JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 12, Number 10, 2005
© Mary Ann Liebert, Inc.
Pp. 1261–1274

An Approximation Algorithm for Haplotype Inference by Maximum Parsimony

YAO-TING HUANG,[1] KUN-MAO CHAO,[1,2] and TING CHEN[3]

ABSTRACT

This paper studies haplotype inference by maximum parsimony using population data. We define the optimal haplotype inference (OHI) problem as given a set of genotypes and a set of related haplotypes, find a minimum subset of haplotypes that can resolve all the genotypes. We prove that OHI is NP-hard and can be formulated as an integer quadratic programming (IQP) problem. To solve the IQP problem, we propose an iterative semi-definite programming-based approximation algorithm, (called SDPHapInfer). We show that this algorithm finds a solution within a factor of $O(\log n)$ of the optimal solution, where n is the number of genotypes. This algorithm has been implemented and tested on a variety of simulated and biological data. In comparison with three other methods, (1) HAPAR, which was implemented based on the branching and bound algorithm, (2) HAPLOTYPER, which was implemented based on the expectation-maximization algorithm, and (3) PHASE, which combined the Gibbs sampling algorithm with an approximate coalescent prior, the experimental results indicate that SDPHapInfer and HAPLOTYPER have similar error rates. In addition, the results generated by PHASE have lower error rates on some data but higher error rates on others. The error rates of HAPAR are higher than the others on biological data. In terms of efficiency, SDPHapInfer, HAPLOTYPER, and PHASE output a solution in a stable and consistent way, and they run much faster than HAPAR when the number of genotypes becomes large.

Key words: algorithm, haplotype inference, integer quadratic programming, maximum parsimony, semi-definite programming.

1. INTRODUCTION

Correlating variations in DNA sequence with phenotypic differences has been one of the grand challenges in biology. Efforts have been made to obtain all common variants in the human population, including single nucleotide polymorphisms (SNPs), deletions, and insertions. Many SNPs have been identified, and these data are now publicly available for researchers. For example, the International HapMap Project (Helmuth, 2001), formed in 2002, aimed to characterize the patterns of linkage disequilibrium

[1]Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan.
[2]Graduate Institute of Networking and Multimedia, National Taiwan University, Taipei, Taiwan.
[3]Department of Biological Sciences, University of Southern California, Los Angeles, CA 90089.

across the human genome using SNPs such that the information can be used for large-scale genetic association studies. As a dense SNP haplotype map is being built (Daly *et al.*, 2001; Helmuth, 2001; Patil *et al.*, 2001), various methods have been proposed to use haplotype information in linkage disequilibrium mapping. Some existing statistical methods for genetic linkage analysis have also shown increased power by incorporating SNP haplotype information (Huang *et al.*, 2004; Seltman *et al.*, 2001; Zhang *et al.*, 2002, 2003). But, the use of haplotype maps has been limited due to the fact that the human genome is a *diploid* and, in practice, *genotype* data instead of *haplotype* data are collected directly, especially in large-scale sequencing projects, because of cost considerations. Although recently developed experimental techniques (Douglas *et al.*, 2001) give the hope of deriving haplotype information directly with affordable costs, efficient and accurate computational methods for haplotype reconstruction from genotype data are still in high demand.

A number of methods have been developed to infer haplotypes based on genotypes of unrelated individuals. These methods can be divided into those based on combinatorics (Bafna *et al.*, 2003; Eskin and Halperin, 2003; Gusfield, 2001, 2002, 2003; Wang and Xu, 2003) and those based on expectation-maximization (EM) algorithms or Bayesian algorithms (Excoffier and Slatin, 1995; Lin *et al.*, 2002; Niu *et al.*, 2002; Qin *et al.*, 2002; Stephens *et al.*, 2001, 2003). The statistical methods first infer haplotype frequencies and then use these frequencies to compute the haplotype configuration (or called *phase*) for each genotype. A recent study by Stephens and Donnelly (2003) compared three statistical approaches, the PL-EM algorithm (Niu *et al.*, 2002), called HAPLOTYPER, and two MCMC algorithms based on Gibbs sampling, one called PHASE (Stephens *et al.*, 2001) and another by Lin *et al.* (2002), using a variety of simulated and real genotype data. Two measures of accuracy were used: the error rate of individuals whose haplotype estimates are not completely correct, and the error rate of a single site. The results showed that both error rates of these algorithms can be as high as 50%.

On the other hand, most combinatorics-based methods consider two models. The first model is based on perfect phylogeny, assuming there is no recombination, and the other model is based on pure parsimony, assuming the number of real haplotypes is minimum. In this paper, we study the pure parsimony model. Gusfield (2003) first formulated the problem and proposed an integer linear programming algorithm to solve this problem. Wang and Xu (2003) proposed a branching and bound algorithm called HAPAR to find the optimal solution. Recently, Brown and Harrower (2004) proposed a new formulation of the problem. Lancia *et al.* (2004) proved the APX-hardness of the problem. That is, there is a constant $\lambda > 1$ such that the existence of a λ-approximation algorithm for this problem would imply P=NP. Sharan *et al.*(2005) showed that it remains APX-hard even in some very restricted cases.

In this paper, we first formulate the haplotype inference based on pure parsimony problem as an optimal haplotype inference (OHI) problem. Then the OHI problem is reformulated as an integer quadratic programming (IQP) problem. Based on the IQP problem, we propose an iterative semi-definite programming-based approximation algorithm that finds a solution within a factor of $O(\log n)$ of the optimal solution, where n is the number of genotypes. We also prove that OHI is NP-hard through a reduction from the problem of Exact Cover By 3-Sets (X3C) (Garey and Johnson, 1979). This algorithm has been implemented and tested on a variety of simulated and biological data. In comparison with three other methods, HAPAR, HAPLOTYPER, and PHASE, the experimental results indicate that this algorithm outputs solutions with high accuracy and efficiency.

2. METHOD

Problem formulation

Suppose we are given n individuals for a local chromosomal region of L linked SNPs. Let $G = \{g_1, g_2, \ldots, g_n\}$ denote the genotypes for the n individuals, where $g_i = \{g_{i1}, \ldots, g_{iL}\}$, g_{ij} denotes the genotype for individual i at locus j, and $g_{ij} = 0$, 1, or 2 denote that this locus is homozygous wild type, homozygous mutant, or heterozygous, respectively. Experimental data may have missing alleles. We let $g_{ij} = 3$, 4, or 5 to denote two missing alleles, one missing allele and one wild type, and one missing allele and one mutant.

Let $H = \{h_1, h_2, \ldots, h_m\}$ denote the set of all possible unobserved haplotypes for G. We denote $|H| = m$ to be the number of elements in a set. If two haplotypes h_r and h_t form a genotype g_i, we denote $h_r \otimes h_t = g_i$, and we also say that h_r and h_t resolve g_i, or a haplotype configuration for g_i is h_r and h_t. Let $S = \{S_1, \ldots, S_n\}$ denote the sets of unobserved haplotype configurations for G, where $S_i = \{(h_r, h_t) : h_r \otimes h_t = g_i\}$ denotes the set of all unobserved haplotype configurations for g_i. We formulate the haplotype inference by maximum parsimony as follows, which is referred to as the optimal haplotype inference (OHI) problem.

