Minimum Conflict Individual Haplotyping from SNP Fragments and Related Genotype

Xiang-Sun Zhang1, Rui-Sheng Wang2, Ling-Yun Wu1 and Wei Zhang3

1Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100080, China. 2School of Information, Renmin University of China, Beijing 100872. 3North Carolina State University, Raleigh, NC 27695-7906, U.S.A.

Abstract: The Minimum Error Correction (MEC) is an important model for haplotype reconstruction from SNP fragments. However, this model is effective only when the error rate of SNP fragments is low. In this paper, we propose a new computational model called Minimum Conflict Individual Haplotyping (MCIH) as an extension to MEC. In contrast to the conventional approaches, the new model employs SNP fragment information and also related genotype information, thereby a high accurate inference can be expected. We first prove the MCIH problem to be NP-hard. To evaluate the practicality of the new model we design an exact algorithm (a dynamic programming procedure) to implement MCIH on a special data structure. The numerical experience indicates that it is fairly effective to use MCIH at the cost of related genotype information, especially in the case of SNP fragments with a high error rate. Moreover, we present a feed-forward neural network algorithm to solve MCIH for general data structure and large size instances. Numerical results on real biological data and simulation data show that the algorithm works well and MCIH is a potential alternative in individual haplotyping.

Keywords: individual haplotyping; minimum conflict individual haplotyping; NP-hard; dynamic programming; feed-forward neural network; reconstruction rate.

Introduction
The availability of complete genome sequence for human beings [Vent et al. 2001] makes it possible to investigate genetic differences and to associate genetic variations with complex diseases [Hohe et al. 2000]. Single nucleotide polymorphism (SNP)—a single DNA base varying from one individual to another, is believed to be the most frequent form to address genetic differences [Chakravarti, 1998; Li et al. 2005]. SNPs are found approximately every 1000 base pairs in the human genome and turn to be promising tools for doing disease association study. Many research works have been carried out for determining SNP sites or designing a detailed SNP map for human genome [Altshuler et al. 2000; Helmuth, 2001].

The nucleotides in a SNP position are called alleles. Almost all SNPs have two different alleles which we denote the wild type as 1 and the mutant type as -1. The SNP sequence information on each copy of a pair of chromosomes in a diploid genome is called a haplotype which is a string over {−1, 1}. A genotype is the conflated information of a pair of haplotypes on homologous chromosomes. For a genotype, if a pair of alleles at a SNP site is made of two identical values, this SNP site is called homozygous, otherwise it is called heterozygous.

Haplotypes generally have more information content than that individual SNPs have in disease association studies [Stephens et al. 2001], but it is substantially more difficult to determine haplotypes than to determine genotypes or individual SNPs through experiments. Hence, computational methods that can reduce the cost of determining haplotypes become attractive alternatives. There are generally two classes of computational methods for determining haplotype. One class concerns with inferring haplotypes from the genotype samples in a population. There are several models based on different assumptions on the biological system under consideration [Clark, 1990; Gusfield, 2002; Halperin and Eskin, 2004; Li et al. 2005; Wang and Xu, 2003]. The second class, called single individual haplotyping or haplotype assembly, is based on the data and methodology of shotgun sequence assembly [Lancia et al. 2001; Lippert et al. 2002]. The input data consists of aligned short genome fragments with SNPs coming from DNA shotgun sequencing or data generated by a resequencing effort for the purpose of large-scale haplotyping. When we focus on SNP positions, these short genome fragments are actually the aligned SNP fragments.

Correspondence: Email: wangrsh@amss.ac.cn
Computationally, the individual haplotyping problem is to determine the “best” pair of haplotypes from data (SNP fragments) which is possibly inconsistent and contradictory. It focuses on partitioning SNP fragments into two sets according to SNP states, 

with each set determining a haplotype [Lancia et al. 2001; Lippert et al. 2002]. For such a problem, there are several models based on different error assumptions [Lancia et al. 2001; Li, L.M. et al. 2004; Lippert et al. 2002; Rizzi et al. 2002; Wang et al. 2005], of which the Minimum Error-Correction model (MEC) is widely adopted. MEC assumes that the inconsistency of the data comes from realistic sequence errors and these errors can be corrected. However, MEC is only effective in the case that SNP fragments have a low error rate. When the error rate of SNP fragments is relatively high, we can not reconstruct original haplotypes with a high accuracy by error correction only [Wang et al. 2005]. To improve the haplotyping quality, we need to either reduce the errors in SNP fragments which will call for improvement of the shotgun experiment, or add extra information to the given SNP fragment set. Since genotype data can be much more easily (and also economically) obtained, a computational model combining both SNP fragments and genotype information will be a realistic strategy. The idea is motivated by the method in [Eskin et al. 2004]. In their paper they phased long genotypes using local haplotype information. In this paper, we propose a new computational model (Minimum Conflict Individual Haplotyping: MCIH) for individual haplotyping by using this strategy, which is also proved to be NP-hard.

