability of the initial mosaic state and is usually assumed to be equal for all possible configurations, denotes the transition probability between two mosaic states and reflects the likelihood of historical recombination events in the interval between j and j−1, denotes the probability of observed genotypes at each..”(2010.Li.MaCH, paragraph 2).
3. 2012.Zhi.HapSeq: “…the transition probability between two states and reflects the likelihood of historical recombination events” (HapSeq, page 2, 2nd to last paragraph on right).
4. 2012.Williams.HAPIUR: “Because one state models several original haplotypes, states can transition without recombination to one or more states at the subsequent window, and our model weights recombination’s to a state by the frequency of the underlying haplotype segment.” (HAPI-UR, page 3, last paragraph on right).
5. 2013.Zhang.DBM: “We use Markov Chain Monte Carlo (MCMC) algorithms to estimate posterior distributions of model parameters of interest. Particularly, DBM outputs genotypes, haplotypes and recombination probabilities between SNPs.” (2013.Zhang.DBM, page 2, last paragraph on left side).
6. 2007.BEAGLE: “2 Unlike haplotype-block–based models, which permit recombination only between haplotype blocks, localized haplotype-cluster models can model the complex recombination patterns found in real data.” (2007.BEAGLE, page 3, 2nd paragraph on right side).

**Mutation Rate (MR):**

1. 2005.Wang: Their Genetic Algorithm incorporated mutation rate as a genetic operation parameter in its algorithm (2005.Wang, page 3 last paragraph on left side).
2. 2005.Zhao: States that the mutation rate of the real seed haplotypes is low (2005.Zhao, page 6 2nd paragraph on right).
3. 2007.BEAGL: in order to specify the HMM they must specify the emissions probability which is associated with mutations, along with other probability variables ( 2007.BEAGLE, page 4, 1st paragraph on left side).
4. 2007.Levy.HuRef: “To examine sequence diversity in the genome, we estimated nucleotide diversity using the population mutation parameter (2007.Levy.HuRef, page 8, paragraph 2 on left side).”
5. 2008.Kang: Their Genetic Algorithm adopts a single-point mutation (2008.Kang, page 6, section 3.5, 2nd to last sentence).
6. 2012.Wang.GAHap: Uses single-point mutation (2012.Wang.GAHap, page 4, 1st paragraph on left).
7. 2009.Howie.IMPUTE2: incorporates emission probabilities into their model that models historical mutations (2009.Howie.IMPUTE2, page 4, 2nd paragraph on right side).
8. 2010.Kang.HapAssembler: uses mutation rate as a parameter in their algorithms (2010.Kang.HapAssembler, page 2, scheme of algorithm).
9. 2010.Li.MaCH: incorporates mutation rate into their algorithm (2010.Li.MaCH, page 15 second to last paragraph).
10. 2012.Delaneau.SHAPEIT: uses mutation parameter in algorithm. (2012.Delaneau.SHAPEIT, page 4 second to last paragraph on right)
11. 2012.Efros.PPHS: mutation is accounted for in algorithm (2012.Efros.PPHs, page 2, 2nd to last paragraph on left side.)
12. 2012.Zhi.HapSeq: uses a parameter in their algorithm that combines mutation, gene conversion and genotyping error into one parameter (2012.Zhi.HapSeq, page 3, last paragraph on right).
13. 2013.He.Hap-seqX:uses emission probabilities to represent the mutation rate in their algorithm (2013.He.Hap\_seqX, page 3, 3rd paragraph from right bottom).
14. 2018Olyaee.AROHap: accounts for mutation probability in their algorithm (2018.Olyaee.AROHap, Page 4, paragra 2 on right).
15. 2018.Tangherloni.GenHap: “mutation rate mr ∈ {0.01, 0.05, 0.1, 0.15}.” (2018.Tangherloni.GenHap, page 12, 1st paragraph).

**Sequencing Errors (SE):**

1. 2005.Wang: “DNA sequencing errors and the diploidy of human genome make the problem complex” (2005.Wang, page 1, 2nd to last paragraph on right side).
2. 2007.Levy.HuRef: “Therefore, our estimate for is likely a combination of both a true higher mutation rate in repetitive regions and sequencing errors.” (2007.Levy.HuRef, page 9, last paragraph on left).
3. 2010.Li.MaCH: incorporates sequencing errors in to their algorithm (2010.Li.Mach, page 18, 2nd paragraph from the top).
4. 2012.Efros.PPHS: implements a sequencing error parameter in their algorithm (2012.Efros.PPHS, page 2 1st paragraph on right side).
5. 2012.Xie.HBOP: Integrates a measurement for sequencing error as they believe sequencing errors are inevitable and should be accounted for (2012.Xie.HBOP, page 6, paragraph 2 on the left side).
6. 2012.Zhi.HapSeq: uses sequencing error as a parameter in their algorithm (2012.Zhi.HapSeq, page 3, last paragraph on right).
7. 2013.Matsumoto.MixSIH: accounts for sequence and mapping errors as a combined parameter in their algorithm (2013.Matsumoto.MixSIH, page 4, last paragraph on left).
8. 2013.Yang.HARSH: uses a standard sequencing error rate of 0.01 within their algorithm (2013.Yang.HARSH, page 5, 3rd paragraph from right bottom).
9. 2013.Zhang.DBM: similarly used a default sequencing error rate of 0.01 within their algorithm (2013.Zhang.DBM, page 3, 2nd paragraph from left top).
10. 2013.Zhang.HapSeq2: includes sequencing error as a parameter in their algorithm (2013.Zhang.HapSeq2, page 4, last paragraph on left).
11. 2013.Aguiar.HapCompass-Tumor: uses sequencing error in their algorithm (2013.Aguiar.HapComapss\_Tumor, page 4 2nd paragraph).
12. 2014.HapTree: accounts for sequencing error in algorithm (2014.Berger.HapTree, page 4, 2nd paragraph on left).
13. 2015.Ahn.ParticleHap: uses a sequencing error probability in the algorithm (2015.Ahn.PaticleHap, page 7, 1st paragraph on left side).
14. 2018.Na.PEATH: The probability of sequencing errors is represented as a Phred Score (2018.Na.PEATH, page 3, 1st paragraph on left).
15. 2018.Tangherloni.GenHap: “GenHap can efficiently solve large instances of the wMEC problem, yielding optimal solutions by means of a global search process, without any a priori hypothesis about the sequencing error distribution in reads.” (2018.Tangherloni.GenHap, page 3, last paragraph).

**Evaluation Using Sequencing Errors:**

1. 2007.Levy.HuRef: “The repetitive regions are likely to have a higher false-positive rate due to sequencing error and misassembly. Further, they are not represented in the current estimate of the false-positive rate” (2007.Levy.HuRefp, page 13, 1st paragraph on left).
2. 2008.Bansal.HASH:
   1. “This correlation between high posterior error probabilities and low sequencing quality values represents an independent confirmation of the quality of the ­­reconstructed haplotypes and also indicates that some of the inconsistencies between the reconstructed ha­­plotypes and the fragments are a result of sequencing error.” (Page 8, 2nd to last paragraph on lift side).
   2. “Instead, simulations show that the error rate depends upon the sequencing error and depth of coverage.” (2010.Bansal.HASH, page 10, second to last paragraph on left)
3. 2010.Kang.HapAssembler: uses sequencing error percentage to evaluate the accuracy of algorithms, stating that: “The result showed that when the sequencing error rate is smaller than or equal to 0.2, the performance of the genetic algorithm is slightly better than that of the dynamic clustering algorithm” (2010.Kang.HapAssembler, page 3, paragraph 1 on left side).
4. 2012.Efros.PPHS: evaluates sequencing error rate as a measurement of accuracy regarding their algorithm (2012.Efros.PPHS, page 6, last paragraph on left).
5. 2012.Xie.HBOP: accounts for sequencing error when generating the simulated data used for evaluations (2012.Xie.HBOP, page 2, last paragraph on left side).
6. 2012.Zhi.HapSeq: uses sequencing error to aid in the measurement of accuracy among compared algorithms (2012.Zhi.HapSeq, page 4, 1st paragraph on right).
7. 2017.WhatsHap: uses sequencing error rates to measure accuracy upon evaluation of their algorithm (2017.WhatsHap, page 9, paragraph 2).

**Sequencing Quality (SQ):**

1. 2007.Levy.HuRef : “We used an experimental design based on very high-quality Sanger-based whole-genome shotgun sequencing, allowing us to maximize coverage of the genome and to catalogue the vast majority of variation within it.” (2007.Levy.HuRef, page 2, 2nd to last paragraph on right side).
2. 2008.Bansal.HASH : “Corresponding to each variant call Xi [j], we have an error probability qi [j], which denotes the probability that the variant call is incorrect. As qi [j] cannot be estimated from the fragment data, we use quality scores si [j] that usually accompany sequence data. For example, the quality scores might be obtained using phred (Ewing and Green 1998).” (2008.Bansal.HASH, page 3, 2nd paragraph from the left)
3. 2014.Mazrouee.FastHap: “The value of needs to be set based on the quality of data. For the dataset used in our experiments on different chromosomes, we set experimentally and found that provides the best performance.” (2014.Mazrouee.FastHap, page 3, last line on right)

**Evaluation Using Sequencing Quality:**

1. 2008.Bansal.HASH : “We also compared the log-likelihood of the haplotype assemblies for the greedy algorithm and HASH. The log-likelihood was computed using the sequencing quality values to estimate the q matrix. We found that the log-likelihood for the haplotypes reconstructed using HASH was consistently higher than that of the greedy haplotypes” (2008.Bansal.HASH, page 8, 2nd paragraph on left)

**Phred Score (PS):**

1. 2008.Bansal.HASH:“Corresponding to each variant call Xi [j], we have an error probability qi [j], which denotes the probability that the variant call is incorrect. As qi [j] cannot be estimated from the fragment data, we use quality scores si [j] that usually accompany sequence data. For example, the quality scores might be obtained using phred (Ewing and Green 1998).” (2008.Bansal.HASH, page 3, 2nd paragraph from the left)
2. 2016.HapCol: “This confidence degree is a combination of the probability that an error occurred during sequencing (phred-based error probability) for that base call and of the confidence of the read mapping to that genome position” (2016.HapCol, page 2, 3rd paragraph from bottom left)
3. 2017.WhatsHap: “A major advantage of MEC is that it can be easily adapted to a weighted version (wMEC) in order to deal with phred-scaled error probabilities.” (2017.WhatsHap, page 3, 3rd paragraph from bottom)
4. 2018.HapCHAT: “Then, for each site, the character that is the result of a merge is chosen applying a majority rule, weighted by the Phred score of each symbol.” (2018.HapCHAT, page 4, 2nd paragraph top right)
5. 2018.Na.PEATH:
   1. The probability of sequencing errors is represented as a Phred Score (2018.Na.PEATH, page 3, 1st paragraph on left).
   2. “Qij is the phred quality score and represents the probability for a sequencing error at the jth position of the ith sequence read.” (2018.Na.PEATH, page 3, 2nd paragraph from bottom left)

**Evaluation Using Phred Score:**

None

**Sequencing Coverage (SC):**

1. 2004.Panconesi.FastHare: “Another important aspect is the number of fragments that cover a given position, or coverage. Clearly, if the coverage for a certain position is very low, it is unlikely that we can reconstruct the two strands in that position with good accuracy. In practice, the coverage can be assumed to be around 10 (with roughly half of the fragments coming from each strand).” (2004.Panconesi.FastHare, Page 2, 3rd sentence in last paragraph).
2. 2007.Levy.HuRef:
   1. “We used an experimental design based on very high quality Sanger-based whole-genome shotgun sequencing, allowing us to maximize coverage of the genome and to catalogue the vast majority of variation within it.” (2007.Levy.HuRef, page 2, 2nd to last paragraph on right side).
   2. “We developed a statistical model based on our assembly read coverage in the single diploid genome and on the filtering criteria used for calling high confidence variants. We assumed that chromosomes containing each of the two alleles are equally likely to be sampled and that allele loci are independent. At a given heterozygous locus, the probability of observing both alleles in at least x reads follows the binomial distribution with p = 0.50 and n = depth of coverage, where x is defined by the filtering criteria. To calculate the false-negative rate genome wide, a Poisson distribution is also incorporated to estimate sequence depth at different loci, where k is set to the genome sequence coverage (7.5 for SNPs, 5.5 for insertions, 4.9 for deletions, after read filtering is taken into account).” (2007.Levy.HuRef, page 28, 3rd paragraph from bottom left).
3. 2008.Bansal.HASH: “For haplotype assembly to be feasible, one requires a high sequence coverage (sufficient overlaps between reads) and reads that are long enough to span multiple variant sites.” (2008.Bansal.HASH, page 2 , second paragraph on right)
4. 2008.Genovese.SpeedHap: “Thus, a critical parameter of the input data is the minimum (or average) coverage of SNPs by fragments. This number is also related to the throughput of the sequencing equipment.” (2008.Genovese.SpeedHap, page 1, 2nd to last paragraph)
5. 2008.Xie.P\_WMLF\_GS: “In our experiments, the parameters are as follows: fragment coverage rate c=10, the difference rate between two haplotypes d = 20%, the minimal length of fragment lMin=3, the maximal length of fragment lMax=7 and empty values probability p=2%.” (2008.Xie.P\_WMLF\_GS, page 6, 3rd to last paragraph on bottom right)
6. 2011.HALLDORSSON: “In this section, we estimate the coverage needed to assemble haplotypes of multiple individuals.” where L: the average length of the read, N: the number of reads, & G: the length of the genome. (2011.HALLDORSSON, page 3, last paragraph of page)
7. 2013.Deng:
   1. “The number of rows covering column i in M is referred to as the coverage of column i.” (2013.Deng, page 2 2nd to last paragraph on bottom right).
   2. “We first design a dynamic programming algorithm that gives an exact solution and runs in O(n × 2t × t) time, where n is the number of columns in M, and t is the maximum coverage of a column in M. The dynamic programming algorithm will be very slow when t is large.” (2013.Deng, page 3, 2nd to last paragraph on left)
8. 2013.Matsumoto.MixSIH: “The coverage of the data was about 3.03. We used the trio-based data and the sequencing data in binary format for our experiment.” (2013.Matsumoto.MixSIH, page 6 1st paragraph on left)
9. 2013.Zhang.DBM: uses sequencing coverage in algorithm as parameter. (2013.Zhang.DBM, page 3, equation 3)
10. 2014.Chen.HGHap: “There are three parameters for the data set: haplotype length l = 100, 350, 700, error rate e == 0, 0.1, 0.2, 0.3 and coverage rate c = 3, 5, 8, 10.” (2014.Chen.HGHap, page 3, 1st paragraph on left)
11. 2014.Kuleshov.ProbHap: “The exact dynamic programming algorithm described above is practical for coverages of up to 10–12. For deeper or for highly uneven coverages, we propose a simple preprocessing heuristic.” (2014.Kuleshov.ProbHap, page 6, 2nd paragraph from bottom left)
12. 2016.HapCol: uses coverage as a parameter in their algorithm (2016.HapCol, page 4, left first 2 paragraphs)
13. 2017.Edge.HapCUT2: “d is the average coverage per variant” (2017.Edge.HapCUT2, page 10, 2nd paragraph from bottom left)
14. 2017.WhatsHap: “Here, we present WhatsHap, a fixed parameter tractable (FPT) approach to wMEC where coverage, that is the number of fragments that cover an SNP position, is the only parameter.” (2017.WhatsHap, page 1, last paragraph)
15. 2018.Tangherloni.GenHap: “This approach is feasible thanks to the long reads with higher coverage produced by the second- and third-generation sequencing technologies. As a matter of fact, highly overlapping reads allow us to partition the problem into easier sub-problems, avoiding the possibility of obtaining incorrect reconstructions during the merging phase.” (2018.Tangherloni.GenHap, page 7, last paragraph).
16. 2018.Satas: “In this article, we demonstrate a positive aspect of amplification bias: since neighboring genomic loci are often co-amplified, the correlation in sequence coverage between neighboring alleles on the same chromosome can be used to phase haplotypes in diploid genomes.” ( 2018.Satas, page 2, 1st paragraph on left).