Definition. Optimal haplotype inference (OHI) problem: *Given a set of genotypes G and a polynomial-sized set of unobserved haplotypes H for G, find a minimum subset of haplotypes, $V \subseteq H$, such that for every genotype g_i, $1 \leq i \leq n$, there exists a pair of haplotypes $h_r \in V$ and $h_t \in V$ such that h_r and h_t resolve g_i (or $(h_r, h_t) \in S_i$).*

We would like to note that m (i.e., the size of H) is fixed in this paper because (1) the idea of haplotype inference is restricted to a high linkage disequilibrium (LD) region, which is usually short (Zhang *et al.*, 2002, 2004), and (2) the number of observed haplotypes in a short chromosomal region in a human population is generally small. Theoretically, a genotype in a short chromosomal region may still contain a large number of ambiguous SNPs due to factors such as missing data and thus corresponds to an exponential number of possible haplotypes. However, poor-quality genotypes of this kind do not provide enough information for haplotypes, so we do not use them for haplotype inference. If each genotype corresponds to a maximum number of K haplotypes, m is bounded by $O(nK)$.

Integer quadratic programming

Define x_i as the variable for haplotype h_i: $x_i = 1$ if $h_i \in V$, and $x_i = -1$ otherwise. Given a set of genotypes G, the OHI problem can be formulated as the following integer quadratic programming problem:

$$\begin{aligned} & & \text{Minimize} \quad & \sum_{i=1}^{m} (1 + x_i)^2/4 \\ & \mathit{IQP}(\mathrm{G}): & \text{subject to:} \quad & \sum_{(h_r,h_t)\in S_j} (1 + x_r)(1 + x_t)/4 \geq 1, \quad \forall j \in [1, n], \\ & & & x_i \in \{-1, 1\}, \quad \forall i \in [1, m]. \end{aligned} \tag{1}$$

The set $V = \{i | x_i = 1\}$ corresponds to the set of selected haplotypes. The jth inequality guarantees that genotype $g_j \in G$ can be resolved.

Semidefinite programming relaxation

Since solving this integer quadratic programming is NP-complete, we consider relaxations of IQP. We can interpret IQP as restricting x_i to be a 1-dimensional vector with unit norm. Thus, we can relax x_i into an $(m + 1)$-dimensional vector $\mathbf{y}_i$ of unit Euclidean norm. We introduce another $(m + 1)$-dimensional unit vector $\mathbf{y}_0$, and relax IQP to

$$\begin{aligned} & & \text{Minimize} \quad & \sum_{i=1}^{m} (\mathbf{y}_0 + \mathbf{y}_i)^2/4 \\ & \mathit{SDP}(\mathrm{G}): & \text{subject to:} \quad & \sum_{(h_r,h_t)\in S_j} (\mathbf{y}_0 + \mathbf{y}_r) \cdot (\mathbf{y}_0 + \mathbf{y}_t) \geq 4, \quad \forall j \in [1, n], \\ & & & |\mathbf{y}_i| = 1, \quad \forall i \in [1, m]. \end{aligned} \tag{2}$$

In fact, SDP becomes IQP if we let

$$\mathbf{y}_0 = (1, 0, \ldots, 0), \mathbf{y}_1 = (x_1, 0, \ldots, 0), \ldots, \mathbf{y}_m = (x_m, 0, \ldots, 0). \tag{3}$$

SDP can be solved by semi-definite programming. Let $Y = (\mathbf{y}_0\mathbf{y}_1 \dots \mathbf{y}_m)^T(\mathbf{y}_0\mathbf{y}_1 \dots \mathbf{y}_m)$, where $y_{ij} = \mathbf{y}_i \cdot \mathbf{y}_j$. Then Y is positive semi-definite. We reformulate SDP into the following semi-definite programming:

$$\begin{aligned} \text{Minimize} \quad & C \cdot Y \\ \text{subject to:} \quad & A_j \cdot Y \geq a_j, \quad \forall j \in [1, n], \\ & y_{ii} = 1, \\ & Y \succeq 0, \end{aligned} \tag{4}$$

where $Y \succeq 0$ means Y is symmetric positive semi-definite. The semi-definite programming is an extension of the linear programming into convex cones. An efficient algorithm for the semi-definite programming is called the interior point method. Let $OPT(SDP)$ be the optimal solution of SDP. For any given $\varepsilon > 0$, the interior point method finds a solution of value less than $OPT(SDP) + \varepsilon$ in time polynomial in the input size and $\log 1/\varepsilon$. Once an almost optimal solution Y is found, we can use an incomplete Cholesky decomposition to obtain vectors $\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_m$.

Algorithm SDPHapInfer

In the following, we introduce an algorithm that iteratively runs a semi-definite programming, finds a solution $\{\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_m\}$, and constructs a solution $\{x_0, x_1, \dots, x_m\}$ by *randomized rounding.*

```
Algorithm SDPHapInfer

1. Initialization
   (a) Let $U = G = \{g_1, \dots, g_n\}$ be the set of unresolved genotypes;
   (b) Let $V = \{\}$ be the set of selected haplotypes;
2. SDP-solving
   (a) Formulate IQP(U) and SDP(U);
   (b) Solve SDP(U), obtaining a solution $\{\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2, \dots\}$;
3. Randomized-rounding
   (a) Randomly pick two multidimensional unit vectors $\mathbf{z}_1$ and $\mathbf{z}_2$;
   (b) Set $x_0 = 1$;
   (c) Set $x_i = 1$ for $i > 0$ if $(\mathbf{z}_1 \cdot \mathbf{y}_i)(\mathbf{z}_1 \cdot \mathbf{y}_0) > 0$ and $(\mathbf{z}_2 \cdot \mathbf{y}_i)(\mathbf{z}_2 \cdot \mathbf{y}_0) > 0$, $x_i = -1$ otherwise;
   (d) Let $V = V \cup \{h_i : x_i = 1\}$;
4. Iteration
   (a) Let U be the set of the genotypes that can not be resolved by $V$.
   (b) If $|U| \neq 0$; go to Step 2;
5. Return $V$.
```

In Step 2, if a pair of haplotypes $h_r \in V$ and $h_t \notin V$ resolve $g_i \in U$, we set variable $\mathbf{y}_r = \mathbf{y}_0$ in SDP(U). Theoretically, we can run the SDP at Step 2 only once and use this result for randomized rounding for all the iterations without changing the time complexity, but practically, running the SDP for each iteration gives better solutions.

Analysis of the algorithm

We show in the following that algorithm SDPHapInfer finds a solution of $O(\log n)$-approximation of the optimal solution of the OHI problem. Let OPT_{OHI} be the optimal solution for the OHI problem solved by IQP, and let OPT_{SDP} be the optimal solution obtained from Step 2 in the algorithm. Let $W = \sum_{i=1}^{m}(1 + x_i)^2/4$ after the randomized rounding in Step 3 in Algorithm SDPHapInfer. In the following, we first show that $E(W)$ is a lower bound of OPT_{OHI} in Lemma 1. Then we show that with a constant probability, each of the n inequalities in IQP will be satisfied by the randomized rounding scheme in Lemmas 2, 3, and 4. Finally, we show that repeating this process $O(\log n)$ times, with a high probability, all the inequalities in IQP will be satisfied and the solution is an $O(\log n)$ approximation in Theorem 1.