There are two ways to show that a new established model is practical or more effective than an existing model: one is to theoretically prove that the solution of the new model is superior to that of the existing model, the other is to numerically solve the problem and compare the solutions obtained by the two models. It is obvious that we can only do it in the second way. For this reason we try to design an exact algorithm for the MCIH problem. A special data structure suited for a dynamic programming algorithm is displayed and used to evaluate the model. Computational results by this exact algorithm confirm the significance of MCIH. Moreover, a feed-forward neural network (FNN) is designed for computing approximate solutions to the problem (MCIH) in general case and of large size. Extensive computational results show the effectiveness of the proposed FNN algorithm and MCIH.

The paper is organized as follows: In Section I, we give the problem definition together with the complexity analysis of MCIH. The model evaluation is shown in Section II. In Section III, a feed-forward neural network algorithm is designed for implementing MCIH. Experiment results are given in Section IV. Section V concludes the paper.

I. Formulation and Problem

For a genotype \( g = (g_1, g_2, \cdots, g_n) \), if the \( j \) th SNP site is wild type homozygous, \( g_j = 2 \); if it is mutant type homozygous, \( g_j = -2 \); when it is heterozygous, \( g_j = 0 \). A pair of haplotypes \( h_1 = (h_{11}, h_{12}, \cdots, h_{1n}) \) and \( h_2 = (h_{21}, h_{22}, \cdots, h_{2n}) \) is called compatible with a genotype \( g \) if the following conditions hold: for each SNP site \( j \) where \( g_j = -2 \), \( h_{1j} = h_{2j} = -1 \); for each SNP site \( j \) where \( g_j = 2 \), \( h_{1j} = h_{2j} = 1 \); for each SNP site \( j \) where \( g_j = 0 \), \( h_{1j} = -h_{2j} = -1 \) or \( h_{1j} = -h_{2j} = 1 \).

Suppose that there are \( m \) SNP fragments from a pair of chromosomes and the length of each corresponding haplotype is \( n \). Define an \( m \times n \) SNP matrix \( M = (m_{ij}) \), whose entry \( m_{ij} \) has the value \(-1, 1, 0\) (for a missing or skipped base, we call it a hole). Each row \( m_i \) corresponds to a SNP fragment \( f_i \) and each column corresponds to a SNP site. Since the given SNP fragments may have different lengths, but are generally less than \( n \), we also assign value \( 0 \) to the uncovered elements in a row.

Let \( x, y \in \{-1, 1, 0\} \) and define

\[
d(x, y) = \begin{cases} 
1 & \text{if } x \neq 0, \ y \neq 0 \text{ and } x \neq y, \\
0 & \text{otherwise},
\end{cases}
\]

then the distance between two SNP fragments \( f_i = (m_{i1}, \cdots, m_{im}) \) and \( f_k = (m_{k1}, \cdots, m_{kn}) \) is defined as \( HD(f_i, f_k) = \sum_{j=1}^{m} d(m_{ij}, m_{kj}) \). If \( HD(f_i, f_k) > 0 \), we say two fragments \( f_i \) and \( f_k \) are in conflict, otherwise we call them compatible. \( HD(f_i, f_k) > 0 \) indicates that either \( f_i \), \( f_k \) are not from the same chromosome copy or there are errors in the data. \( HD(f_i, f_k) \) is similar to the Hamming distance, i.e., the number of mismatches (conflicts) between two fragments. The distance between a fragment and a haplotype is defined in the similar way.

The MEC problem [Lippert et al. 2002; Wang et al. 2005] is defined as: Given a set of SNP fragments, correct a minimum number of SNP states
(−1 into 1 and vice versa), such that the modified SNP fragments can be divided into two disjoint sets of pairwise compatible fragments, and each set determines a haplotype. In order to add the information of genotype into MEC, we propose the following combinatorial optimization model for individual haplotyping:

MCIH (Minimum Conflict Individual Haplotyping): Given a set of SNP fragments (a 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.

MCIH is in fact to correct a minimum number of SNP states under the guidance of the genotype information so that the modified SNP fragments can be divided into two disjoint sets of pairwise compatible fragments which determine a pair of haplotypes compatible with the genotype. The computational complexity of the MCIH problem (similar spirit with [Eskin et al. 2004]) is discussed in Appendix A where we prove it to be NP-hard. It indicates that the MCIH problem may have no efficient algorithm for exact solutions.