**Evaluation Using Sequencing Coverage:**

1. 2010.MaxSat:
   1. “The basic parameters of sequencing technology that we explore are the sequence coverage ratio (the number of times that each base pair in the sequence is covered)” (2010.MaxSat, page 6, 2nd paragraph on left side)
   2. “As we can see, the number of connected components in each block decreases as coverage ratio increases and as SD (standard deviation) increases.” (2010.MaxSat, page 7, 1st paragraph on left)
2. 2010.Xu: “In addition, because each fragment covers a number of SNPs in the range roughly [3, 7][10], thus the length of each SNP fragment is randomly set in this range.” (2010.Xu, page 5, 1st paragraph on left)
3. 2012.Zhi.HapSeq: “For each set, we generated sequencing reads of 4× coverage” (2012.Zhi.HapSeq, page 4, 2nd paragraph on right)
4. 2012.Bayzid.HMEC: “Here coverage rate indicates the percentage of the total columns of the SNP matric that sampled out” (2012.Bayzid.HMEC, page 6, 3rd paragraph on right)
5. 2012.Xie.HBOP: “Given fragments generated as above, the average call coverage c is calculated by dividing the total number of alleles of the fragments by the haplotype length n.” (2012.Xie.HBOP, page 2, 1st paragraph on right)
6. 2013.He.Hap\_seqX: “We randomly choose one target individual, generate simulated sequence reads with an error rate of 1% and coverage as 1X. The reads are of size 1000 bp in each end.” (2013.He.Hap\_seqX, page 4, 3rd paragraph from bottom left)
7. 2013.Chen.Zhi:“We use part of the simulated datasets of Geraci (2010). To generate a read matrix, three parameters ‘, c and e are used, where ‘is the number of SNPs, c is the coverage and e is the error rate.” (2013.Chen.Zhi, page 6, last paragraph on left)
8. 2013.Yang.HARSH: “we evaluate our method using six levels of sequencing coverages: 1, 2, 4, 6, 8 and 10.” (2013.Yang.HARSH, page 6, 2nd paragraph on bottom left)
9. 2013.Zhang.HapSeq2: “We fixed the rest of the simulation parameters: sequencing error rate to 0.5%, sequencing depth of coverage to 4X, as we have shown that the trends observed from different settings of these parameters tend to be similar” (2013.Zhang.HapSeq2, page 5, 1st paragraph on left)
10. 2014.Berger.HapTree: “To do so, we simulated reads with error rate 0:02 from a pair of phased k-ploid SNP loci for different coverages (5X, 10X, 20X, 100X) and for (2014.Berger.HapTree, page 6, 2nd to last paragraph on right)
11. 2014.Chen.HGHap: use coverage as a parameter for evaluation (2014.Chen.HGHap, page 5-6, tables 1-3)
12. 2014.Matsumoto.CSP: uses SNP fragment coverage as an evaluation parameter. (2014.Matsumoto.CSP, page 6)
13. 2014.Mazrouee.FastHap: uses coverage a parameter for evaluation. (2014.Mazrouee.FastHap, page 5 top right paragraph)
14. 2015.Ahn.ParticleHap: “To explore the performance of the algorithm over a broad range of experimental parameters, we generate datasets with different SNP lengths (n = 100, 200 and 300) and vary the coverage rate (c = 4, 6, 8 and 10) for each genotype calling error rate (ge = 0.04 and ge = 0.08).” (2015.Ahn.ParticleHap, page 8, 1st paragraph on right)
15. 2016.HapCol: “We also assessed accuracy and performances of the tools while varying coverage, read length and sequencing/indel error rate on simulated long read datasets with characteristics similar to those of the ‘future generation’ sequencing technologies that are currently (or soon) available (coverage up to 25)” (2016.HapCol, page 5, 3rd paragraph from bottom right)
16. “Table 1 shows the runtimes of the four tools for coverages 5X, 10X, and 15X” (2017.WinHap, page 8, last paragraph)
17. 2018.Olyaee.AROHap: uses a fixed coverage of 3 on simulated data for evaluation (2018.Olyaee.AROHap, page 7, left side)
18. 2019.Majidian:
    1. Simulated Data:
       1. “The average number of reads are N = 561, 936, and 1873 for coverage values of c = 3, 5, and 10, respectively. The number of SNPs covered in each read is a constant value equal to.” (2019.Majidian, page 7, 2nd paragraph)
       2. “Furthermore, as a consequence of increasing the coverage value, a better performance is achieved by a lower SWER and a higher reconstruction” (2019.Majidian, page 7, 2nd paragraph)
    2. Fosmid Data: NA12878
       1. “The coverage of this data set is c = 3 and the average read length is 40 kb, and hence, is a low-coverage and long-read dataset” ( 2019.Majidian, page 8, 1st paragraph).

**Genotype Errors/SNP Calling Errors:**

1. 2010.Li.MaCH: Accounts for genotype errors in their algorithm (2010.Li.MaCH, page 15 second to last paragraph).
2. 2012.Zhi.HapSeq: uses a parameter in their algorithm that combines mutation, gene conversion and genotyping error into one parameter (2012.Zhi.HapSeq, page 3, last paragraph on right).

**Evaluation using Genotype/SNP calling Errors:**

1. 2012.Dewal.HATS: Uses a genotype error correction measurement to increase performance of algorithm upon evaluation. (2012.Dewal.Hats, page 7, 3rd paragraph on right from bottom).
2. 2012.Efros.PPHS: Notes under their evaluation section that the genotype error rate in the data was relatively high, due to low coverage (2012.Efros.PPHS, page 6 send to last paragraph on left).
3. 2013.Yang.HARSH: Uses an error rate to evaluate the accuracy of their algorithm, where the error rate is the sum of the genotyping and switching error rates (2013.Yang.HARSH, page 5, 3rd paragraph on right).
4. 2013.Zhang.DBM: Uses percentage of genotyping error to compare accuracy of their algorithm among others (2013.Zhang.DBM, page 6, last paragraph on left).

**Sequencing/Read Length:**

1. 2007.Levy.HuRef: “The average separation of these variants on the genome was ;1500 bp (twice the average read length)” (2007.Levy.HuRef, page 26, 2nd to last paragraph on right)
2. 2008.Bansal.HASH: “Sanger sequencing, continued enhancements in technology are improving the read lengths; e.g., 454 Life Sciences (Roche) read lengths have increased from 100 bp to over 400 bp” (2008.Bansal.HASH, page 10 1st paragraph on left).
3. 2013.Chen.Zhi: “The length of a read is je - js + 1, where je and js are the end and the start positions of the read, respectively.” (2013.Chen.Zhi, page 2, 3rd paragraph on right).

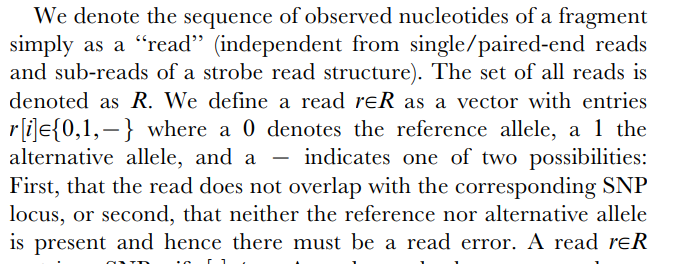
**Evaluation Using Sequencing/Read Length:**

1. 2008.Chen.HapRec: “A random matrix of SNP fragments is created as follows: (1) Haplotype 1 is generated at random with length m (m ∈ {50, 100, 150}).” (2008.Chen.HapRec, page 8, 4th to paragraph
2. 2008.Genovese.SpeedHap: “the average distance in bits per seconds of two SNPs in the DNA sequence is quantified as 300 bps on average, and each fragment is of 650 bps. Each fragment covers a number of SNPs in the range roughly [3... 7], thus we chose the length of each fragment in this range.” (Ranging from 300 to 650 bps). (2008.Genovese.SpeedHap, page 7-8 (last to first sentence).
3. 2009.Wu: “two haplotypes with given length, n = 1000, are randomly generated” (2009.Wu, page 8, 1st paragraph)
4. 2010.Duitama.ReFHap: simulated data fragment length ranged from 6 to 12. (2010.Duitama.ReFHap, page 7, table 7).
5. 2010.MaxSAT: “We show the read matrix for the first block of chromosome 22 in Figure 2 as an example, where we have around 2300 reads spanning a block of length around 400.” (2010.MaxSAT, page 5, 2nd to last paragraph on left).
6. 2010.Xu:
   1. (Simulated Data): “Here, we use ms program to generate 40 haplotypes with length 100, and combine them into 20 pairs randomly.” (2010.Xu, page 4, last paragraph on left)
   2. (real Data): “the following procedure is used to select 60 pairs of haplotypes with length 500.” (2010.Xu, page 4, last paragraph on right).
7. 2010.HALLDORSSON:
   1. read length was set at either 100 or 200 to determine desired coverage of a particular percentage of SNPs. (2010.HALLDORSSON, page 5, table 1)
   2. figures 3 & 4 are fragment conflict graphs used for evaluation with read length equal to 50 & 1000, respectively. (2010.HALLDORSSON, page 10 & 11, figures 3&4).
8. 2011.Mousavi.HapSAT: “Tables 1–3 show the results for fragment lengths of 100, 350, and 700” (2011.Mousavi.HapSAT, page 4, top right paragraph)
9. 2012.Efros.PPHS:
   1. (evaluation) “Read length was set to 400” (2012.Efros.PPHS, page 7, figure 4)
   2. evaluation): “The read length of tests 1,2 was 400 while for test 3,4 it was 2000 bases”. (2012.Efros.PPHS, page 7, table 1)
10. 2012.Wang.GAHap: “This data set is presented using three parameters: (i) haplotype length l (100, 350 and 700), (ii) the coverage c (3, 5, 8 and 10), and (iii) the error rate e (10%, 20% and 30%).” (2012.Wang.GAHap, page 5, 1st paragraph on left)
11. 2012.Zhi.HapSeq: “We then generated 16 sets of chromosomes, representing all combinations of sample sizes (K =60 or K =100), read lengths (36 or 75 bp), sequencing error rates (0.2 or 0.5%) and paired end setting (‘paired’ or ‘unpaired’)” (2012.Zhi.HapSeq, page 4, 1st paragraph on right).
12. 2012.Aguiar.HapCompass: read length is set to 100 bps. (2012.Aguiar.HapCompass, page 10, table 2)
13. 2012.Dewal.HATS: simulation parameter: mean length of a recurrent amplicon, default set to 390 kb. (2012.Dewal.HATS, page 3, table 1)
14. 2012.Xie.H-BOP: simulated data was generated with an average fragment length = 3. While the real data had an average fragment length at 18.03. (2012.Xie.HBOP, page 4, 1st paragraph on left (simulated data), last paragraph on left (real data)).
15. 2013.Matsumoto.MixSIH:
    1. Simulated Data: “randomly dividing them into subsequences of length between l1 and l2. We then randomly flipped the binary values of the fragments from 0(1) to 1(0) with probability e. In the following, we use M = 1000, c = 5, l1 = 3, l2 = 7 and e = 0.1” (2013.Matsumoto.MixSIH, page 5, 2nd paragraph on bottom right).
    2. Real Data: “the diploid genomic DNA was fragmented into pieces of length about 40 kilo-bases” (2013.Matsumoto.MixSIH, page 5, last paragraph on right).
16. 2013.Yang.HARSH:
    1. “Through extensive simulations we show that the gain in performance of our approach over existing models extends to realistic read lengths (e.g. 100–400 bp), making our approach readily applicable to existing sequencing datasets” (3012.Yang.HARSH, page 2, 2nd paragraph bottom left).
    2. “we can observe that the lower bound of haplotype assembly achieves similar performance as haplotype phasing only under the unrealistic read length 4000 bp.” (2013.Yang.HARSH, page 6, 2nd paragraph from bottom left).
    3. “For pair-end reads with fixed length 1000 bp mean and 100 bp standard deviation” (2013.Yang.HARSH, page 6, last paragraph on right).
    4. : “We also evaluate HARSH with different read lengths. At fixed coverage 4, we simulated pair-end reads with 1000, 2000, 3000 and 4000 bp in each end.” (2013.Yang.HARSH, page 7, last paragraph on left).
17. 2013.Zhang.HapSeq2:” These 60 sets are simulated with 6 scenarios, each with 10 repetitions: (i) 36 bp reads, unpaired (coded as 36-0); (ii) 100 bp reads, unpaired (100-0); (iii) 36 bp reads, paired with 250 bp insert (36-250); (iv) 100 bp reads, paired with 250 bp insert (100-250); (v) 100 bp reads, paired with 500 bp insert (100-500); and (vi) 100 bp reads, paired with 1000 bp insert (100-1000).” (2013.Zhang.HapSeq2, page 4, 1st paragraph on left).
18. 2014.Aquiar.HapCompass\_Tumor: “The reads simulated from HapCompass include medium (200bp) and long (2000bp) read lengths with error rates of 2% and 5% respectively to model the higher error rates associated with long-read high-throughput sequence technologies.” (2014.Aguiar.HapCompass\_Tumor, page 8, last paragraph).
19. 2014.Berger.HapTree:” The fragment length (frag\_len) is normally distributed with a mean of 550 and standard deviation of 30, but with min and max lengths of 500 and 600 respectively.” (2014.Berger.HapTree, page 8 1st paragraph on right).
20. 2014.Kuleshov.ProbHap: “The long reads have an average length of 40 kb and cover the genome at a depth of 3.” (2014.Kileshov.Probhap, page 2, 2nd paragraph on right).
21. 2014.Matsumoto.CSP:” The reads cluster which is significantly large (> 30 kb for Kaper’s data and > 45 kb for Duitama’s data) are divided into multiple reads cluster so that each cluster length is below threshold (30 kb and 45 kb, respectively).” (2014.Matsumoto.CSP, page 6, 2nd paragraph on left).
22. 2014.Mazrouee.FastHap: “The variant matrix used for haplotype assembly was generated based on aligned short reads with paired-end method for each pair of various length (from 15 to 200 bp each end)” (2014.Mazrouee.FastHap, page 5, last paragraph on left).
23. 2014.Pan.WinHap2: “We use well-known Hudson’s software “ms” [26] to generate simulated genotype sets with N = 50, N = 100, N = 200, N = 500 and M = 10,000, M = 20,000, M = 50,000, M = 100,000, M = 1,000,000. Here N means the number of sequences and M means the length of sequences.” (2014.Pan.WinHap2, page 5, 1st paragraph on left).
24. 2016.HapCol: “We also assessed accuracy and performance of HAPCOL on a large collection of realistically simulated datasets reflecting the characteristics of ‘future-generation’ sequencing technologies that are currently (or soon) available (coverage up to 25, read length from 10,000 to 50,000 bases, substitution error rate up to 5% and indel rate equal to 10%)” (2016.HapCol, Page 2, last paragraph on right).
25. 2017.Edge.HapCUT2: “Hi-C-like reads of length 150 bp were simulated in pairs” (2017.Edge.HapCUT2, Page 11, last paragraph on left).
26. 2017.WhatsHap: “The distribution of the internal segment size (i.e., fragment size minus size of read ends) was chosen to be 100 bp and 250 bp, respectively, which reflects current library preparation protocols. Longer reads with 1,000 bp, 5,000 bp, 10,000 bp, and 50,000 bp were simulated with two different uniform error rates of 1 % and 5 %.” (2017.WhatsHap, page 7, last paragraph).
27. 2018.HapCHAT:
    1. “for chromosome 1 of the Ashkenazim individual, which has an average coverage of 60.2× and an average mapped read length of 8687 bp (basepairs)” (2018.HapCHAT, page 5, 3rd paragraph on right).
    2. As for individual NA12878, which comprises chromosomes 1–22, the average read length ranges from 4746 to 5285 bp. (2018.HapCHAT, page 5, 2nd to last paragraph on right).
28. 2018.Tangherloni.GenHap: “Read lengths were set to 600bp and 5000bp for the Roche/454 and the PacBio RS II sequencers, respectively.” (2018.Tangherloni.GenHap, page 11, 2nd to last paragraph).
29. 2019.Majidian:
    1. Simulated Data:
       1. “The average number of reads are N = 561, 936, and 1873 for coverage values of c = 3, 5, and 10, respectively. The number of SNPs covered in each read is a constant value equal to.” (2019.Majidian, page 7, 2nd paragraph)
       2. “Furthermore, as a consequence of increasing the coverage value, a better performance is achieved by a lower SWER and a higher reconstruction” (2019.Majidian, page 7, 2nd paragraph)
    2. Fosmid Data: NA12878
       1. “The coverage of this data set is c = 3 and the average read length is 40 kb, and hence, is a low-coverage and long-read dataset” ( 2019.Majidian, page 8, 1st paragraph).

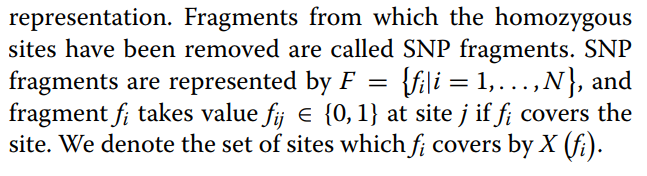
**Alignment info [AI]:**

1. 2007.Levy.HuRef: “We used open-source software (http://sourceforge.net/projects/kmer/) [40,93,94] to generate a one-to-one comparison between HuRef and NCBI human genome reference assembly.” (2007.Levy.HuRef, page 27, 2nd paragraph bottom left)
2. 2008.Genovese.SpeedHap: “Moreover, implicitly, it is assumed that all reads are roughly aligned, which is a very strong assumption within the current shotgun sequencing technology”. (2008.Genovese.SpeedHap, page 10 2nd from top paragraph on right).
3. 2013.Deng: “We assume that all the fragments have been pre-aligned to a reference DNA sequence.” (2013.Deng, page 2, 2nd paragraph on right).
4. 2013.Matsumoto.MixSIH: “Let F = { fi|i = 1, . . . , N} be the set of input fragments which are supposed to be aligned to the reference genome” (2013.Matsumoto.MixSIH, page 3, last paragraph on right).
5. 2014.Matsumoto.CSP: “We aligned Kaper’s data and Duitama’s data to a human reference genome (hg18) using bowtie (version 1.0.0) and bfast (version 0.7.0), respectively” (2014.Matsumoto.CSP, page 5, last paragraph on right).
6. 2017.WhatsHap: “All data sets were created to have 30 · average coverage and were mapped to the human genome using BWA MEM (Li, 2013).” (2017.WhatsHap, page 7, last paragraph).

