Advances and Applications in Bioinformatics and Chemistry

Dovepress

open access to scientific and medical research

Open Access Full Text Article

ORIGINAL RESEARCH

Construction of random perfect phylogeny matrix

This article was published in the following Dove Press journal:
Advances and Applications in Bioinformatics and Chemistry
15 November 2010
Number of times this article has been viewed

Mehdi Sadeghi[1,2]
Hamid Pezeshk[4]
Changiz Eslahchi[3,5]
Sara Ahmadian[6]
Sepideh Mah Abadi[6]

[1]National Institute of Genetic Engineering and Biotechnology, Tehran, Iran; [2]School of Computer Science, [3]School of Mathematics, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran; [4]School of Mathematics, Statistics and Computer Sciences, Center of Excellence in Biomathematics, College of Science, University of Tehran, Tehran, Iran; [5]Department of Mathematics, Shahid Beheshti University, G.C., Tehran, Iran; [6]Department of Computer Engineering, Sharif University of Technology, Tehran, Iran

Purpose: Interest in developing methods appropriate for mapping increasing amounts of genome-wide molecular data are increasing rapidly. There is also an increasing need for methods that are able to efficiently simulate such data.

Patients and methods: In this article, we provide a graph-theory approach to find the necessary and sufficient conditions for the existence of a phylogeny matrix with k nonidentical haplotypes, n single nucleotide polymorphisms (SNPs), and a population size of m for which the minimum allele frequency of each SNP is between two specific numbers a and b.

Results: We introduce an $O(\max(n^2, nm))$ algorithm for the random construction of such a phylogeny matrix. The running time of any algorithm for solving this problem would be $\Omega(nm)$.

Conclusion: We have developed software, RAPPER, based on this algorithm, which is available at http://bioinf.cs.ipm.ir/softwares/RAPPER.

Keywords: perfect phylogeny, minimum allele frequency (MAF), tree, recursive algorithm

Introduction

With the widespread availability of molecular data, computational methods for gene mapping are being developed. Often, the statistical properties and the behavior of these methods need to be assessed and tested by simulation. By increasing the number of computational methods for gene mapping, there is an increasing need for tools that can simulate data for long genomic regions. One of the most popular models used to infer haplotypes from genotype data is perfect phylogeny.[1–3] This model reconstructs haplotype sequences with the assumptions of infinite sites and no recombination. Given a set of genotypes, the goal is to find a set of haplotypes that fit a perfect phylogeny. The solution divides haplotypes into disjoint blocks that are all compatible with a perfect phylogeny tree. Each block can be seen as a region of genome with different evolutionary history. Simulation of genotype or haplotype data based on a coalescent model is central to estimation methods and testing new methodologies. Coalescent simulation can be used to understand the statistical properties of DNA sequences under different evolutionary scenarios and also evaluate and compare different methods for haplotype analysis. A number of simulation programs have been developed under this model and are currently being used.[4–11] We suggest a haplotype simulation to produce haplotype data with predefined allele frequencies with coalescent property. By using the set of haplotypes that satisfy the coalescent property, we can simulate a long genomic region, which can be used as an approximation to the evolutionary process that produce the real data. This simulation constructs a random perfect phylogeny matrix (PPM) with k

Correspondence: Changiz Eslahchi
Department of Mathematics, Shahid Beheshti University, G.C., Tehran, Iran
Tel/Fax +98 21 22431652
Email ch-eslahchi@sbu.ac.ir

submit your manuscript | www.dovepress.com
Dovepress
DOI: 10.2147/AABC.S13397

nonidentical haplotypes, n single nucleotide polymorphisms (SNPs), and a population size of m in which the minimum allele frequency (MAF) of each SNP is between two predefined numbers. A simulated data set for generating a long DNA sequence can be constructed based on assumptions about recombination rate and distribution in an evolutionary model among these perfect phylogeny blocks. The phylogenetic tree is represented by a matrix A in which a_{ij} is the state of character j in sequence i, and the ith row is the character vector of sequence i. In this article, we assume that characters are binary and directed, ie, only $0 \rightarrow 1$ changes may occur on any path from the root to a leaf of the tree. For the output, the ancestral state of an allele is represented by zero. We suggest a haplotype simulation approach that produces haplotype data with prespecified allele frequencies that satisfy coalescent model, ie, it produces a phylogenetic tree in which branches model the changes through the time of evolution based on the model. By above discussions, finding a method to construct random PPM with k nonidentical haplotypes, n SNPs, and a population size of m in which the MAF of each SNP is between two specific numbers a and b will be very useful for data simulation (We consider MAF of column c in A as the number of 1's in column c.) In this article, we take a graph-theory approach to the problem and show that there is a one-to-one correspondence between the set of perfect phylogeny matrices with certain conditions and some rooted trees. We find the necessary and sufficient conditions for the existence of such trees with respect to input parameters. We present an $O(\max(n^2, nm))$ algorithm for generating a random matrix with the above conditions. We have developed a software based on this algorithm, RAPPER, which generates these matrices in a reasonable time. This article is organized as follows: we provide preliminaries and formulate the problem; in sections 3 and 4, Matrices and trees, and Extended tree following Gusfield,[12] we construct a tree for every matrix and discuss its properties. In Necessary conditions, we discuss the necessary conditions. In Sufficient conditions, we find some sufficient conditions and present an algorithm to generate a random sample of the abovementioned matrices.