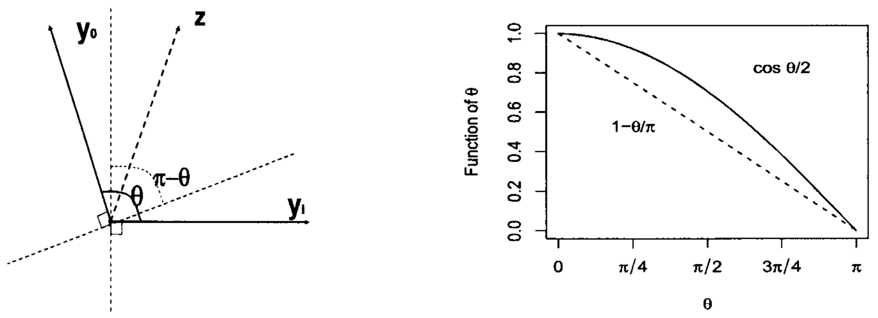

FIG. 1. Left: the case for $(\mathbf{z}\cdot\mathbf{y}_0)(\mathbf{z}\cdot\mathbf{y}_i) > 0$; **right:** comparison of two curves $\cos(\theta/2)$ and $1-\theta/\pi$.

Lemma 1.

$$OPT_{OHI} \geq OPT_{SDP} \geq E(W). \tag{5}$$

Proof. Through Equation (3), we can convert the optimal solution of OHI obtained by IQP into a feasible solution of SDP. Therefore, $OPT_{OHI} \geq OPT_{SDP}$. Let $\{\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_m\}$ be the optimal solution obtained by SDP. Let the angle between two unit vectors $\mathbf{y}_0$ and $\mathbf{y}_i$ be θ_i, $0 \leq \theta_i \leq \pi$ and $1 \leq i \leq m$. Thus, $\mathbf{y}_0 \cdot \mathbf{y}_i = \cos\theta_i$. Given two random vectors $\mathbf{z}_1$ and $\mathbf{z}_2$, the probability for $(\mathbf{z}_1 \cdot \mathbf{y}_0)(\mathbf{z}_1 \cdot \mathbf{y}_i) > 0$ and $(\mathbf{z}_2 \cdot \mathbf{y}_0)(\mathbf{z}_2 \cdot \mathbf{y}_i) > 0$ (or the probability for setting $x_i = 1$) is $((\pi - \theta_i)/\pi)^2 = (1 - \theta_i/\pi)^2$, as shown in Fig. 1. Thus,

$$E[(1+x_i)^2] = (1+1)^2 \times (1-\theta_i/\pi)^2 + (1-1)^2 \times (1-(1-\theta_i/\pi)^2) = 4(1-\theta_i/\pi)^2.$$

At the same time,

$$(\mathbf{y}_0 + \mathbf{y}_i)^2 = 2 + 2\mathbf{y}_0 \cdot \mathbf{y}_i = 2 + 2\cos\theta_i = 4\cos^2(\theta_i/2).$$

Obviously,

$$\cos(\theta_i/2) \geq 1 - \theta_i/\pi,$$

as shown in Fig. 1. Thus,

$$(\mathbf{y}_0 + \mathbf{y}_i)^2 \geq E[(1+x_i)^2],$$

$$OPT_{SDP} = \sum_{i=1}^{m} (\mathbf{y}_0 + \mathbf{y}_i)^2/4 \geq \sum_{i=1}^{m} E[(1+x_i)^2/4] = E(W). \qquad \blacksquare$$

After each iteration (Step 4) in algorithm SDPHapInfer, $E(W)$ becomes smaller and smaller. For any three vectors $\mathbf{y}_0$, $\mathbf{y}_s$, and $\mathbf{y}_t$, we let α denote the angle between $\mathbf{y}_0$ and $\mathbf{y}_s$, β denote the angle between $\mathbf{y}_0$ and $\mathbf{y}_t$, and γ denote the angle between $\mathbf{y}_s$ and $\mathbf{y}_t$, as shown in Fig. 2a. Obviously,

$$0 \leq \alpha + \beta + \gamma \leq 2\pi, \qquad \text{for } 0 \leq \alpha, \beta, \gamma \leq \pi. \tag{6}$$

Lemma 2. *After randomized rounding, the probability for $(1+x_s)(1+x_t) = 4$ is*

$$\left(1 - \frac{\alpha+\beta+\gamma}{2\pi}\right)^2,$$

and the probability for $(1+x_s)(1+x_t) = 0$ is

$$1 - \left(1 - \frac{\alpha+\beta+\gamma}{2\pi}\right)^2.$$

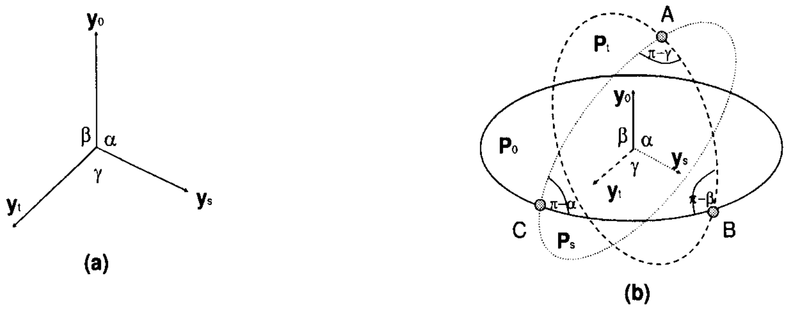

FIG. 2. (**a**) Three angles for vectors $\mathbf{y}_0$, $\mathbf{y}_s$, and $\mathbf{y}_t$; (**b**) the sphere triangle ABC formed by three planes to which $\mathbf{y}_0$, $\mathbf{y}_s$, and $\mathbf{y}_t$ are norm vectors.

Proof. After randomized rounding, $(1+x_s)(1+x_t) = 4$ or $= 0$ and nothing else, and $(1+x_s)(1+x_t) = 4$ indicates that two random vectors $\mathbf{z}_1$ and $\mathbf{z}_2$ satisfy the following two conditions: (1) $\mathbf{z}_1 \cdot \mathbf{y}_0 > 0$, $\mathbf{z}_1 \cdot \mathbf{y}_s > 0$, $\mathbf{z}_1 \cdot \mathbf{y}_t > 0$, or $\mathbf{z}_1 \cdot \mathbf{y}_0 < 0$, $\mathbf{z}_1 \cdot \mathbf{y}_s < 0$, $\mathbf{z}_1 \cdot \mathbf{y}_t < 0$; (2) $\mathbf{z}_2 \cdot \mathbf{y}_0 > 0$, $\mathbf{z}_2 \cdot \mathbf{y}_s > 0$, $\mathbf{z}_2 \cdot \mathbf{y}_t > 0$, or $\mathbf{z}_2 \cdot \mathbf{y}_0 < 0$, $\mathbf{z}_2 \cdot \mathbf{y}_s < 0$, $\mathbf{z}_2 \cdot \mathbf{y}_t < 0$. We project all unit vectors $\mathbf{y}_0$, $\mathbf{y}_s$, and $\mathbf{y}_t$ into unit vectors on a three-dimensional sphere. Let P_0 be the plane to which $\mathbf{y}_0$ is a normal vector, and similarly, we define plane P_s to $\mathbf{y}_s$ and plane P_t to $\mathbf{y}_t$. As shown in Fig. 2b, the three planes P_0, P_s, and P_t intersect the sphere and thus define a sphere triangle within which any random vector $\mathbf{z}$ (i.e., $\mathbf{z}_1$ or $\mathbf{z}_2$) satisfies $\mathbf{z} \cdot \mathbf{y}_0 > 0$, $\mathbf{z} \cdot \mathbf{y}_s > 0$, and $\mathbf{z} \cdot \mathbf{y}_t > 0$. In fact, the three surface angles of this sphere triangles are $\pi - \alpha$, $\pi - \beta$, and $\pi - \gamma$, and the surface area of the sphere triangle is equal to

$$(\pi - \alpha) + (\pi - \beta) + (\pi - \gamma) - \pi = 2\pi - \alpha - \beta - \gamma.$$

