A Heuristic Algorithm for Minimum Conflict Individual Haplotyping

Md. Shamsuzzoha Bayzid, Md. Maksudul Alam and Md. Saidur Rahman
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology
Dhaka-1000, Bangladesh

Abstract—**Haplotype is a pattern of SNPs (Single Nucleotide Polymorphism) on a single chromosome. Constructing a pair of haplotypes from aligned and overlapping but intermixed and erroneous fragments of the chromosomal sequences is a nontrivial problem. Minimum error correction (MEC) model, which is the mostly used model, minimizes the number of errors to be corrected so that the pair of haplotypes can be constructed through consensus of the fragments. However, this model is effective only when the error rate of SNP fragments is low. To overcome this problem, Zhang *et al.* proposed a new model called Minimum Conflict Individual Haplotyping (MCIH) as an extension to MEC [1]. This new model uses both SNP fragment information and related genotype information for haplotype reconstruction. MCIH has already been proven to be a potential alternative in individual haplotyping. In this paper, we give a heuristic algorithm for MCIH that searches through alternative solutions using a gain measure and stops whenever no better solution can be achieved. Experimental results on real data show that our algorithm performs better than the best known algorithm for MEC and the algorithm for MCIH proposed by Zhang *et al.* [1].**

I. Introduction

A single DNA molecule is a long chain of nucleotides called base pairs. There are four such nucleotides which are represented by the set of symbols {A,T,G,C}. Hence, a DNA can be thought of as a string of symbols taken from this set. Every diploid organism has a set of pairs of DNA molecules. Each pair contains a paternal copy and a maternal copy of almost identical sequence of nucleotides considering no recombination. The copies differ at only a few positions with respect to their total length. Most of the times the variations occur at single nucleotide positions (on average 1 in every 600 base pairs) which are separated by a non-empty identical sub-sequence. Such variation is called *"Single Nucleotide Polymorphism"* and abbreviated as "SNP" [2], [3]. SNP is believed to be the most frequent form to address genetic differences [4], [5].

The nucleotide in a SNP site is called *allele*. A SNP site is called bi-allelic if it can have only two nucleotides. It is called multi-allelic if it can have more than two alleles. Almost all SNPs have two different alleles which we denote by 0 (wild type) and 1 (mutant type). From now on, we will consider the simplest case where only bi-allelic SNPs occur in a specific pair of DNA. Since the two copies DNA molecules are identical except at the SNP sites, we can describe the two copies by two shorter sequences containing information only for SNPs. These two sequences consisting of nucleotides at SNP sites only are called the *haplotypes*. A *genotype* is the conflation of two haplotypes on the homologous chromosome. In a genotype, an SNP site is called *homozygous* if the pair of alleles at the SNP site is made up of two identical values, otherwise it is called *heterozygous*.

Haplotyping an individual deals with determining a pair of haplotypes, one for each copy of a given chromosome. A chromosome is a complicated structure of a DNA molecule bound by proteins. This pair of haplotypes completely define the SNP fingerprints of an individual for a specific pair of chromosomes. Given the two sequences of bases, haplotyping is straight forward and just needs to iterate through both the sequences and remove all the common alleles from them. But, haplotyping is complicated if we want to construct haplotypes from sequencing data for more reliability. Sequencing data for a genome does not contain the complete sequences of bases for a specific chromosome, rather it provides a set of fragments of arbitrary length for the whole genome. Therefore, the actual problem of haplotyping is to find two haplotypes from the set of overlapping fragments of both the chromosomes where fragments might contain errors and to which copy of the chromosome a fragment belongs is not determined.

