A Hidden Markov Technique for Haplotype Reconstruction

Pasi Rastas, Mikko Koivisto, Heikki Mannila, and Esko Ukkonen

Department of Computer Science & HIIT Basic Research Unit,
P.O. Box 68,(Gustaf Hällströmin katu 2b), FIN-00014, University of Helsinki, Finland
firstname.lastname@cs.helsinki.fi

Abstract. We give a new algorithm for the genotype phasing problem. Our solution is based on a hidden Markov model for haplotypes. The model has a uniform structure, unlike most solutions proposed so far that model recombinations using haplotype blocks. In our model, the haplotypes can be seen as a result of iterated recombinations applied on a few founder haplotypes. We find maximum likelihood model of this type by using the EM algorithm. We show how to solve the subtleties of the EM algorithm that arise when genotypes are generated using a haplotype model. We compare our method to the well-known currently available algorithms (PHASE, HAP, GERBIL) using some standard and new datasets. Our algorithm is relatively fast and gives results that are always best or second best among the methods compared.

1 Introduction

The DNA differences between individuals of the same species are typically on single nucleotide locations in which more than only one nucleotide (allele) occurs in the population. Such differences, due to point mutations, and their sites are called *single nucleotide polymorphisms* (SNPs). SNPs can be used as genetic markers that can be utilized, for example, in finding disease causing mutations. For a diploid species, when an SNP is *typed* (observed) for an individual, the following problem arises: There are two near-identical copies of each chromosome of a diploid organism, but the common techniques for SNP typing do not provide the allele information separately for each of the two copies. Instead, they just give genotype information, i.e., for each SNP an unordered pair of allele readings is found, one from each copy. The alleles coming from the *same* chromosome copy are called a haplotype, while a genotype combines alleles from the two copies. So a genotype {A, C}, {T, T}, {G, T} could result from two haplotype pairs: (ATG, CTT) and (ATT, CTG).

A genotype with two identical alleles in a site is called *homozygote*, while a genotype with two different alleles is called *heterozygote* in that site. Given a set of genotypes, the problem of finding the corresponding two haplotypes for each genotype is called *phasing* or *resolving* the genotypes. Resolving is done simultaneously for all genotypes, based on some assumptions on how the haplotypes have evolved. The first approach to resolve haplotypes was Clark's method

R. Casadio and G. Myers (Eds.): WABI 2005, LNBI 3692, pp. 140–151, 2005.

[1] based on a greedy resolution rule. Clark's method is sometimes referred to as parsimony-based, but pure parsimony was investigated later in [2]. In pure parsimony one asks for finding a smallest set of haplotypes able to resolve all the genotypes. Different probabilistic approaches, still without recombination, have been proposed by e.g. [3,4,5,6]. Yet another combinatorial method was proposed by Gusfield [7], aiming at finding a set of resolving haplotypes that admits a *perfect phylogeny*. Gusfield's method works on genotype blocks within which no recombination is allowed; the block structure has to be uncovered separately. Greenspan and Geiger [8] were able to combine block finding and haplotype resolution by using a Bayesian network model. Very recently, Kimmel and Shamir [9,10] gave another such method, with improved phasing results.

In this paper we describe an approach to the phasing problem based on looking at the haplotypes as a result of recombinations applied on some small number of underlying founder haplotypes. This can be formalized as a simple hidden Markov model. The model has transitions along each founder and between the founders. A haplotype is generated along the transition paths: at each state of the model some allele is emitted, according to the emission probability distribution of the state. Transitions are taken according to the associated distributions, the transitions between different founders (i.e., transitions with low probability) indicating recombination events.

To solve the phasing problem for a given set of genotypes, we learn a maximum likelihood hidden Markov model from the genotype data, and then for each genotype in the data we find a resolving pair of haplotypes that has the highest probability in this model. In practice we use the EM algorithm for estimating the parameters of the model and the Viterbi algorithm for finding the resolving haplotype pairs. We need to modify the standard versions of these algorithms [11], as the data does not contain haplotypes but unphased genotypes.

We have tested the method on some real datasets and compared its performance to the state-of-art phasing softwares PHASE [5] (version 2.1.1), HAP [12] (version 3.0), and GERBIL [10]. A prototype implementation of our method, called HIT (a Haplotype Inference Technique), gives results that are always best or second best among the methods compared, when the phasing accuracy is measured by using the switch distance [13]. PHASE is the strongest competitor but it is clearly slower than our method.

