International Journal of Electrical and Computer Engineering (IJECE)
Vol. 12, No. 3, June 2022, pp. 2856~2866
ISSN: 2088-8708, DOI: 10.11591/ijece.v12i3.pp2856-2866

HapPart: partitioning algorithm for multiple haplotyping from haplotype conflict graph

Abu-Bakar Muhammad Abdullah, Md. Monowar Hossain, Pintu Chandra Shill
Department of Computer Science and Engineering, Khulna University of Engineering and Technology, Khulna, Bangladesh

Article Info

Article history:

Received Jun 10, 2021
Revised Jan 28, 2022
Accepted Feb 8, 2022

Keywords:

Conflict graph
DNA sequence
Haplotype
Minimum error correction
Polyploidy

ABSTRACT

Each chromosome in the human genome has two copies. The haplotype assembly challenge entails reconstructing two haplotypes (chromosomes) using aligned fragments genomic sequence. Plants viz. wheat, paddy and banana have more than two chromosomes. Multiple haplotype reconstruction has been a major research topic. For reconstructing multiple haplotypes for a polyploid organism, several approaches have been designed. The researchers are still fascinated to the computational challenge. This article introduces a partitioning algorithm, HapPart for dividing the fragments into *k*-groups focusing on reducing the computational time. HapPart uses minimum error correction curve to determine the value of *k* at which the growth of gain measures for two consecutive values of *k*-multiplied by its diversity is maximum. Haplotype conflict graph is used for constructing all possible number of groups. The dissimilarity between two haplotypes represents the distance between two nodes in graph. For merging two nodes with the minimum distance between them this algorithm ensures minimum error among fragments in same group. Experimental results on real and simulated data show that HapPart can partition fragments efficiently and with less computational time.

Corresponding Author:

Md. Monowar Hossain
Department of Computer Science and Engineering, Khulna University of Engineering and Technology
Khulna 9203, Bangladesh
Email: murad0904045@gmail.com

1. INTRODUCTION

Domestication is crucial in agriculture because it meets up the human interest to a great extent. Its major goal is to exert significant control over an organism's reproduction so that newer organisms can survive in difficult environments. It is gaining popularity on a daily basis since it ensures a consistent supply of resources. Scientists are quite interested in coming up with innovative ideas in this field. Several studies based on animal and plant domestication are existed. This paper focuses on plant domestication. Majority of significant plants that we need in our daily life possess polyploid cells which have more than one pair of each chromosome. Polyploidy can be achieved through plant genetic material multiplication or hybridization. Study on multiple haplotyping of a polyploid genome becomes necessary for creating new organisms.

A haplotype is a collection of deoxyribonucleic acid (DNA) variants inherited from a single parent. Polyploid organisms have several haplotypes, whereas diploid species have two. It's possible that a DNA sequence can contain insertion of wrong alleles or deletion of actual alleles. The haplotype assembly challenge is concerned with reassembling haplotypes from a set of DNA sequence that may contain a significant number of errors. Determination of haplotypes is more challenging than determination of genotypes from DNA reads of a polyploid genome. There have been a lot of research efforts on this computational problem to increase the accuracy and reduce the computational cost so that it can be used

widely on large-scale datasets. To reconstruct haplotypes, Wang *et al*. [1] suggested a clustering algorithm which can divide all fragments from the set of DNA sequence into two disjoint subsets solving the minimum error correction (MEC) model and each subset constructs the haplotype. Kargar *et al*. [2] proposed a model where the most observed allele is inserted into two constructed haplotypes and the corresponding column is deleted from the single nucleotide polymorphisms (SNP) matrix. A semi-supervised competitive neural network, proposed by XXu [3] divides all DNA fragments into two groups correcting minimum number of SNPs. The HapCompass model [4] is made up of a compass graph with SNPs as nodes, and the weight of each edge indicates the difference in phasing between two pieces. It removes the edges containing minimum weights from the graph. A novel Bayesian framework, HapTree designed by [5] performs SNP-pair phasing and full haplotype assembly based on a probabilistic framework. Xie *et al*. [6] suggested a heuristic algorithm H-PoP which is a dynamic programming model and reduces computational cost at each iteration. Individual haplotyping with minimum error correction proposed by [7] uses heuristic algorithm to find haplotypes for diploid organisms. An extension of this algorithm proposed by [8] finds multiple haplotypes for polyploid organisms. Fuzzy conflict graph is introduced by [9] to develop a fast heuristic partitioning model. Extension of compass graph and HapCompass framework has been performed by [10] to implement optimization in haplotype assembly. Moeinzadeh *et al*. [11] represents a new tool Rainbow for reconstructing haplotypes for polyploid genomes using short read sequence data from a highly heterozygous hexaploid genome. A greedy heuristic approach has been introduced by [12] to compute max-cuts in a graph derived from DNA fragments. It is shown by [13] that haplotyping problem is NP-hard and polynomial time algorithms have been designed for fragments assembly. A heuristic solution introduced by [14] is faster and more accurate than a dynamic programming. Some algorithmic strategies for haplotype determination have been suggested by [15] from localized polymorphism data. A novel computational model proposed by [16] improves the MEC model. A survey on single individual haplotyping problems has been performed by [17] on real haplotype data and its complexity has been explained by [18]. Phasing of variants in haplotypes proposed by [19] from overlapping sequenced fragments. Zhang *et al.* [20] presented reconstruction for haplotypes by identifying the compatible ones with the observed genotypes. Li *et al.* [21] reconstructed the haplotypes by computing the neighboring SNP phases and connecting them. Useful measures for genotype imputation accuracy and its several statistical methods have been suggested by [22] and [23] respectively. The relation of phase information with human genome has been illustrated by [24]. Baaijens and Schönhuth [25] introduced an overlap graph based on the interaction of DNA reads. This method performs better than the other reconstructing haplotype approaches. The importance of non-SNP genetic variation for defining human genome has been explained by [26].

This paper proposes a novel partitioning algorithm, HapPart which can partition a sequence of *k*-ploid organism into *k* groups faster than the heuristic algorithm H-PoP. Analyzing the minimum error correction (MEC) values for each value of *k* i.e., from 1 to total number of fragments, the expected partition is obtained. We use haplotype conflict graph which contains the haplotypes as its nodes and their distances as its edges. Merging two groups in order to move from *k*-groups to *k*-1 groups ensures construction of haplotypes with minimum error. Furthermore, this algorithm requires less amount of memory than H-PoP.