**Input:**

1. 2001.Lancia: “A class of inputs in which the SNP matrix has gaps but that can still be solved in polynomial time is the following.” (2001.Lancia, page 9, 2nd paragraph)
2. 2004.Panconesi.Fast:
   1. “We have also developed a dynamic programming algorithm of complexity O(32kn) that solves Min Element Removal exactly, where n is the number of rows of the input matrix and k is the maximum length of a fragment.” (2004.Panconesi.FastHare, page 5, 3rd paragraph from top) (1/14)
   2. “The input to Fast Hare is a SNP-fragment matrix M with n rows and m columns” (2004.Panconesi.FastHare, page 5, 5th paragraph from top) (2/14)
3. 2005.Wang:
   1. “Input: as SNP matrix with m rows and n columns” (2005.Wang, page 3, table 1.)
   2. “The input data are short genome fragments with SNPs coming from DNA shotgun sequencing or generated by a resequencing effort for the purpose of large-scale haplotyping.” (2005.Wang, page1, 2nd paragraph on right).
4. 2005.Zhao:
   1. “The input data to this problem is a set of aligned weighted short genome fragments, which can come directly from a shotgun sequencing project or might be generated by a resequencing effort for the purpose of large-scale haplotyping.” (2005.Zhao, page 1, right paragraph)
   2. “INPUT: A SNP matrix M = (mij)m×n and a weight matrix W = (wij)m×n
      1. NOTES: mi: the ith row of the SNP matrix. k: the iteration index. (2005.Zhao, page 3, table 1)
5. 2007.BEAGLE:
   1. “The Beagle phasing algorithm is conceptually simple: at each iteration of the algorithm, phased input data are used to build a localized haplotype-cluster model as described elsewhere” (2007.BEAGLE, page 4, last paragraph on right)
   2. “The randomly phased data are the input for the first iteration of the phasing algorithm.” (2007.BEAGLE, page 5, 3rd paragraph on left).
6. 2007.Levy.HuRef: “An SNP matrix (rows = reads or mate pairs, columns =variants) was constructed as follows: for each variant location, reads whose sequence matched the consensus sequence were assigned state ‘‘0,’’ while reads not matching the consensus were assigned state ‘‘1.’’ A pair of mated reads was merged into a single row only if they were in the same scaffold, with the expected orientation and separated by the expected distance (+ or - 3 SD). Thus, a row in the matrix correspond to one of the following: (i) a pair of mated reads with consistent placements and (ii) a single unmated read or single mated read whose mate is not consistently placed.” (2007.Levy.HuRef, page 29, 2nd paragraph).
7. 2008.Kang: “Input: SNP fragments matrix M, weight matrix W, genotype g population size PS, crossover rate CR, mutation rate MR, the maximum number of population generation GN” (2008.Kang, page 7, table 1)
8. 2008.Bansal.HASH: “The complete data can be represented by a fragment matrix X with m rows and n columns, where each row represents a fragment and each column corresponds to a variant site.” (2008.Bansal.HASH, page 3, 2nd paragraph on right).
9. 2008.Chen.HapRec “Input: M, an n × m matrix of SNP fragments.”(2008.Chen.HapRec, page 4, last paragraph/proof outline).
10. 2008.Genovese.SpeedHap: “The input to the problem is a set of fragments F and a set of SNP positions S” (2008.Genovese.SpeedHap, page 4 last paragraph on right)
11. 2008.Xie.P\_WMLF/GS, page 6, figure 6: an m x n SNP matrix M, an m xn weight matrix W, a 3 x n GenoSpectrum F, and a weighted coefficient gw.
12. **2009.Howie.IMPUTE2:?**
13. 2009.Wu: “The classical SOM algorithm we described in the last section uses SNP values on fragments as input data.” (2009.Wu, page 5, 2n paragraph).
14. 2010.Kang.HapAssembler: “Input: SNP fragments matrix M, weight matrix W, genotype g, population size PS, crossover rate rc, mutation rate rm, the maximum number of population generation GN” (2010.Kang.HapAssembler, page 2, 2nd paragraph on right)
15. 2010.Duitama.ReFHap: “we represent the input of the problem as a matrix M of size mxn where m is the number of fragments and n is the number of variant loci.” (2010.Duitama.ReFHap, page 2, last paragraph on right).
16. 2010.Li.MaCH: “Our model resolves a set of unphased genotypes G into an imperfect mosaic of several template haplotypes.” (2010.Li.MaCH, page 15, 2nd paragraph).
17. 2010.MaxSAT: “ A matrix X of size m×n can be built from the alignment, where m is the number of reads and n is the number of heterozygous sites” (2010.MaxSAT, page 2, last paragraph on right)
18. 2010.Xu: “There are m input neurons which represent m SNP fragments.” (2010.Xu, page 2, 2nd paragraph on right).
19. 2011.HALLDORSSON:
    1. “The input for the workflows that we consider are any of the combination of: (A) genotype data (B) next generation sequencing (NGS) (C) pedigree information.” (2011.HALLDORSSON, page 1, first paragraph)
    2. “Thus, the input to the haplotype assembly problem is an m ×n SNP matrix M whose m rows correspond to fragments f1, ..., fm and each fragment fi covers at least 2 of the n SNP sites.” (2011.HALLDORSSON, page 2, last paragraph)
20. 2011.Mousavi.HapSAT: “Input instance: a read matrix R m x n, where n is length of the read and m is the number of reads a row is comprised of. (2011.Mousavi.HapSAT,page 2, 3rd paragraph from bottom left)
21. 2012.Efros.PPHS: “The input for our algorithm is the sequence data, i.e., the set of reads obtained from the sequencer, where each read is assumed to be generated by randomly picking a position in the genome, randomly picking one of the copies of the chromosome in that position, and adding noise using the parameter ε in each position of the read independently” (2012.Efros.PPHS, page 2, 2nd paragraph on right)
22. 2012.Wang.GAHap: “ a m n matrix X, where m and n represent the total number of fragments and the length of the reference genome sequence, respectively.” (2012.Wang.GAHap, page 2, 2nd paragraph on right)
23. 2012.Aguiar.HapCompass: “We define an m x n SNP-fragment matrix M whose m rows correspond to fragments” (2012.Aguiar.HapCompass, page 3, 2nd paragraph)
24. 2012.Bayzid.HMEC: SNP Matrix M, an ordered set of fragments (2012.Bayzid.HMEC, page 2, 2nd paragraph on right)
25. 2012.Xie.B-HOP: “the input aligned fragments are encoded as an m×n SNP matrix M [15,16,25], where m is the number of fragments and n the number of SNP loci.” (2012.Xie.BHOP, page 6, 2nd to last paragraph on left).
26. 2012.Delaneau.SHAPEIT: “It takes as input a set of genotypes and a genetic map, and produces as output, either a single set of estimated haplotypes, or a haplotype graph that encapsulates the uncertainty about the underlying haplotypes.” In manual: [http://mathgen.stats.ox.ac.uk/genetics\_software/shapeit/shapeit.html#gettingstarted](http://mathgen.stats.ox.ac.uk/genetics_software/shapeit/shapeit.html" \l "gettingstarted)
27. 2012.Dewal.HATS: “HATS jointly considers input regarding n tumor and matched normal samples (1 j n). Specifically, input data corresponding to a particular sample j includes the input data from the naive model” (2012.Dewal.HATS, page 10, 2nd paragraph on right bottom).
28. **2012.Williams.HAPIUR:?**
29. 2012.Zhi.HapSeq:
    1. “HapSeq uses three input files: the count file, the site file, and the haplotype count (“jump”) file from jumping reads that cover two consecutive potential polymorphic sites.” (Manual: <file:///C:/Users/Allison%20Bertie/Downloads/HapSeq-Manual.pdf>, page 2, 2nd paragraph).
30. **2013.Delaneau.SHAPEIT2:?**
31. **2013.He.Hap\_SeqX:?**
32. 2013.Yang.HARSH: “We aim to use both the reference panel and sequencing data to perform haplotype phasing as shown in Figure 1. Formally, suppose that we are only considering L biallelic SNPs, M reads and N reference haplotypes. Each read is represented by X…” (2013.Yang.HARSH, page 3, 3rd paragraph on right).
33. 2013.Zhang.HapSeq2:
    1. **“**HapSeq2 uses four input files: the count file, the site file, and the haplotype count (“jump”) file from jumping reads that cover two consecutive potential polymorphic sites, and the sequencing read file that contains the sequencing reads that cover two or more adjacent and non-adjacent sites.” (Manual: <file:///C:/Users/Allison%20Bertie/Downloads/HapSeq2-Manual%20(2).pdf>, page 3, 1st paragraph)
34. 2013.Chen.Zhi:
    1. “In the haplotype assembly problem, we are given a matrix of X whose entries each belong to {0,1,-} (i.e. each row of X is a ternary string).” (2013.Chen.Zhi, page 2, 2nd paragraph on right).
    2. Each row of X corresponds to a read, whereas each column corresponds to an SNP site.” (2013.Chen.Zhi, page 2, 2nd paragraph on right).
35. 2013.Deng: “The input to the haplotype assembly problem is a set of fragments sequenced from the two copies of a chromosome of a single individual.” (2013.Deng, page 2, paragraph 2 on right).
36. 2013.Matsumoto.MixSIH: “The SIH problem takes a set of aligned SNP fragments F as input and outputs a hidden phase vector” (2013.Matsumoto.MixSIH, page 4, first paragraph on left).
37. 2013.Zhang.DBM:” DBM requires input of read counts of two alleles per putative SNP per individual. The SNPs should be ordered by their positions. DBM can also work for partially ordered (e.g. when reads are aligned to contigs) or unordered SNPs, but the accuracy of genotype calling and haplotype phasing will be affected due to loss of LD information” (2013.Zhang.DBM, page 2, 2nd paragraph on right).
38. 2014.Aguiar.HapCompass\_Tumor: “input : Sequence reads, variant calls, and number of distinct haplotypes k” (2014.Aguiar.HapCompass\_Tumor, page 17, algorithm 1 outline).
39. 2014.Chen.HGHap:” Suppose that there are m SNP fragments from a pair of chromosomes and that the length of the corresponding haplotypes is n. These fragments can be represented by an SNP matrix S, in which each row corresponds to an SNP fragment and each column corresponds to an SNP site.” (2014.Chen.HGHap, page 2 last sentence on left and first on right).
40. 2014.Kuleshov.ProHap:” Formally, an instance of the SIH problem is defined by a pair of n m matrices M, Q” where M corresponds to a heterozygous position and reads matrix and Q is a q-score matrix. (2014.Kuleshov.ProbHap, page 4, 2nd paragraph on left).
41. 2014.Mazrouee.FastHap:” We assume that the input to the haplotype assembly algorithm is a 2D array containing only heterozygous sites of the aligned fragments, called variant matrix, X, of size m n, where m denotes the number of fragments (aligned DNA short reads) and n represents the number of SNPs that the union of all fragments cover.” (2014.Mazrouee.FastHap, page 2, 1st paragraph on right).
42. 2014.Pan.WinHap2: “The input to WinHAP2 consists of n genotype vectors, each with m coordinates corresponding to m SNPs” (2014.Pan.WinHap2, page 2, 2nd to last paragraph on right).
43. 2014.Berger.HapTree:
    1. 

(2014.Berger.HapTree, page 3, 1st paragraph on right).

1. 2014.Matsumoto.CSP:
   1. 

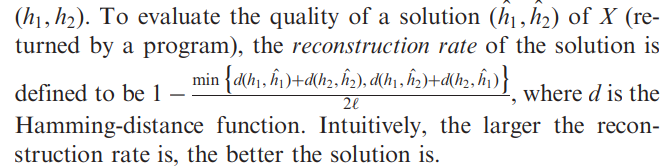
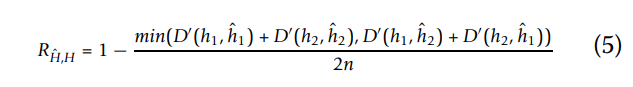
(2014.Matsumoto.CSP, page 3, 2nd paragraph on right).

1. 2015.Ahn.ParticleHap: “. Assume there are m paired-end short reads covering n remaining SNP sites. Such data can be represented by an m × n matrix where the rows contain information provided by the reads while the columns correspond to the SNP sites.” (2015.Ahn.ParticleHap, Page 3, 2nd to last paragraph on left).
2. 2016.HapCol: “A fragment matrix is a matrix M consisting of n rows (fragments) and m columns (SNPs).” (2016.HapCol, page 3, 2nd paragraph on left).
3. 2017.WhatsHap: “The input to the MEC problem is a matrix F with entries in {0, 1, - }. Each row of F corresponds to a fragment/read. Each column of F corresponds to an SNP position.” (2017.WhatsHap, page 4, 2nd paragraph).
4. 2017.Edge.HapCUT2: “The input to HapCUT2 consists of haplotype fragments (sequence of alleles at heterozygous variant sites identified from aligned sequence reads) and a list of heterozygous variants (identified from WGS data).” (2017.Edge.HapCUT2, page 2, last paragraph on right).
5. 2018.Olyaee.AROHap: “The input data which is called SNP matrix is a M x N matrix S = (Sij); M is the number of fragments with length N that are provided from a pair of chromosomes” (2018.Olyaee.AROHap,page 2, last paragraph on right).
6. 2018.HapCHAT: “For each read set, we provide to HapCHAT the corresponding BAM and VCF file. We run HapCHAT on this input pair on otherwise default settings, with the exception of providing it the reference genome (hg37) via the optional parameter --reference.” (2018.HapCHAT, page 13, last paragraph on left).
7. 2018.Na.PEATH: “ Suppose that two n x m matrices are M and Q.” M is the input matrix. Where M is sequence read by heterozygous variant position matrix and Q is a quality score matrix. (2018.Na.PEATH, page 2-3, last paragraph on left and first paragraph on top right).
8. 2018.Tangherloni.GenHap: “GenHap exploits a very simple and efficient structure for individuals, which encodes as a binary string a partition of the fragment matrix M.” (2018.Tangherloni.GenHap, page 6, last paragraph).
9. 2018.Tangherloni.GenHap: “The resulting SAM file was converted into the compressed Binary Alignment/Map (BAM) format for a more efficient manipulation [34]. In order to store the SNPs, we exploited the Variant Call Format (VCF) [35], which is the most used format that combines DNA polymorphism data, insertions and deletions, as well as structural variants. Lastly, the BAM and VCF files were processed to produce a WhatsHap Input Format (WIF) file [5], which is the input of GenHap.” (2018.Tangherloni.GenHap, page 11, 2nd paragraph).
10. 2019.Majidian:
    1. “input: N aligned” (2019.Majidian, page 5, algorithm 1).