2 A Hidden Markov Model for Recombinant Haplotypes

We consider m SNP markers from the same chromosome, numbered $1, \ldots, m$ from left to right in the physical order of appearance along the chromosome. Let A_j be the set of possible alleles (values) of marker j. Then a *haplotype* is a sequence in $A_1 \times \ldots \times A_m$. A *genotype* is an unphased haplotype pair and can be defined as a sequence in $A'_1 \times \ldots \times A'_m$, where each $A'_j = A_j \times A_j$. A genotype g is *homozygous* at marker j, if $g_j = (x, y)$ and $x = y$, and *heterozygous* if $x \neq y$. We use the encoding $A_j = \{1, 2\}$ where 1 and 2 refer, respectively, to the most frequent and the second frequent allele of the SNP j.

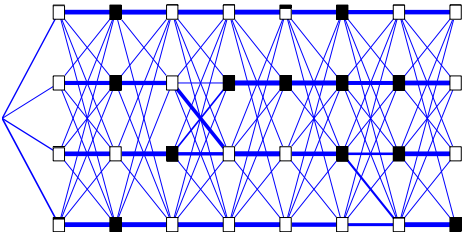

Fig. 1. An example HMM for $m = 8$ markers and $K = 4$ founders. The states are represented as boxes, the jth column of four states corresponding to the jth marker. The black area within each state encodes the emission probability of allele 1 from that state (each marker has only two alleles). The thickness of each transition line encodes the corresponding transition probability.

Our hidden Markov model (HMM) model is a pair $M = (S, \theta)$ where S is the set of states and $\theta = (\tau, \varepsilon)$ consists of the state transition probabilities τ, and the allele emission probabilities ε. The set of states $S = \{s_0\} \cup S_1 \cup \ldots \cup S_m$ consists of disjoint sets S_j, the states at marker j. The transition probabilities $\tau(s_{j-1}, s_j)$ are defined for all $s_{j-1} \in S_{j-1}$ and $s_j \in S_j$, i.e., only transitions from states in S_{j-1} to states in S_j are allowed for all $j = 1, \ldots, m$. The transition probabilities from each fixed s_j form a probability distribution, i.e., their sum equals 1. Each state $s_j \in S_j$ has a probability distribution emitting the alleles in A_j, i.e., probability $\varepsilon(s_j, a)$ of emitting $a \in A_i$. We restrict our consideration to the case that all sets S_j contain the same number K of states. The parameter K, called the *number of the founders* of M and the number m of the markers determine the topology of the HMM. The *initial state* s_0 is a dummy state from which the HMM does not emit any letter. Any path from the dummy state to a state in S_m generates a haplotype in $A_1 \times \ldots \times A_m$, with a probability determined as the product of the transition and emission probabilities along the path.

Our HMM can also handle missing data. We assume that the unobserved values are missing at random, i.e., the fact that a value is unobserved provides no information about the underlying allele; if a data point is missing, the probability of emitting is considered to be 1.

The connection to the idea of the founder sequences is as follows: The current haplotype sequences are seen as results of iterated recombinations on the haplotypes of some ancient founder population whose offspring the observed population is. The current sequences should therefore be built of fragments of the founder sequences, and some such preserved fragments should be seen in several current sequences. Our model M represents the current sequences, based on K founders. A high transition probability $\tau(s_{j-1}, s_j)$ suggests that states s_{j-1} and s_j refer to the same haplotype, i.e., there is a conserved piece of some founder. A low transition probability suggests a cross-over (recombination) between the two states.

A HMM with similar topology appears in [14,15]. Our HMM can also be seen as a probabilistic generalization of the combinatorial approach of [16] to parse haplotypes with respect An example of our model is given in Figure 1.

3 HMM Estimation and Haplotype Reconstruction

In this section we show how, given a set of unphased genotypes, the popular expectation-maximization algorithm can be efficiently applied to maximum likelihood estimation of the parameters of the hidden Markov model. We also show how the estimated HMM can be used for haplotype reconstruction.

3.1 The EM Algorithm for Maximum Likelihood Estimation

We use the maximum likelihood principle to fit our hidden Markov model to the observed genotype data G we want to phase. That is, for a fixed number of founders, we search for the parameters $\theta = (\tau, \varepsilon)$ so as to maximize the likelihood $P(G \mid \theta)$. This estimation problem is known to be hard in general HMMs, and this seems to be the case also in our application. Therefore we resort to the commonly adopted family of expectation-maximization (EM) algorithms, which are guaranteed to converge to a *local* optimum [17].