2. RESEARCH METHOD

2.1. Preliminaries

DNA sequencing techniques deal with overlapping fragments which consist of nucleotides. The smallest portion of the DNA sequence is called base. For a diploid organism, a pair of DNA molecules comes from the parents' copies. These copies are different in few positions and they are known as SNP. SNPs are considered as bi-allelic, let two alleles in an SNP site be 0 and 1. They can represent any two elements of the set {A, T, G, C}. A fragment is considered as a sequence of SNP sites. It is built up with symbols {0, 1, –} of length *n*. '–' is usually known as a gap. It means an undetermined SNP.

The input of the problem is aligned fragments of a DNA sequence. Let *m* be the numbers of fragments. Then $m \times n$ denotes the SNP matrix *M*. Each entry is denoted by *M* [*i, j*] which defines the allele of the i^{th} fragment at j^{th} SNP site. We need to determine the dissimilarity between two alleles $a_1, a_2 \in \{0, 1, -\}$. The dissimilarity function $d(a_1, a_2)$ [6] is defined as (1).

$$d(a_1, a_2) = \begin{cases} 1, & if\ a_1, a_2 \neq - \ and\ a_1 \neq a_2 \\ 0, & otherwise \end{cases} \quad (1)$$

2.2. Haplotype construction

Partitioning the given fragments into some sets of fragments and constructing the haplotypes from those sets is the main objective of the haplotype assembly problem. Let *C* be the total sets of fragments, then

$C=\{C_1, C_2, C_3, \ldots, C_k\}$. The haplotypes are constructed in such a way that they conflict with one another in minimum number of positions in the SNP site. The methodology of construction according to [7] is:

$$H_{ij}= \begin{cases} 1, & if\ N_j^1(C_i) > N_j^0(C_i); \\ 0, & if\ N_j^0(C_i) \geq N_j^1(C_i)\ and\ N_j^0(C_i) \neq 0 \\ -, & if\ N_j^0(C_i) = N_j^1(C_i) = 0 \end{cases} \quad (2)$$

where $i \in \{1, 2, \ldots, k\}$ and $j \in \{1, 2, \ldots, n\}$. $N_j^1(C)$ is the number of fragments in a cluster C where the fragments contain 1's in the $j^{\text{-th}}$ position of SNP matrix, similarly $N_j^0(C)$ means the number of fragments containing 0's. Partition $P(C_1,\ldots C_k)$ is made by one or more sets and the haplotypes for all sets are constructed then the number of errors $E(P)$ [6] is calculated as (3):

$$E_k(P)= \sum_{i=1}^{k} \sum_{f \in C_i} \sum_{j=1}^{n} d(f_j, H_{ij}) \quad (3)$$

where k is total number of sets or groups that are to be constructed. Partitioning and haplotype construction should not consider only the minimum number of errors rather it should also consider the maximum diversity among the haplotypes. According to [6], the diversity measure among the haplotypes can be calculated as (4):

$$D(P) = \sum_{i_1,\, i_2=1\ldots k; i_1 \neq i_2} \sum_{j=1}^{n} d(H_{i_1 j}, H_{i_2 j}) \quad (4)$$

In this approach, we use the diversity measure as $D_k'(P)= \frac{1}{k} D(P)$. Here k is the normalization factor which is the number of total sets.

2.3. Pairwise haplotype distance

The distance between two fragments f_1 and f_2, D (f_1, f_2), is generally defined as the number of SNPs where both fragments have different alleles. Therefore,

$$D(f_1, f_2) = \sum_{i=1}^{n} d(f_{1i}, f_{2i}) \quad (5)$$

two fragments f_1 and f_2 are said to be conflicting if D (f_1, f_2)> 0. In this approach, we are defining the pairwise haplotype distance which is similar to the (5) as (6):

$$\Delta(H_1, H_2)= \sum_{i=1}^{n} d(H_{1i}, H_{2i}). \quad (6)$$

2.4. Haplotype conflict graph

The haplotype conflict graph is built based on the pairwise haplotype distances. This graph is used to partition the fragments into k groups. Initially, for m number of fragments, there are m haplotypes which results ${}^{m}C_2$ edges in the graph with m nodes. The pairwise haplotype distance represents the edge and each haplotype represents a node. An example with 6 fragments and their conflict graph has been shown in Figure 1. First, each fragment is mapped to a string of {0, 1, –}. Then the distances among them are calculated.

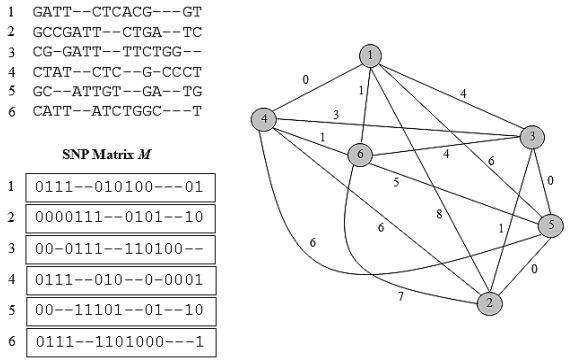

Figure 1. SNP Matrix and haplotype conflict graph for 6 fragments

2.5. Method

In the haplotype conflict graph, the two nodes having the least distance are merged together to create a new group and the haplotypes are updated as shown in the Figure 2. At each stage of the graph contraction, the MEC value is calculated and the conflict graph is updated according to the new distances among the haplotypes for further calculation. In Figure 2(a), 5 groups are formed out of 6 groups. Nodes 3 and 5 are chosen to be merged and their haplotype is updated. In the next, the distances among the 5 nodes are updated according to the distances among the haplotypes. The updated distances are indicated by red color. In the following figures from Figures 2(b) to 2(f), each iteration is demonstrated. From the MEC values found in each iteration, the MEC curve is formed and the best partition for particular *k* is determined according to the equations provided in the next subsection.