**Output:**

1. **2001.Lancia: ?**
2. 2004.Panconesi.FastHare: The output consists of three objects:
   1. – Two haplotypes of length n, i.e. two n-bit strings h1 and h2 over the alphabet {−, A, B}. These are the algorithm’s guesses of the blue and red haplotype.
   2. – A SNP-fragment matrix M with m columns and n rows. In this matrix the non null entries of each row of M can be modified.
   3. – A partition of the rows of M (fragments) into two groups. Each group corresponds to one of the two haplotypes in output. (2004.Panconesi.FastHare, page 5, 4th paragraph from top)
3. **2005.Wang: ?**
4. 2005.Zhao: “OUTPUT: Stable haplotypes h1, h2”. (2005.Zhao, page 3, table 1)
5. 2007.BEAGLE: “Beagle will output the most-likely haplotype pairs by default but also includes an option for sampling haplotype pairs conditional on an individual’s genotypes at the end of the phasing algorithm.” (2007.BEAGLE, page 11, 2nd paragraph on right)
6. **2007.Levy.HuRef: ?**
7. 2008.Kang: “Output: a pair of haplotypes h1, h2” (2008.Kang, page 7, table1)
8. 2008.Chen.HapRec: “ Output: two haplotypes H1 and H2.” (2008.Chen.HapRec, page 4, Proof outline/ last paragraph).
9. 2008.Genovese.SpeedHap: “The output is a pair of consensus strings S1 and S2.” (2008.Genovese.SpeedHap, page 4, last paragraph on right).
10. 2008.Xie.P\_WMLF/GS, page 6, figure 6: a solution to the WMLF/GS problem for M. the minimal E[P] and the corresponding SE[P] Eq.2
11. **2008.Chen.HapRec: ?**
12. **2008.Bansal.HASH: ?**
13. 2009.Wu: “The output layer is same, i.e. two output neurons represent a pair of the desired haplotypes.” (2009.Wu, page 5, 3rd paragraph)
14. 2009.Howie.IMPUTE2:
    1. <https://sourceforge.net/projects/tumorhats/files/>
15. 2010.Kang.HapAssembler: “Output: a pair of haplotypes h1, h2” (2010.Kang.HapAssembler, page 2, 2nd paragraph on right)
16. 2010.Duitama.ReFHap: “ Since we assume that all loci are heterozygous, the output of ReFHap is just one haplotype h and h¯ is just the haplotype obtained by flipping every allele call in h.”(2010.Duitama.ReFHap, page 4, 2nd paragraph on right).
17. **2010.Li.MaCH: ?**
18. 2010.MaxSAT: “ The haplotypes can be represented as an unordered pair of binary strings H =(h1,h2), each of length n. Since all the sites are heterozygous, h2 is the bit-wise complement of h1.” (2010.MaxSAT, page 3 first paragraph on left).
19. 2010.Xu: “The two output neurons represent two subsets of fragments from which two haplotypes can be assembled.” (2010.Xu, page 2, 2nd paragraph on right).
20. 2011.HALLDORSSON: “Output a pair of haplotypes for each individual such that each individual sharing a haplotype do so.” (2011.HALLDORSSON, page 7, 3rd paragraph from bottom).
21. 2011.Mousavi.HapSAT: “Outputs: (1) a pair of haplotypes H = (h0, h1), and (2) a covering function f(.) from {1, ..., m} to {0, 1}” (2011.Mousavi.HapSAT, page 2, 3rd paragraph from bottom left)
22. 2012.Efros.PPHS: “The output of the algorithm results in another tree with a possibly different set of haplotypes, HPPHS.” (2012.Efros.PPHS, page 5, 1st paragraph on right)
23. 2012.Aguiar.HapCompass: “Output the phasing corresponding to any spanning tree of GC and the number of weighted edges corrected as the score of this phasing (or output the weight of all remaining edges in GC).” (2012.Aguiar.HapCompass, page 6, Algorithm 1 outline)
24. 2012.Bayzid.HMEC: outputs current haplotypes (2012.Bayzid.HMEC, page 4, Algorithm 1: HMEC Algorithm)
25. 2012.Delaneau.SHAPEIT:
    1. output the most likely pairs of haplotypes for each sample” (<http://mathgen.stats.ox.ac.uk/genetics_software/shapeit/shapeit.html#output>)
26. **2012.Dewal.HATS:** 
    1. <https://sourceforge.net/projects/tumorhats/files/>
27. 2012.Wang.GAHap:
    1. “GAHap is written in JAVA and is free to access for all educational and non-commercial purposes. GAHap’s source code can be easily obtained by contacting the authors.” (2012.Wang.GAHap, page 8, 2nd to last paragraph on left).
28. **2012.Williams.HAPIUR:** 
    1. **“**HAPI-UR produces output phased results in either Eigenstrat (the default) or IMPUTE2 format as described below.” (<https://code.google.com/archive/p/hapi-ur/downloads>)
29. **2012.Xie.HBOP: ?**
30. 2012.Zhi.HapSeq:
    1. **“**There are two main output files: the imputed genotypes at each site for each individual and the inferred pair of haplotypes for each individual. These files have the same format as those from Thunder. We have included two example files.” (<https://github.com/ZhiGroup/HapSeq2/blob/master/HapSeq-Manual.pdf>)
31. 2013.Chen.Zhi: “Given X, we want to compute the unknown haplotypes, which are an unordered pair H=(h,h’), of binary strings each of length n, where n is the number of columns in X. Such a pair is called a solution of X.” (2013.Chen.Zhi, page 2, 3rd paragraph on right).
32. 2013.Delaneau.SHAPEIT2:
    1. output the most likely pairs of haplotypes for each sample” (<http://mathgen.stats.ox.ac.uk/genetics_software/shapeit/shapeit.html#output>)
33. 2013.Deng: “The refining procedure stops when, at the end of some iteration, the obtained haplotypes no longer change, or when a certain number of iterations have been finished. The two haplotypes output in the last iteration are the output of the refining procedure.” (2013.Deng, page 4, 2nd to last paragraph on right).
34. **2013.He.Hap\_SeqX: ?**
35. 2013.Matsumoto.MixSIH: “The SIH problem takes a set of aligned SNP fragments F as input and outputs a hidden phase vector” (2013.Matsumoto.MixSIH, page 4, first paragraph on left).
36. 2013.Yang.HARSH:
    1. “Randomly initialize haplotype H … Collect samples by repeating steps 2 and 3, and output the one with highest probability” (2013.Yang.HARSH, page 4, algorithm 1).
    2. HARSH will output the most probable haplotypes given the sequencing reads and haplotype references. As an initial version, we only output the most probable prediction, the confidence and multiple predictions will be available in later versions. (<http://genetics.cs.ucla.edu/harsh/manual.html>)
37. 2013.Zhang.DBM: “. The output of DBM includes the inferred haplotypes (and genotypes and SNP calls), the recombination probabilities at each SNP and the underlying haplotype structures.” (2013.Zhang.DBM, page 4, 2nd paragraph on left).
38. 2013.Zhang.HapSeq2: “Construct the consensus haplotypes and genotypes of each individual based on haplotypes and genotypes obtained from each outer iteration.” (2013.Zhang.HapSeq2, page 2, 2nd to last paragraph on right).
39. 2014.Aguiar.HapCompass\_Tumor: “output: k haplotypes” (2014.Aquiar.HapCompass\_Tumor, page 17, algorithm 1 outline).
40. **2014.Berger.HapTree: ?**
41. 2014.Chen.HGHap: “Step 4: Haplotype determination: Determine the haplotypes from two groups of vertices.” (h1 & h2) (2014.Chen.HGHap, page 5, 1st paragraph on right).
42. 2014.Kuleshov.ProbHap:” The output of PROBHAP is a set of haplotype blocks in the format of RefHap and HapCUT. In addition, PROBHAP also produces at each position three confidence scores that can be used to identify locations where the phasing results are less accurate” (2014.Kuleshov.ProbHap, page 2, 4th paragraph on left).
43. 2014.Matsumoto.CSP:
    1. Output\_file1:This contains the haplotypes and these probabilities of the target individual for each SNP fragment.Because we use sliding-window calculation, a SNP fragment appears many times.
    2. Output\_file2:This contains the CSP values for each SNP fragment. (<https://sites.google.com/site/hmatsu1226/software/csp>)
44. 2014.Mazrouee.FastHap:
    1. **Availability**: An implementation of FastHap is available for sharing on request. **Contact**: [ude.alcu.sc@hedipes](mailto:dev@null) (2014.Mazrouee.FastHap, page 1, under abstract).
45. 2014.Pan.WinHap2: “After we get the haplotype results of all segments, we must merge them to obtain the whole haplotype result.” (2014.Pan.WinHap2, Page 7, last paragraph on right).
46. 2015.Ahn.ParticleHap:
    1. “reconstructs a pair of haplotypes” (2015.Ahn.ParticleHap, page 8, 1st paragraph on right)
    2. “The output file contains phased haplotypes for each haplotype block” (<https://sites.google.com/site/asynoeun/particlehap>)
47. 2016.HapCol: “Output: a conflict free matrix M’ obtained from M with the minimum number of corrections.” (2016.HapCol, Page 3, paragraph 4 on left).
48. 2017.Edge.HapCUT2:
    1. “If the maximum posterior probability is less than a user-defined threshold (0.8 by default), then the variant is pruned from the output haplotypes (see Supplemental Methods for details).” (2017.Edge.HapCUT2, page 10, 1st paragraph on right).
    2. “HapCUT2 now outputs the phased variants to a VCF file "output\_haplotype\_file.phased.vcf" Haplotype blocks are printed to the output file. For a given block, column 2 represents the allele on one chromosome copy (0 for reference, 1 for variant), while column 3 represents the allele on the other copy.” (<https://github.com/vibansal/HapCUT2/blob/master/README.md>).
49. 2017.WhatsHap: “It is then straightforward to derive the two haplotypes h1 and h2” (2017.WhatsHap, page 7, 2nd to last paragraph).
50. 2017.WhatsHap:
    1. **“**In this work we have concentrated on assembling SNP haplotypes from reads of a sequenced genome.” (2017.WhatsHap, page 11, 3rd paragraph).
    2. <https://whatshap.readthedocs.io/en/latest/guide.html> – manual/used guide
51. 2018.HapCHAT: “It is the resulting set of phasings by WhatsHap, in the form of phased VCF, that we use for the basis of comparison with the other methods.” (2018.HapCHAT, page 11, last paragraph on right).
52. 2018.Olyaee.AROHap: “Then, H1 and H2 as output haplotypes are reconstructed by merging fragments residing in C1 and C2, respectively.” Where C1 & C2 are partitioned fragment clusters. (2018.Olyaee.AROHap, page 3, 3rd paragraph on left).
53. 2018.Na.PEATH: “output: haplotype with the best fitness” (2018.Na.PEATH, page 2, right side of page).
54. 2018.Tangherloni.GenHap: “GemSIM can in principle simulate data from any sequencing technology producing output data encoded in the FASTQ format [32], for raw reads, and Sequence Alignment/Map (SAM), for aligned reads. In this work, we exploited the error model for the Roche/454 sequencer, already available in GemSIM, and defined an additional error model for the PacBio RS II technology.” (2018.Tangherloni.GenHap, page 11, last paragraph).
55. 2019.Majidian:
    1. “output: Haplotypes” (2019.Majidian, page 5, algorithm 1).

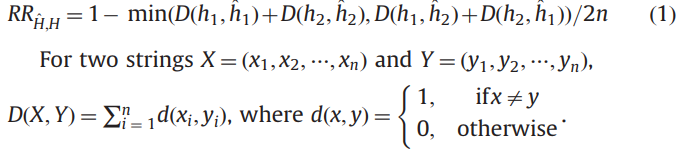
**Evaluations using Reconstruction Rate:**

1. 2005.Wang: “In our experiments, we use reconstruction rate (RR), the similarity degree between the original haplotypes and the reconstructed haplotypes, to measure performance of an algorithm or a model” (2005.Wang, page 3, 3rd paragraph on right).
2. 2008.Chen.HapRec: “The reconstruction rate is defined as the ratio of the total number of correctly reconstructed bits to the total number of bits in two haplotypes.” (2008.Chen.HapRec, page 8, 3rd paragraph from bottom)
3. 2008.Xie.P\_WMLF\_GS: “The reconstruction rate of haplotypes is defined as the ratio of the number of the SNP sites that are correctly inferred out by an algorithm to the total number of the SNP sites of the haplotypes.” (2008.Xie.P\_WMLF\_GS, page 6, 2nd paragraph on right).
4. 2008.Genovese.SpeedHap:
   1. “The quality of the output is usually measured by the reconstruction rate that is essentially the percentage of the output strings correctly reconstructed” (2008.Genovese.SpeedHap, page 4, last paragraph on left).
   2. “Simulation experiments over a haplotype of 600 SNPs embedded in a 180,000 DNA strand with an average coverage of 6.88 shows that 512 positions could be recovered and of these 490 are determined correctly. Thus, the reconstruction rate is 0.957.” (2008.Genovese.SpeedHap, page 4, last paragraph on left).
5. 2009.Wu:
   1. “The reconstruction rate expresses the similarity between the original haplotypes and the reconstructed haplotypes.” (2009.Wu, page 7, last paragraph).
   2. “When the error rate of SNP fragments is large, the reconstruction rate by NSOM (novel self-organizing map) under different parameter settings is still identical for most of instances since the reconstruction rate is obtained by averaging over all instances.” (2009.Wu, page 11, 2nd paragraph).
6. 2012.Wang.GAHap:”
   1. Thus, most approaches (including our approach in this work) use reconstruction rate (RR) – besides MEC – to measure quality of their results.” (2012.Wang.GAHap, page 3, 2nd paragraph on left bottom).
   2. “Table 3 shows the average reconstruction rate for results calculated by baseline, 2d-mec and GAHap; baseline and 2d-mec results are borrowed from Gearci’s research [3]. This table shows that reconstruction rate of solutions found by baseline and GAHap were almost very similar despite the fact that baseline was advantaged by having all information a priori.” (2012.Wang.GAHap, page 6, last paragraph on right)
7. 2012.Bayzid.HMEC: “Experimental results indicate that the reconstruction rates of both HapCUT and HMEC are reasonably good. For very low coverage, reconstruction rate of HapCUT is slightly better than HMEC. However, as the coverage rate increases, HMEC begins to outperform HapCUT.” (2012.Bayzid.HMEC, page 9, 1st paragraph on left).
8. 2013.Chen.Zhi (page 6, 1st paragraph on right
   1. 
   2. “In particular, the average reconstruction rate achieved by our exact program for the general case is better than the best reconstruction rate reported by Geraci (2010), but the average reconstruction rate achieved by our exact program for the all-heterozygous case is worse.” (2013.Chen.Zhi, page 6, 2nd paragraph on right)
9. 2013.Deng:
   1.  “Throughout our experiments, we measure the performance of our algorithm by the reconstruction rate, a frequently used criterion in the haplotype assembly problem. Given a problem instance in the benchmark, the reconstruction rate is defined as follows:

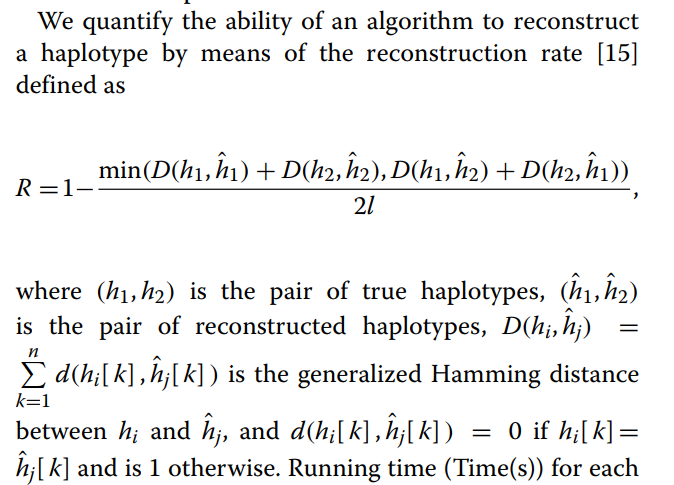
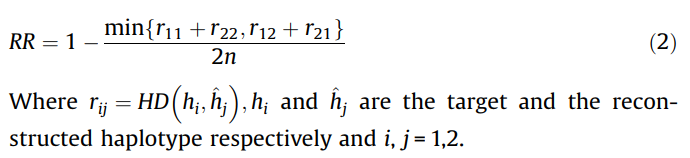
(2013.Deng, page 5, last paragraph on right)

* 1. “Intuitively speaking, the reconstruction rate measures the ability of an algorithm to reconstruct the correct haplotypes.” (2013.Deng, page 6, 2nd paragraph on left).

1. 2014.Chen.HGHap: “The performance of the algorithm is measured by the reconstruction rate (RR), which is defined as:

”

(2014.Chen.HGHap, page 2, 2nd paragraph on right)

1. 2015.ParticleHap:
   1. (2015.Ahn.ParticleHap, page 8, 1st paragraph on right)
   2. “As can be seen in those tables, ParticleHap assembles haplotypes with the reconstruction rates of 97.85 % and 95.68 % when the data is affected by the genotype calling error rates of 4 % and 8 %, respectively” (2015.Ahn.ParticleHap, page 8, 2nd paragraph on right).
2. 2018.Olyaee.AROHap: (page 3, 2nd paragraph on left)
   1.  “After reconstructed haplotypes are obtained, reconstruction rate (RR) as a useful metric is used for evaluation (Wang et al., 2005; Zhao et al., 2007; Geraci, 2010). This relation is defined as follows:
   2. “Specially, for the datasets with e = 20 and 30%, AROHap achieves highest reconstruction rates or the second highest reconstruction rates in most cases.” (2018.Olyaee.AROHap, page 7, last paragraph on right).
   3. “Although, the performance of the proposed method on the reconstruction rates of gained haplotypes is very significant, it may be slow when encountered with large datasets in some cases. To address this problem, the method can be parallelized to run on GPUs as a future improvement.” (2018.Olyaee.AROHap, page 8, last paragraph on right)
3. 2019.Majidian:
   1. “Next, in the section of Results, these algorithms are compared to some benchmark methods in terms of the reconstruction rate and the switch error rate.” (2019.Majidian, page 2, 4th paragraph).
   2. “As seen in both Tables 3 and 4, the proposed HapOPT algorithm outperforms the others in terms of the reconstruction rate as well as the SWER.” (2019.Majidian, page 7, last paragraph).
   3. “From the above results, one can observe that HapOPT outperforms SDhaP and AltHap in terms of the reconstruction rate…” (2019.Majidian, page 8, 2nd to last paragraph).