Haplotypes have more information content than individual SNPs in disease association studies [6], but at the same time it is substantially more difficult to determine haplotypes than to determine genotypes or individual SNPs through experiments [1]. For this reason, computational methods that can reduce the cost of determining haplotypes become important research area. The problem of haplotyping has been studied extensively. There are generally two classes of computational methods for determining haplotype namely haplotype inference and haplotype assembly or individual haplotyping. There are several models for haplotype inference based on different assumptions [5], [7]–[10]. On the other hand, haplotype assembly is based on the data and methodology of shotgun sequence assembly [11], [12]. There are several models for individual haplotyping based on different error assumptions [11]–[14], among which minimum error correction (MEC) model is the most widely used. MEC is based on the assumption that the inconsistencies of the data comes from realistic sequence errors – that can be corrected. However, MEC is not much effective when SNP fragments have a high error rate. Hence, to improve the haplotyping quality, we need to either reduce the errors in SNP fragments which requires the improvement of the shotgun experiment, or add extra information to the given SNP fragment set. Zhang *et al.* have proposed a new model combining both SNP fragments and genotype information, which they call minimum conflict individual haplotyping (MCIH), since genotype data can be much more easily and economically obtained [1]. They also presented a dynamic programming based exact algorithm for MCIH and showed that it performs better than MEC. Moreover, they proved that MCIH is NP-hard.

In this paper we present a heuristic algorithm for MCIH. Our heuristic algorithm for MEC, which we call the HMEC, has the highest reconstruction rate compared to the other approaches of MEC [15]. Although HMEC can reconstruct the haplotypes with very high reconstruction rate (almost 100%); in some cases, with very high error rates and very low coverage value (high "hole" rate), its reconstruction rate falls down. In this paper we customize our HMEC to incorporate the genotype data that leads to new heuristic algorithm for MCIH which we call HMCIH. Experimental results confirms that HMCIH performs very well in high error rate and it performs better than HMEC and the feed-forward neural network (FNN) based algorithm proposed by Zhang *et al.*

II. Preliminaries

In this section we give some definitions and preliminary ideas.

Let S be the set of k bi-allelic SNP sites over which the haplotypes will be constructed. Let F be the set of m fragments produced from two copies of the chromosome. Each fragment contains information of nonzero number of SNPs in S. Because the SNPs are bi-allelic, let the two possible alleles for each SNP site be 0 and 1 where they can be any two elements of the set $\{A, T, G, C\}$. Since all the nucleotides are the same at the sites other than SNP sites, we can remove these extraneous sites from all the fragments and consider the fragments as sequences of SNP sites only. Thus each fragment $f \in F$ is a string of symbols $\{0, 1, -\}$ of length k where '$-$' denotes an undetermined SNP named as *hole*. All the fragments can be arranged in an $m \times k$ matrix $M = \{M_{ij}\}, i = 1, \ldots, m, j = 1, \ldots k$, where row i is a fragment of F and column j is a SNP of S. This matrix is called SNP matrix.

TABLE I
A SNP Matrix.

----1101------------
-----0001110101-----
11010010011---------
---10100---010------
---------10110101011
010111---------01011

The consecutive sequence of '$-$'s that lie between two non-hole symbols is called a *gap*. A *gapless* SNP matrix is the one that has no gap in any of the fragments. In Table I, the first, second and third rows have no gaps while each of the fourth and sixth rows has one gap.

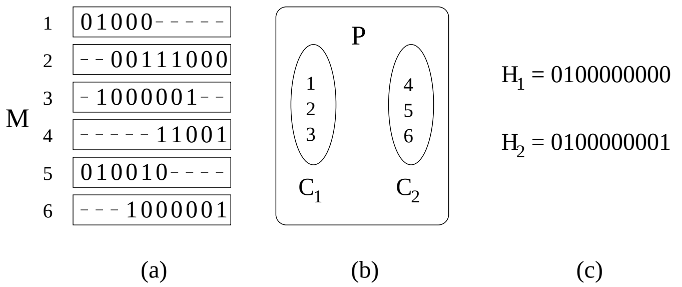

Fig. 1. SNP matrix and its partition.