The generic EM algorithm approaches an intractable optimization problem by completing the original data with auxiliary hidden data. Then the expected log-likelihood of the complete data – where the expectation is with respect to the distribution of the hidden data given the current parameter values – is maximized in an iterative fashion. Usually the choice of the hidden data is natural and direct from the problem. For the standard HMMs the hidden data contains the unobserved hidden states.

In our case it is natural to treat the hidden state sequences, two per genotype, as the hidden data. This is, in essence, the choice that has been made in a number of related applications of EM to the haplotype reconstruction problem; e.g., [8,9,10]. While this approach works nicely when a state is deterministically related to an allele, computational problems will arise as soon as emission parameters are included in the model [9]. In such a case Kimmel and Shamir [9,10] use a (multivariate) numerical maximization routine within each EM iteration.

We propose an alternative instantiation of the EM algorithm that yields efficient closed-form expressions for the maximizing parameter values within each EM iteration. The idea is simple: in the hidden data we include not only the hidden states but also indicators which for any pair of states and the corresponding observed pair of alleles determine which one of the two states emitted the first allele in the pair, the second allele being emitted by the other state. We next provide some technical details.

3.2 Hidden Data and the Maximization Step

Let $G = \{g_1, \ldots, g_n\}$ be a set of n genotypes over m markers. We suppose the topology (the state space S) of our HMM $M = (S, \theta)$ is fixed and we wish to find parameter values $\theta = (\tau, \epsilon)$ that maximize the probability of the genotype data, $P(G \mid \theta)$.

In this setting, the EM algorithm is as follows. Starting from some initial values $\theta^{(0)}$ the algorithm iteratively improves the current values $\theta^{(r)}$ by setting

$$\theta^{(r+1)} := \arg\max_{\theta} \sum_{Z} P(Z \,|\, G, \theta^{(r)}) \ln P(G, Z \,|\, \theta)\,, \tag{1}$$

where Z runs through a chosen set of additional (hidden) data. In words, the new parameter values are obtained by maximizing the expected log-likelihood of the complete data. For a large enough r the increment in the likelihood becomes negligible and the algorithm terminates.

We choose the hidden data Z such that the complete likelihood $P(G, Z \,|\, \theta)$ factorizes into a product of individual transition and emission probabilities, as described below. This is the key to obtain a computationally efficient evaluation of (1). Recall that our HMM $M = (S, \theta)$ defines a probability distribution over singleton haplotypes. A genotype is obtained as a pair of two independent haplotypes, each generated by M along a path through some m states of M. From this generative model we extract the hidden data Z as the the combination of (a) the two state sequences per observed genotype and (b) the alleles emitted from the states.

The paths are given by an $n \times m \times 2$ matrix $T = (t_{ijk})$ of states of M. The entry $t_{ijk} \in S_j$ gives the state from which the jth allele for the first ($k = 1$) or the second ($k = 2$) haplotype for building g_i is to be emitted. The emitted allele from the possible alternatives that are consistent with g_i is indicated by an $n \times m \times 2$ matrix $U = (u_{ijk})$. The entries of U are selector variables that take values in $\{1, 2\}$. Recall that g_i consists of observed genotypes $g_{i1}, \ldots, g_{im}$ over the m markers, each genotype being a pair $g_{ij} = (g_{ij1}, g_{ij2})$ of alleles; note that we do not know which of the two alleles comes from which of the two underlying haplotypes. Here we only have arbitrarily fixed the order of the two observations. Element u_{ijk} of U specifies the jth allele of the first ($k = 1$) or of the second ($k = 2$) haplotype for building g_i: if $u_{ijk} = 1$ then the allele is g_{ij1} and if $u_{ijk} = 2$ then the allele is g_{ij2}. Both alleles must always be used, so we require that $\{u_{ij1}, u_{ij2}\} = \{1, 2\}$.