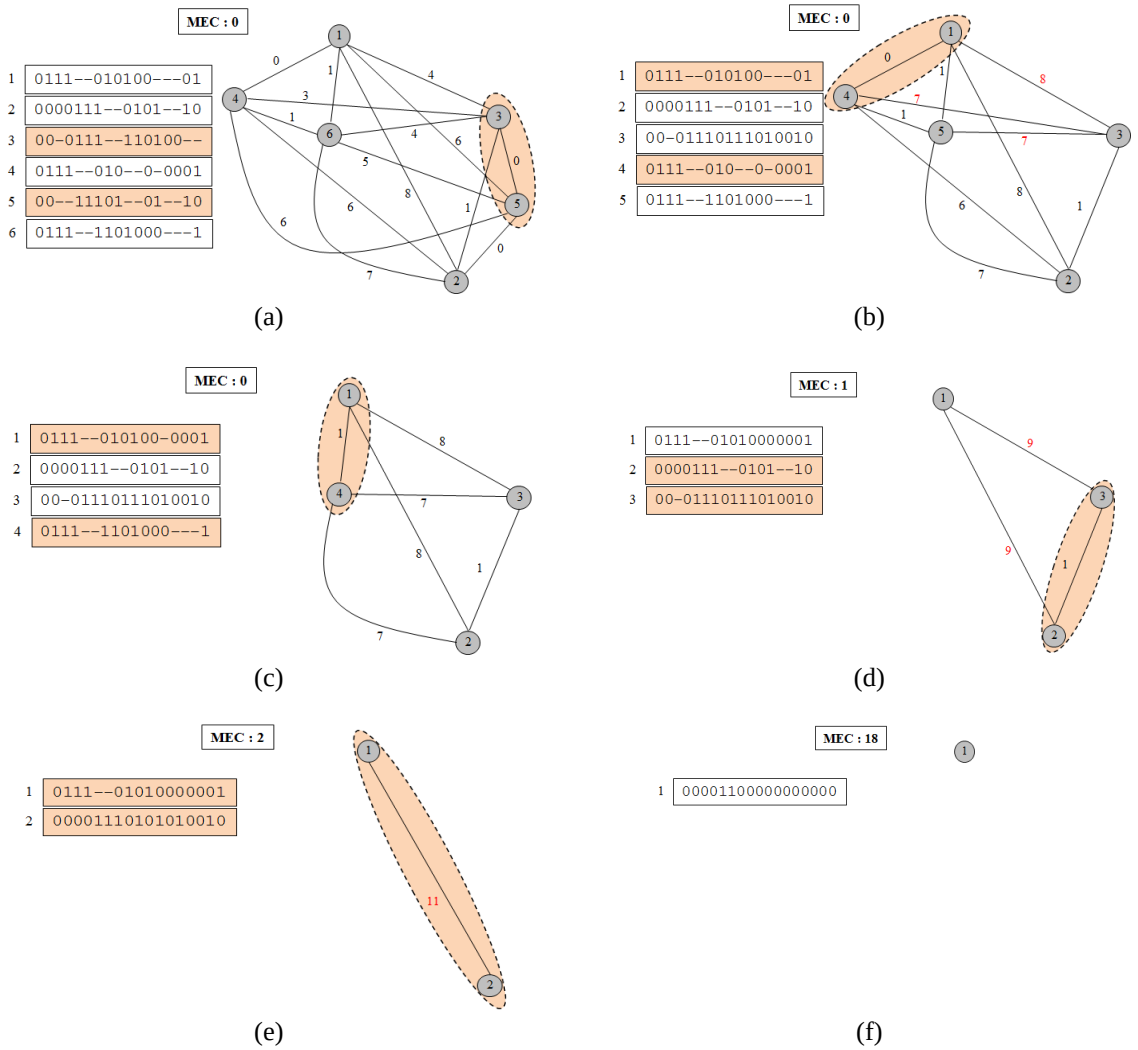

Figure 2. Steps of haplotype conflict graph contraction to find MEC values (a) initially there are 6 groups and MEC=0; 3 and 5 are merged together, (b) MEC=0; 1 and 4 are merged together, (c) MEC=0; new 1 and 4 are merged together, (d) MEC=1; 2 and 3 are merged together, (e) MEC=2; new 1 and 2 are together, and (f) MEC=18 when there is only 1 group

2.6. Algorithm

In order to find out the value of *k* at which the best partition can be achieved, we look into the errors among the fragments in all groups by varying *k*. The number of errors is 0 for *k*=total fragments, *m* because each haplotype contains 1 fragment. The number of errors becomes the highest for *k*=1 because only one haplotype contains all the fragments. Varying the value of *k* from 1 to *m*, we obtain the MEC curve which is similar to an exponential curve.

Figures 3(a), 3(b) and 3(c) show the MEC curves for partitioning 5, 90 and 400 fragments respectively. An important feature of these curves is: an abrupt change occurs at certain value of k. This certain value leads towards the best partition. To find out the abrupt change in the MEC curve the difference between two consecutive MEC values i.e., the gain, ΔE is calculated and then growth of this gain, $\Delta' E$ is calculated.

$$\Delta E_i = E_{i-1}(P) - E_i(P) \tag{7}$$

$$\Delta' E_i = \Delta E_i - \Delta E_{i+1} \tag{8}$$

where i is total number of sets. To perform these calculations, one complication will arise. To find the values of ΔE_1 and $\Delta' E_m$, we need $E_0(P)$ and $\Delta E_{m+1}(P)$. For $E_0(P)$, we need to extend the exponential curve upward solving the general exponential equation, $f(x)=ab^x$. For, $\Delta E_{m+1}(P)$, let it be equal to 0. Considering the MEC values $E_1(P)$ and $E_2(P)$, $E_0(P)$ can be calculated by (9).

$$E_0(P) = \frac{E_1(P)^2}{E_2(P)} \tag{9}$$

The diversity among the haplotypes for each value of k is calculated by the formula of $D'_k(P)$. Finally, the final score S_k for each k is computed as:

$$S_k = \Delta' E_k * D'_k(P) \tag{10}$$

The value of k for which we can get maximum score S_k is the required number of sets or groups and the profiles for each group i.e., the haplotypes are the output of this algorithm.