Considering the symmetric case that $\mathbf{z} \cdot \mathbf{y}_0$, $\mathbf{z} \cdot \mathbf{y}_s$, and $\mathbf{z} \cdot \mathbf{y}_t$ are all negative, we can calculate the probability of $\mathbf{z}_1$ and $\mathbf{z}_2$ for $(1 + x_s)(1 + x_t) = 4$,

$$\left(2 \times \frac{2\pi - \alpha - \beta - \gamma}{4\pi}\right)^2 = \left(1 - \frac{\alpha + \beta + \gamma}{2\pi}\right)^2,$$

and the probability of $\mathbf{z}_1$ and $\mathbf{z}_2$ for $(1 + x_s)(1 + x_t) = 0$,

$$1 - \left(1 - \frac{\alpha + \beta + \gamma}{2\pi}\right)^2. \qquad \blacksquare$$

Lemma 3. *If* $0 \le \alpha + \beta + \gamma \le 2\pi$ *and* $0 \le \alpha, \beta, \gamma \le \pi$,

$$1 - \frac{\alpha + \beta + \gamma}{2\pi} \ge \frac{1 + \cos\alpha + \cos\beta + \cos\gamma}{2\pi}. \tag{7}$$

Proof. We define a function $f(x) = x + \cos x$ for $x \in [0, \pi]$. Obviously,

$$f' = 1 - \sin x \ge 0, \quad f(0) = 1 \quad \text{and} \quad f(\pi) = \pi - 1. \tag{8}$$

Thus, f increases monotonically over $[0, \pi]$. We define another function

$$g = 2\pi - 1 - (\cos\alpha + \alpha) - (\cos\beta + \beta) - (\cos\gamma + \gamma) \tag{9}$$

for $0 \le \alpha + \beta + \gamma \le 2\pi$ and $0 \le \alpha, \beta, \gamma \le \pi$. Equation (7) holds if and only if $g \ge 0$. We first show that $g \ge 0$ for the case of $\alpha + \beta + \gamma = 2\pi$. For $\alpha + \beta + \gamma \le 2\pi$, we show $g > 0$ by increasing α, β, and γ to satisfy both $0 \le \alpha, \beta, \gamma \le \pi$ and $\alpha + \beta + \gamma = 2\pi$. In this process, g decreases because f increases monotonically over $[0, \pi]$. Combining these two cases, the lemma holds.

Assume $\alpha + \beta + \gamma = 2\pi$. Then, we replace α by $2\pi - \beta - \gamma$,

$$\begin{aligned}
g &= 2\pi - 1 - (\cos(2\pi - \beta - \gamma) + 2\pi - \beta - \gamma) - (\cos\beta + \beta) - (\cos\gamma + \gamma) \\
&= -1 - \cos(\beta + \gamma) - \cos\beta - \cos\gamma \\
&= -1 - (\cos\beta\cos\gamma - \sin\beta\sin\gamma) - \cos\beta - \cos\gamma \\
&= \sin\beta\sin\gamma - (\cos\beta + 1)(\cos\gamma + 1) \\
&= 2\sin\frac{\beta}{2}\cos\frac{\beta}{2} \times 2\sin\frac{\gamma}{2}\cos\frac{\gamma}{2} - 2\cos^2\frac{\beta}{2} \times 2\cos^2\frac{\gamma}{2} \\
&= 4\cos\frac{\beta}{2}\cos\frac{\gamma}{2}\left(\sin\frac{\beta}{2}\sin\frac{\gamma}{2} - \cos\frac{\beta}{2}\cos\frac{\gamma}{2}\right) \\
&= -4\cos\frac{\beta}{2}\cos\frac{\gamma}{2}\cos\frac{\beta+\gamma}{2} \\
&= 4\cos\frac{\beta}{2}\cos\frac{\gamma}{2}\cos\frac{\alpha}{2} \\
&\geq 0. \qquad \blacksquare
\end{aligned}$$

By Lemmas 2 and 3, the probability for $(1 + x_s)(1 + x_t) = 0$ is

$$1 - \left(1 - \frac{\alpha + \beta + \gamma}{2\pi}\right)^2 \leq 1 - \left(\frac{1 + \cos\alpha + \cos\beta + \cos\gamma}{2\pi}\right)^2. \tag{10}$$

Lemma 4. *The probability that each inequality in IQP (Equation (1)) is not satisfied after the randomized rounding of SDP is at most* $e^{-4/k\pi^2}$.

Proof. Let $(\mathbf{y}_0 + \mathbf{y}_s) \cdot (\mathbf{y}_0 + \mathbf{y}_t)$ be the ith term in the jth inequality in SDP (Equation (2)). Let k be the number of terms in the jth inequality. Let α_i, β_i, and γ_i be the angles for vectors $\mathbf{y}_0$ and $\mathbf{y}_s$, vectors $\mathbf{y}_0$ and $\mathbf{y}_t$, and vectors $\mathbf{y}_s$ and $\mathbf{y}_t$, respectively. By Lemma 2, the probability that the jth inequality in IQP (Equation (1)) is not satisfied is

$$\amalg_{i=1}^{k}\left(1 - \left(1 - \frac{\alpha_i + \beta_i + \gamma_i}{2\pi}\right)^2\right).$$

By Equation (10),

$$\begin{aligned}
\amalg_{i=1}^{k}\left(1 - \left(1 - \frac{\alpha_i + \beta_i + \gamma_i}{2\pi}\right)^2\right) &\leq \amalg_{i=1}^{k}\left(1 - \left(\frac{1 + \cos\alpha_i + \cos\beta_i + \cos\gamma_i}{2\pi}\right)^2\right) \\
&\leq \left(1 - \frac{\sum_{i=1}^{k}(1 + \cos\alpha_i + \cos\beta_i + \cos\gamma_i)^2/k}{4\pi^2}\right)^k.
\end{aligned}$$

In fact,

$$\begin{aligned}
\sum_{(h_s,h_t)\in S_j}(\mathbf{y}_0 + \mathbf{y}_s) \cdot (\mathbf{y}_0 + \mathbf{y}_t) &= \sum_{(h_s,h_t)\in S_j}(1 + \mathbf{y}_0 \cdot \mathbf{y}_t + \mathbf{y}_0 \cdot \mathbf{y}_s + \mathbf{y}_s \cdot \mathbf{y}_t) \\
&= \sum_{i=1}^{k}(1 + \cos\alpha_i + \cos\beta_i + \cos\gamma_i) \geq 4.
\end{aligned}$$