The point in introducing the hidden data $Z = (T, U)$ is that the complete likelihood factorizes into

$$P(G, T, U \,|\, \theta) = \Big(\frac{1}{2}\Big)^n \prod_{i=1}^{n} \prod_{j=1}^{m} \prod_{k=1,2} \tau(t_{i(j-1)k}, t_{ijk}) \varepsilon(t_{ijk}, g_{iju_{ijk}})\,.$$

Here the coefficient $(1/2)^n$ appears, since all the 2^n values for U are a priori equally likely (independently of θ). Thus, the expected log-likelihood is

$$\sum_{T,U} P(T, U \,|\, G, \theta^{(r)}) \ln P(G, T, U \,|\, \theta) = \sum_{j=1}^{m} A_j(\tau) + \sum_{j=1}^{m} B_j(\varepsilon) - n \ln 2\,,$$

where

$$A_j(\tau) = \sum_{i=1}^{n} \sum_{k=1,2} \sum_{T,U} P(T, U \,|\, G, \theta^{(r)}) \ln \tau(t_{i(j-1)k}, t_{ijk})\,,$$

$$B_j(\varepsilon) = \sum_{i=1}^{n} \sum_{k=1,2} \sum_{T,U} P(T, U \,|\, G, \theta^{(r)}) \ln \varepsilon(t_{ij}, g_{iju_{ijk}}) \,.$$

Furthermore, each A_j only depends on the transition probability parameters for transitions from a state in S_{j-1} to a state in S_j. Similarly B_j only depends on the emission probability parameters for states in S_j. Thus, the maximizing parameter values can be found separately for each A_j and B_j.

Standard techniques of constrained optimization (e.g., the general Lagrange multiplier method [18] or the more special Kullback–Leibler divergence minimization approach [17]) now apply. For the transition probabilities $\tau(a, b)$, with $a \in S_{j-1}, b \in S_j$, we obtain the update equation

$$\tau^{(r+1)}(a, b) = c \sum_{i=1}^{n} \sum_{k=1,2} P(t_{i(j-1)k} = a, t_{ijk} = b \,|\, G, \theta^{(r)}) \,, \tag{2}$$

where c is the normalization constant of the distribution $\tau^{(r+1)}(a, \cdot)$. That is, $\tau^{(r+1)}(a, b)$ is proportional to the expected number of transitions from a to b. Note that the hidden data U plays no role in this expression. Similarly, for the emission probabilities $\varepsilon(b, y)$, with $b \in S_j, y \in A_j$, we obtain

$$\varepsilon^{(r+1)}(b, y) = c \sum_{i=1}^{n} \sum_{k=1,2} P(t_{ijk} = b, g_{iju_{ijk}} = y \,|\, G, \theta^{(r)}) \,, \tag{3}$$

where c is the normalization constant of the distribution $\varepsilon^{(r+1)}(b, \cdot)$. That is, $\varepsilon^{(r+1)}(b, y)$ is proportional to the expected number of emissions from b to y. Note that the variable u_{ijk} is free meaning that the expectation is over both its possible values.

3.3 Computation of the Maximization Step

We next show how the well-known forward–backward algorithm of hidden Markov Models [11] can be adapted to evaluation of the update formulas (2) and (3).

Let a_j and b_j be states in S_j. For a genotype $g_i \in G$, let $L(a_j, b_j)$ denote the (left or backward) probability of emitting the initial segment $g_{i1} \dots g_{i(j-1)}$ and ending at (a_j, b_j) along the pairs of paths of M that start from s_0. It can be shown that

$$\begin{aligned} L(a_0, b_0) &= 1 \qquad \text{and} \\ L(a_{j+1}, b_{j+1}) &= \sum_{a_j, b_j} P(g_{ij} \,|\, a_j, b_j, \varepsilon) L(a_j, b_j) \tau(a_j, a_{j+1}) \tau(b_j, b_{j+1}) \end{aligned} \tag{4}$$

where

$$P(g_{ij} \,|\, a_j, b_j, \varepsilon) = \frac{1}{2} \varepsilon(a_j, g_{ij1}) \varepsilon(b_j, g_{ij2}) + \frac{1}{2} \varepsilon(a_j, g_{ij2}) \varepsilon(b_j, g_{ij1}) \,.$$

(Recall that here we treat g_i as an ordered pair, though the ordering of the alleles is arbitrary.) Then the probability of the genotype g_i is obtained as $P(g_i \,|\, \theta) = \sum_{a_m, b_m} L(a_m, b_m) P(g_{im} \,|\, a_m, b_m, \varepsilon)$ and the probability of the entire data set is $P(G|\theta) = \prod_{g_i \in G} P(g_i \,|\, \theta)$. Note that for each g_i we have its own $L(\cdot, \cdot)$.