A SNP matrix $M =< M_1, M_2, \ldots, M_m >$ can be viewed as an ordered set of m fragments where a fragment $M_i =< M_{i1}, M_{i2}, \ldots, M_{ik} >$ is an ordered set of k alleles. A fragment M_i is called to *cover* the jth SNP if $M_{ij} \in \{0, 1\}$ and called to *skip* the jth SNP if $M_{ij} = -$. Let M_s and M_t be two fragments. The distance between two fragments, $D(M_s, M_t)$, is defined as the number of SNPs that are covered by both of the fragments and have different alleles. Hence,

$$D(M_s, M_t) = \sum_{j=1}^{k} d(M_{sj}, M_{tj}) \tag{1}$$

where $d(x, y)$ is defined as

$$d(x, y) = \begin{cases} 1 & \text{if } x \neq - \text{ and } y \neq - \text{ and } x \neq y; \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

In Table. I, the distance between second and third fragment is two, as they differ in the seventh and ninth SNP sites (columns).

Two fragments M_s and M_t are said to be *conflicting* if $D(M_s, M_t) > 0$, otherwise they are *compatible*. Let $P(C_1, C_2)$ be a *partition* of M, where C_1 and C_2 are two sets of fragments taken from M so that $C_1 \bigcup C_2 = M$ and $C_1 \bigcap C_2 = \phi$ [14]. In Fig. 1(b), an arbitrary partition, corresponding to the SNP matrix of Fig. 1(a), is shown. A SNP matrix M is an *error-free* matrix if and only if there exists a partition $P(C_1, C_2)$ of M such that for any two fragments $x, y \in C_i, i \in \{1, 2\}$, x and y are non-conflicting, i.e., $D(x, y) = 0$. Such a partition is called an error-free partition.The partition in the Fig. 1(b) is not error free since $D(M_1, M_2) > 0$ in C_1 and $D(M_5, M_6) > 0$ in C_2 . A haplotype $H_i, i \in \{1, 2\}$ is constructed from its corresponding fragment class C_i using the formula,

$$H_{ij} = \begin{cases} 1 & \text{if at least one fragment in } C_i \\ & \text{has a 1 in } jth \text{ SNP;} \\ 0 & \text{if at least one fragment in } C_i \\ & \text{has a 0 in } jth \text{ SNP;} \\ - & \text{if all the fragments in } C_i \text{ skips} \\ & jth \text{ SNP.} \end{cases} \tag{3}$$

where C_i is called the defining class of haplotype H_i and $H_{ij}, i \in \{1, 2\}$ and $j = 1, \ldots k$, denotes the jth element of haplotype H_i.

A genotype g is represented as $g = (g_1, g_2, \cdots, g_n)$, where $g_j = 0$ if the jth SNP site is wild type homozygous; when it is mutant type homozygous, $g_j = 1$; and $g_j = 2$ if it is heterozygous. We call a pair of haplotypes $H_1 = (H_{11}, H_{12}, \cdots, H_{1n})$ and $H_2 = (H_{21}, H_{22}, \cdots, H_{2n})$ *compatible* with a genotype g if for each each SNP site j where $g_j \neq 2$, $H_{1j} = H_{2j} = g_j$; and for each SNP site j where $g_j = 2$, $H_{1j} = 0, H_{2j} = 1$ or $H_{1j} = 1, H_{2j} = 0$ [14].

Now we take a focus to the general minimum error correction problem. If a matrix M is *not* error-free, there will be no error-free partition P. For such M there will be at least one conflicting pair of fragments in each of the classes of all possible partitions. Therefore it is impossible to construct a haplotype that is non-conflicting with all the fragments in its defining class of fragments. If we are given a partition $P(C_1, C_2)$ and two haplotypes H_1 and H_2 constructed from P then the number of errors $E(P)$ that must be corrected can be readily calculated by the following formula,

$$E(P) = \sum_{i=1}^{2} \sum_{f \in C_i} D(f, H_i) \tag{4}$$

The MEC problem asks to find a partition P that minimizes the error function $E(P)$ over all such partitions of a SNP matrix M. Minimum conflict individual haplotyping (MCIH) incorporates the genotype information into MEC, and can be defined as: Given a set of SNP fragments (SNP matrix M) from an individual's DNA and the related genotype g, reconstruct a pair of haplotypes compatible with g and involving a minimum number of conflicts with the given SNP fragments [1].