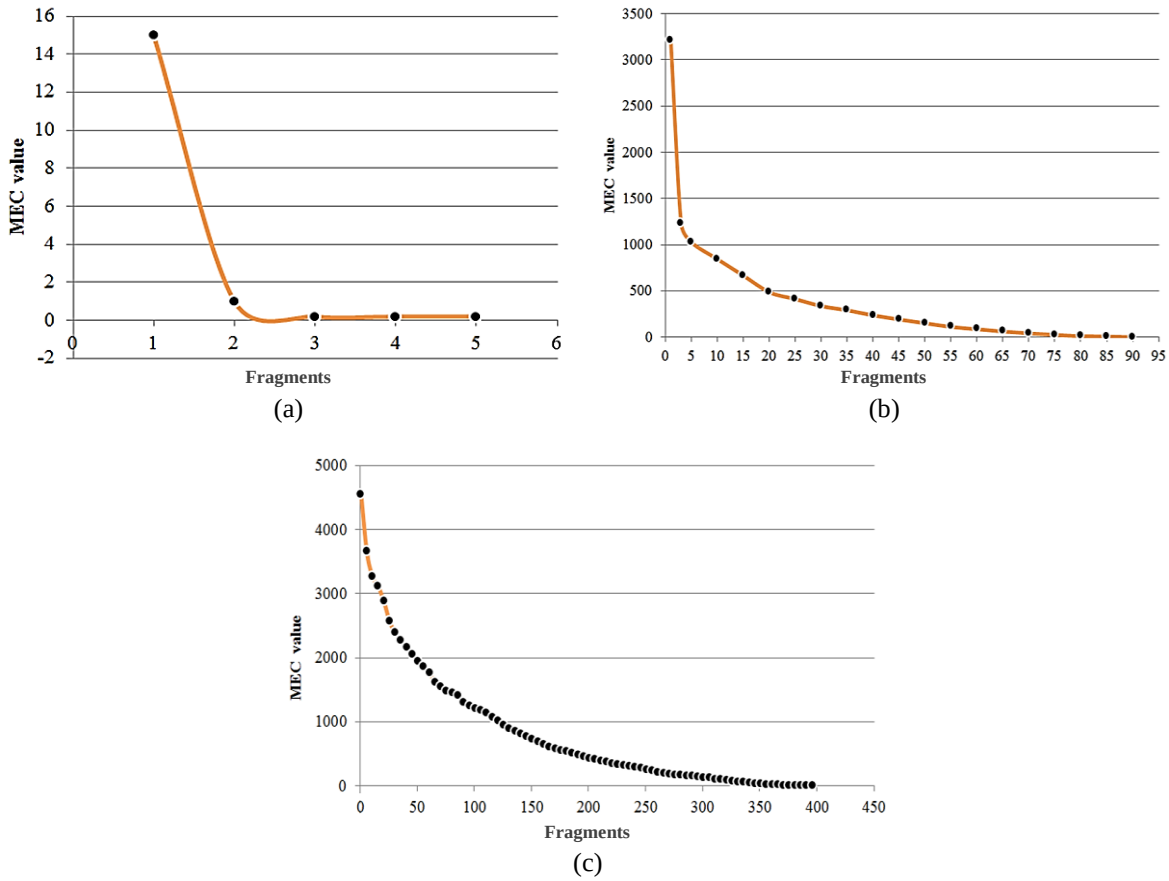

Figure 3. MEC curves in several events (a) partitioning 5 fragments, (b) partitioning 90 fragments, and (c) partitioning 400 fragments

Algorithm 1: HapPart

```
Input: an m x n matrix M.
Ouput: k number of sets of fragments and their haplotypes.
Initiate a heap H of size m
Assign each fragment to C_i where i ε {1, 2, …, m}
j = 1
while j <= m do
   begin
      Compute all haplotypes H_t for the partition P(C) where t ε {1, 2, …, (m−j+1) }
      Compute MEC E_j(P) and diversity D'_j(P)
     Insert all haplotypes H_t, E_j(P) and D_j(P) into H
     Initiate adjacency matrix G of size (m−j+1) x (m−j+1)
  if j < m
     begin
         Build or update the graph G of which each cell G_ab = Dissimilarity between H_a and H_b
         Find the minimum distance d_pq in the graph G
         Merge the sets C_p and C_q
  end if
  j++
end while
Compute E_0(P) according to (9)
for each element at k in heap H do
   Compute Δ'E_k and S_k
end for
Find the maximum score S_k according to (10)
Let L be the element in H with maximum score S_k
output k and H_1, …, H_k according to haplotypes in L.
```

For avoiding disruptions, diversity in case of k=1 is considered as 0.001 or a value that tends to 0. Similarly, the MEC values for higher values of k are considered as inverse of total number of fragments. While merging two sets, it may happen that there are several distances which are minimum. For that case we cannot choose any one of them and merge their sets. To ensure that the total error among the fragments with the haplotypes is minimum and the diversity is maximum, we have to choose all the minimum distances and build all possible partitions of fragments. We take that partition for which (MEC–diversity) is minimum.

In Figure 4, there are 4 haplotypes of which haplotype H_1 contains fragments 1 and 3, H_2 contains 2 and 5, H_3 contains only 4 and H_4 contains only 6. Among the distances there are 2 which are minimum i.e., 3. H_3 has to be merged with either H_1 or H_2. We consider H_1 to be merged with H_3 and calculate (MEC-diversity). Again, we consider H_2 to be merged with H_3 and do the same. H_3 must be merged with that haplotype for which (MEC – diversity) is minimum.

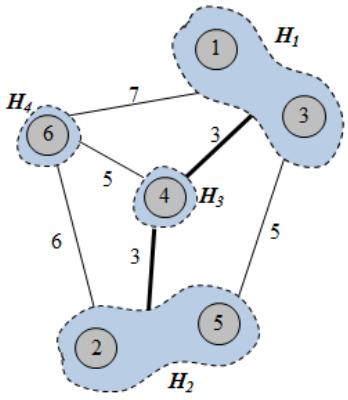

Figure 4. An instance where two sets are to be merged

Theorem 1. Choosing the minimum distance in the graph and merging the groups having that distance ensures the best partition when k-1 groups are built from k groups.

Proof. Let there be four fragments p, q, r, and s. The distances between p and q is d_1, between q and s is d_2 and between p and r is d_3. The relation among the distances is $d_1<d_2<d_3$. As d_1 is the minimum, we choose p and q be merged together and haplotype, H_{pq} be constructed. Usually, haplotype is constructed in such a way that it represents most of the characteristics of its fragments.