Direct evaluation of (4) would use $O(|G| \sum_j |S_j|^4) = O(nmK^4)$ time in total. By noting that

$$L(a_{j+1}, b_{j+1}) = \sum_{a_j} \tau(a_j, a_{j+1}) \sum_{b_j} L(a_j, b_j) P(g_{ij} \,|\, a_j, b_j, \varepsilon) \tau(b_j, b_{j+1})$$

and by storing the sum $\sum_{b_j} L(a_j, b_j)\tau(b_j, b_{j+1})$ for each a_j and b_{j+1} the running time reduces to $O(nmK^3)$. The space requirement is $O(mK^2)$.

We call $L(\cdot, \cdot)$ the forward (or left) table. Similarly, we define the backward (or right) table $R(\cdot, \cdot)$. For a genotype $g_i \in G$, let $L(a_j, b_j)$ denote the probability of emitting the end segment $g_{i(j+1)} \dots g_{im}$ along the pairs of paths of M that visit (a_j, b_j).

We are now ready to show how formulas (2) and (3) can be evaluated. We consider the latter formula; the former is handled similarly. First notice that it is sufficient to consider evaluation of

$$P(t_{ijk} = b, g_{iju_{ijk}} = y \,|\, G, \theta^{(r)}) = P(t_{ijk} = b, g_{iju_{ijk}} = y, g_i \,|\, \theta^{(r)}) \Big/ P(g_i \mid \theta^{(r)}) \,.$$

We already described a way to compute the denominator. The numerator can be written as

$$\sum_{a_j} \sum_{u_{ijk}=1,2} I(g_{iju_{ijk}} = y) \frac{1}{2} L(a_j, b) \varepsilon(a_j, g_{iju_{ijk}}) \varepsilon(b, g_{ij(3-u_{ijk})}) R(a_j, b) \,,$$

where $I(\cdot)$ is the 0, 1-valued indicator function. Note that both u_{ijk} and $3 - u_{ijk}$ take values in $\{1, 2\}$. For update (3) a similar forward–backward expression is found. Thus, the total time complexity of an EM iteration is the above given $O(nmK^3)$.

3.4 Initialization and Model Training

As the EM algorithm is guaranteed to find only a local optimum, it is important to find a good initial configuration of the model parameters. Our initialization routine greedily finds a promising region in the parameter space. It consists of three steps.

First, we fix the transition probabilities and emission probabilities without looking at the data, as follows. Let $s_{j1}, \dots, s_{jK}$ be the states in S_j. For the first transition we set $\tau(s_0, s_{1l}) = 1/K$ for $l = 1, \dots, K$. Then for each $j = 1, \dots, m$, we set $\tau(s_{(j-1)l}, s_{1l'})$ to $1 - \rho$, if $l = l'$, and to $\rho/(K-1)$ otherwise. The emission probabilities for $s_j \in S_j$ are initialized by setting $\varepsilon(s_j, b) = 1 - \nu$ for a selected major allele b specific to s_j, and $\varepsilon(s_j, a) = \nu/(|A_j| - 1)$ for the other alleles $a \neq b$.

Second, we select the major alleles in a greedy manner based on the observed data. We traverse the sets S_j from left to right and assign to the states in S_j the

major alleles that locally maximize the likelihood of the initial segments of G up to marker j. This is done by simply trying all $|A_j|^K$ possible choices. Using dynamic programming the pass takes time $O(nmK^3 2^K)$ for SNP markers. We then make another pass from left to right and again choose the locally optimal major alleles but now in the context of the current solution on both sides of S_j.

Finally, the probability distributions are perturbed a bit by multiplying each parameter value by e^X, where X is drawn uniformly from $[-\eta, \eta]$, independently for each parameter, and η is a noise parameter. The perturbed distributions are obtained by normalizing the perturbed values. The constants ρ, ν, and η are specified by the user. In our tests, reported in Section 4, we used $\rho = 0.1$, $\nu = 0.01$, and $\eta = 0.8$.

Starting from the perturbed initial model, we then apply the EM algorithm to find a maximum likelihood HMM for the genotype data G. In practice, we repeat this training scheme several times, and then pick the highest likelihood HMM as the final model from which the haplotypes for G are read, as will be described in Section 3.5.

Another parameter to be fixed is the number of founders, K. Our experiments show that too small a K gives poor results, but as long as K is sufficiently large (for our test data typically K should be at least 5) varying K has a rather small effect on the quality of haplotyping result.