II. Model Evaluation

The purpose of presenting the new model is to get a high-quality solution of the haplotype assembly problem. To show that MCIH is a potential alternative, we evaluate the model by studying its exact solutions. A special data structure with Markov property suited for a dynamic programming algorithm is considered as follows.

Let $l_i$ and $r_i$ be the beginning and ending positions of the $i$th SNP fragments $f_i$ on the SNP matrix respectively, $1 \leq i \leq m$. For any two fragments $f_i$ and $f_j$, we assume

$$
\text{if } l_i \leq l_j, \text{ then } r_i \leq r_j.
$$

The rows of the SNP matrix is reordered according to the beginning positions and the ending positions of SNP fragments. For two rows $m_i$ and $m_j$, $i < j$ if and only if $l_i < l_j$. If $l_i = l_j$ and $r_i < r_j$, the $i$th row will also be put before the $j$th row. From the proof of the computational complexity in Appendix A, we know that the MCIH with this special structure remains NP-hard.

To solve MCIH with this special structure, we give a dynamic programming (DP) algorithm. The input data of the DP algorithm are a genotype $g$ with length $n$ and a SNP matrix of the type (2). The outputs of the DP algorithm are a partition of the SNP fragments and a pair of haplotypes ($h_1$, $h_2$) generated from the partition. In fact, the DP algorithm is used to reconstruct one haplotype, say $h_1$. Clearly, $h_2$ can be obtained immediately from $g$ and $h_1$.

Suppose that $x(j) = (x_1, \ldots, x_r)$ is an assignment to the positions $l_j, \ldots, r_j$ of $h_1$, and $\bar{x}(j)$ is the corresponding assignment to $h_2$ at the same positions. If $g_{k} \neq 0$, $x_k = \bar{x}_k = g_{k}/2$, otherwise $x_k + \bar{x}_k = 0$ for $k = l_j, \ldots, r_j$. Let $f_j(\bar{x}(j))$ denote the number of conflicts between the SNP fragment $f_j$ and $h_1$ with assignment $x(j)$, $f_\bar{x}(j, \bar{x}(j))$ denote the number of conflicts between the SNP fragment $f_j$ and $h_2$ with assignment $\bar{x}(j)$. Define $f(j, x(j)) = \min\{f_j(j, x(j)), f_\bar{x}(j, \bar{x}(j))\}$ which implies the haplotype that the $j$th fragment belongs to. The main steps of the dynamic programming algorithm are as follows:

**Step 1 Initialization.**

Let $j = 1$. $N(1, x(1)) = f(1, x(1)) = \min\{f(1, x(1)), f(1, \bar{x}(1))\}$ for all possible $x(1)$.

**Step 2 Follow the recurrence formula.**

Define $N(j, x(j))$ as the minimum associated number of conflicts between the first $j$ fragments and haplotypes ($h_1$, $h_2$) at positions 1, 2, …, $r_j$, with the assignment $x(j)$.

$$
N(j+1, x(j+1)) = \min_{(b_{j+1}, \ldots, b_{j+r_{j+1}-1})} \
\{h(j, (b_{i_1}, \ldots, b_{i_{r_{j+1}}}, x_{l_{j+1}}), x_{r_{j+1}})) \
+ f(j+1, x(j+1))\},
$$

where $b_k = g_{k}/2$ for $g_{k} \neq 0$, otherwise $b_k \in \{1, -1\}$, $k = l_{j}, \ldots, l_{j+1}-1$.

**Step 3 Trace the solution to obtain a pair of haplotypes ($h_1$, $h_2$) and a partition of the SNP fragments.**

When all $N(j, x(j))$ for $1 \leq j \leq m$ are computed by the recurrence formula, we can find $N(m, x(m))$ for all $x(m) = (x_{l_m}, \ldots, x_{r_m})$. Thus, the solution of the MCIH problem can be obtained by tracing the solution forwardly which leads to a minimal value at each $j$ from the following formula:

$$
\min_{(x_{l_m}, \ldots, x_{r_m})} N(m, x(m)).
$$
For all $j, f(j, x(j)) = \min\{ f_1(j, x(j)), f_2(j, x(j)) \}$ determines a partition of the SNP fragments.

Assume that $L$ is the maximum length of SNP fragments. It can be shown that the dynamic programming algorithm solving the special MCIH problem has the complexity $O(2^L m)$. Hence, the algorithm is exponential with the maximum length of SNP fragments. However, when the maximum length of SNP fragments is fixed, this algorithm is linear to $m$, i.e., the number of fragments. In real applications, the value of $L$ is generally between 3 and 8.