In Figure 5(a), d_1 is considered as minimum and p and q are merged. In Figure 5(b), the distances between H_{pq} and r and between H_{pq} and s have been updated as d_3' and d_2' respectively where $d_2'<d_3'$. Now we have to prove that merging p, q and s results better partition than merging p, q, and r. In Figure 5(c) p, q and s are merged. Haplotype H_{pqs} will be at their center to make sum of the distances between H_{pqs} and the fragments minimum. Similarly, in Figure 5(d) H_{pqr} will be at the center of p, q, and r. The distance d_1 is same in both figures but because of d_3' being greater than d_2' the area of triangle pqr is larger than that of triangle pqs. Therefore, the sum of distances between H_{pqr} and the fragments of its group is greater than that of distances between H_{pqs} and its fragments. That means merging p, q and s ensures better partition. In the similar way, for more fragments a polygon is formed. The area of the polygon has to be minimum. This can be achieved when minimum distance is considered. In Figure 5(e), 4 haplotypes are constructed using the minimum distances between two haplotypes. Let us consider that this construction is not correct and according to Figure 5(f) haplotypes H_1 and H_4 are reconstructed for better partition. The fragment which is included with the group of H_4 has the smaller distance from the group of H_1 than that of H_4. Since HapPart considers the minimum distance. Therefore, the area of the new polygons is larger than the previous. This means total error has become larger which is against the objective. On the other hand, merging the groups having minimum distance ensures maximum diversity among haplotypes. Moreover, when there are more than one minimum distance, all possible partitions are checked for minimum (MEC-diversity) measure. It makes clear that HapPart ensures the best partition when k-1 groups are built from k groups.

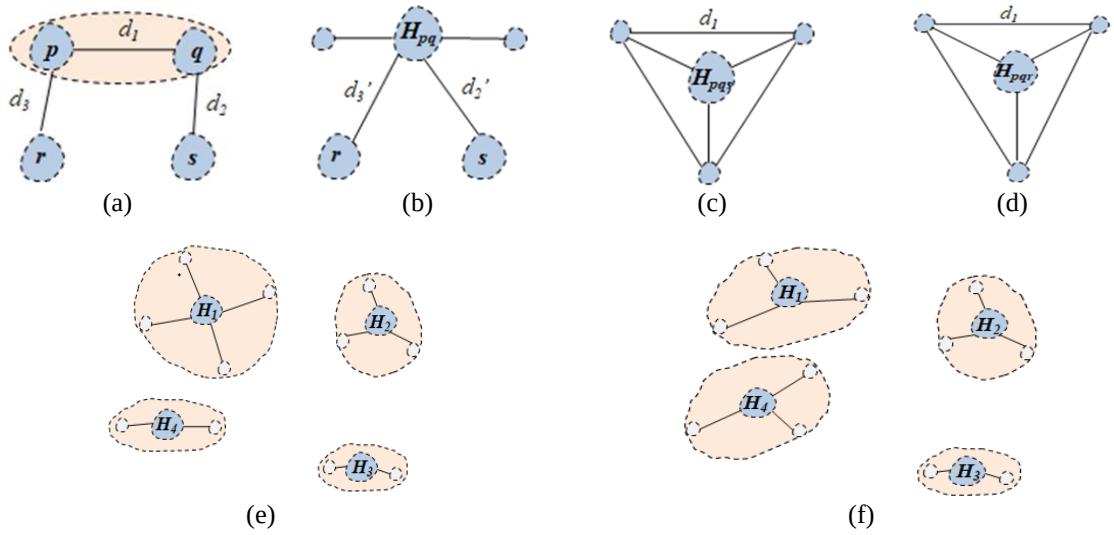

Figure 5. Several cases of merging 2 groups are shown (a) p and q are merged, (b) the graph is updated considering the distances between new haplotype h_{pq} and other fragments, (c) p, q and s are merged together, (d) p, q and r are merged together, (e) 4 haplotypes are constructed according to HapPart, and (f) haplotypes h_1 and h_4 are reconstructed.

3. RESULTS AND DISCUSSION

This research compares the performance of this approach with that of a heuristic partitioning technique, H-PoP, utilizing both actual and simulated data. We perform the simulation using the data of *oryza sativa* cDNA 5', mRNA sequence and *triticum aestivum* cDNA, mRNA sequence collected from national center for biotechnology information (NCBI) nucleotide database. Simulated data is used varying the coverage rates and error rates. The value of k is also varied to find out the correctness for tetraploid, pentaploid and hexaploid containing errors. All experimental tests are carried on a Windows 64-bit operating system node containing processing unit 2.40 GHz and memory unit 4.00 GB.

3.1. Experiments on *oryza sativa* cDNA 5', mRNA sequence

From the *oryza sativa* cDNA 5', mRNA sequence containing 490 SNPs, numerous samples of reads are created with 75% coverage rate. During the overlapping reads generation, the polyploidy is maintained. Using the multiple sequence alignment tool, the reads are first aligned. The data is then used to run the proposed model. This model varies the total number of SNP sites and its ploidy. Figure 6 shows the result from this model that runs on a sequence of triploid *oryza sativa*.

Figure 6 shows final score Δ′*E* at different values of *k* for a sequence consisting of 90 fragments from a triploid *oryza sativa*. At *k* = 3, the score is the highest which indicates the best partition of the fragments. Table 1 shows comparisons in execution time and memory use between HapPart and H-PoP. Table 1 shows that as the value of *k* rises, the computational cost of H-PoP rises faster than that of HapPart in proportion to the total number of SNPs. When the total number of SNPs for *k* = 3 and *k* = 2 are the same, H-PoP takes more than 20 minutes. In terms of memory utilization, HapPart requires far less than H-PoP. Memory use for H-PoP increases dramatically as the value of *k* increases.