**Evaluation using Switch Error:**

1. 2008.Bansal.HASH:
   1. “Average switch distance or switch error rate (Lin et al. 2002) is defined as the fraction of positions for which the phase between the two haplotypes is different relative to the previous position.” (2008.Bansal.HASH, page 6, last paragraph on left).
   2. “The absolute accuracy can be expressed in terms of the “switch error rate” (Lin et al. 2002) or the fraction of adjacent pairs of variants whose phase in the HuRef haplotypes is incorrect.” (2008.Bansal.HASH, page 8, last paragraph on left).
   3. “We have computed two independent estimates of the switch error rate: one based on the haplotypes samples generated by our MCMC algorithm and another through comparison of the HuRef haplotypes to the population haplotypes from the HapMap project.” (2008.Bansal.HASH, page 8, last paragraph on left).
   4. “Based on haplotypes sampled by the Markov chain, the switch error probability for a pair (i, j) was estimated as the fraction of times the less frequent haplotype pair was observed” (2008.Bansal.HASH, page 8, 1st paragraph on right).
   5. “The switch error rate for a chromosome can be approximated as the average of the switch error probabilities for adjacent pairs. For chromosome 22 of HuRef, the switch error rate was estimated to be 0.009 using 1000 samples.” (2008.Bansal.HASH, page 8, 1st paragraph on right).
   6. “If the inferred HuRef phasing does not match the preferred population phasing, one can infer a switch error with some probability (the probability value depends upon the strength of LD between the pair of variants).” (2008.Bansal.HASH, page 8, 2nd paragraph on right).
2. 2010.Duitama. ReFHap: (ONLY WITH SIMULATED DATA)
   1. “The second measure is the switch error (SE) which is calculated by traversing the resulting haplotypes from left to right and computing the number of times needed to jump from one haplotype to its complement to reconstruct the real haplotype.” (2010.Duitama.ReFHap, page 5, 1st paragraph on right).
   2. “For our simulations we can calculate the number of switch errors because we know the true haplotype for each instance.” (2010.Duitama.ReFHap, page 5, 1st paragraph on right).
   3. “ReFHap consistently produces lower MEC and switch errors.” (2010.Duitama.ReFHap, page 5, 2nd paragraph on right).
   4. “However, switch errors are the true measure of quality, not MEC.” (2010.Duitama.ReFHap, page 5, 3rd paragraph on right).
3. 2010.Li.MaCH: “After a pre-specified number of rounds are completed, we generate a pair of consensus haplotypes for each individual. This consensus haplotype pair is defined as the pair that minimizes total switch error when compared to the haplotypes sampled at each round.” (2010.Li.MaCH, page 16, 1st paragraph).
4. 2012.PPHS:
   1. “Generally, there are two types of errors; the first type is regular switch errors (for example two haplotypes 11 and 00 are phased as 10 and 01), and the second type is mismatch errors (two haplotypes 11,00 are phased as 11 and 01). We used the sum of all switch and mismatch errors (S + M when S is the number of switch errors in the data and M is the number of mismatch errors). We refer to this as the SM error metric.” (2012.Efros.PPHS, page 5, 2nd paragraph on left).
5. 2012.Zhi.HapSeq:
   1. “The average genotypic discordance rate, the percentage of imputed genotypes that are inconsistent with the true genotypes and the average switch error which is defined as number of switches between the original haplotype and the reconstructed haplotype, are used as criteria to quantify the performance of Thunder and HapSeq.” (2012.Zhi.HapSeq, page 4, 3rd paragraph on right bottom).
6. 2012.Xie.H-BOP:
   1. “A switch error is an inconsistency between the reconstructed haplotype pair and the real haplotype pair over two contiguous SNPs.” (2012.Xie.HBOP, page 2, 2nd paragraph on right)
   2. “the number of switch errors divided by the phased haplotype length is called switch error rate.” (2012.Xie.HBOP, page 2, 2nd paragraph on right)
   3. “Though the switch errors of HBOP is larger than that of ReFHap, the switch error rates of H-BOP and ReFHap are both 0.014.” (2012.Xie.HBOP, page 5, 1st paragraph on left).
7. 2012.Aquiar.HapCompass:
   1. “The haplotype switch error metric is defined as the number of switches in haplotype orientation required to reproduce the correct phasing (Lin et al., 2002). It was originally developed for the haplotype phasing problem and was among the metrics used in the Marchini et al., 2006 phasing benchmark.” (2012.Aguiar.HapCompass, page 3, section 3.1 1st paragraph)
   2. “Switch error is generally more favorable than pure edit distances for haplotypes because it more accurately models the phase relationship between adjacent SNPs.” (2012.Aguiar.HapCompass, page 3, section 3.1, 1st paragraph)
   3. “… the switch error metric accurately captures the close range relationship between adjacent SNP phase.” (2012.Aguiar.HapCompass, page 3, section 3.1, 2nd paragraph).
8. 2013.Yang.HARSH:
   1. “The genotyping error rate is the proportion of wrongly predicted genotypes, and the switching error is the proportion of switches in the inferred haplotypes to recover the correct phase in an individual. The total error rate is the sum of genotyping error rate and switching error rate.” (2013.Yang.HARSH, page 5 3rd paragraph from bottom right)
   2. “The error rate consists of both genotyping error for all SNPs and switch error within heterozygous SNPs” (2013.Yang.HARSH, page 6, fig. 5)
9. 2013.He.Hap\_SeqX: “. For t=0.01, we observed that for more than 50% chunks, Hap-seqX achieved the optimal solution, namely the switch error rate of Hap-seqX is identical to that of Hap-seq on these chunks.” (2013.He.Hap\_SeqX, page 4, last paragraph on left).
10. 2014.Berger.HapTree:
    1. “Because real polyploid data is hard to come by, we also evaluate HapTree on real human diploid data and find that, when compared to the more accurate trio-based data as the ground truth [17], HapTree significantly reduces the number of switch errors, while remaining on par in terms of MEC score over existing single-individual haplotype assembly methods for diploid genomes.” (2014.Berger.HapTree, page 2, last paragraph on right).
    2. “A widely used measure in diploid phasing is switch error, which is calculated as the number of positions where the two chromosomes of a proposed phase must be switched in order to agree with the true phase. For polyploid phasing, we generalize switch error to vector error.” (2014.Berger.HapTree, page 6, last paragraph on left).
    3. “Notably, when comparing to the ground-truth phase as determined by trio-based phasing, we found HapTree significantly outperforms HapCUT in terms of switch error rate for the phasing experiments on the NA12878 genome for 454 and Illumina datasets.” (2014.Berger.HapTree, page 8, 3rd paragraph on left).
11. 2014.Pan.WinHap2:
    1. “Usually, the individual error rate (IER) [15] and the switch error rate (SER) [13] are used to evaluate the performance of phasing algorithms [27,28]. IER is defined as the percentage of individuals whose genotypes are incorrectly resolved and SER is defined as the ratio between the numbers of switch errors and all the heterozygous loci.” (2014.Pan.WinHap2, page 5, 2nd paragraph on left).
    2. “So we just use switch error rate (SER) to evaluate the performance of WinHAP2 in this paper.” (2014.Pan.WinHap, page 5, 1st paragraph on right).
12. 2014.Kuleshov.ProbHap:
    1. “We measure performance using the concept of a switch error (Browning and Browning, 2011). A switch error is said to occur when the true parental provenance of SNPs on a haplotype changes with respect to the previous position” (2014.Kuleshov.ProbHap, page 2, 3rd paragraph on right).
    2. “Given comparable phasing rates and N50 block lengths, PROBHAP produced haplotype blocks with more accurate longrange phase: the long-range switch error of PROBHAP was 11% lower than that of the second best algorithm, RefHap (Table 1). In addition, PROBHAP also produced 6% fewer short switch errors than RefHap.” (2014.Kuleshov.ProbHap, page 2, last paragraph on right).
13. 2014.Matsumoto.CSP:
    1. “We also used other two accuracy measures, switch error rate and QAN50 [17]. The switch error rate is defined as the frequency of switch errors which are inconsistency between inferred haplotypes and true haplotypes.” (2014.Matsumoto.CSP, page 7, 1st paragraph on left).
    2. “The switch error rate improved after removing suspicious CFs in all conditions. This result is consistent with the result based on pairwise accuracy measure and shows the usefulness of removing CFs with CSP.” (2014.Matsumoto.CSP, page 9, last paragraph on left).
    3. “Switch error rates of MixSIH were lowest in all conditions and this suggests that MixSIH succeeds to extract reliable haplotype regions with MC values.” (2014.Matsumoto.CSP, page 9, last paragraph on left).
14. 2016.HapCol: “Accuracy of the reconstructed haplotypes has been evaluated in terms of (switch) error rate (Browning and Browning, 2011) (i.e. the number of inconsistencies over contiguous phased variants) and in terms of phased positions (i.e. the number of positions for which the tool gave a phase prediction over the total number of positions that can be phased using the fragments given as input).” (2016.HapCol, page 5, last paragraph on right).
15. 2017.WhatsHap: “Switch errors are two consecutive SNP positions whose phases have been mistakenly predicted, and which cannot be interpreted as flip errors. For example, if the correct haplotype is 000111|111000 and the predicted haplotype is 000000|111111, then we count one switch error between positions 3 and 4.” (2017.WhatsHap, page 10, 2nd paragraph).
16. 2017.Edge.HapCUT2:
    1. “. The accuracy of the haplotypes was assessed by comparing the assembled haplotypes to gold-standard trio-phased haplotypes and using the switch error rate and mismatch error rate metrics (see Methods).” (2017.Edge.HapCUT2, page 4, 2nd to last paragraph on right).
    2. “On this data (Fosmid-based dilution pool data), the switch error and the mismatch error rates for HapCUT2 were virtually identical or slightly better than ProbHap, the second best performing method, across all chromosomes” (2017,Edge.HapCUT2, page 4, last paragraph on right).
    3. “. Further, the switch error rates for haplotypes assembled using these two technologies decrease rapidly as coverage is increased initially and saturate quickly after that” (2017.Edge.HapCUT2, page 7, 2nd paragraph on left).
17. 2018.HapCHAT:
    1. “An experimental analysis on real and simulated sequencing reads with up to 60× coverage reveals that we are able to leverage high coverage towards better predictions in terms of both accuracy (switch error rate) and recall (QAN50 score — the Quality Adjusted N50 score, see Discussion Section), as we see an upward trend in both, as coverage increases” (2018.HapCHAT, page 3, 2nd paragraph on left).
    2. “A switch error is the boundary (that is two consecutive SNV positions) between two portions of such a mosaic. The switch error percentage is the ratio between the number of switch errors and the number of phased SNVs minus one (expressed as a percentage).” (2018.HapCHAT, page 13, last paragraph on right).
18. 2018.Na.PEATH:
    1. “Next we measured the switch error rate (SWER) and minimum error correction (MEC) score, according to the noise rate of the phasing matrix, to evaluate the accuracy of the present approach (Fig. 3). The SWER is measured by the ratio of the number of switch errors to the length of the entire haplotype. We analyzed the SWER based on the single switch error (SSWER) and range switch error (RSWER). SSWER was calculated as the number of switch errors at a single position per megabase (sw/mb).” (2018.Na.PEATH, page 4, last paragraph on left). (Figure 3)
19. 2019.Majidian:
    1. “To consider another criterion for performance evaluation, we make use of the SWitch Error Rate (SWER), defined as the number of switches divided by the haplotype length.” (2019.Majidian, page 7, 1st paragraph).
    2. “From the above results, one can observe that HapOPT outperforms SDhaP and AltHap in terms of the reconstruction rate and SWER.” (2019.Majidian, page 8, 2nd to last paragraph).

**Evaluations using MEC Score:**

1. 2008.Bansal.HASH:
   1. “For each chromosome, we compared the haplotype assembly against the fragment matrix and computed the MEC (minimum error correction) score (Bafna et al. 2005), defined as the minimum number of variant calls in the fragment matrix that need to be modified for every fragment to perfectly match one of the two haplotypes.” (2008.Bansal.HASH, page 7, last paragraph on right).
   2. “The MEC score represents a parsimonious estimate of the discordance between the haplotypes and the fragment matrix.” (2008.Bansal.HASH, page 8, 1st paragraph on left).
   3. The haplotype assembly derived using HASH has a lower MEC score for each chromosome, reflecting the greater accuracy of the haplotypes.” (2008.Bansal.HASH, page 8, 1st paragraph on left).
2. 2010.Duitama.ReFHap:
   1. “Different strategies to remove conflicts lead to optimization objectives studied in previous works like finding the minimum number of fragments to remove (MFR), the minimum number of loci to remove (MSR) or the minimum number of allele calls to correct (MEC).” (2010.Duitama.ReFHap, page 3, 2nd paragraph on bottom left).
   2. “The first one is the Minimum Error Correction (MEC), which is the minimum number of changes within the matrix to make it consistent with the answer haplotypes. This measure divided by the total number of allele calls in the input matrix is a good estimator of the allele calling error rate” (2010.Duitama.ReFHap, page5, 1st paragraph on right).
   3. “ReFHap consistently produces lower MEC and switch errors.” (2010.Duitama.ReFHap, page5, 2nd paragraph on right)
3. 2010.MaxSAT:
   1. “Out of these objective functions, MEC, which is the number of conflicts between the sequence reads and the constructed haplotypes, is the most difficult one to optimize. The haplotype assembly problem with MEC as the object function is NP-hard even for gapless reads of length 2, while polynomial algorithms exist for solving the problem with MFR and MSR as the objective function” (2010.MaxSAT, page 2, 2nd paragraph on left).
   2. “The objective function we use is MEC, which is the minimum number of changes, or corrections, that need to be made in the read matrix such that the resulting matrix admits a perfect bi-partition, where each corrected read maps to either haplotype perfectly” (2010.MaxSAT, page 3, 2nd paragraph on left).
   3. “The global MEC score is the sum of the scores from each block and the optimal haplotypes are the concatenation of the haplotypes from each block.” (2010.MaxSAT, page 5, 1st paragraph).
4. 2013.Chen.Zhi:
   1. “. To our knowledge, this is the first time that optimal haplotypes under the MEC model are completely obtained for the filtered HuRef dataset.” (2013.Chen.Zhi, page 2, last paragraph on left).
   2. “With the increase of coverage, the number of errors increases and so does the MEC score.” (2013.Chen.Zhi, page 6, last paragraph on right).
5. 2014.HapTree:
   1. “Among various formulations suggested for this problem, the most commonly used is an NP-hard minimum error correction (MEC) definition [14,15], which aims to identify the smallest set of nucleotide changes required within mapped fragments that would allow a conflict-free separation of reads into two separate homologous chromosomes (or a bipartite separation of the fragment conflict graph)” (2014.Berger.HapTree, page 1, last paragraph on right).
   2. “Determining the quality of a phasing solution depends on whether the true phase is known. When no such information is available, the Minimum Error Correction (MEC) score [15] is a widely used scoring function to measure the quality of phasing solutions.” (2014.Berger, HapTree, page 6, 1st paragraph).
   3. “The MEC score is defined as the minimum (amongst chromosomes) number of mismatches between a phase H and the read set R.” (2014.Berger, HapTree, page 6, 1st paragraph).
6. 2014. Mazrouee .FastHap:
   1. “The average MEC (normalized by number of variant calls) was 2.48, 2.56 and 2.86 for FastHap, HapCUT and Greedy, respectively.” (2014.Mazrouee.FastHap, page 6, last paragraph on right).
   2. “The amount of improvement in MEC using FastHap was 13 and 2.8% compared with Greedy and HapCUT, respectively.” (2014.Mazrouee.FastHap, page 6, last paragraph on right).
   3. “We ran both FastHap and random partitioning algorithms on the same variant matrix 10 times and calculated percentage of improvements in MEC achieved by FastHap. The improvement numbers ranged from 12.17 to 31.64%, with an average improvement of 19.13%.” (2014.Mazrouee.FastHap, page 7, 2nd to last paragraph on left).
7. 2015.Ahn.ParticleHap:
   1. “The minimum error correction (MEC) criterion, in particular, has received a considerable amount of attention and has been broadly used in practice.” (2015.Ahn.ParticleHap, page 2, last paragraph on left).
   2. To evaluate the performance of haplotype assembly, we adopt three measures: the number of phased SNPs (nPhased), the minimum error correction (MEC) score, and running time (Time).” (2015.Ahn.ParticleHap, page 6, last paragraph on right).
   3. “MEC score is the smallest number of entries in the data matrix which need to be changed so that the sequencing information is consistent with an error-free haplotype pair (we report the total MEC score evaluated as the sum of the MEC scores obtained for each haplotype block)” (2015.Ahn.ParticleHap, page 6, last paragraph on right).
   4. “ParticleHap simultaneously provides longer lengths of phased haplotypes as well as lower MEC scores, demonstrating the high accuracy of the proposed algorithm (note that the total number of reads and allele calls involved in the MEC calculation of ParticleHap is larger than those for HapCUT and ReFHap).” (2015.Ahn.ParticleHap, page 7, last paragraph on right).
8. 2018.Na.PEATH:
   1. “Next we measured the switch error rate (SWER) and minimum error correction (MEC) score, according to the noise rate of the phasing matrix, to evaluate the accuracy of the present approach” (2018.Na.PEATH, page 4, 2nd paragraph on left).
   2. “The results showed that the SWERs and MEC of the PEATH method were comparable to those for the ProbHap method and better than those for MixSIH.” (2018.Na.PEATH, page , 1st paragraph on left).
   3. “Moreover, for MEC, the PEATH method showed better results than ProbHap-O. The MECs of PEATH and ProbHap-O were 121 430–121 714 and 122 001, respectively” (2018.Na.PEATH, page 5, 1st paragraph on left).