Thus,

$$\left(1 - \frac{\sum_{i=1}^{k}(1+\cos\alpha_i + \cos\beta_i + \cos\gamma_i)^2/k}{4\pi^2}\right)^k \leq \left(1 - \frac{16/k^2}{4\pi^2}\right)^k = \left(1 - \frac{4}{k^2\pi^2}\right)^k \approx e^{-4/k\pi^2}.$$

The probability that the jth inequality is not satisfied is at most $e^{-4/k\pi^2}$. ■

Theorem 1. *Algorithm SDPHapInfer gives a solution of $O(\log n)$ approximation.*

Proof. After t iterations in the algorithm, the probability that the ith inequality in IQP (Equation (1)) is not satisfied is at most $p_i = (e^{-4/k\pi^2})^t = e^{-4t/k\pi^2}$. Therefore, the probability that at least one of the inequalities in IQP is not satisfied is less than $p = \sum_{i=1}^{n} p_i = n \times e^{-4t/k\pi^2}$. Let $t = c(k\pi^2/4)\ln n$, where c is a constant. Then $p = 1/n^{c-1}$. Thus, after t iterations, the probability that all the inequalities are satisfied is at least $1 - 1/n^{c-1}$. Also, the expected total number of haplotypes selected is

$$E\left[\sum_{i=1}^{m}(1+x_i)^2/4\right] \leq t \times E(W) \leq t \times OPT_{IQP}.$$

With a high probability, algorithm SDPHapInfer stops after $O(\log n)$ iterations and finds a solution of $O(\log n)$ approximation. ■

OHI is NP-hard

Theorem 2. *The optimal haplotype inference (OHI) problem is NP-hard.*

Proof. We make a reduction from an NP-complete problem called Exact Cover By 3-Sets (X3C) (Garey and Johnson, 1979) to OHI. The X3C is defined as follows: given a set $X = \{x_1, x_2, \ldots, x_n\}$ with $n = 3q$ elements and a collection $C = \{C_1, C_2, \ldots, C_m\}$ of 3-element subsets of X, find a subcollection $C' \subseteq C$ such that every element of X occurs in exactly one member of C'.

Example. Given $X = \{1, 2, 3, 4, 5, 6\}$, $C = \{C_1 = \{1, 2, 3\}, C_2 = \{2, 3, 4\}$, and $C_3 = \{4, 5, 6\}\}$, then $C' = \{C_1, C_3\}$.

Genotype construction. Assume every subset in C is distinct. We construct genotypes and haplotypes with $m+1$ SNP loci, where the jth SNP corresponds to collection C_j. We will explain the purpose of the $(m+1)$th SNPs in the following paragraphs. In this reduction, we construct a set of genotypes $G = G_1 \cup G_2$ and a set of haplotypes $H = H_1 \cup H_2$, such that genotypes in G can be resolved by haplotypes in H.

We first design n genotypes for G_1, where each genotype g_i corresponds to x_i and the value of g_{ij} is determined by whether $x_i \in C_j$:

$$G_1 = \{g_i \mid \quad g_{ij} = 2 \text{ if } x_i \in C_j \text{ or } j = m+1, \quad \text{and } g_{ij} = 0 \text{ otherwise}, \quad i = 1, \ldots, n\}.$$

Example. For the given X and C, we construct $G_1 = \{g_1 = (2, 0, 0, 2),\ g_2 = (2, 2, 0, 2),\ g_3 = (2, 2, 0, 2),\ g_4 = (0, 2, 2, 2),\ g_5 = (0, 0, 2, 2),\ g_6 = (0, 0, 2, 2)\}$.

At the same time, we consider a specific set H_1 of m haplotypes, where each haplotype h_j corresponds to C_j:

$$H_1 = \{h_j \mid \quad h_{jj} = 1,\ h_{j\ m+1} = 1, \text{ and } \ h_{jk} = 0 \ \text{ otherwise}, \quad j = 1, \ldots, m\}.$$

Example. For the given X and C, $H_1 = \{h_1 = (1, 0, 0, 1),\ h_2 = (0, 1, 0, 1),\ h_3 = (0, 0, 1, 1)\}$.

Obviously, H_1 itself cannot resolve G_1, so we add another set of haplotypes H_2, such that $H_1 \cup H_2$ resolves G_1. Let D_i be the set of heterozygous loci (value 2) in g_i except the $(m + 1)$th SNP. Let $H_2 = H_2^{(1)} \cup H_2^{(2)} \cup \ldots \cup H_2^{(n)}$, where

$$H_2^{(i)} = \{h_j^{(i)},\ j \in D_i \mid \quad h_{jk}^{(i)} = 1 \text{ if } \ k \in D_i \ \&\ j \neq k, \quad \text{and } h_{jk}^{(i)} = 0 \text{ otherwise}\},$$

and $h_j^{(i)} \otimes h_j = g_i$.

Example. For the sample example, $D_1 = \{1\}$, $D_2 = D_3 = \{1, 2\}$, $D_4 = \{2, 3\}$, $D_5 = D_6 = \{3\}$, and $H_2 = H_2^{(1)} \cup H_2^{(2)} \cup H_2^{(3)} \cup H_2^{(4)} \cup H_2^{(5)} \cup H_2^{(6)}$, where $H_2^{(1)} = \{h_1^{(1)} = (0, 0, 0, 0)\}$, $H_2^{(2)} = \{h_1^{(2)} = (0, 1, 0, 0), h_2^{(2)} = (1, 0, 0, 0)\}$, $H_2^{(3)} = \{h_1^{(3)} = (0, 1, 0, 0), h_2^{(3)} = (1, 0, 0, 0)\}$, $H_2^{(4)} = \{h_2^{(4)} = (0, 0, 1, 0), h_3^{(4)} = (0, 1, 0, 0)\}$, $H_2^{(5)} = \{h_3^{(5)} = (0, 0, 0, 0)\}$, $H_2^{(6)} = \{h_3^{(6)} = (0, 0, 0, 0)\}$.
Clearly, $h_1 \otimes h_1^{(1)} = g_1$, $h_1 \otimes h_1^{(2)} = g_2$, $h_2 \otimes h_2^{(2)} = g_2$, etc.

To make this reduction complete, we construct the final set of genotypes G_2 where each genotype is homozygous and corresponds to one haplotype in H_2. Let $G_2 = G_2^{(1)} \cup G_2^{(2)} \cup \ldots \cup G_2^{(n)}$, where $G_2^{(i)} == H_2^{(i)}$, or

$$G_2^{(i)} = \{g_j^{(i)} == h_j^{(i)}, \quad j \in D_i \ \&\ h_j^{(i)} \in H_2^{(i)}\}.$$

Here we use "==" to indicate that a genotype is numerically equal to a haplotype.