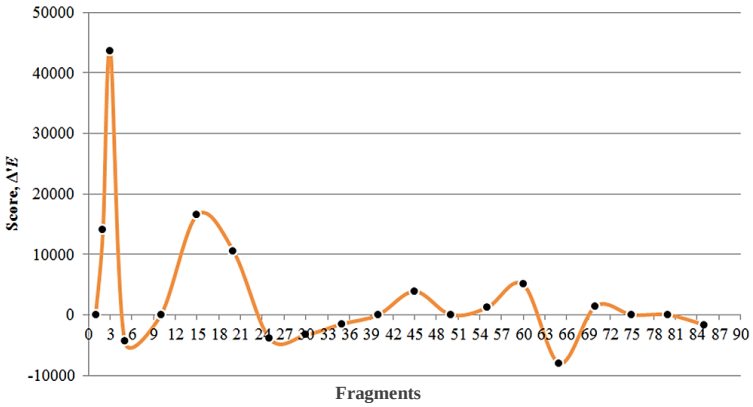

Figure 6. Score Δ′E against different values of *k* for a triploid *oryza sativa* sequence

Table 1. Comparisons considering dataset *oryza sativa* cDNA 5′, mRNA sequence

k-ploidy	Total SNPs	Execution time (sec)		Memory use (MB)	
		HapPart	H-PoP	HapPart	H-PoP
k = 2	25200	5.033	61.504	15.36	35.83
k = 3	5016	0.129	34.169	9.6	61.51
k = 4	3444	0.051	58.631	9.3	194.59

3.2. Experiments on *triticum aestivum* cDNA, mRNA sequence

This experiment is performed exactly in the same way that we have done for *oryza sativa* cDNA 5′, mRNA sequence. The *triticum aestivum* cDNA, mRNA sequence contains 770 SNPs. Result of executing the sequence for tetraploid *triticum aestivum* has been shown in Figure 7. The sequence consists of 400 fragments. The scores Δ′E for some of first 50 values of *k* have been provided. Table 2 shows the comparisons between HapPart and H-PoP for the data varying *k* and the total number of SNPs.

Each fragment has a longer length than the previous data. As a result, we must consider a smaller number of fragments. Otherwise, H-PoP takes several minutes to complete. On the other hand, this algorithm takes a fraction of 1 second. It takes about 6 minutes to execute a sequence of 400 fragments.

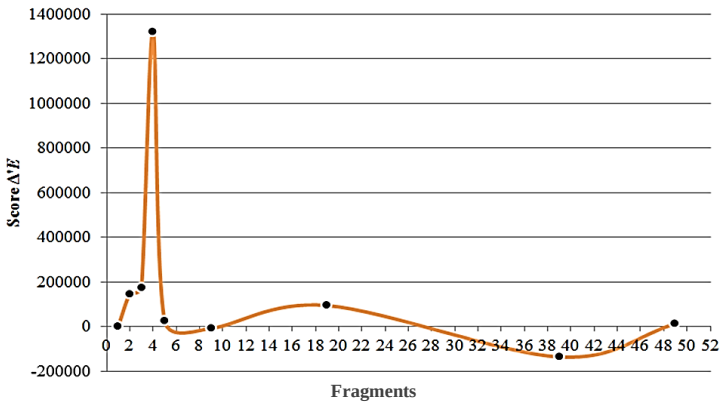

Figure 7. Score Δ′E against different values of *k* for a tetraploid *triticum aestivum* sequence

Table 2. Comparisons considering dataset *triticum aestivum* cDNA, mRNA sequence

k-ploidy	Total SNPs	Execution time (sec)		Memory use (MB)	
		HapPart	H-PoP	HapPart	H-PoP
k = 2	12680	0.279	11.712	10.24	30.709
k = 3	6944	0.109	14.916	9.78	43.103
k = 4	6780	0.116	142.670	12.28	256.06

3.3. Experiments on simulated data

For further evaluation, this research uses simulated data. Experiments on the real data show that the proposed model takes less time to execute than the existing H-PoP model. Several overlapping reads are generated with 50% and 75% coverage rates. In addition, this model varies the error rates by 5%, 10%, 15% and 20%. The value of *k* is varied from 4 to 6 to show the performance of HapPart regarding the reconstruction of haplotypes. Phasing accuracy is the degree to which the SNPs in the reconstructed haplotypes match those in the genuine haplotypes. Correct phasing rate [6] is used to measure this degree for *k* haplotypes. The comparisons between HapPart and H-PoP are shown in Figure 8.

In Figures 8(a) and 8(b), though the correct phasing rates are better for H-PoP in most of the cases, HapPart shows results close to H-PoP in many cases. They remain within 0.89 to 0.91 for coverage 75% and within 0.95 to 0.96 for coverage 50%. In Figure 8(c), we show the change of correct phasing rate with the change of value *k* from 4 to 6. When *k*=4, the rate is 0.94 and it decreases to 0.91 at *k* = 6. It is observed that for long reads HapPart performs better.