**Evaluations using Runtime:**

1. 2004.Panconesi.FastHare: Running time is an evaluation parameter. as seen in Fig. 4, Fast Hare shows no increased in running time when the number of fragments increases from 100 to 800. (2004.Panconesi.FastHare, page 9, 1st paragraph).
2. 2005.Wang: “In addition, the running time of the algorithm heavily depends on the error rate of fragments, which is obvious in Table 2.” (2005.Wang, page 4, 1st paragraph on right).
3. 2005.Zhao: “The running time of our algorithms for every example is no more than 1 min.” (2005.Zhao, page 4, 1st paragraph on left).
4. 2007.BEAGLE:
   1. “We must estimate the scaling of our algorithm with respect to sample size empirically because the running time depends on the rate at which the number of nodes and edges at each level grows as the sample size increases.” (2007.BEAGLE, page 5, 1st paragraph on right).
   2. “Depending on the data set, HaploRec-S was 5–172 times slower than Beagle with R = 1, with the largest relative differences in running time resulting from the real data sets genotyped with the Affymetrix 500K SNP array.” (R is the sample per individual) (2007.BEAGLE, page 8, 2nd to last paragraph on right).
5. 2008.Chen.HapRec: “In our tables, the results are the average time and the reconstruction rate of the 1000 executions of algorithm SHR-Three.” (2008.Chen.HapRec, page 8, 3rd paragraph from bottom).
6. 2008.Xie.P-WMLF/GS:
   1. “When n increases, the haplotype reconstruction rate of the three algorithms decreases, and their running time increases accordingly.” (2008.Xie.P\_WMLF\_GS, page 7, 2nd paragraph on right).
   2. “In Figure 8a, when n=20, the haplotype reconstruction rates of P-WMLF/GS, GA-MEC/GI and WMLF are 96.6, 93.0 and 84.9%, respectively; and their running time are 1.3, 0.4 and 0.0007 s.” (2008.Xie.P\_WMLF\_GS, page 7, 2nd paragraph on right).
   3. When n increases to 120, their haplotype reconstruction rates decrease to 93.5, 88.1 and 79.8%; and their running time increases to 10.2, 6.5 and 0.008 s. (2008.Xie.P\_WMLF\_GS, page 7, 2nd paragraph on right).
7. 2009.Howie.IMPUTE2:
   1. “Table 1 shows that IMPUTE v1 was the fastest of the methods considered here, followed by BEAGLE (default), MACH, IMPUTE v2 (k= 40), BEAGLE (50 iterations), fastPHASE (K= 20), IMPUTE v2 (k= 80), and fastPHASE (K= 30).” (2009.Howie.IMPUTE2, page 7, 2nd paragraph on right).
   2. “Another advantage of our approach can be seen by comparing the running times of the restricted and full datasets for BEAGLE and IMPUTE v2. The average BEAGLE run took 3.3 times longer in the full dataset than in the restricted dataset, whereas the IMPUTE v2 running time increased by factor of just 1.1.” (2009.Howie.IMPUTE2, page 12, 2nd paragraph on right)
8. 2010.Duitama. ReFHap:
   1. “Our experiments indicate that although the algorithm is reliable, its running time is too large for whole genome haplotyping.” (2010.Duitama.ReFHap, page 2, 2nd paragraph on right).
   2. “We show through extensive simulation experiments that ReFHap represents an improvement in running time compared to previous algorithms without loss of accuracy.” (2010.Duitama.ReFHap, page 2, 3rd paragraph on right).
   3. “MEC percentage and running time of ReFHap and HapCUT for a real instance with 32347 SNPs and 13905 fragments in chromosome 22

ReFHap HapCUT HapCUT (50 It)

%MEC: 6.32% 6.26% 6.24%

Time: 73.04 Sec 0.99 Hours 50.4 Hours” (2010.Duitama.ReFHap, page 7, table 2).

1. 2010.MaCH: (2010.Li.MaCH, page 25, Table I, under computation time)
2. 2012.Wang.GAHap:
   1. “Table 5 reflects its average processing time and shows that each data set in our experiments was processed within a reasonable time of less than 20 min.” (2012.Wang.GAHap, page 7, last paragraph on right).
3. 2012.Bayzid.HMEC:
   1. “Clearly, HMEC is much faster than GMEC. For example, while HMEC can reconstruct a haplotype with 936 sites in a fraction of a second, GMEC takes 72 seconds.” (2012.Bayzid.HMEC, page 2, 2nd paragraph on left).
   2. “Extensive simulations indicate that HMEC outperforms the genetic algorithms of Wang et al. [5] in terms on both reconstruction rate and running time, and it has better (in most cases) or comparable accuracy and significantly smaller running time than that of HapCUT [6], which is the most accurate heuristic algorithm available.” (2012.Bayzid.HMEC, page 2, 2nd paragraph on left).
4. 2013.He.Hap\_seqX:
   1. “In reality, although the time complexity increased, as we avoid the heavy IO to write the intermediate results on hard disk and read them in, the running time of Hap-seqX is indeed faster than that of Hap-seq.” (2013.He.Hap\_seqX, page 4, 1st paragraph on left).
   2. “The running time of Hap-seq is 9 h and the running time of Hap-seqX is 5 h on a cluster of 300 nodes.” (2013.He.Hap\_seqX, page 4, 1st paragraph on right).
5. 2013.Zhang.DBM:
   1. “Figure 4 shows the computing time of the three programs for the simulated CEU datasets at 3. At N = 10, DBM was the slowest program among the three.” (2013.Zhang.DBM, page 5, 1st paragraph on left).
   2. “At N 20, however, DBM ran up to 8 X faster than THUNDER. Both DBM and THUNDER ran in time complexity , i.e. proportional to the square of the number of states and linear to the number of individuals and SNPs.” (2013.Zhang.DBM, page 5, 1st paragraph on left).
   3. “In Figure 4, we also observed that the computing time of DBM is almost linear with respect to the sample size.” (2013.Zhang.DBM, page 5, last paragraph on right).
6. 2013.Zhang.HapSeq2:
   1. “The interlaced MH-flipping procedure results in longer running time. The time increased is linear to the number of flips and to the number of potential MCMC moves in each round. In total, the running time for HapSeq2 is 2–3 times that of Thunder, and is thus still practical.” (2013.Zhang.HapSeq2, page 7, 1st paragraph on left).
7. 2013.Chen.Zhi:
   1. “The running time of our program is also related to the coverage. With the increase of coverage, the number of errors increases and so does the MEC score. As the result, the running time increases accordingly as shown in Table 3.” (2013.Chen.Zhi, page 6, last paragraph on right).
8. 2013.Deng:
   1. “From Table 1, we can see that for a fixed coverage rate c, when increasing the size of boundOfCoverage, the reconstruction rate of the obtained initial solution gets higher, and the running time increases accordingly.” (2013.Deng, page 6, last paragraph on left).
9. 2013.Matsumoto.MixSIH:
   1. “Our method applies the VBEM algorithm repeatedly and hence is rather slow. It is comparative to HapCUT(previous versoin) and about 10-fold slower than both ReFHap and HapCUT(latest versoin), and from 50-fold to 500-fold slower than both FastHare and DGS.” (2013.Matsumoto.MixSIH, page 12, last paragraph on right).
10. 2014.Berger.HapTree: “Not only does HapTree outperform HapCompass on phasing quality, it is also significantly faster, especially for longer block length. The median runtimes for block length 10 and 106 coverage were (0:00702,0:633) seconds for HapTree and HapCompass, respectively; for block length of 40 and 406 coverage, they were (0:0279,13:099) seconds, respectively.” (2014.Berger.HapTree, page 7, 2nd to last paragraph on right).
11. 2014.Mazrouee.FastHap:
    1. “That is, a higher read coverage allows FastHap to generate better accuracy without significant impact on its running time. In contrast, as the haplotype length grows, HapCUT algorithm runs very slowly compared with FastHap.” (2014.Mazrouee.FastHap, page 6, 1st paragraph on right).
12. 2014.Pan.WinHap2:
    1. “But the running time and memory consumption of WinHAP2 are both much lower than others’. While SHAPEIT2- gets the lowest error rate, its running time is about 60 times and its memory consumption is 30 times than WinHAP2.” (2014.Pan.WinHap2, page 6, 2nd paragraph on left).
13. 2014.Kuleshov.ProbHap:
    1. “Although the three heuristics ran faster than PROBHAP and MixSIH, a major reason for their speed was due to not having to compute confidence scores.” (2014.Kuleshov.ProbHap, page 3, last paragraph on right).
    2. “In fact, PROBHAP spends roughly two-thirds of its running time computing such scores. Nonetheless, it phases chromosome 22 in just under a minute; the total time for phasing a human genome was under 30 minutes.” (2014.Kuleshov.ProbHap, page 4, 1st paragraph on left).
14. 2015.Ahn.ParticleHap:
    1. “Running time (Time(s)) for each algorithm is evaluated along with the reconstruction rate (ReconRate).” (2015.Ahn.ParticleHap, page 8, 2nd paragraph on right).
    2. “As expected, the running time of ParticleHap increases with both the haplotype length and sequencing coverage.” (2015.Ahn.ParticleHap, page 8, last paragraph on right).
15. 2016.HapCOL:
    1. “Performances of the tools have been evaluated in terms of running time and peak memory usage, as reported by the Unix utility time.” (2016.HapCOL, page 5, last paragraph on right).
    2. “. In terms of performance, HAPCOL is slower than WHATSHAP on the single instance with coverage 15 and it has a similar running time of WHATSHAP on the two instances with coverage 20 (21 min, on average).” (2016.HapCOL, page 7, 2nd to last paragraph on right).
16. 2017.WhatsHap:
    1. “Hence, the runtime of our approach is polynomial (in fact, linear) in the number of SNPs. A linear-runtime solution for the wMEC addresses both that future sequencing technologies generate reads of tens of thousands of base pairs (bp), and that those will likely suffer from elevated sequencing error rates.” (2017.WhatsHap, page 3, last paragraph).
    2. “The runtimes of both WhatsHap and the DP approaches by Deng et al. (2013) are low for low coverages with a slight advantage for Deng et al. (2013).” (2017.WhatsHap, page 8, last paragraph).
    3. “WhatsHap is the first exact approach for the weighted MEC problem, which aims at statistically sound handling of sequencing errors, with runtime linear in the number of SNPs, which is essential for processing long reads.” (2017.WhatsHap, page 11, 2nd paragraph).
    4. “Since SNPs comprise roughly 5% of positions, and the runtime of our method is on the order of 10 min on average (for sufficient 15 · coverage), such a de novo haplotype could be generated in about 3 hours.” (2017.WhatsHap, page 11, last paragraph).
17. 2017.Edge.HapCUT2:
    1. “Further, it implements a number of optimizations to enable fast runtimes on diverse types of sequence data sets.” (2017.Edge.HapCUT2, page 2, 2nd paragraph on right).
    2. “Standard deviations of runtimes and error rates between replicates were small” (2017.Edge.HapCUT2, page 3, 1st paragraph on right).
    3. “While HapCUT2 has the same asymptotic behavior as HapCUT, it improves upon the memory usage and runtime significantly in practice.” (2017.Edge.HapCUTt2, page 4, 1st paragraph).
18. 2018.HapCHAT:
    1. “Both plots confirm that HapCHAT computes predictions that are at least as good as those of WhatsHap (and clearly better in terms of Hamming distance) with a comparable runtime.” (2018.HapCHAT, page 14, last paragraph on right).
    2. “We decided to include in the Tables the comparison of WhatsHap at both 20x and 15x max coverage, while 20x is the maximum coverage that we could test for WhatsHap – 15x is suggested by the authors as the default value for running WhatsHap and achieve the best trade-off between accuracy and running time.” (2018.HapCHAT, page 15, first paragraph on left).
19. 2018.Olyaee.AROHap:
    1. “The table (Table 7) reveals that running time of the proposed method is appropriate against the other approaches and in worst cases, it can solve the samples in less than 5 min.” (2018.Olyaee.AROHap, page 9, 1st paragraph on left).
    2. “ It should be noted that fast convergence speed of ARO and starting from initial promising solutions are the main reasons that improve execution time performance.” (2018.Olyaee.AROHap, page 8, 1st paragraph on right).
20. 2018.Tangherloni.GenHap: “For what concerns the running time, GenHap outperformed HapCol in all tests except in the case of #SNPs = 10000, as shown in Figure 5, being around 4× faster in reconstructing the haplotypes.” (2018.Tangherloni.GenHap, page 15, 1st paragraph).
21. 2019.Majidian:
    1. HapOPT: compares runtime (2019.Majidian,page 10, table 6).

Evaluation Using Error Rates:

1. 2005.Wang:
   1. “the error rate of fragments e: from 0.1 to 0.4” (2005.Wang, page 3, 2nd to last paragraph on right)
   2. “This indicates that the MEC model cannot reconstruct haplotypes with high accuracy when error rate of SNP fragments is high, even if an exact algorithm is employed.” (2005.Wang, page 4, 1st paragraph on left).
2. 2009.Wu:
   1. “Note that when error rate is no more than 0.2, NSOM can almost correctly reconstruct all original haplotypes.” (2009.Wu, page 8, 2nd to last paragraph).
   2. When the error rate of SNP fragments is large, the reconstruction rate by NSOM under different parameter settings is still identical for most of instances since the reconstruction rate is obtained by averaging over all instances. (2009.Wu, page 11).
3. 2012.GAHap:
   1. “This data set is presented using three parameters: (i) haplotype length l (100, 350 and 700), (ii) the coverage c (3, 5, 8 and 10), and (iii) the error rate e (10%, 20% and 30%).” (2012.GAHap, page 5, 1st paragraph on left).
   2. “We also notices that as the error rate of our experiments increases, the convergence difference between GAHap with small and large population sizes decreases. It implies that for data sets with high error rates, information is too noisy to conclude any concrete solution; even with large population sizes and a large number of population regenerations.” (2012.GAHap, page 5, last paragraph on left).
4. 2014.Mazrouee.FastHap:
   1. “Such variant matrices were then used to examine how performance of different algorithms (i.e. FastHap, Greedy, HapCut) changes as a result of changes in error rate, coverage and haplotype length.” (2014.Mazrouee.FastHap, page 2, last paragraph on right).
   2. “. We observe that the accuracy numbers are always larger than what one may expect owing to the error rate. For example, when the error rate is 20%, one may expect an absolute accuracy of 80%, but the measured accuracy is 85.7%. This can be interpreted as follows. As the error rate (i.e. number of flipped variant calls) increases, some variant calls may become consistent with a different haplotype of higher accuracy.” (2014.Mazrouee.FastHap, page 6, 2nd paragraph on right)
5. 2014.Aguiar.HapCompass\_Tumor:
   1. “Finally, while the error rate does affect haplotype assembly accuracy, as long as the error rate is less than 0.2%, the haplotype assemblies are similar in quality.” (2014.Aguiar.HapCompass\_Tumor, page 8, 1st paragraph)
   2. “The reads simulated from HapCompass include medium (200bp) and long (2000bp) read lengths with error rates of 2% and 5% respectively to model the higher error rates associated with long-read high-throughput sequence technologies.” (2014.Aguiar.HapCompass\_Tumor, page 8, last paragraph)
6. 2018.Olyaee.AROHap:
   1. “The dataset includes three parameters which are haplotype length l = 100, 350 and 700, the coverage c = 3, 5, 8 and 10, and the error rate e = 0, 10, 20 and 30%. For each combination of these parameters, there are 100 SNP matrices and each of them contains several fragments.” (2018.Olyaee.AROHap, page 6, 2nd paragraph on left)
7. 2018.Tangherloni.GenHap:” The performance was then evaluated by computing (i) the average haplotype error rate (HE), which represents the percentage of SNPs erroneously assigned with respect to the ground truth” (2018.Tangherloni.GenHap, page 14, 2nd to last paragraph).