III. A Heuristic Algorithm

In this section we present our heuristic algorithm for MCIH which we call HMCIH. Here we incorporate the genotype information into our proposed heuristic algorithm for MEC (HMEC) in [15]. HMCIH works in two steps. First, it uses the HMEC to reconstruct a pair of haplotypes from a set of SNP fragments by dividing the given SNP fragments into two disjoint sets of pairwise compatible fragments, with each set determining a haplotype. HMEC have already been proven to be the best algorithm in this purpose. We do not incorporate the genotype information in gain calcualation during this partitioning phase relying on the extra ordinary high reconstruction rate of HMEC and also considering the chance of getting trapped at local minima; rather we use the genotype information to refine the pair of haplotypes, generated by HMEC, in the next step which we call the *refinement step*. We now describe the refinement step after giving a brief overview of HMEC. The details of HMEC can be found in [15].

A. HMEC

Construction of a haplotype from an erroneous class C requires correction of SNP values, i.e., alleles, in the fragments. Since, we want to correct minimum number of errors, we have to construct a haplotype which is minimum conflicting with the fellow fragments of its defining class. Therefore, for each SNP site, the haplotype should take the allele that is present in majority of the fragments. Let $N_j^0(C)$ be the number of fragments of a collection C that have 0 in jth SNP. Similarly, $N_j^1(C)$ defines the number of 1s [14]. Therefore, to minimize the number of errors $E(P)$ for a specific partition P, the haplotype should be constructed according to the following methodology

$$H_{ij} = \begin{cases} 1 & \text{if} \quad N_j^1(C_i) > N_j^0(C_i); \\ 0 & \text{if} \quad N_j^0(C_i) \geq N_j^1(C_i) \text{ and} \\ & \qquad N_j^0(C_i) \neq 0; \\ - & \text{if} \quad N_j^1(C_i) = N_j^0(C_i) = 0. \end{cases} \tag{5}$$

where $i \in \{1, 2\}$ and $j = 1, 2, \ldots, k$. In Fig. 1(c) the two haplotypes H_1 and H_2, associated with the partition P in Fig. 1(b), are constructed by this method.

To find the best partition we will use a heuristic search. This algorithm starts with a current partition $P_c = P(M, \phi)$ and iteratively searches a better partition. In each iteration the algorithm performs a sequence of transfer of fragments from their present collection to the other one so that the partition becomes less erroneous. A fragment's transfer of collection can both increase or decrease the error function $E(P)$. Let the partition before transferring a fragment f be P_p and the partition resulted is P_n. We define the gain of the transfer as $Gain(f) = E(P_p) - E(P_n)$. Fig. 2 demonstrates an example calculation of gain measure. Let $F =< f_i >, i = \{1, 2, \ldots, m\}$ be an ordering of all the fragments in a partition P in such a way that fragment f_i will precede fragment f_j if all the fragments before f_i in F have already been transferred to form an intermediate partition P_i and $Gain(f_i) \geq Gain(f_j)$ over P_i. Hence, $P_1 = P_c$ at the start of each iteration. We also define the cumulative gain of a fragment ordering F upto nth fragment as $CGain(F, n) = \sum_{i=1}^{n} Gain(f_i)$. Here $Gain(f_i) = E(P_i) - E(P_{i+1})$. The maximum cumulative gain, $MCGain(F)$ is defined as

$$MCGain(F) = \max_{1 \leq i \leq m} CGain(F, i).$$

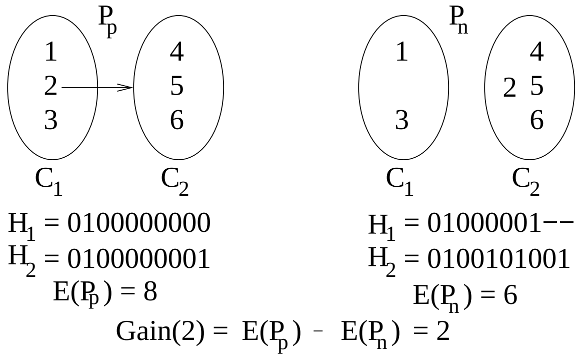

Fig. 2. An example of Gain measurement calculation.