As expected, numerical results (see Appendix B for details) of DP on the special data structure display that MCIH at the cost of genotype information improves the reconstruction rate greatly at various error rates of SNP fragments. An extreme case is considered where every SNP site is heterozygous. It means that the model has no homozygous site information available from the given genotype. The numerical results show that in this case MCIH still has a higher reconstruction rate than MEC. This indicates that MCIH is not trivial, i.e., a higher reconstruction rate does not only depend on homozygous site information. Heterozygous site information has much contribution to the accuracy of the reconstructed haplotypes.

III. A Feed-Forward Neural Network Algorithm

As discussed in Introduction, the MCIH problem is actually a classification problem. That is, given a set of SNP fragments, we want to classify it into two fragment subsets such that each subset determines a haplotype to solve the problem with minimum conflicts. It is well known that feed-forward neural network (FNN) is a powerful tool for classification. In general, an FNN maps a set of objects into classes $C_1, \cdots, C_s$ characterized by the attributes of the objects through repeatedly learning the objects and adjusting its parameters (neuron connection weights). In our problem, a set of SNP fragments are to be divided into two classes such that each class can be assembled as one of the haplotypes which will solve the problem with as few conflicts as possible.

The proposed FNN (see Figure 1) consists of three layers. The $m$ input neurons represent $m$ SNP fragments. Each input neuron accepts an $n$-dimensional vector on $\{1, -1, 0\}$. Two hidden neurons represent two subsets of fragments corresponding to one pair of haplotypes. The input dimension of the hidden neuron is also $n$ and the outputs of the two neurons are a pair of tentative haplotypes. There is only one neuron in the last layer, which simply conflates the tentative haplotypes and outputs a tentative genotype. Comparing this tentative genotype with the given genotype will provide us the information to adjust the neuron connection weights by the popular back-propagation algorithm (see Appendix B and related literatures for feed-forward neural networks [Rumelhart et al. 1986; Zhang, 2000]). The main characteristic of the designed neural network is trying to achieve two objectives, i.e., compatibility and minimum number of conflicts simultaneously.

Forwarding process of FNN

The forwarding process of FNN can be stated as follows.

1) The inputs and outputs of the first layer are $m$ rows of the SNP matrix, $m_1, \cdots, m_m$, i.e., $m$ SNP fragments $f_i$, $i = 1, \cdots, m$. In fact, the neurons in the first layer are trivial identity maps $I$. 

2) There are $2m$ parameters, i.e., the weights from the first layer to the second layer,

$$W_{m \times 2} = \begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ \cdots & \cdots \\ w_{m1} & w_{m2} \end{pmatrix}.$$

3) The inputs to the second layer are $y_1 = (y_{11}, \cdots, y_{1n})$ and $y_2 = (y_{21}, \cdots, y_{2n})$, where

$$y_{ik} = \sum_{i=1}^{m} m_{ik} w_{ik} \quad l = 1, 2, \quad k = 1, \cdots, n. \quad (3)$$
Two neurons in the hidden layer have the same sigmoidal neuron function:
\[ \phi(\lambda x) = \tanh(\lambda x) = \frac{1 - e^{-2\lambda x}}{1 + e^{-2\lambda x}}. \] (4)

4) The outputs of the second layer neurons are \( h_1 = (h_{11}, \ldots, h_{1n}) \) and \( h_2 = (h_{21}, \ldots, h_{2n}) \), where
\[ h_{lk} = \phi(\lambda y_{lk}), \quad l = 1, 2, \quad k = 1, \ldots, n. \] (5)

5) The inputs to the third layer are \( h_1 \) and \( h_2 \). The neuron function of the third layer is a linear summation, i.e., the output of the third layer is
\[ z_k = h_{1k} + h_{2k}, \quad k = 1, \ldots, n. \] (6)

Learning rules
The MCIH problem is to find a pair of haplotypes \((h_1, h_2)\) to

Objective-1: minimize
\[ \sum_{f_i \in F_1} HD(h_1, f_i) + \sum_{f_i \in F_2} HD(h_2, f_i) \] (7)

for all partitions \((F_1, F_2)\) of \( f_1, \ldots, f_n \) and all possible pairs of haplotypes \((h_1, h_2)\), and

Objective-2: satisfy
\[ h_{1k} + h_{2k} = g_k, \quad k = 1, \ldots, n, \] (8)

or minimize
\[ \sum_{k=1}^{n}(h_{1k} + h_{2k} - g_k), \quad k = 1, \ldots, n, \] (9)

for all possible pairs of haplotypes \((h_1, h_2)\), where \( g = (g_1, g_2, \ldots, g_n) \) is the given genotype.