X3C to OHI reduction. Obviously, we need H_2 to resolve G_2. The $(m + 1)$th SNPs prevent the case that a genotype can be resolved by two haplotypes from H_1 or two haplotypes from H_2. A solution C' to X3C defines a set of haplotypes $H' \subseteq H_1$, where $h_j \in H'$ if $C_j \in C'$. For any genotype $g_i \in G_1$, since for any $x_i \in X$ there exists $C_j \subseteq C'$ such that $x_i \in C_j$, we have a haplotype configuration for g_i: $h_j \otimes h_j^{(i)} = g_i$. Thus $H' \cup H_2$ resolves every genotype in G.

On the other hand, we show that $H' \cup H_2$ is the minimum set. Let $S \cup H_2$ be another haplotype configuration (solution) for G; S may contain haplotypes outside H. Let $s_i \in S$, and the value at the kth locus of s_i is 1. Since the kth SNP corresponds to a subset C_k, there are exactly three genotypes, corresponding to elements of C_k, with a nonzero value (2) at the kth locus. Then s_i can resolve at most three genotypes in G_1, and thus $|S| \geq n/3 = |H'|$. H' is optimal.

Example. Following the example above, $C' = \{C_1 \cup C_3\}$ covers X, and the corresponding haplotypes are $\{h_1, h_3\} \cup H_2$. The optimal haplotype configuration is $h_1 \otimes h_1^{(1)} = g_1$, $h_1 \otimes h_1^{(2)} = g_2$, $h_1 \otimes h_1^{(3)} = g_3$, $h_3 \otimes h_3^{(4)} = g_4$, $h_3 \otimes h_3^{(5)} = g_5$, and $h_3 \otimes h_3^{(6)} = g_6$.

In summary, OHI is NP-hard. ■

3. RESULTS AND DISCUSSION

The SDPHapInfer algorithm has been implemented in MatLab. In practice, because of numerous heterozygous SNP loci, the number of possible haplotypes could be very large. As a result, solving the semi-definite programming may be inefficient. To improve the efficiency of SDPHapInfer, we discard the haplotypes whose removal does not affect the optimal solution. For details of these methods, please refer to Gusfield (2003) and Wang and Xu (2003).

Furthermore, after obtaining a set of haplotypes, there could be more than one haplotype pair that can resolve the same genotype. As a consequence, we have to decide which haplotype pair should be assigned to the genotype. In our implementation, each haplotype is associated with a *frequency*, which is defined as the number of genotypes resolved by the haplotype. When a genotype can be resolved by multiple haplotype pairs, we first calculate the product of frequencies for each haplotype pair. Then, the haplotype pair with maximun product is assigned to the genotype. For example, suppose that the problem input consists of three genotypes (g_1, g_2, and g_3) and SDPHapInfer finds four haplotypes (h_1, h_2, h_3, and h_4). If $g_1 = h_1 \otimes h_2 = h_3 \otimes h_4$, $g_2 = h_1 \otimes h_3$, and $g_3 = h_1 \otimes h_4$, the frequencies for all haplotypes are $h_1 = 3$, $h_2 = 1$, $h_3 = 2$, and $h_4 = 2$. In this example, SDPHapInfer will output the solution $g_1 = h_3 \otimes h_4$ (instead of $h_1 \otimes h_2$) for genotype g_1.

The SDPHapInfer algorithm has been tested on a variety of simulated and biological data. The experimental results of SDPHapInfer are compared with a branching and bound algorithm called HAPAR (Wang and Xu, 2003), a PL-EM algorithm called HAPLOTYPER (Niu *et al.*, 2002), and PHASE (Stephens *et al.*, 2003) which combined the Gibbs sampling algorithm with an approximate coalescent prior.

Comparison of the number of haplotypes

We randomly generate m haplotypes with k SNPs. Based on these haplotypes, a genotype is created by randomly pairing two haplotypes. Denote n as the number of genotypes which are created in this way. SDPHapInfer, HAPAR, HAPLOTYPER, and PHASE are then applied to resolve these n genotypes. We first compare the number of haplotypes found by each algorithm under two parameter settings: (1) $m = 10$, $k = 10$, and n ranges from 5 to 25; (2) $m = 20$, $k = 10$, and n ranges from 5 to 25.

Table 1 lists the numbers of haplotypes found by each algorithm. When $m = 10$, all algorithms output similar numbers of haplotypes. When $m = 20$, the number of haplotypes found by PHASE is significantly larger than the others. This is because PHASE incorporates the coalescent model which tends to select the haplotype that is close to an included one. However, random data are less likely to have close haplotypes. Due to the lack of coalescent information, PHASE tends to use more haplotypes than do other methods.

In addition, we observe that HAPAR requires substantially longer execution time than other methods as m and n become larger. For example, when $m = 20$ and $n \geq 18$, HAPAR fails to output a solution in two hours, while other methods usually output solutions in less than 10 minutes. We also test these programs on large datasets containing a few hundreds of genotypes generated from a limited number of haplotypes (i.e., $m = 10$ and $n = 100$, 200, and 300). The experimental result indicates that all programs can still handle large datasets as long as the numbers of haplotypes and SNPs are limited.

Experiments on simulated data

We now evaluate these algorithms by the *error rate*, which is a commonly used criterion in the haplotype inference study (Stephens *et al.*, 2003; Wang and Xu, 2003). The error rate is defined as the proportion of genotypes whose original haplotype pairs are inferred incorrectly. Two kinds of simulated data, random data and Hudson's data (Hudson, 2002), are tested in our experiments.

In the experiment on random data, we adopt the first parameter setting (i.e., $m = 10$ and $k = 10$) and randomly generate 100 datasets for each parameter n (i.e., the number of genotypes). Define e_a as

TABLE 1. THE NUMBER OF HAPLOTYPES FOUND BY EACH ALGORITHM[a]

$m = 10$, $k = 10$, $n =$ 5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
SDPHapInfer 7	6	7	8	7	8	10	10	9	9	9	10	7	10	10	9	10	10	10	10	10
HAPAR 7	6	7	8	7	8	10	10	9	9	9	10	7	10	10	10	10	10	10	10	10
HAPLOTYPER 7	6	7	8	7	8	10	10	9	9	9	10	7	10	10	10	10	10	10	10	10
PHASE 10	6	11	11	7	8	15	11	16	9	10	12	7	10	10	9	10	12	10	34	24
$m = 20$, $k = 10$, $n =$ 5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
SDPHapInfer 8	8	11	9	14	13	13	15	13	15	14	15	18	18	17	16	15	15	20	17	18
HAPAR 9	8	11	9	14	13	13	14	13	*f*	14	*f*	18	*f*	*f*	*f*	*f*	*f*	*f*	*f*	*f*
HAPLOTYPER 8	8	11	9	14	13	13	15	13	14	14	14	18	18	15	16	15	15	20	17	18
PHASE 9	9	14	15	17	16	20	20	25	26	26	28	24	29	32	33	24	26	32	39	35

[a] *f*: fail to find a solution in two hours.