In each iteration the algorithm finds the current ordering F_c of P_c and transfers only those fragments of F_c that can achieve the $MCGain(F_c)$ and the fragment that is the last to be transferred is referred as f_{max}. Thus the algorithm moves from one partition to another reducing the error function by an amount of $MCGain(F_c)$. The algorithm continues as long as $MCGain(F_c) > 0$ and stops whenever $MCGain(F_c) \leq 0$.

B. Refinement Step

HMEC can reconstruct haplotypes with very high reconstruction rate. For SNP fragments with low or medium error rate the reconstruction rate is near about 100%. However, its reconstruction rate slightly falls down for very high error rate and low coverage value. We investigate that there are some potential opportunities to improve the reconstruction rate with the help of genotype information. Only in some few cases, genotype may also fail to correct errors.

We now describe our refinement strategies. Let M and g be a given SNP matrix and the relative genotype respectively, and H_1 and H_2 be the original pair of haplotypes. Let HMEC divides M into two disjoint sets of fragments C_1 and C_2 that determine H_1' and H_2' respectively, i.e, H_1' and H_2' are the reconstructed haplotypes. The refinement procedure for jth SNP site, using the genotype information, is as follows. When $g_j = i, i \neq 2$; we set $H_{1j}' = H_{2j}' = i$ to make H_1' and H_2' compatible with g. We now consider the case $g_j = 2$. Let for any variable $x \in \{m, n\}$, x^T is defined as follows.

$$x^T = \begin{cases} m & \text{if } x = n; \\ n & \text{if } x = m. \end{cases} \qquad (6)$$

Let $n_{k,j} = \max\limits_{l \in \{0,1\}} N_j^l(C_k)$; $k \in \{1, 2\}$. First, we calculate $n_{1,j}$ and $n_{2,j}$. Then we need to consider the following cases.
Case 1: $n_{1,j} = n_{2,j} = 0$.
This is the case where both H_1' and H_1' contain a hole at the jth SNP site. Therefore, we do not have enough information to refine it.
Case 2: $n_{1,j} > n_{2,j}$.
In this case, we assign $H_{2j}' = p^T$, where $p = H_{1j}'$.
Case 3: $n_{2,j} > n_{1,j}$.
In this case, we assign $H_{1j}' = q^T$, where $q = H_{2j}'$.
Case 4: $n_{1,j} = n_{2,j}$.
Let $p = H_{1j}'$ and $q = H_{2j}'$. We now assign $H_{2j}' = p^T$ if $N_j^{p^T}(C_1) < N_j^{q^T}(C_2)$; otherwise we assign $H_{1j}' = q^T$.

One can observe that the aforementioned refinement strategy for $g_j = 2$ will not always refine the jth SNP site of H_1 and H_2 correctly, but we can expect a high probability of correct refinement in this way. Experimental results also confirm this. For an $m \times k$ SNP matrix, time complexity of each iteration of HMEC is $O(m^3k)$ which can be reduced to $O(m^2k)$ by using special data structure [15]. Refinement step is executed only once and it takes $O(m)$ time.

IV. Experimental Results

In this section, we use real biological data as well as simulated data to demonstrate the performance of our algorithm. We performed the simulation using the data from angiotensin-converting enzyme (ACE) [16] and public Daly set [17]. We compared the performance of our algorithm with that of HMEC and the feed-forward neural network (FNN) based algorithm of MCIH proposed by Zhang *et al.* [1]. The simulation was conducted on a computer with 1.80 GHz Core 2 Duo processor. We used Microsoft Visual C++ compiler 6 for implementation.