**Assumptions:**

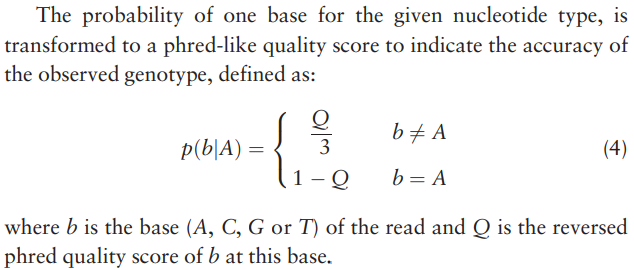
1. 2001.Lancia:
   1. “we assume that there are no fragment inclusions, i.e., denoting by and the first and last snip of a fragment implies .” (2001.Lancia, page 5, 2nd paragraph).
   2. “Without loss of generality, we assume that a variable appears at most once in a clause.” (2001.Lancia, page 9, 2nd to last paragraph.)
2. 2004.Fast Hare:
   1. “Without loss of generality we can assume that the two haplotypes are strings of A’s and B’s of the same length, rather than strings over the four letter alphabet A, C, G, T [5].” (2004.Panconesi.FastHare, page 1, last paragraph).
   2. “In practice, the coverage can be assumed to be around 10 (with roughly half of the fragments coming from each strand).” (2004.Panconesi.FastHare, page 2, last paragraph).
3. 2005.Wang: “This model assumes that all the fragments come from one organism but there are sequencing errors to be corrected (this case is very practical).” (2005.Wang, page 1, last paragraph on right).
4. 2005.Zhao:
   1. “. In this model, it is assumed that all SNP fragments come from the same region of a pair of chromosomes but there are sequencing error letters in these fragments to be corrected by flips.” (2005.Zhao, page 2, 2nd paragraph on left).
   2. “In this paper, without loss of generality, it is assumed that the confidence level (weight) of a letter is between 0 and 1.” (2005.Zhao, page 2, last paragraph on left).
5. 2007.BEAGLE: “We assume Hardy-Weinberg equilibrium, so that the diploid initial and transition probabilities are the product of the corresponding haploid probabilities…” (2007.BEAGLE, page 4, 2nd to last paragraph on left).
6. 2007.HuRef: “We assumed that chromosomes containing each of the two alleles are equally likely to be sampled and that allele loci are independent.” (2007.Levy.HuRef, page 28, 3rd paragraph on left).
7. 2008.SpeedHap:
   1. “we assume that the locus is a homozygote and the data in the smaller set are reading errors.” (2008.Genovese.SpeedHap, page 5, last paragraph on left).
   2. “We can now safely assume that if the ratio between the cardinalities of the sets of an unused group is far enough from the mean, the locus represented from that group is a homozygote and the elements in the smaller set of that group are all errors and can be corrected updating M accordingly.” (2008.Genovese.SpeedHap, page 6, last paragraph on left).
   3. “One exception combining both approaches is the algorithm in [26] that uses a Markov chain model for single individual haplotype reconstruction where it is assumed that both the individual genotype and the shotgun fragments of the two haplotype strings are available.” (2008.Genovese.SpeedHap, page 9, last paragraph on left).
   4. “Moreover, implicitly, it is assumed that all reads are roughly aligned, which is a very strong assumption within the current shotgun sequencing technology.” (2008.Genovese.SpeedHap, page 10, 2nd to last paragraph on right).
8. 2008.Kang:
   1. “all SNP fragments are from one organism, and they involve some sequencing errors.” (2008.Kang, page 2, 1st paragraph)
   2. “The WMLF model has another assumption that for each SNP site, there is a weight for its flipping, which represents the confidence level of the sequencer machine’s reading.” (2008.Kang, page 2, 1st paragraph).
   3. The confidence level (weight) of a value is assumed to be between 0 and 1. (2008.Kang, page 3, 1st paragraph)
9. 2008.Xie.P\_WMLF\_GS: Alignment (2008.Xie.P\_WMLF\_GS, page 2, 2nd paragraph on left).
10. 2008.Bansal.HASH:
    1. “We assume that a list of genetic variants such as SNPs, short insertions/deletions, etc., is available.” (2008.Bansal.HASH, page 3, 2nd to last paragraph on left).
    2. “If information about the sequencing quality values is not available or for performing simulations, we assume a uniform error probability for all variants.” ( 2008.Bansal.HASH, page 3, 2nd paragraph on right).
    3. “We assume that the variant calls for a fragment Xi are independent of each other.” (2008.Bansal.HASH, page 3, last paragraph on right).
    4. “In our approach, we assume that a list of variants generated from the sequenced reads is available.” (2008.Bansal.HASH, page 10, last paragraph on left).
11. 2008.Chen.HapRec:?
12. 2009.Howie.IMPUTE2:
    1. “if we assume that both haplotypes were sampled from a population that conforms to Hardy-Weinberg Equilibrium" (2009.Howie.IMPUTE2, page 4 , 2nd to last paragraph on right).
    2. “The underlying theory assumes that the haplotypes in question were sampled randomly from a population, which is clearly not the case when we select k haplotypes in the manner described above.” (2009.Howie.IMPUTE2, page 5, 2nd to last paragraph on right).
13. 2009.Wu: ?
14. 2010.Duitama.ReFHap:
    1. “ReFHap assumes that all input loci are heterozygous.” (2010.Duitama.ReFHap, page 3, 1st paragraph on left).
    2. “ that real fragments are equally likely to come from either of the two haplotypes.” (2010.Duitama.ReFHap, page 5, 1st paragraph on left).
    3. “Assuming absence of genotyping errors” (2010.Duitama.ReFHap, page 5, 1st paragraph on right).
15. 2010.Kang.HapAssembler: ?
16. 2010.Li.MaCH: ?
17. 2010.MaxSAT:
    1. “We also assume all the reads have already been correctly aligned to the reference genome by some mapper, which may not be true since the mapper may introduce mapping errors and the reads may come from repeat-rich regions.” (2010.MaxSAT, page 2, last paragraph on right).
    2. “we assume all reads are of length k” (k being the length of the longest read) (2010.MaxSAT, page 3, 3rd paragraph from bottom right).
18. 2010.Xu: ?
19. 2011.HALLDORSSON:
    1. "distance between SNPs and sequence reads follow a Poisson distribution (λ = 400 and λ = 10000 respectively)” (2011.HALLDORSSON , page 4, 1st paragraph)
    2. “We assume the maximum distance between fragments is 2 SNPs.” (2011.HALLDORSSON , page 9, figure 2).
20. 2011.Mousavi.HapSAT:
    1. “We assume that all reads are of the same length n (since they have already been aligned).” (2011.Mousavi.HapSAT, page 2, 2nd paragraph on left).
21. 2012.Aguiar.HapCompass:
    1. “Because DNA sequence reads originate from a haploid chromosome, the alleles spanned by a read are assumed to exist on the same haplotype.” (2012.Aguiar.HapCompass, page 2, 3rd paragraph).
22. 2012.Bayzid.HMEC: ?
23. 2012.Dewal.HATS:
    1. “We assume that only one of the chromosomes in a homologous pair undergoes amplification along an amplicon, as the majority of amplifications were observed to be monoallelic versus biallelic in earlier work.” (2012.Dewal.HATS, page 2, last paragraph on left).
    2. “Only one chromosome is assumed to be amplified along a homologous region.” (2012.Dewal.HATS, page 6, last paragraph on right).
    3. “we assume one of the haplotypes is (A)mplified while the other is (U)namplified” (2012.Dewal.HATS, page 8, 3rd paragraph on right).
    4. “The concept behind this is that, at a given locus, a germline haplotype of a tumor sample is assumed to be identical to one of the training haplotypes, randomly switching the training sample that is locally identical to the tumor haplotype along the genome.” (2012.Dewal.HATS, page 10, 2nd paragraph on left).
    5. “assuming an underlying Poisson distribution for the read counts” (2012.Dewal.HATS, page 10, 1st paragraph on right).
24. 2012.Efros.PPHS:
    1. “The basic assumption of our algorithm is that within a short region, the history of the genetic variants (SNPs or deletions) follows the perfect phylogeny model,” (2012.Efros.PPHS, page 2, 2nd paragraph from bottom left).
    2. “. We will assume each read is a copy of l bases extracted from the genome starting at a random position.” (2012.Efros.PPHS, page 2, 1st paragraph on right).
    3. “Under simplifying assumptions, we can assume that the error rate does not depend on the genomics position.” (2012.Efros.PPHS, page 2, 1st paragraph on right).
    4. “each read is assumed to be generated by randomly picking a position in the genome, randomly picking one of the copies of the chromosome in that position, and adding noise using the parameter ε in each position of the read independently.” (2012.Efros.PPHS, page 2, 2nd paragraph on right).
    5. “we assume that the alleles of each SNP are represented by the {0, 1} notation, where 0 is the more common allele.” (2012.Efros.PPHS, page 2, last paragraph on right).
    6. “We thus assume that the root of the tree is the haplotype with all 0 values.” (2012.Efros.PPHS, page 3, 1st paragraph on left).
25. 2012.Wang.GAHap:
    1. “Assume a set of shotgun fragments are obtained from a chromosome and are aligned against the reference genome sequence.” (2012.Wang.GAHap, page 2, 2nd paragraph on right).
    2. “assume that this information is stored in a m n matrix X, where m and n represent the total number of fragments and the length of the reference genome sequence, respectively.” (2012.Wang.GAHap, page 2, 2nd paragraph on right).
    3. “we assume that soft-removed nucleotides belong to sequence errors and they should belong to the major allele” (2012.Wang.GAHap, page 4, 2nd paragraph on right).
    4. SNPs are all assumed biallelic” (2012.Wang.GAHap, page 7, 2nd paragraoh on right).
26. 2012.Xie.HBOP:
    1. “we assume that all SNP loci are heterozygous and every fragment covers at least two heterozygous SNP loci” (2012.Xie.HBOP, page 6, 2nd paragraph from bottom left).
27. 2012.Zhi.HapSeq:
    1. “we assume the makers are biallelic SNPs, with alleles labeled as A and B.” (2012.Zhi.HapSeq, page 2, 1st paragraph on right).
    2. “For paired end settings, we assumed an ‘insert fragment’ length of 200 bp.” (2012.Zhi.HapSeq, page 4, 2nd paragraph on right).
28. 2012.Delaneau.SHAPEIT: ?
29. 2012.Williams.HAPIUR:
    1. “Our implementation currently assumes that markers are biallelic and deduces the complementary haplotype segment by inverting the allele values at all heterozygous sites in the haplotype segment for the known haploid state.” (2012.Williams.HAPIUR, page 5, 2nd paragraph on right).
    2. “we estimated their runtimes for all chromosomes from these values by assuming that their runtimes scale linearly in the number of markers; this scaling matches the scaling we observed on the four chromosomes.” (2012.Williams.HAPIUR, page 8, last paragraph on right).
30. 2013.Chen.Zhi: ?
31. 2013.Deng:
    1. “the authors assumed that the two constructed haplotypes are complementary with each other, i.e. there are only 2 choices at a SNP site in the reconstructed haplotypes.” (2013.Deng, page 2, last paragraph on left).
    2. “We assume that all the fragments have been pre-aligned to a reference DNA sequence” (2013.Deng, page 2, 1st paragraph on right).
    3. “We assume that a column with more than two distinct nucleotides in M (fragment matrix) must contain errors.” ” (2013.Deng, page 2, 2nd paragraph on right).
32. 2013.He.Hap\_seqX:
    1. “The gap size is assumed to follow a normal distribution of mean 1000 bp and the standard deviation of 100 bp.” (2013.He.Hap\_seqX , page 4, 3rd paragraph on left).
    2. “We also assume the reads are uniformly generated.” (2013.He.Hap\_seqX , page 4, 3rd paragraph on left).
33. 2013.Matsumoto.MixSIH:
    1. “We assume this binary representation throughout the paper.” (2013.Matsumoto.MixSIH , page 3, 2nd paragraph from bottom right).
    2. “. We further assume the mixture probabilities are equal” (2013.Matsumoto.MixSIH, page 4, last paragraph on left).
34. 2013.Yang.HARSH: ?
35. 2013.Zhang.DBM:
    1. “We assume that each state has its own allele distribution per SNP, and the alleles are independently generated from the states at each SNP.” (2013.Zhang.DBM, page 2, 2nd paragraph on bottom right).
36. 2013.Zhang.HapSeq2:
    1. “with the assumption of fixed genotypes for that individual, we perform the MH sampling to update the haplotype pair of that individual.” (2013.Zhang.HapSeq2, page 2, 2nd paragraph on right).
    2. “Throughout, we assume biallelic sites, with alleles labeled as 0 (the reference allele) and 1 (the alternative alleles). (2013.Zhang.HapSeq2, page 2, last paragraph on right).
    3. “we assume that the genotypes are fixed, as they are already determined from the HMM procedure so only haplotypes of that individual are updated according to the sequencing reads and haplotypes from the reference haplotypes (internal and/or external reference haplotypes).” 2013.Zhang.HapSeq2, page 4, 1st paragraph on left).
37. 2013.Delaneau.SHAPEIT2: ?
38. 2014.Aguiar.HapCompass\_Tumor:
    1. “We assume that each sequence read is sampled from a single haploid fragment generated from one of the k haplotypes” (2014.Aguiar.HapCompass\_Tumor, page 3, last paragraph).
39. 2014.Berger.HapTree:
    1. “all heterozygous loci are biallelic” (2014.Berger.HapTree, page 2, 2nd paragraph on right).
    2. “We further assume that for each SNP locus s, the genotype of s is known and is defined to be the number of chromosomes carrying the alternative allele” (2014.Berger.HapTree, page 3, last paragraph on left).
    3. “our method assumes uniform error rates with respect to the SNP position” (2014.Berger.HapTree, page 3, 1st paragraph on right).
    4. “We assume the sequencing errors are independent of each otherWe assume the sequencing errors are independent of each other” ((2014.Berger.HapTree, page 4, 2nd paragraph on left).
40. 2014.Chen.HGHap:?
41. 2014.Kuleshov.ProbHap?
42. 2014.Matsumoto.CSP:
    1. “Our method is based on the assumption that chimeric fragments are derived artificially and differ from the biological conserved haplotypes in the population” (2014.Matsumoto.CSP, page 3, 2nd paragraph on left).
    2. “CFs correspond to an artificially recombinant haplotype and differ from the biological haplotypes in the population.” (2014.Matsumoto.CSP, page 10, last paragraph on right).
    3. “t left and right parts of the fragment are derived from different haplotypes in a haplotype pair” (2014.Matsumoto.CSP, page 10, 1st paragraph on left).
43. 2014.Mazrouee.FastHap:
    1. "Input to the haplotype assembly algorithm is a 2D array containing only heterozygous sites of the aligned fragments" (2014.Mazrouee.FastHap, page 2, 1st paragraph right).
44. 2014.Pan.WinHap2:
    1. “any SNP mutation happened just once in the whole evolutionary history.” (2014.Pan.WinHap2, page 2, 1st paragraph on left).
    2. “we assume that the length of the scalable sliding window ranges from l\_min to l\_max” (2014.Pan.WinHap2, page 8, 1st paragraph on left).
45. 2015.Ahn.ParticleHap:
    1. “the SNP sites are assumed to be bi-allelic (i.e., each SNP site contains one of only two possible nucleotides), the alleles are labelled as 0 and 1 and the haplotypes are represented by binary sequences.” (2015.Ahn.ParticleHap, page 2, 2nd paragraph on left).
    2. Composition probabilities are mutually independent and constant across haplotypes. (2015.Ahn.ParticleHap, page 3, 1st paragraph on right).
    3. “We assume that each read is generated from one of the two haplotypes” (2015.Ahn.ParticleHap, page 3, 1st paragraph on right).
    4. “We assume that the bases in the sequence are equally likely, i.e., the composition probabilities of the 4 nucleotides are equal, 0.25.” (2015.Ahn.ParticleHap, page 7, 1st paragraph on left).
    5. “we assume the error rate e = 0.01, which is consistent with the typical sequencing accuracy of the 454 platform.” (2015.Ahn.ParticleHap, page 7, 1st paragraph on left).
    6. “we assumed equal prior probabilities of all genotypes.” (2015.Ahn.ParticleHap, page 8, last paragraph on right).
46. 2016.HapCol (Future Generation tech):
    1. “HAPCOL is able to work without the all-heterozygous assumption.” (2016.HapCol, page 2, 2nd paragraph on right).
    2. “we implicitly assume that there exists a dummy empty column M0 in position 0 of the input M.” (2016.HapCol, page 4, 1st paragraph on right).
    3. “sequencing errors are uniformly distributed with a substitution error rate (given as input), an assumption which reflects the characteristics of future-generation sequencing technologies.” (2016.HapCol, page 5, 1st paragraph on right).
47. 2017.Edge.HapCUT2:
    1. “Reads are assumed to be sorted by starting position.” (2017.Edge.HapCUT2, page 3, table 1).
    2. “The heterozygous variants are assumed to have been identified separately from WGS data for the same individual.” (2017.Edge.HapCUT2, page 9, 2nd paragraph on left).
    3. “independence of fragments” (2017.Edge.HapCUT2, page 9, 1st paragraph on right).
    4. “We assume that the h-trans error probability, represented as τ(I), is the same for all reads with the same insert length I.” (2017.Edge.HapCUT2, page 10, last paragraph on left).
    5. “HapCUT2 assumes that the heterozygous sites are known in advance” (2017.Edge.HapCut2, 10, 1st paragraph on right).
48. 2017.WhatsHap:
    1. “all-heterozygous assumption, where all columns correspond to heterozygous sites” (2017.WhatsHap, page 4, 3rd paragraph).
    2. “WhatsHap is an exact dynamic programming approach that solves wMEC instances in linear time if we assume bounded coverage.” (2017.WhatsHap, page 5, 1st paragraph).
    3. “t. We ran each method under the assumption that the haplotypes contain only heterozygous positions (all-het, for short).” (2017.WhatsHap, page 8, 3rd paragraph).
49. 2018.Na.PEATH:
    1. “The low concordant sequence reads (concordance ratio < 0.3) were assumed to be complementary sequence reads since the true answer was represented as only one haplotype among the two reads.” (2018.Na.PEATH, page 3, 4th paragraph on right).
    2. “all input variables are independent” (2018.Na.PEATH, page 6, 2nd paragraph on right).
50. 2018.Olyaee.AROHap: ?
51. 2018.HapCHAT:
    1. “we assume that we have no error in the shared part and exactly one error on the other sites.” (2018.HapCHAT, page 4, 1st paragraph on right).
    2. “we assume that there is exactly one error in the sites with same value and at most an error in the sites with different values.” (2018.HapCHAT, page 4, 1st paragraph on right).
52. 2018.Tangherloni.GenHap: “we did not assume the all-heterozygosity of the phased positions [19]. Under this assumption, every column corresponds to heterozygous sites, implying that h1 must be the complement of h2.” (2018.Tangherloni.GenHap, page 7, 2nd to last paragraph).
53. 2019.Majidian:
    1. “ Thus, it is usual to assume that the majority of SNPs are bi-allelic, meaning that each SNP can be chosen from just two of the four possible nucleotides, i.e., A, T, C, and G [1].” (2019.Majidian, page 1, 1st paragraph).