1) Learning to satisfy objective-1
The distance between \( h \) and \( m \) (i.e., \( f_i \)) is defined as
\[ HD(h, m) = \sum_{k=1}^{n} d(\text{sgn}(h_{lk}), m_{ik}), \]
where \( d \) is defined by the formula (1) and

\[ \text{sgn}(x) = \begin{cases} 1, & x \geq 0, \\ -1, & x < 0. \end{cases} \]

Classify all SNP fragments into two disjoint sets according to their distances to \( h_1 \) and \( h_2 \), i.e.,
\[ F_1 = \{ f_i : HD(h_1, m_i) < HD(h_2, m_i), i = 1, \ldots, m \}, \] (10)
\[ F_2 = \{ f_i : HD(h_2, m_i) \leq HD(h_1, m_i), i = 1, \ldots, m \}. \] (11)

For the neuron corresponding to \( h_1 \) in the second layer, the network learns to minimize the following error function between \( h_1 \) and the SNP fragments in \( F_1 \):
\[ R_{11} = \sum_{f_i \in F_1} \sum_{k=1}^{n} (h_{1k} - m_{ik})^2 |m_{ik}|. \] (12)

For the neuron corresponding to \( h_2 \), the network learns to minimize the error function between \( h_2 \) and the SNP fragments in \( F_2 \):
\[ R_{12} = \sum_{f_i \in F_2} \sum_{k=1}^{n} (h_{2k} - m_{ik})^2 |m_{ik}|. \] (13)

2) Learning to satisfy objective-2
The objective that the third layer adjusts to is to minimize the difference between the tentative genotype \( z \) and the original genotype \( g \):
\[ R_2 = \sum_{k=1}^{n} (z_k - g_k)^2. \] (14)

A Back-Propagation-like procedure
The main steps of the algorithm is as follows:

Initialization
Set parameter values \( L_1, L_2, \rho, \lambda \) and \( \varepsilon \). Randomly set the initial values of the weight matrix \( W(0) \) with entries \( w_{il} \in [0, 1] \), \( i = 1, \ldots, m, \) \( l = 1, 2, t = 0 \).

Step 1—Feed-forwarding
Input a SNP matrix. Do computation of (3), (5) and (6) to get a pair of tentative haplotypes \((h_1, h_2)\).
Step 3—Output

Substep 2.1: Obtain a fragment partition \( F_1, F_2 \) using \( \hat{h}_1, \hat{h}_2 \) according to the formulae (10) and (11).

Substep 2.2: Calculate the gradients of the error functions \( \nabla_{w_1} R_{i_1}, \nabla_{w_2} R_{i_2}, \nabla_{w_1} R_2, \nabla_{w_2} R_2 \) according to (17), (18), (19), and (20).

Substep 2.3: Update the current weight matrix \( W (t) \) by

\[
\begin{align*}
{w}_1(t + 1) &= {w}_1(t) - \rho \Delta w_1, \\
{w}_2(t + 1) &= {w}_2(t) - \rho \Delta w_2,
\end{align*}
\]

where

\[
\begin{align*}
\Delta w_1 &= L_1 \nabla_{w_1} R_{i_1} + L_2 \nabla_{w_2} R_2, \\
\Delta w_2 &= L_1 \nabla_{w_1} R_2 + L_2 \nabla_{w_2} R_2,
\end{align*}
\]

\( \rho \) is the step length along the negative gradient. \( L_1 \) and \( L_2 \) are parameters to be selected.

Substep 2.4: If \( \| W (t + 1) - W (t) \| < \varepsilon \), go to Step 3, otherwise return to Step 1.

Step 3—Output

Record a pair of haplotypes \( (\hat{h}_1, \hat{h}_2) \) and a genotype \( \hat{g} \) as the output.

The formulae to compute the derivatives in the algorithm are given in Appendix B. In contrast to DP algorithm that gives an exact solution, the algorithm based on the neural network can not ensure an exact optimal solution but is proved very efficient for large-scale problems by the numerical results in next section.

IV. Simulation and Results

In this section, we will use both real data and simulation data to test MCIH and the FNN algorithm for haplotype reconstruction. The computation is implemented on a 2.26G Hz Pentium 4 processor PC using Microsoft Visual C++ compiler 6.