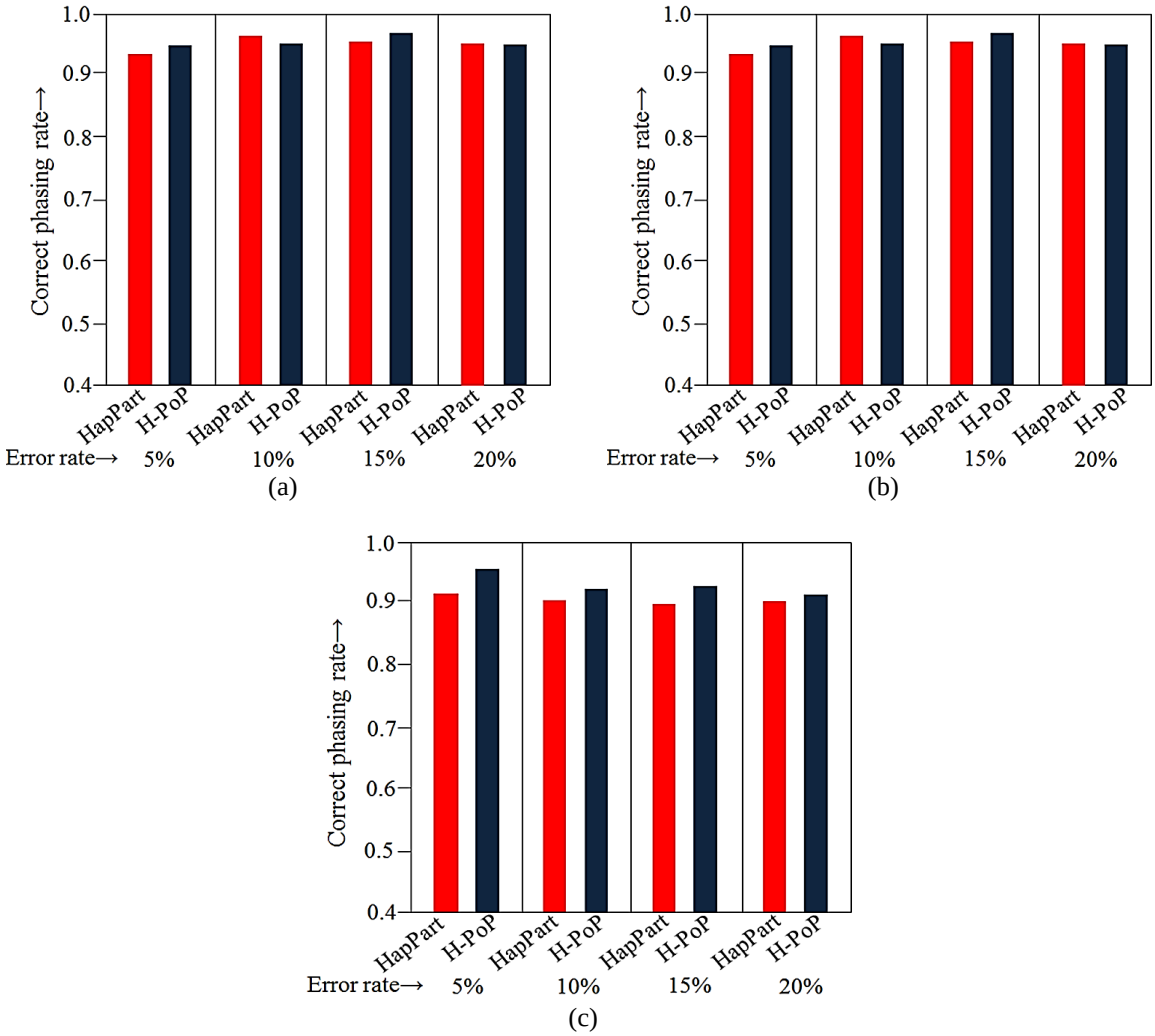

Figure 8. Correct phasing rate of HapPart and H-PoP (a) considering 50% coverage, (b) considering 75% coverage with error rates 5%, 10%, 15% and 20%, and (c) varying *k* from 4 to 6

4. CONCLUSION

The task of haplotype assembling is challenging. Many studies have been conducted in order to find an accurate and quick solution. The polyploid haplotyping problems search on large number input reads to find optimal partitioning. These are called polyploid balanced optimal partition (PBOP) problems and they are NP-hard. The heuristic algorithm, H-PoP based on distance between the consensus haplotypes of different groups can compute haplotypes faster than SDhaP, HapTree and HapCompass. Nevertheless, it requires huge amount of memory. In this paper, we developed a new partitioning algorithm for reconstructing multiple haplotypes for a polyploid organism. Performance study has been carried out with a variety of real-world and hypothetical comparisons on simulated data. It shows that HapPart computes the haplotypes faster. It also requires less amount of memory than H-PoP. However, novel direction on the calculation of pairwise haplotype distance may lead to better accuracy of the result.

REFERENCES

[1] Y. Wang, E. Feng, and R. Wang, "A clustering algorithm based on two distance functions for MEC model," *Computational Biology and Chemistry*, vol. 31, no. 2, pp. 148–150, Apr. 2007, doi: 10.1016/j.compbiolchem.2007.02.001.

[2] M. Kargar, H. Poormohammadi, L. Pirhaji, M. Sadeghi, H. Pezeshk, and C. Eslahchi, "Enhanced evolutionary and heuristic algorithms for haplotype reconstruction problem using minimum error correction model," *MATCH Communications in Mathematical and in Computer Chemistry MATCH Commun. Math. Comput. Chem*, vol. 62, pp. 261–274, 2009.

[3] X. Xu, "A semi-supervised style method for haplotype assembly problem based on MEC model," in *Proceedings - 2010 6th International Conf. on Natural Computation, ICNC 2010*, Aug. 2010, vol. 3, pp. 1508–1512, doi: 10.1109/ICNC.2010.5582649.

[4] D. Aguiar and S. Istrail, "HapCompass: a fast cycle basis algorithm for accurate haplotype assembly of sequence data," *Journal of Computational Biology*, vol. 19, no. 6, pp. 577–590, Jun. 2012, doi: 10.1089/cmb.2012.0084.

[5] E. Berger, D. Yorukoglu, J. Peng, and B. Berger, "HapTree: a novel bayesian framework for single individual polyplotyping using NGS data," *PLoS Computational Biology*, vol. 10, no. 3, Mar. 2014, doi: 10.1371/journal.pcbi.1003502.

[6] M. Xie, Q. Wu, J. Wang, and T. Jiang, "H-PoP and H-PoPG: heuristic partitioning algorithms for single individual haplotyping of polyploids," *Bioinformatics*, vol. 32, no. 24, pp. 3735–3744, Aug. 2016, doi: 10.1093/bioinformatics/btw537.

[7] M. S. Bayzid, M. M. Alam, A. Mueen, and M. S. Rahman, "HMEC: A heuristic algorithm for individual haplotyping with minimum error correction," *ISRN Bioinformatics*, vol. 2013, pp. 1–10, Jan. 2013, doi: 10.1155/2013/291741.

[8] M. M. Hossain, A.-B. M. Abdullah, and P. C. Shill, "An extension of heuristic algorithm for reconstructing multiple haplotypes with minimum error correction," in *2020 Emerging Technology in Computing, Communication and Electronics (ETCCE)*, Dec. 2020, pp. 1–6, doi: 10.1109/ETCCE51779.2020.9350874.

[9] S. Mazrouee and W. Wang, "FastHap: fast and accurate single individual haplotype reconstruction using fuzzy conflict graphs," *Bioinformatics*, vol. 30, no. 17, pp. i371–i378, Aug. 2014, doi: 10.1093/bioinformatics/btu442.

[10] D. Aguiar and S. Istrail, "Haplotype assembly in polyploid genomes and identical by descent shared tracts," *Bioinformatics*, vol. 29, no. 13, pp. i352–i360, Jun. 2013, doi: 10.1093/bioinformatics/btt213.

[11] M. H. Moeinzadeh *et al.*, "Ranbow: a fast and accurate method for polyploid haplotype reconstruction," *PLoS Computational Biology*, vol. 16, no. 5, May 2020, doi: 10.1371/journal.pcbi.1007843.