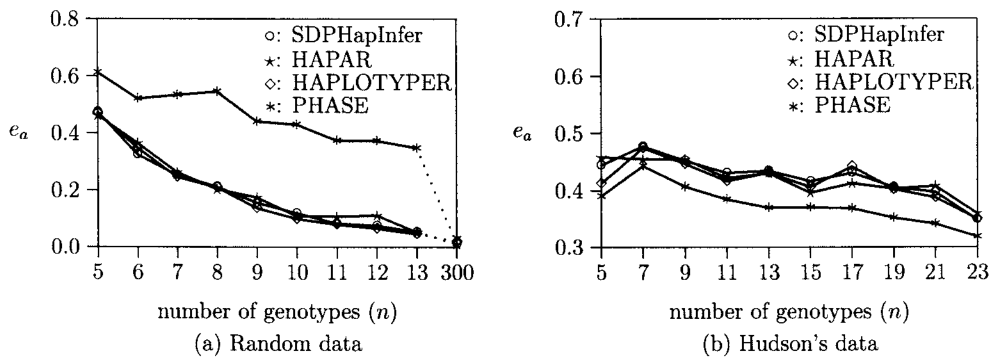

FIG. 3. The comparison of four algorithms on simulated data.

the average error rate of an algorithm over 100 datasets. Figure 3(a) plots e_a with respect to n for each algorithm. When n is small, these genotypes can be resolved by many possible combinations of haplotypes. Thus, the error rates of all algorithms are very high (Wang and Xu, 2003). The error rate for each program is significantly improved as n grows. When applying these programs on large datasets with $n \geq 100$, all error rates are very close to zero. For example, when $n = 300$, the error rates of all programs are only around 0.01.

We observe that the error rates of PHASE are higher than those of SDPHapInfer, HAPAR, and HAPLOTYPER in this experiment. This is because random data does not fit the coalescent model of PHASE. The coalescent information in random data is inadequate for PHASE to infer correct haplotypes. On the other hand, the methods based on other models perform well on random data (e.g., the error rates are less than 0.1 when $n > 12$).

In the experiment on Hudson's data, we generate coalescent haplotypes by Hudson's program (Hudson, 2002) which can simulate a set of haplotypes under the assumption of neutral evolution and a uniformly distributed recombination rate using the coalescent model. When using this program, the recombination parameter is set to 100. Similarly, 100 datasets (for each parameter n) are generated by randomly pairing these haplotypes. Figure 3(b) plots e_a with respect to n for all algorithms. In this experiment, PHASE outperforms all other methods. This is because these simulation haplotypes are generated using the coalescent method and PHASE incorporates the approximate coalescent prior in their algorithm. The error rates of all algorithms again gradually decrease as n becomes larger.

Experiments on biological data

We also test these algorithms on biological data of β_2-Adrenergic receptors (β_2ARs) from Drysdale *et al.* (2000) and cystic fibrosis from Kerem *et al.* (1989). The β_2ARs data contain 15 haplotypes with 13 SNPs, and the cystic fibrosis data consist of 28 haplotypes with 23 SNPs. We again generate 100 datasets for each parameter n by randomly pairing these haplotypes.

Figure 4(a) plots e_a with respect to n for the experiment on the β_2ARs data. The experimental result indicates that the error rates of SDPHapInfer are similar to those of HAPLOTYPER. On the other hand, PHASE slightly outperforms other methods when n is small, and the error rates are close to those of SDPHapInfer and HAPLOTYPER when n becomes large. It is a surprise to see that HAPAR is outperformed by other programs, because theoretically the solution found by SDPHapInfer is an approximation to the optimal solution found by HAPAR. We observed that sometimes HAPAR may fail to resolve all genotypes in one dataset and thus obtains higher error rates. One possible reason is that HAPAR discards many haplotypes to increase efficiency in the implementation, but it might discard critical haplotypes and then cannot resolve all genotypes.

Figure 4(b) plots e_a with respect to n for the experiment on the cystic fibrosis data. In this experiment, many genotypes cannot be resolved by HAPAR in two hours and are thus discarded. Because of insufficient data, the error rates do not decrease smoothly as in previous experiments. The performance of each algorithm does not differ much on these data.

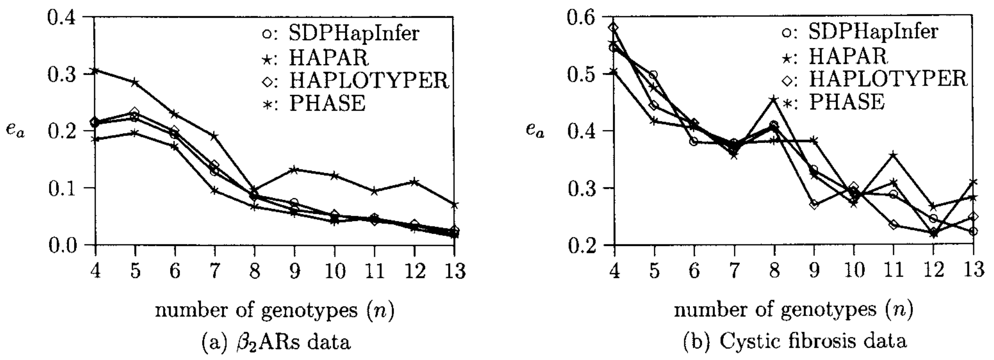

FIG. 4. The comparison of four algorithms on biological data.

Discussion and comparison

PHASE outperforms other methods in Hudson's data because they are generated using a coalescent process and PHASE uses an approximate coalescent prior in the Gibbs sampling algorithm. This is also the reason that the performance of PHASE is slightly better than that of others in biological data of β_2ARs and of cystic fibrosis, but much worse in random data.

The performance of SDPHapInfer is very similar to that of HAPLOTYPER in all experiments. One possibility is that SDPHapInfer infers haplotypes based on a criterion which mixes the maximum parsimony model with the maximum likelihood method used in HAPLOTYPER. Consider an alternative relaxation of IQP (Equation (1)) to the following quadratic programming:

$$\text{Minimize } \sum_{i=1}^{m}(1+x_i)^2/4$$

$$\text{subject to: } \sum_{(h_r,h_t)\in S_j}(1+x_r)(1+x_t)/4 \geq 1, \quad \forall j \in [1,n],$$

$$-1 \leq x_i \leq 1, \quad \forall i \in [1,m].$$

Define $w_i = (1+x_i)/2$. We can think of w_i as the weight of the ith haplotype that contributes to the resolution of a genotype. Since $0 \leq w_i \leq 1$, the meaning of w_i is similar to the ith haplotype frequency expressed in the likelihood function of HAPLOTYPER (Niu *et al.*, 2002). Note that the SDP relaxation (Equation (2)) is simply a further relaxation of IQP. Therefore, this mixed criterion implies that SDPHapInfer can find solutions as good as HAPLOTYPER.

4. CONCLUSION

In this paper, we studied haplotype inference by maximum parsimony using population data. We formulated this problem as an integer quadratic problem and proposed an iterative semi-definite programming-based approximation algorithm called SDPHapInfer. In addition, we proved that SDPHapInfer finds a solution within a factor of $O(\log n)$ of the optimal solution. The SDPHapInfer algorithm has been implemented, tested, and compared with three other methods on a variety of simulated and biological data. The experimental results indicate that the error rates of SDPHapInfer and HAPLOTYPER are similar on all kinds of data. PHASE has lower error rates than other methods on Hudson's data but has higher error rates on random data. In the experiment on the β_2ARs data, the performance of HAPAR is worse than the others. In terms of efficiency, HAPAR is the fastest of the four algorithms when the numbers of haplotypes and genotypes are small. However, the execution time of HAPAR is unstable. Especially on the datasets of larger haplotypes and genotypes, it is substantially slower than the others. On the other hand, SDPHapInfer, HAPLOTYPER, and PHASE output solutions in a stable and consistent way.