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. Assume that \( h = (h_1, h_2) \) is the original pair of haplotypes, and \( \hat{h} = (\hat{h}_1, \hat{h}_2) \) is the reconstructed pair of haplotypes. We define RR as:

\[
RR(h, \hat{h}) = 1 - \frac{\min \{r_{i_1} + r_{i_2}, r_{i_2} + r_{i_1}\}}{2n},
\]

where \( r_{i_1} = HD(h_i, \hat{h}_i), i = 1, 2, j = 1, 2 \). We use compatibility rate (CR) to measure the similarity degree between the original genotype \( g \) and the reconstructed genotype \( \hat{g} \):

\[
CR(g, \hat{g}) = \frac{HD(g, \hat{g})}{n}.
\]

For a partition \( P = (F_1, F_2) \) and a pair of haplotypes \( (h_1, h_2) \), the corresponding conflict number is defined by (7).

In this paper, we set step length \( \rho = 0.05 \), \( \varepsilon = 0.01 \) and \( \lambda = 0.1 \) (in fact, the algorithm is robust with these parameters) in the algorithm. In (15) and (16), we set \( L_1 = 0.2 \) and \( L_2 = 0.8 \).

Experiment 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 systemic blood pressure. Because it has a key function in the renin-angiotensin system, many association studies have been performed with DCP1 (encode angiotensin-converting enzyme). Literature [Rieder et al. 1999] completed the genomic sequencing of the DCP1 gene from 11 individuals and reported 78 SNP sites in 22 chromosomes. Out of the 78 varying sites, 52 are non-unique polymorphic sites.

Among these 11 individuals, there are two identical genotypes. We omit one of them. In addition, we omit the genotypes with no more than one heterozygous site whose haplotypes can be inferred immediately. Now each of the 8 pairs of haplotypes is used to generate 15 instances in which SNP fragments are randomly generated according to different parameter settings: the number of SNP fragments \( m = 20 \); other parameters, the hole rate of fragments \( h_r \): 0.25, 0.5, 0.75; the error rate of fragments \( e \): 0.05, 0.1, 0.15, 0.2 and 0.25.

The results of MCIH and MEC (solved by an exact algorithm in [Wang et al. 2005]) averaged on eight individuals is illustrated in Figure 2. All the results are obtained by running the algorithms only once. When the error rate of SNP fragments is low, the neural network is robust and efficient. When the error rate of SNP fragment is high, the network may get into a plight of local minima. The genotype compatibility rate returned by the algorithm for most instances is 100% or at least larger than 98%. Only several instances with error
rate 0.25 and hole rate 0.75 have genotype compatibility rate between 94% and 96%. In addition, the neural network algorithm solves each of these instances in several seconds.

Figure 2 shows that MCIH has a much higher reconstruction rate at various parameter settings, which indicates that with additive genotype information MCIH is effective. Even if an approximate algorithm is employed, it is more effective for haplotype reconstruction than MEC.

Experiment on data from chromosome 5q31
Now we performed simulations using the data from public Daly set [Daly et al. 2001]. They 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. The haplotypes of 129 children from the trios can be inferred from the genotypes of their parents through pedigree information and the nontransmitted chromosomes as an extra 129 (pseudo) haplotype pairs. Markers for which both alleles could not be inferred are marked as missing. Among the resulting set of 258 haplotype pairs, the ones with more than 20% missing alleles are removed, which leaves us 147 haplotype pairs. Among these pairs, 18 genotypes with no more than one heterozygous site are omitted, then 129 pairs of haplotypes are left as the test set.

Each of the 129 pairs of haplotypes is used to generate 15 instances in which SNP fragments are randomly generated according to different parameter settings: \( m = 30 \); other parameters, \( hr: 0.25, 0.5, 0.75 \); \( e: 0.05, 0.1, 0.15, 0.2 \) and 0.25. The results of MCIH and MEC averaged on 129 pairs of haplotypes are illustrated in Figure 3.

The picture again shows that MCIH is quite effective and the designed back-propagation-like algorithm has a very good performance. The genotype compatibility rate returned by the algorithm for most instances is 100% or greater than 98%. Only a few instances with error rate 0.25 and hole rate 0.75 have genotype compatibility rate between 92% and 96%. In addition, the neural network algorithm solves instances with a low error rate and hole rate in several seconds. For a few instances with a high error rate and hole rate, however, it takes the algorithm several minutes to stop.

Experiments on Hudson’s simulation data
We use a well-known program ms [Hudson, 2002] that uses coalescent theory to generate a simulated...
population of haplotypes. \textit{ms} has a parameter $r$ as the recombination rate of population haplotypes. Firstly, let $r = 0$, then 20 haplotypes with 80 SNP sites are generated using \textit{ms} and form a haplotype set. Then we randomly combine two haplotypes in the haplotype set into a pair of individual haplotypes. 15 pairs of haplotypes are obtained by this way. Each of the 15 pairs of haplotypes is used to generate 15 instances according to the parameters as those in the last subsection. All of these instances form a dataset. The results of the two models averaged on 15 pairs of haplotypes are shown in Figure 4. To further evaluate MCIH, let $r = 100$ and other parameters be the same. The results of two models on this dataset are summarized in Figure 5. The genotype compatibility rate for almost all instances is 100\% or larger than 98\%.