We now proceed to our testing methodologies. We first sample the original haplotype pair into many fragments with certain coverage rate and error rate. Each fragment works as distinct sample of the same specimen. Here coverage rate indicates how many columns of SNP matrix have been sampled out. The remaining sites are gaps/holes. We then introduce some specific amount of error into these samples. The number of fragments, coverage and error rate are user given input for our simulated sequencing technique. The simulation was controlled in several ways. We varied the error rate while number of fragments and coverage rate were kept constant. We also performed this procedure for different coverage values.

A. Simulation on angiotensin-converting enzyme (ACE)

Angiotensin-converting enzyme catalyses the conversion of angiotensin I to the physiologically active peptide angiotensin II, which controls fluid-electrolyte balance and systematic blood pressure. Because it has a key function in the renin-angiotensin system, many association studies have been performed with DCP1 (encode angiotensin-converting anzyme) [1]. Rieder *et al.* completed the genomic sequencing of the DCP1 gene from 11 individuals and reported 78 SNP sites in 22 choromosomes [16].

We take six pairs of haplotypes to perform the simulation. We generate 50 fragments from each of these haplotype pairs with varying coverage and error rates. We perform the simulation for three different coverage rates (25%, 50% and 75%). For each of these coverage rates, we perform our simulation for different error rates: 5%, 10%, 15%, 20%, 25%, 30%, 40%, 50% and 60%. Figure 3 demonstrates the results of HMEC and HMCIH averaged on six individuals. The reconstruction rate achieved by our algorithm for most instances is 100% or around 98% $\sim$ 99%. Only for a few cases with very high error rate (50%, 60%) and very low coverage rate (25%), the reconstruction rate falls below 95%. The comparison clearly confirms the superiority of HMCIH over HMEC. Moreover, HMCIH works much better than the FNN based algorithm for MCIH. When the reconstruction rate achieved by FNN falls down to around 90% with 25% coverage and 25% error rate [1], HMCIH achieves 97% reconstruction rate with that parameter settings. Even with 60% error and 25% coverage, reconstruction rate of HMCIH does not fall below 90%. Moreover, our algorithm solves each of these instances in no more than 0.015 sec.

B. Simulation on data from chromosome 5q31

In this section we demonstrate our simulation result conducted on the data from public Daly set. Daly *et al.* reported a high-resolution analysis of a haplotype structure across 500kb on chromosome 5q31 using 103 SNPs in a European derived population which consists of 129 trios [1], [17].