3.5 Reading the Haplotypes from a HMM

We reconstruct from a trained HMM $M = (S, \theta)$ the haplotypes of each $g \in G$ as follows. First we find for g the Viterbi path from M, that is, a pair (p, p') of paths through M such that emitting g from (p, p') has the highest probability among all path pairs (q, q'), i.e.,

$$P(g, p, p' \,|\, \theta) = \max_{(q,q')} P(g, q, q' \,|\, \theta) \,.$$

This can be done by a variant of (4) followed by standard trace-back. Then generate from p a haplotype h and from p' a haplotype h' such that they together give genotype g and $P(h \mid p, \theta)P(h' \mid p', \theta)$ is largest possible. This is simple local maximization at heterozygous markers of g. Haplotypes $\{h, h'\}$ are the reconstructed haplotypes for g according to our method.

4 Test Results

We have implemented the presented phasing method in a prototype program HIT (Haplotype Inference Technique). In this section, we report the results we have obtained on a few real datasets. We compare the performance of HIT against HAP version 3.0 [12], PHASE version 2.1.1[5], and GERBIL [10].

4.1 Datasets

We tested our method on five real datasets. Daly's et al. [19] commonly used benchmark dataset is a sample from a European-derived population and spans

a 500-kb region on human chromosome 5q31 which contains a genetic risk factor for Crohn disease. From that area there are genotypes for 103 SNP markers, collected from 129 trios (of mother, father, and child). The trios were used to infer the true haplotypes for the 129 genotypes of the children.

The second dataset is a fragment of the data provided recently by Hinds et al. [20]. This data consists of 71 haplotype pairs over 1,586,383 SNPs that cover the entire human genome. We took the haplotypes for the SNPs 1,000–1,199 of chromosome 2. We notice that these haplotypes were inferred from genotypes with some version of HAP [12].

The rest three datasets are genotype samples over 68 SNP markers from three datasets from Finland [21]. We call these datasets Population1 (32 haplotypes), Population2 (108 haplotypes), and Population3 (108 haplotypes).

The latter four datasets were available in a haplotyped form. For our tests we constructed the input genotypes simply by merging the two haplotypes of each individual.

4.2 Switch Distance

We measure the quality of haplotyping results using the commonly adopted switch distance [13]. Switch distance between two pairs of haplotypes $\{h, h'\}$ and $\{f, f'\}$ is the minimum number of phase shifts needed to turn $\{h, h'\}$ into $\{f, f'\}$. For example if the true haplotypes for a genotype are $\{111111, 222222\}$, then the switch distance is 1 to $\{111222, 222111\}$ and 5 to $\{121212, 212121\}$.

Unfortunately, the basic switch distance is undefined when no phase shifts can transform a haplotype pair into another pair. This may happen when the data has missing values or genotyping errors. For example, suppose the observed genotype is $\{1,2\}\{1,2\}\{-,-\}\{1,2\}\{1,2\}$, where a "$-$" stands for a missing allele. Then it is possible that our model gives haplotypes $\{112111, 222222\}$, thus imputing the missing values. However, if the underlying true pair of haplotypes is $\{111111, 222222\}$, the distance between these haplotype pairs is not defined.

In such a situation one needs a generalized switch distance. Define $errors(\{h, h'\}, \{f, f'\})$ as the minimum number of allele substitutions to $\{f, f'\}$ that are needed to make the switch distance defined, and let J be the markers where no changes are needed. Then our generalized switch distance is defined as $sd'(\{h, h'\}, \{f, f'\}) = errors(\{h, h'\}, \{f, f'\}) + sd_J((\{h, h'\}, \{f, f'\})$ where sd_J is switch distance restricted on J.

Another possibility, used by some authors [13,22], is just to ignore inconsistent markers and report the basic switch distance on the remaining markers; denote this distance by sd''. In our tests, we needed sd' and sd'' only for the Daly et al. dataset [19].

The relative versions of these distances are obtained by dividing by the total number of heterozygote sites of the genotypes minus the number of genotypes.