[12] V. Bansal and V. Bafna, "HapCUT: an efficient and accurate algorithm for the haplotype assembly problem," *Bioinformatics*, vol. 24, no. 16, pp. i153–i159, Aug. 2008, doi: 10.1093/bioinformatics/btn298.

[13] G. Lancia, V. Bafna, S. Istrail, R. Lippert, and R. Schwartz, "SNPs problems, complexity, and algorithms," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 2161, Springer Berlin Heidelberg, 2001, pp. 182–193.

[14] A. Panconesi and M. Sozio, "Fast hare: a fast heuristic for single individual SNP haplotype reconstruction," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3240, Springer Berlin Heidelberg, 2004, pp. 266–277.

[15] R. Lippert, R. Schwartz, G. Lancia, and S. Istrail, "Algorithmic strategies for the single nucleotide polymorphism haplotype assembly problem.," *Briefings in bioinformatics*, vol. 3, no. 1, pp. 23–31, Jan. 2002, doi: 10.1093/bib/3.1.23.

[16] R. S. Wang, L. Y. Wu, Z. P. Li, and X. S. Zhang, "Haplotype reconstruction from SNP fragments by minimum error correction," *Bioinformatics*, vol. 21, no. 10, pp. 2456–2462, Feb. 2005, doi: 10.1093/bioinformatics/bti352.

[17] F. Geraci, "A comparison of several algorithms for the single individual SNP haplotyping reconstruction problem," *Bioinformatics*, vol. 26, no. 18, pp. 2217–2225, Jul. 2010, doi: 10.1093/bioinformatics/btq411.

[18] R. Cilibrasi, L. Van Iersel, S. Kelk, and J. Tromp, "The complexity of the single individual SNP haplotyping problem," *Algorithmica (New York)*, vol. 49, no. 1, pp. 13–36, Aug. 2007, doi: 10.1007/s00453-007-0029-z.

[19] S. R. Browning and B. L. Browning, "Haplotype phasing: existing methods and new developments," *Nature Reviews Genetics*, vol. 12, no. 10, pp. 703–714, Sep. 2011, doi: 10.1038/nrg3054.

[20] K. Zhang, F. Sun, and H. Zhao, "HAPLORE: a program for haplotype reconstruction in general pedigrees without recombination," *Bioinformatics*, vol. 21, no. 1, pp. 90–103, Jul. 2005, doi: 10.1093/bioinformatics/bth388.

[21] L. M. Li, J. H. Kim, and M. S. Waterman, "Haplotype reconstruction from SNP alignment," *Journal of computational biology : a journal of computational molecular cell biology*, vol. 11, no. 2–3, pp. 505–516, Mar. 2004, doi: 10.1089/1066527041410454.

[22] B. L. Browning and S. R. Browning, "A unified approach to genotype imputation and haplotype-phase inference for large data sets of trios and unrelated individuals," *American Journal of Human Genetics*, vol. 84, no. 2, pp. 210–223, Feb. 2008, doi: 10.1016/j.ajhg.2009.01.005.

[23] J. Marchini and B. Howie, "Genotype imputation for genome-wide association studies," *Nature Reviews Genetics*, vol. 11, no. 7, pp. 499–511, Jun. 2010, doi: 10.1038/nrg2796.

[24] R. Tewhey, V. Bansal, A. Torkamani, E. J. Topol, and N. J. Schork, "The importance of phase information for human genomics," *Nature Reviews Genetics*, vol. 12, no. 3, pp. 215–223, Feb. 2011, doi: 10.1038/nrg2950.

[25] J. A. Baaijens and A. Schönhuth, "Overlap graph-based generation of haplotigs for diploids and polyploids," *Bioinformatics*, vol. 35, no. 21, pp. 4281–4289, Apr. 2019, doi: 10.1093/bioinformatics/btz255.

[26] S. Levy *et al.*, "The diploid genome sequence of an individual human," *PLoS Biology*, vol. 5, no. 10, pp. 2113–2144, Sep. 2007, doi: 10.1371/journal.pbio.0050254.

BIOGRAPHIES OF AUTHORS

Abu-Bakar Muhammad Abdullah received the BSc. Eng. degree in Computer Science and Engineering from Bangladesh University of Engineering and Technology, Bangladesh, in 2015 and is pursuing the MSc. Eng. degree in Computer Science and Engineering from Khulna University of Engineering and Technology, Khulna, Bangladesh. Currently, he is an Assistant Professor at the Department of Computer Science and Engineering, Bangabandhu Sheikh Mujibur Rahman Science and Technology University, Gopalgonj, Bangladesh. His research interests include Bioinformatics, Algorithms, Large Scale Data Management and Artificial Intelligence. He can be contacted at email: abdullahcse09@gmail.com.

Md. Monowar Hossain completed B.Sc. Eng. degree in Computer Science and Engineering from Chittagong University of Engineering & Technology (CUET) in 2014 and M.Sc. Eng. degree in Computer Science and Engineering from Khulna University of Engineering & Technology (KUET) in 2021 with outstanding result. His current research interests are Bioinformatics, Natural Language Processing, Cryptography and Machine Learning. He was a solution delivery engineer at Systems Solutions & Development Technologies Ltd. Now, he is an Assistant Professor in the Department of Computer Science & Engineering (CSE), Bangabandhu Sheikh Mujibur Rahman Science and Technology University (BSMRSTU), Gopalganj, Bangladesh. He can be contacted at email: murad0904045@gmail.com.

Pintu Chandra Shill received the B.Sc. degree in Computer Science and Engineering (CSE) from Khulna University of Engineering & Technology (KUET), Bangladesh in 2003, M.Sc. degree in Computer Engineering from Politecnico di Milano, Italy in 2008 and a Ph.D degree in Intelligent Information Systems from University of Fukui, Japan in 2013. He is currently a Professor in the Department of Computer Science and Engineering, Khulna University of Engineering & Technology (KUET), Bangladesh, where he is a member of the Computational Intelligence Research Group. He has authored or co-authored more than 50 papers in international journals, and conferences. His current research interests include multi-objective genetic algorithms and genetic fuzzy systems, knowledge extraction based on evolutionary algorithms, neural computing, and bioinformatics. He is a member of the Institution of Engineers, Bangladesh (IEB). He can be contacted at email: pintu@cse.kuet.ac.bd.