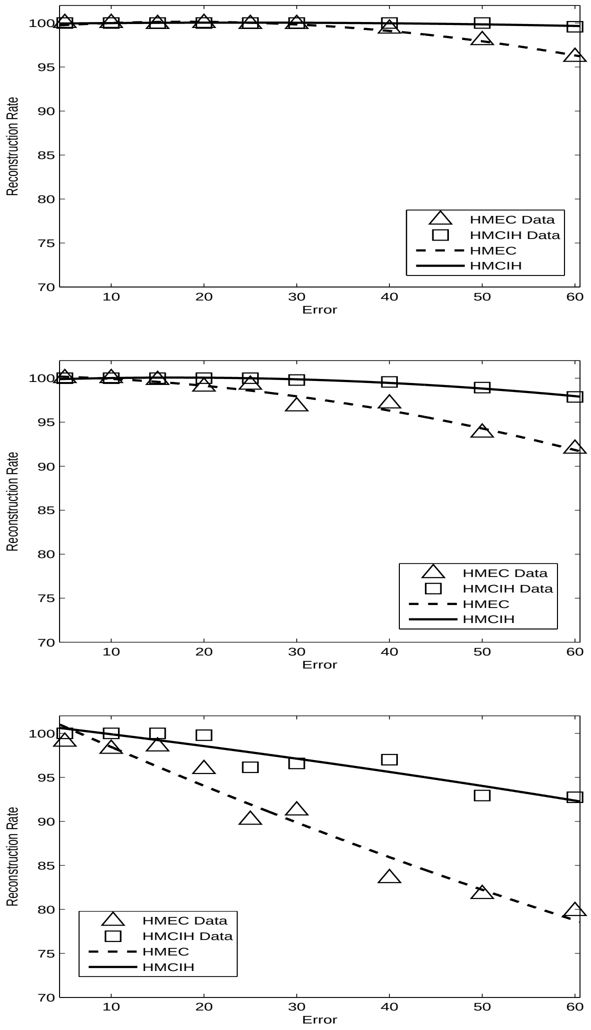

Fig. 3. The comparison of HMCIH and HMEC on ACE. From top to bottom, coverage = 75%, coverage = 50%, coverage = 25%.

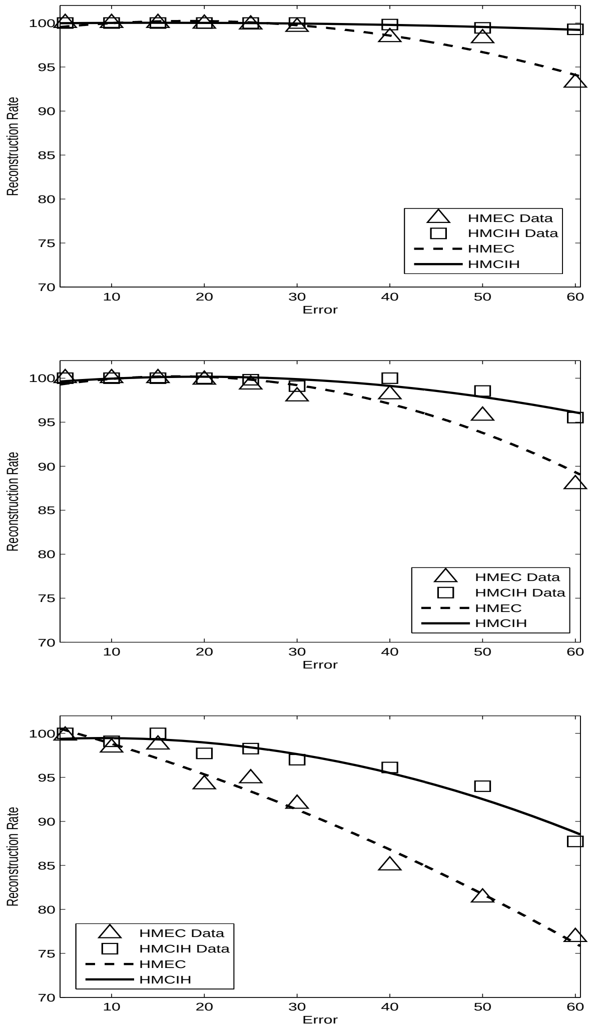

Fig. 4. The comparison of HMCIH and HMEC on Daly set. From top to bottom, coverage = 75%, coverage = 50%, coverage = 25%.

Here we follow the same simulation procedure that we followed for angiotensin-converting enzyme. Figure 4 demonstrates the result. The figure again shows that HMCIH is much better than the HMEC and the FNN algorithm. For some instances with high error and hole rate, FNN takes several minutes to stop [1], whereas HMCIH does not take more than 1 sec.

V. Conclusion

In this paper, we gave a heuristic algorithm for minimum conflict individual haplotyping (MCIH). MCIH is an extension of MEC, proposed by Zhang *et al.*, that can solve the haplotyping problem with higher reconstruction rate than the MEC with the cost of additive genotype information [1]. Considering the facts that genotypes can be achieved easily and economically, and MCIH performs much better than MEC (specially in the case of high error rate), it is fairly effective to use MCIH as an alternate way of individual haplotyping at the cost of genotype information. Since MCIH is computationally intractable, it may not have any efficient exact algorithm. Hence, we present a heuristic algorithm for MCIH (HMCIH) that performs fairly well. Since our previous heuristic algorithm for MEC (HMEC) [15] performs very well, we rely on that heuristic and customize it for MCIH by incorporating the additive genotype information. Experimental results confirms that HMCIH is much better than HMEC and the FNN based algorithm proposed by Zhang *et al.* The extra ordinary fast converging time, deterministic nature, and of course the high reconstruction rate prove the potential of HMCIH to be a practical tool for individual haplotyping.