From Figure 4 and Figure 5 we can see that MCIH is effective on haplotypes not only without recombination but also with a high recombination rate.

\section*{V. Conclusion}
Individual haplotyping is an important problem in computational biology. In this paper, we proposed a new computational model for individual haplotyping—MCIH as an improvement of MEC and as an alternative way for biologists to solve the haplotyping problem more efficiently. The model is proved to be NP-hard. To evaluate the new model, we displayed a special SNP matrix structure for which a dynamic programming algorithm can be used to solve MCIH exactly. Comparing the exact solutions of MCIH and MEC, we argue that the proposed MCIH is worth further studying. Due to the computational intractability of the MCIH problem, a feed-forward neural network is designed and a back-propagation-like procedure is formed as an efficient approximate algorithm. Computational results on multiple data sets show that the designed algorithm performs well and MCIH at the cost of additive genotype information has a higher accuracy of haplotype reconstruction than MEC, especially in the case of SNP fragments with a high error rate. Since genotype information can be obtained much easily and economically, the new model is practical as a supporting tool for reconstruction of haplotypes.
Appendix

A. The MCIH problem is NP-hard

Proof. The NP-hardness of the MCIH problem can be obtained by reduction from MAX-CUT. Let $G = (V, E)$ be an instance of MAX-CUT with a cut of at least size $r$. Then an instance of the MCIH problem is constructed as follows: Define $|E|$ SNP fragments with all of the SNP fragments having length of $|V| + |E|$. The genotype $g$ has length of $|V| + 2|E| - 1$. Let $|E| + 1, |E| + 2, \ldots, |E| + |V|$ be the set of positions on the genotype that corresponds to vertices of $G$. Define the genotype at these positions as ambiguous sites, i.e., the genotype has values of 0 at these positions. Let other positions be homozygous sites having a value of 2. Each edge $e$ corresponds to a SNP fragment $f_e$. For convenience, we label the edges as $e_1, e_2, \ldots, e_{|E|}$. The SNP fragment $f_e$ indicated by the edge $e_k$ begins at the $k$th position of the genotype. Note that all SNP fragments share the positions indicated by vertices of $G$. If $e_k = (v_p, v_q)$, then we define $f_e(v_p) = 1$, $f_e(v_q) = -1$. For other vertex positions or positions that do not cover, we let $f_e(v) = 0$, i.e., a hole site. For each non-vertex position $l$ covered by $f_e$, $f_e(l) = 1$.

We now prove that $G$ has a cut of at least size $r$ if and only if the above MCIH instance has a solution with at most $|E| - r$ conflicts. Firstly, assume that the above MCIH instance has a solution with at most $|E| - r$ conflicts. Let $(h_1, h_2)$ be a pair of haplotypes which is compatible with the genotype $g$ and has minimum number of conflicts with the given SNP fragments. We know $h_1(v) \neq h_2(v)$ for each vertex position $v$. Let $V_1 = \{v \in V \mid h_1(v) = 1\}$ and $s$ be the number of edges crossing the cut created by $V_1$. For each edge $e_k = (v_p, v_q)$, if $h_1(v_p) \neq h_1(v_q)$, then the number of conflicts between $(h_1, h_2)$ and $f_e$ is zero. If $h_1(v_p) = h_1(v_q)$, the number of conflicts is 1. So the total number of conflicts is:

$$|\{(v, v) \in E \mid h_1(v) = h_1(v)\}|$$
$$= (|E| - |\{(v, v) \in E \mid h_1(v) \neq h_1(v)\}|)$$
$$= |E| - s \leq |E| - r.$$

Hence $s \geq r$, i.e., $G$ has a cut with size at least $r$. On the other hand, if $(V_1, \overline{V_1})$ be a cut of $G$ of at least size $r$, then we define $h_1$ to have value of 1 at non-vertex positions. For a vertex position $v$, if $v \in V_1$, then $h_1(v) = 1$, else $h_1(v) = -1$. $h_2$ can be obtained from $g$ and $h_1$. We can see that such a pair of haplotypes is compatible with $g$ and has at most $|E| - r$ conflicts with the SNP fragments.