4.3 Comparison Results

Figure 2 shows how the performance of HIT depends on the number of founders. We see that increasing the number of founders consistently increases the good-

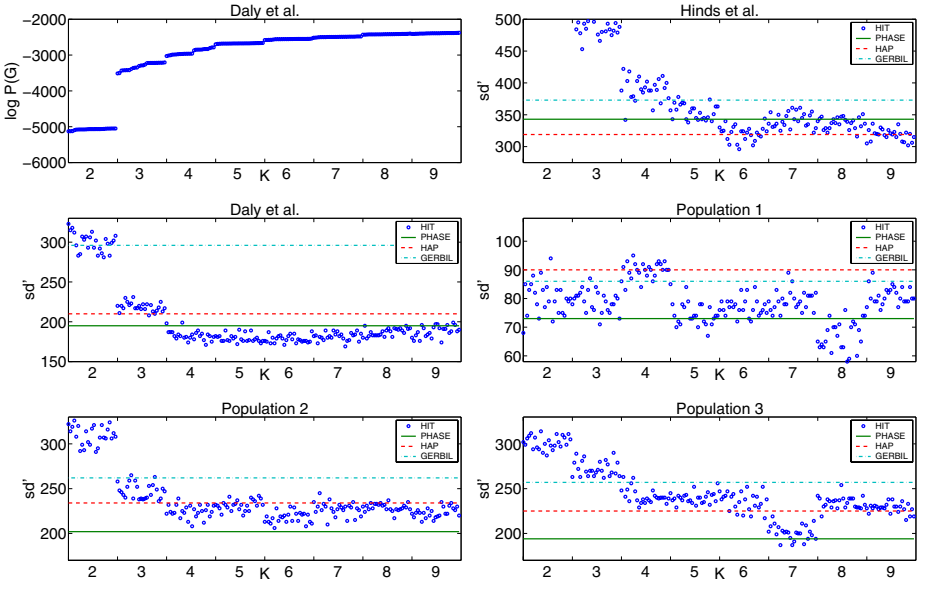

Fig. 2. The phasing accuracy (vertical axis) as a function of the number of founders (horizontal axis) for five real data sets. Shown the achieved total switch distance for 25 random restarts of HIT (in increasing order of likelihood), for 2 to 9 founders. For the Daly et al. data also shown the growth of likelihood (top left); for the other datasets the curves behave similarly (not shown). The results for PHASE [5,22], GERBIL [9,10], and HAP [12], shown as vertical lines, were obtained with their default parameter values.

ness of fit to the data, as expected. However, overfitting does not seem to impede the performance of HIT in phasing. For example, for the Daly et al. data HIT gives the best result consistently for $K \geq 4$.

The effect of starting the EM algorithm from slightly different initial settings is also shown in Figure 2, indicating a fairly robust behaviour. We note that the correlation of the achieved data likelihood and switch distance is surprisingly small. Thus the plain likelihood is perhaps not the best criterion for choosing the model for haplotyping.

Table 1 summarizes the phasing accuracy of HIT when we set the number of founders K to 7, performed 25 restarts of the EM algorithm, and used the highest likelihood HMM for haplotyping. We note that in the Daly et al. dataset the handling of the missing data (sd' or sd'') has a clear effect on the results, yet the relative differences are about the same for both measures. The fact that GERBIL treats an allele pair where one allele is missing as completely missing data, explains its relatively poor performance w.r.t. sd'' on the Daly et al. dataset. We note that HIT always gives the best or second best result.

Table 2 displays the running times of the compared methods. Clearly, GERBIL and HAP are very fast, whereas PHASE becomes rather slow for the largest

Table 1. Phasing accuracy of HIT ($K = 7$ founders), PHASE [5,22], GERBIL [9,10], and HAP [12] on five real data sets, measured using switch distance. For the Daly et al. dataset the first and the second line show switch distance sd'' and sd', respectively; for the other cases all variants coincide. The relative distances are given in parentheses

Dataset	HIT	PHASE	GERBIL	HAP
Daly et al.	**80 (0.021)**	*86 (0.023)*	*86 (0.023)*	89 (0.024)
Daly et al.	**185 (0.049)**	*195 (0.052)*	296 (0.079)	210 (0.056)
Hinds et al.	*329 (0.093)*	343 (0.097)	373 (0.11)	**319 (0.090)**
Population1	*82 (0.24)*	**73 (0.21)**	86 (0.25)	90 (0.26)
Population2	*219 (0.17)*	**202 (0.15)**	262 (0.20)	234 (0.18)
Population3	**194 (0.16)**	**194 (0.16)**	257 (0.22)	225 (0.19)

Table 2. The running time in seconds for HIT ($K = 7$ founders, median over 25 EM restarts), PHASE [5,22], GERBIL [9,10], and HAP [12] on five real data sets. HAP was run on its own server, the other programs on a Pentium IV 3.0 GHz with 1 GB of RAM.