**Single Cell HAAlgorithms/Papers:**

Sequencing Error:

1. SC.2018.Satas: “We introduced sequencing error into the spiked-in alleles based on the Phred quality scores of individual reads. Further details on the simulation can be found in Supplementary Material S1” (2018.Satas, page 4, 3rd paragraph from bottom left).
2. SC.2017.Chu:
   1. “There is not sufficient information to determine whether that the variants are genuine de novo SNVs or are due to errors introduced by the SISSOR procedures. This bounds the overall sequencing error rate of the SISSOR technology below 1 × 10−8 (four possible errors in 351 Mb).” (SC.2017.Chu, page 4, 2nd paragraph on right).
   2. “The rates of single-chamber MDA-based sequencing error (10−5 ) and single-cell de novo mutation (10−7 ) were calculated for SISSOR.” (SC.2017.Chu, page 4, fig. 3)

Phred Score:

1. SC.2018.Guo:
   1. (2018.Gou, page 3, 1st paragraph on left).

Sequencing Coverage:

1. SC.2018.Guo: “ In particular, when the read coverage is more than 30, both false discovery rate and false negative rate are 0.” (2018.Guo, page 2, 2nd paragraph on right).

Read Length:

1. SC.2018.Guo: “Recently, single-molecule sequencing technologies can provide SMS reads (average length more than 10K bps) and cover about 90% of positions over chromosomes.” (2018.Guo, page 1, right paragraph).

Alignment Info.:

1. SC.2018.Guo:
   1. “We align the reads to GRCh37 (also known as hg19 reference) (Church et al., 2011), using the read mapping tool BWA\_MEM” (2018.Guo, page 4, 2nd to last paragraph on left).
   2. “BWA\_MEM is an efficient read mapping tool for aligning query sequences against a large reference genome, from http://github.com/ lh3/bwa. This alignment method automatically chooses the local alignment or the end-to-end alignment. It is robust to deal with sequencing errors and applicable to a wide range of sequence lengths from 70 bps to 1M bps.” (2018.Guo, page 4, last paragraph on left).
2. SC.2017.Chu: “Sequencing reads from the individual chambers were identified using the barcodes and mapped to the reference human genome hs37d5 (GRCh37/b37 + decoy sequences) using the default setting of BWA-MEM with Burrows– Wheeler Aligner (BWA)” (SC.2017.Chu, page 2, 1st paragraph on right).

Input:

1. SC.2018.Satas: “We use pairs of SNPs exhibiting high rates of concurrent dropout to define amplification fragments that we input into an existing haplotype assembler.” (2018.Satas, page 3, 1st paragraph on left).

Evaluation using Switch Error:

1. SC.2018.Satas: “Switch error is the proportion of phase connections between adjacent SNPs that are incorrect.” (2018.Satas, page 4, 2nd paragraph on right).
2. SC.2018.Satas: “As the phasing score threshold c decreases, the block lengths increase by several orders of magnitude with only relatively small corresponding increases in switch error rate With phasing score threshold c =2.25, we obtain a block length of N50 = 10.2 kb, with a switch error rate of 0.02. At lower values of phasing score threshold, we see larger increases in error, which corresponds to the observed decrease fragment accuracy at the same values” .” (2018.Satas, page 4, 2nd paragraph on right).
3. SC.2018.Guo:
   1. “The overall switch errors for our method are 7.26 and 5.21 with average 3378 and 5736 SNP sites per block on NA12878 and NA24385, respectively.” (2018.Guo, page 2, 2nd paragraph on right).
   2. “We use the simulated NA12878 date with 25 coverage. Here, we have 901 977 SNP sites on chromosome 1. Among them, there are 1931 switch errors in total. Thus, the switch error rate is 0.0021. We can see that when the coverage is large enough, these methods have comparable switch error rate. It is worth to point out that our results are only based on long reads.” (2018.Guo, page 6, last paragraph on right).
4. SC.2017.Chu:
   1. “First, a switch error was defined as two or more SNPs in a row flipped.” (SC.2017.Chu, page 3, 3rd paragraph on right).
   2. “Splitting longer fragments with detectable switch errors and poor variant calls from mixed homologous reads at the unique genomic position reduced the overall haplotyping errors. (SC.2017.Chu, page 3, 2nd paragraph from bottom right).

Evaluation Using MEC Score:

1. SC.2018.Guo:
   1. Nanopore Data—NA12878: “The MEC score is 5 562 358 before error correction and 341 795 after error correction, over total 20 316 257 bases. The MEC rate is 27.3% before error correction and 1.6% after error correction. For sites with coverage more than 15, the MEC score is 113 252 before error correction 6826 after error correction, over total 481 079 bases.” (2018.Guo, page 6, 2nd paragraph on left).
   2. SMRT Data—NA12878: “The MEC score is 3 151 975 before error correction and 179 686 after error correction, over total 12 891 712 bases. The MEC rate is 24.5% before error correction and 1.4% after error correction. For sites with coverage more than 15, the MEC score is 100 176 before error correction 6475 after error correction, over total 407 250 bases. (2018.Guo, page 6, 3rd paragraph on left).
   3. SMRT Data—NA24385: “The MEC score is 4 923 765 before error correction and 258 956 after error correction, over total 27 690 779 bases. The MEC rate is 17.8% before error correction and 0.9% after error correction. For sites with coverage more than 15, the MEC score is 90 812 before error correction 5139 after error correction, over total 560 973 bases.” (2018.Guo, page 6, 1st paragraph on right).

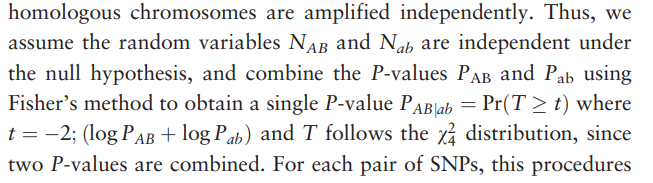
Evaluation using N50 Length:

1. SC.2018.Satas:
   1. “The N50 is the length of a haplotype block such that half of all phased variants are in a block at least as long.” (2018.Satas, page 4, 2nd paragraph on right).
   2. “Using only read fragments Fr, the haplotype assemblies have a median block length of N50 ¼ 41 bp, which, as expected, is shorter than the length of a single exon.” (2018.Satas, page 5, 1st paragraph on left).
   3. “For example, when the phasing score threshold c ¼ 2.75, we obtain a median block length N50 ¼ 9.3 kb, with a corresponding switch error rate of 0.04. This indicates that we are able to phase across multiple exons. Indeed this haplotype block length is of the same order of magnitude of the typical gene length.” (2018.Satas, page 5, 1st paragraph on left).
2. SC.2017.Chu: “If a higher switch and mismatch error rate (1.6%) could be tolerated in an application, a large N50 haplotype length (>15 Mb) was directly produced by HMM-derived SISSOR fragments.” (SC.2017.Chu, page 3, 3rd paragraph on right).

Evaluation using Running Time:

1. SC.2018.Guo: “The running time of our method is related to the size of a block. With the increase of block size, the running time increases linearly. With average coverage 25, the linear growth rate is about 0.15 min/Kbps on NA12878. With average coverage 69, the linear growth rate is about 0.4 min/Kbps on NA24385.” (2018.Guo, page 6, 2nd paragraph on left).

Assumptions:

1. SC.2018.Satas:
   1. “We obtain DNA-sequencing data from n single cells from the same individual and assume that these cells share m heterozygous SNPs.” (2018.Satas, page 3 last paragraph on left).
   2. (2018.satas, page 3, last paragraph on right).
2. SC.2017.Chu: “We expect sequencing accuracy equivalent to the strand–strand consensus in the same cell, assuming error rate was identical in all SISSOR libraries.” (CS.2017.Chu, page 5, 3rd paragraph on left).

**Polyploid HA Algorithms/Papers:**

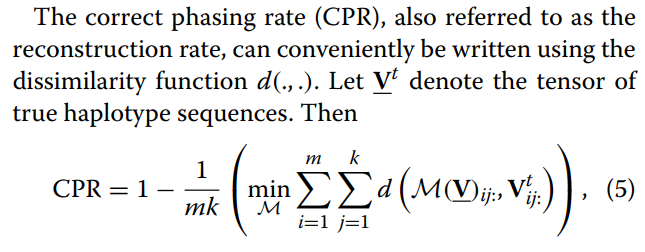
Assumptions:

1. Poly.2018.Hashemi.AltHap: “It should be noted that while state-of-the-art haplotype assembly methods for polyploids assume haplotypes may only have biallelic sites, AltHap is capable of reconstructing polyallelic haplotypes which are common in many plants and some animals, are of particular importance for applications such as crop cultivation [34], and may help in reconstruction of viral quasispecies [35].” (Poly.Hashemi.AltHap, page 3, last paragraph on left).

Input:

1. Poly.2018.Hashemi.AltHap: “let R be an n×m SNP fragment matrix where n denotes the number of sequencing reads and m is the length of haplotype sequences.” (Poly.2018.Hashemi.AltHap, page 3, 2nd paragraph on right).

Reconstruction Rate:

1. Poly.2018.Hashemi.AltHap:
   1. “correct phasing rate (CPR) – also referred to as reconstruction rate” (Poly.2018.Hashemi.AltHap, page 3, last paragraph on left).
   2.  Poly.2018.Hashemi.AltHap, page 4, last paragraph on left).

Evaluations using Reconstruction Rate:

1. Poly.2018.Hashemi.AltHap:
   1. “CPR is related to the generalization error that is calculated on unobserved SNPs positions, i.e., the test data points.” (Poly.2018.Hashemi.AltHap, page 9, last paragraph on right).
   2. “AltHap achieves the best CPR for most of the chromosomes (thirteen out of 22).” (Poly.2018.Hashemi.AltHap, page 11, right paragraph).

Sequencing Errors:

1. Poly.2015.Das.SDhaP: “Then G = (V, E, W) is a correlation graph where the edges connecting vertices associated with similar reads (i.e., the reads that belong to the same haplotype) should have positive weights, while the edges connecting vertices associated with dissimilar reads should have negative weights. In the absence of sequencing errors, that is indeed the case and thus separating the reads into K different clusters corresponding to K distinct haplotypes is trivial.” (Poly.2015.Das.SDhaP, page 4, 2nd paragraph on left).

Evaluation using Sequencing errors:

1. Poly.2015.Das.SDhaP : “To simulate a sequencing process capable of facilitating reconstruction of long haplotype blocks, we randomly generate pairedend reads of length 500 with average insert length of 10,000 bp and standard deviation of 10%; sequencing errors are inserted using realistic error profiles [27] and genotyping is performed using a Bayesian approach [28].” (Poly.2015.Das.SDhaP, page 9, 1st paragraph on left).

Evaluation using Sequencing Coverage:

1. Poly.2015.Das.SDhaP:
   1. “We first tested SDhaP on the HuRef dataset [6] which contains single and mated reads sequenced using a dideoxy Sanger sequencing technology with an average coverage of ≈ 8X.” (Poly.2015.Das.SDhaP, page 8, 2nd paragraph on right).
   2. “Table 3 compares the MEC, SWER and running times of SDhaP with those of HapCUT, HapTree and RefHap. We make these comparisons for haplotype block lengths of 103 and 104 at coverages of 10, 20 and 30.” (Poly.2015.Das.SDhaP, page 9, 1st paragraph on right).
   3. “The algorithm is tested for coverages 5KX, 10KX and 5K2X, where K denotes the ploidy. From the simulation results, it appears that the required coverage increases approximately with the square of the ploidy.” (Poly.2015.Das.SDhaP, page 12, paragraph on left).
   4. “For example, the coverage needed to achieve SWER below 1% for triploids (K = 3) is approximately 45X, for tetraploids (K = 4) the required coverage is around 80X, and for hexaploids (K = 6) the algorithm requires coverage of ≈ 180X.” (Poly.2015.Das.SDhaP, page 12 right paragraph).
2. Poly.2018.Hashemi.AltHap:
   1. “As evident from this table, AltHap outperforms other algorithms for nearly all the combinations of data error rates and sequencing coverage and is also much faster than SCGD, ILP, BP and HapTree while being slightly slower than H-PoP.” (Poly.2018.Hashemi.AltHap, page 11, last paragraph on left).

Evalauation using Read Length:

1. Poly.2018.Hashemi.AltHap:
   1. “To investigate how the performance and complexity of AltHap vary with coverage and read length, in Table 6 we report its CPR, MEC, and runtimes obtained by simulating assembly of biallelic triploid haplotypes using paired end reads of length 2 × 250, 2 × 300, and 2 × 500 and coverage 10, 20 and 30 (block length is set to m = 1000 and data error rate is pe = 0.002).” (Poly.2018.Hashemi.AltHap, page 12, 2nd paragraph on right).

Evaluation using Switch Error:

1. Poly.2015.Das.SDhaP:
   1. “SDhaP’s MEC score is lower and its SWER is nearly half that of the competing schemes. The running times of SDhaP are at least 10 times lower for haplotype block lengths of 104 (although for block lengths of 103 the difference in running times is not as appreciable). Overall, SDhaP clearly outperforms the other considered methods.” (Poly.2015.Das.SDhaP, page 9, 1st paragraph on right).
   2. “For block lengths of 103 and error rates 1%, CPLEX achieves the best MEC scores and SWER but its runtimes are significantly slower than those of SDhaP.” (Poly.2015.Das.SDhaP, page 10, 1st paragraph on right).
   3. “The coverages used were 10X, 20X and 30X. As can be seen in the figure, when the error rate is 1%, the SWER of SDhaP for coverages greater than 20X is very small for all block lengths. When the error rate is 5%, we observe that higher coverage is needed to ensure low SWER.” (Poly.2015.Das.SDhaP, page 11, left paragraph).

Evaluation using Running Time:

1. Poly.2015.Das.SDhaP:
   1. “The running times of SDhaP are at least 10 times lower for haplotype block lengths of 104 (although for block lengths of 103 the difference in running times is not as appreciable). Overall, SDhaP clearly outperforms the other considered methods.” (Poly.2015.Das.SDhaP, page 9, 1st paragraph on right).
2. Poly.2018.Hashemi.AltHap:
   1. “The results of tests conducted on simulated biallelic tetraploid genomes are summarized in Table 5, where we observe that AltHap outperforms the competing schemes in terms of both accuracy and running time.” (Poly.2018.Hashemi.AltHap, page 12, last paragraph on right).

Evaluation using MEC score:

1. Poly.2015.Das.SDhaP:
   1. “As can be seen from the table, the MEC scores obtained with SDhaP are significantly better than those of the competing algorithms except for CPLEX.” (Poly.2015.Das.SDhaP, page 8, 2nd paragraph on right). -HuRef data
   2. “As can be seen from the table, the MEC scores of SDhaP are better than those of HapCUT, HapTree and RefHap” (Poly.2015.Das.SDhaP, page 8, 3rd paragraph on right). -Fosmid data
   3. “SDhaP’s MEC score is lower and its SWER is nearly half that of the competing schemes.” (Poly.2015.Das.SDhaP, page 9, 1st paragraph on right). -simulated data

Genotype data algorithms:

1. SHAPEIT:
   1. “**SHAPEIT** is primarily a tool for inferring haplotypes from SNP genotypes. It takes as input a set of genotypes and a genetic map, and produces as output, either a single set of estimated haplotypes, or a haplotype graph that encapsulates the uncertainty about the underlying haplotypes.” (From Manual: <http://mathgen.stats.ox.ac.uk/genetics_software/shapeit/shapeit.html#gettingstarted>)
2. SHAPEIT2:
3. MaCH:

Review Papers:

1. 2001.Lippert
2. 2016.Rhee.Survey
3. 2019.Tangherloni.Review
4. 2010.Schwartz.Review

Citations:

Ewing B; [*Hillier L*](https://en.wikipedia.org/wiki/Ladeana_Hillier); Wendl MC; [*Green P.*](https://en.wikipedia.org/wiki/Philip_Palmer_Green) (1998). "Base-calling of automated sequencer traces using phred. I. Accuracy assessment". Genome Research. **8** (3): 175–185. [*doi*](https://en.wikipedia.org/wiki/Digital_object_identifier):[*10.1101/gr.8.3.175*](https://doi.org/10.1101%2Fgr.8.3.175)

Crow JF (August 1997). ["The high spontaneous mutation rate: is it a health risk?"](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC33757). *Proceedings of the National Academy of Sciences of the United States of America*. **94** (16): 8380–6. [Bibcode](https://en.wikipedia.org/wiki/Bibcode):[1997PNAS...94.8380C](http://adsabs.harvard.edu/abs/1997PNAS...94.8380C)