B. Evaluation MCIH on special SNP matrix—DP algorithm

In this part, we use simulation data to evaluate MCIH with an exact algorithm—dynamic programming algorithm. 100 pairs of haplotypes with 25 SNP sites are randomly generated according to a parameter similarity degree $s$ (the similarity degree between two haplotypes in one pair). Then, each of 100 pairs of haplotypes is used to generate 5 instances in which SNP fragments are randomly generated according to error rate $e$: 0.05, 0.1, 0.15, 0.2 and 0.25. The lengths of SNP fragments are between 6 and 8. The number of SNP fragments in each instance is between 20 and 40. We first let $s = 0.5$. The results of MCIH and MEC (solved by an exact algorithm in [Wang et al. 2005]) averaged on 100 pairs of haplotypes are summarized in Table 1. The dynamic programming algorithm solves each instance in no more than one second.

As expected, MCIH, which employs genotype information, raises the reconstruction rate greatly at various error rates of SNP fragments. In order to evaluate MCIH more objectively, we let $s = 0$, i.e., every SNP site in the genotypes corresponding to 100 pairs of haplotypes is a heterozygous site (this is an extreme case). Table 1 shows that in this case MCIH has a higher reconstruction rate than MEC even without employing homozygous site information. This indicates that MCIH is not trivial, i.e., a higher reconstruction rate does not only depend on homozygous site information. Heterozygous site information also has much contribution to the accuracy of the reconstructed haplotypes.

Table 1. The results of two models on simulated data.

<table>
<thead>
<tr>
<th>error rate $e$</th>
<th>MEC</th>
<th>MCIH</th>
<th>MEC</th>
<th>MCIH</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>0.941</td>
<td>1.000</td>
<td>0.965</td>
<td>0.996</td>
</tr>
<tr>
<td>0.1</td>
<td>0.904</td>
<td>0.969</td>
<td>0.950</td>
<td>0.984</td>
</tr>
<tr>
<td>0.15</td>
<td>0.863</td>
<td>0.969</td>
<td>0.890</td>
<td>0.946</td>
</tr>
<tr>
<td>0.2</td>
<td>0.786</td>
<td>0.908</td>
<td>0.834</td>
<td>0.922</td>
</tr>
<tr>
<td>0.25</td>
<td>0.763</td>
<td>0.863</td>
<td>0.766</td>
<td>0.830</td>
</tr>
</tbody>
</table>
C. Calculation of the gradients of error functions

Since \( \phi(\lambda x) = \lambda (1 - \phi^2(\lambda x)) \), the gradient of \( R_{11} \) at \( w_1 = (w_{11}, w_{21}, \ldots, w_{m1})^T \) consists of

\[
\frac{\partial R_{11}}{\partial w_{1i}} = \begin{cases} 
\sum_{k=1}^n 2[\phi(\lambda y_{1k}) - m_{a}] \cdot \lambda [1 - \phi^2(\lambda y_{1k})] m_{a} & \text{for } f_i \in F_1 \quad (17) \\
0 & \text{for } f_i \notin F_1
\end{cases}
\]

where \( i = 1, 2, \ldots, m \). The gradient of \( R_{12} \) at \( w_2 = (w_{12}, w_{22}, \ldots, w_{m2})^T \) has

\[
\frac{\partial R_{12}}{\partial w_{2i}} = \begin{cases} 
\sum_{k=1}^n 2[\phi(\lambda y_{2k}) - m_{a}] \cdot \lambda [1 - \phi^2(\lambda y_{2k})] m_{a} & \text{for } f_i \in F_2 \quad (18) \\
0 & \text{for } f_i \notin F_2
\end{cases}
\]

where \( i = 1, 2, \ldots, m \). Similarly, we have

\[
\frac{\partial R_2}{\partial w_{ii}} = \sum_{k=1}^n 2[\phi(\lambda y_{1k}) + \phi(\lambda y_{2k}) - g_{i}] \cdot \lambda (1 - \phi^2(\lambda y_{1k})) m_{a},
\]

and

\[
\frac{\partial R_2}{\partial w_{1i}} = \sum_{k=1}^n 2[\phi(\lambda y_{1k}) + \phi(\lambda y_{2k}) - g_{i}] \cdot \lambda (1 - \phi^2(\lambda y_{2k})) m_{a},
\]

where \( i = 1, 2, \ldots, m \).

Acknowledgement

This work is supported by the National Natural Science Foundation of China under grant No.10471141 and No. 60503004, and the Important Research Direction Project of CAS “Some Important Problems in Bioinformatics,” CAS, China.

The first author’s work is partially supported by the Informatics Research Center for Development of Knowledge Society Infrastructure, Graduate School of Informatics, Kyoto University, Japan. The second author is also supported by the Research Fund of School of Information, Renmin University of China.

References