References

[1] X. S. Zhang, R. S. Wang, L. Y. Wu and W. Zhang, *Minimum Conflict Individual Haplotyping from SNP Fragments and Related Genotype*, Evolutionary Bioinformatics, 2, pp. 261–270, 2006.

[2] V. Bafna, S. Istrail, G. Lancia and R. Rizzi, *Polynomial and APX-hard cases of the individual haplotyping problem*, Theoretical Computer Science, 335, pp. 109–125, 2005.

[3] P. Bonizzoni, G. D. Vedova, R. Dondi and J. Li, *The haplotyping problem: an overview of computational models and solutions*, Journal of Computer Science and Technology, 18(6), pp. 675–688, 2003.

[4] A. Chakravarti, *It's raining SNPs, hallelujah?*, Nature Genetics, 19, pp. 216–217, 1998.

[5] Z. Li, W. Zhou, X. Zhang and L. Chen, *A parsimonious tree-grow method for haplotype inference*, Bioinformatics, 21(17), pp. 3475–3481, 2005.

[6] J. C. Stephens and J. A. Schneider et al., *Haplotype variation and linkage disequilibrium in 313 human genes*, Science, 293, pp. 489–493, 2001.

[7] A. G. Clark, *Inference of haplotypes from PCR-amplified samples of dipoid populations*, Molecular Biology and Evolution, 7(2), pp. 111–122, 1990.

[8] D. Gusfield, *Haplotyping as perfect phylogeny: conceptual framework and efficient solutions*, Proc. of the 6th International Conference on Research in Computational Molecular Biology (RECOMB), pp. 166–175, 2002.

[9] E. Halperin and E. Eskin, *Haplotype reconstruction from genotype data using imperfect phylogeny*, Bioinformatics, 20(12), pp. 1842–1849, 2004.

[10] L. Wang and Y. Xu, *Haploptye inference by maximum parsimony.*, Bioinformatics, 19(14), pp.1773–1780, 2003.

[11] G. Lancia, V. Bafna, S. Istrail and R. Schwartz, *SNP Problems, complexity and algorithms*, Proc. of Annual European Symposium on Algorithms (ESA), Lecture Notes in Computer Science, 2161, pp. 182–193, 2001.

[12] R. Lippert, R. Schwartz, G. Lancia and S. Istrail, *Algorithmic stratagies for the SNPs haplotype assembly problem*, Breifing in Bioinformatics, 3(1), pp. 23–31, 2002.

[13] R. Rizzi, V. Bafna, S. Istrail and G. Lancia, *Practical algorithms and fixed-parameter tractability for the single individual SNP haplotyping problem*, Proc. of the 2nd International Workshop on Algorithms in Bioinformatics (WABI), pp. 29–43, 2002.

[14] R. S. Wang, L. Y. Wu, Z. P. Li and X. S. Zhang, *Haplotype reconstruction from SNP fragments by minimum error correction*, Bioinformatics, 21(10), pp. 2456–2462, 2005.

[15] A. A. Mueen, M. S. Bayzid, M. M. Alam and M. S. Rahman, *A Heuristic Algorithm for Individual Haplotyping with Minimum Error Correction*, Proc. of International Conference on Biomedical Engineering and Informatics (BMEI), IEEE Computer Society Press, pp. 792–796, 2008.

[16] M. J. Rieder, S. L. Taylor, A. G. Clark and D. A. Nickrson, *Sequence variation in the human angiotensin converting enzyme*, Nature genetics, 22, pp. 59–62, 1999.

[17] M. Daly, J. Rioux, T. Hudson and E. Lander, *High-resolution Haplotype Structure in Human Genome*, Nature Genetics, 29, pp. 229–232, 2001.