Dataset	HIT	PHASE	GERBIL	HAP
Daly et al.	126	9290	45	25
Hinds et al.	88	71100	52	29
Population1	9	773	19	2
Population2	28	5180	89	7
Population3	29	4520	10	7

datasets. The speed of HIT (Java implementation) per EM restart is comparable to the speed of GERBIL and HAP, but slower when tens of restarts are used. Yet, HIT scales nicely to large datasets, opposite to PHASE.

References

1. Clark, A.G.: Inference of haplotypes from PCR-amplified samples of dipoid populations. Molecular Biology and Evolution **7** (1990) 111–122
2. Gusfield, D.: Haplotype inference by pure parsimony. Technical Report CSE-2003-2, Department of Computer Science, University of California (2003)
3. Excoffier, L., Slatkin, M.: Maximum-likelihood estimation of molecular haplotype frequencies in a diploid population. Molecular Biology and Evolution **12** (1995) 921–927
4. Long, J.C., Williams, R.C., Urbanek, M.: An E-M algorithm and testing strategy for multiple-locus haplotypes. American Journal of Human genetics **56** (1995) 799–810
5. Stephens, M., Smith, N., Donnelly, P.: A new statistical method for haplotype reconstruction from population data. American Journal of Human Genetics **68** (2001) 978–989
6. Niu, T., Qin, Z., Xu, X., Liu, J.: Bayesian haplotype inference for multiple linked single nucleotide polymorphisms. American Journal of Human Genetics **70** (2002) 157–169

7. Gusfield, D.: Haplotyping as perfect phylogeny: conceptual framework and efficient solutions. In: Research in Computational Molecular Biology (RECOMB '02), ACM Press (2002) 166–175
8. Greenspan, G., Geiger, D.: Model-based inference of haplotype block variation. In: Research in Computational Molecular Biology (RECOMB '03), ACM Press (2003) 131–137
9. Kimmel, G., Shamir, R.: Maximum likelihood resolution of multi-block genotypes. In: Research in Computational Molecular Biology (RECOMB '04), ACM Press (2004) 2–9
10. Kimmel, G., Shamir, R.: Genotype resolution and block identification using likelihood. Proceeding of the National Academy of Sciences of the United States of America (PNAS) **102** (2005) 158–162
11. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE **77** (1989) 257–285
12. Halperin, E., Eskin, E.: Haplotype reconstruction from genotype data using imperfect phylogeny. Bioinformatics **20** (2004) 104–113
13. Lin, S., Cutler, D.J., Zwick, M.E., Chakravarti, A.: Haplotype inference in random population samples. American Journal of Human Genetics **71** (2002) 1129–37
14. Schwartz, R., Clark, A.G., Istrail, S.: Methods for inferring block-wise ancestral history from haploid sequences. In: Workshop on Algorithms in Bioinformatics (WABI '02), Springer (2002) 44–59
15. Jojic, N., Jojic, V., Heckerman, D.: Joint discovery of haplotype blocks and complex trait associations from snp sequences. In: Proceedings of the 20th conference on Uncertainty in artificial intelligence (UAI '04), AUAI Press (2004) 286–292
16. Ukkonen, E.: Finding founder sequences from a set of recombinants. In: Algorithms in Bioinformatics (WABI '02), Springer (2002) 277–286
17. McLachlan, G.J., Krishnan, T.: The EM Algorithm and Extensions. John Wiley and Sons (1996)
18. Bertsekas, D.P.: Constrained Optimization and Lagrange Multiplier Methods. Academic Press, New York (1982)
19. Daly, M.J., Rioux, J.D., Schaffner, S.F., et al.: High-resolution haplotype structure in the human genome. Nature Genetics **29** (2001) 229–232
20. Hinds, D.A., Stuve, L.L., Nilsen, G.B., et al.: Whole-genome patterns of common dna variation in three human populations. Science **307** (2005) 1072–1079
21. Koivisto, M., Perola, M., Varilo, T., et al.: An MDL method for finding haplotype blocks and for estimating the strength of haplotype block boundaries. In: Pacific Symposium on Biocomputing (PSB '03), World Scientific (2003) 502–513
22. Stephens, M., Scheet, P.: Accounting for decay of linkage disequilibrium in haplotype inference and missing-data imputation. Americal Journal of Human Genetics **76** (2005) 449–462