ACKNOWLEDGMENTS

We thank Hsueh-I Lu for his help in semi-definite programming. We thank Thiruvarangan Ramanaj for pointing out a mistake in Lemma 4. We also thank the referees for their valuable comments. Yao-Ting Huang and Kun-Mao Chao were supported in part by NSC grants 92-2213-E-002-059 and 93-2213-E-002-029. Ting Chen was supported in part by NIH CEGS: Implications of Haplotype Structure in the Human Genome, Grant No. P50 HG002790.

REFERENCES

Bafna, V., Gusfield, D., Lancia, G., and Yooseph, S. 2003. Haplotyping as perfect phylogeny: A direct approach. *J. Comp. Biol.* 10, 323–340.

Brown, D., and Harrower I. 2004. A new integer programming formulation for the pure parsimony problem in haplotype analysis. *Proc. WABI '04*, 254–265.

Daly, M.J., Rioux, J.D., Schaffner, S.F., Hudson, T.J., and Lander, E.S. 2001. High-resolution haplotype structure in the human genome. *Nature Genet.* 29(2), 229–232.

Douglas, J.A., Boehnke, M., Gillanders, E., Trent, J.M., and Gruber, S.B. 2001. Experimentally-derived haplotypes substantially increase the efficiency of linkage disequilibrium studies. *Nature Genet.* 28(4), 361–364.

Drysdale, C., McGraw, D., Stack, C., Stephens, J., Judson, R., *et al.* 2000. Complex promoter and coding region β_2-adrenergic receptor haplotypes alter receptor expression and predict *in vivo* responsiveness. *Proc. Natl. Acad. Sci.* 97, 10483–10488.

Eskin, E., and Halperin, E. 2003. Large scale recovery of haplotypes from genotype data using imperfect phylogeny. *Proc. RECOMB '03*, 104–113.

Excoffier, L., and Slatkin, M. 1995. Maximum-likelihood estimation of molecular haplotype frequencies in a diploid population. *Mol. Biol. Evol.* 12, 921–927.

Gabriel, S.B., Schaffner, S.F., Nguyen, H., Moore, J.M., Roy, J., Blumenstiel, B., Higgins, J., DeFelice, M., Lochner, A., Faggart, M., Liu-Cordero, S.N., Rotimi, C., Adeyemo, A., Cooper, R., Ward, R., Lander, E.S., Daly, M.J., and Altshuler, D. 2002. The structure of haplotype blocks in the human genome. *Science* 296(5576), 2225–2229.

Garey, M.R., and Johnson, D.S. 1979. *Computers and Intractability*, Freeman, New York.

Gusfield, D. 2001. Inference of haplotypes from samples of diploid populations: Complexity and algorithms. *J. Comp. Biol.* 8, 305–323.

Gusfield, D. 2002. Haplotyping as perfect phylogeny: Conceptual framework and efficient solutions. *Proc. RECOMB '02*, 166–175.

Gusfield, D. 2003. Haplotyping by pure parsimony. *Proc. CPM '03, Lecture Notes in Computer Science* 2676, 144–155.

Helmuth, L. 2001. Genome research: Map of the human genome 3.0. *Science* 293(5530), 583–585.

Huang, Y.-T., Zhang, K., Chen, T., and Chao, K.-M. 2004. Approximation algorithms for the selection of robust tag SNPs. *Proc. WABI '04*, 278–289.

Hudson, R.R. 2002. Generating samples under a Wright–Fisher neutral model of genetic variation. *Bioinformatics* 18, 337–338.

Kerem, B., Rommens, J., Buchanan, J., Markiewicz, D., Cox, T., Chakravarti, A., Buchwald, M., and Tsui, L.C. 1989. Identification of the cystic fibrosis gene: Genetic analysis. *Science* 245, 1073–1080.

Lancia, G., Pinotti, C., and Rizzi., R. 2004. Haplotyping populations by pure parsimony: Complexity of exact and approximation algorithms. *INFORMS J. Comp.* 16, 348–359.

Lin, S., Cutler, D.J., Zwick, M.E., and Chakravarti, A. 2002. Haplotype inference in random population samples. *Am. J. Human Genet.* 71, 1129–1137.

Niu, T., Qin, Z., Xu, X., and Liu, J.S. 2002. Bayesian haplotype inference for multiple linked single-nucleotide polymorphisms. *Am. J. Human Genet.* 70, 157–159.

Patil, N., Berno, A.J., Hinds, D.A., Barrett, W.A., Doshi, J.M., Hacker, C.R., Kautzer, C.R., Lee, D.H., Marjoribanks, C., McDonough, D.P., *et al.* 2001. Blocks of limited haplotype diversity revealed by high-resolution scanning of human chromosome 21. *Science* 294, 1719–1723.

Qin, Z., Niu, T., and Liu, J. 2002. Partitioning-ligation-expectation-maximization algorithm for haplotype inference with single-nucleotide polymorphisms. *Am. J. Human Genet.* 71, 1242–1247.

Seltman, H., Roeder, K., and Devlin, B. 2001. Transmission/disequilibrium test meets measured haplotype analysis: Family-based association analysis guided by evolution of haplotypes. *Am. J. Human Genet.* 68(5), 1250–1263.

Sharan, R., Halldorsson, B.V., and Istrail, S. 2005. Islands of tractability for parsimony haplotyping. *Proc. CSB '05*, to appear.

Stephens, M., and Donnelly, P. 2003. A comparison of Bayesian methods for haplotype reconstruction from population genotype data. *Am. J. Human Genet.* 73, 1162–1169.

Stephens, M., Smith, N.J., and Donnelly, P. 2001. A new statistical method for haplotype reconstruction from population data. *Am. J. Human Genet.* 68(4), 978–989.

Wang, L., and Xu, Y. 2003. Haplotype inference by maximum parsimony. *Bioinformatics* 19(14), 1773–1780.

Zhang, K., Deng, M., Chen, T., Waterman, M.S., and Sun, F. 2002. A dynamic programming algorithm for haplotype partitioning. *Proc. Natl. Acad. Sci.* 99(11), 7335–7339.

Zhang, K., Sun, F., Waterman, M.S., and Chen, T. 2003. Haplotype block partition with limited resources and applications to human chromosome 21 haplotype data. *Am. J. Human Genet.* 73, 63–73.

Zhang, K., Qin, Z.S., Liu, J.S., Chen, T., Waterman, M.S., and Sun, F. 2004. Haplotype block partitioning and tag SNP selection using genotype data and their applications to association studies. *Genome Res.* 14(5), 908–916.

Address correspondence to:
Kun-Mao Chao
Department of Computer Science
National Taiwan University
#1 Roosevelt Rd. Sec. 4
Taipei, Taiwan

E-mail: kmchao@csie.ntu.edu.tw

or to

Ting Chen
Department of Biological Sciences
University of Southern California
1042 West 36th Place, DRB 290
Los Angeles, CA 90089-1113

E-mail: tingchen@usc.edu