Preliminaries

To find some necessary and sufficient conditions for the existence of a PPM with m rows, k nonidentical rows, and n columns such that in every column, number of 1s are between a and b, we need some definitions. We consider the cases that $k \geq 2$.

Definition 1. Let a and b be 2 integers and assume $a \leq b$. The matrix $B_{m\times n}$ is called a (k,a,b)-PPM if

1. B is a PPM
2. The number of 1′s in each column is between a and b
3. B has k nonidentical rows

Example 1. The following matrix B is a $(3,2,3)$-PPM.

$$B = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

Definition 2. The matrix $A_{k\times n}$ is called (m,a,b)-*extendable* if

1. A is a PPM
2. A has k nonidentical rows
3. There exists a matrix $B_{m\times n}$ that is a (k,a,b)-PPM, and the rows of A and B are identical. (In this case, we say that $A_{k\times n}$ is extendable to $B_{m\times n}$.)

Example 2 A is $(5,2,3)$-*extendable* to B.

$$A_{3\times 3} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \rightarrow B_{5\times 3} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

It is obvious that there exists a (k,a,b)-PPM matrix $B_{m\times n}$ if and only if there exists an (m,a,b)-extendable matrix $A_{k\times n}$.

Definition 3. A matrix B is called *good* if it can be decomposed as follows:

1. The entries of its leftmost column are all 1′s.
2. There exist good matrices $B_1, B_2, \ldots, B_d$ such that the rest (0 or more) of the columns of B form the block structure, as illustrated in Figure 1.

Definition 4. A matrix A is called *canonical* if it satisfies the second condition of the good matrix definition.

In the following definition, we assign a root to each good matrix.

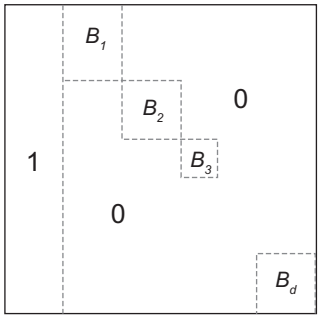


Figure 1 Block structure of a good matrix.

Definition 5. In a good matrix B, we consider the leftmost all-one column as the *root* of the B.

Matrices and trees

Constructing a tree from a matrix

According to Theorem 8 of Pe'er et al,[13] every PPM has an ordering of its rows and columns, which yields a canonical matrix.

Theorem (Pe'er et al).[13] Let B be a binary matrix. The following are equivalent:

1. B has a phylogenetic tree.
2. There exists an ordering of the rows and columns of B, which yields a canonical matrix.

Let $A_{k\times n}$ be a PPM that consists of $B_1, B_2, \ldots, B_d$ *good* blocks and c_i be the corresponding *root* of B_i. Following Gusfield,[12] we construct a labeled tree $T_{(A)}$ by using the following steps (see Figure 2):

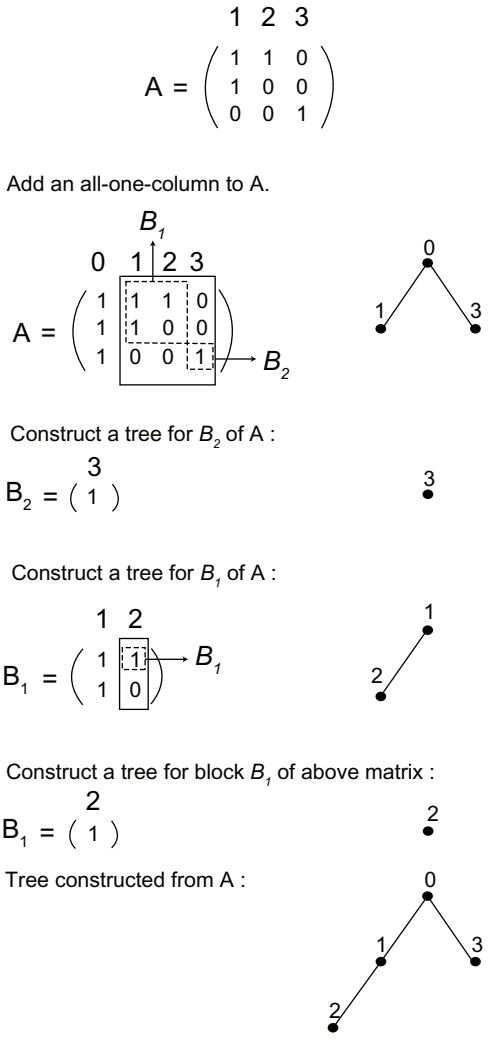


Figure 2 Constructing a labeled tree TA from a perfect phylogenetic matrix, A.

1. Add an all-one column to A as the leftmost column. Index this column by 0.
2. Let vertex 0 be the parent of c_i for all $1 \le i \le d$.
3. Construct a tree from canonical form of every B_i in a recursive manner. (Note that B_i is a good matrix and has an all-one column.)

The vertex set of $T_{(A)}$ is $\{0, 1, 2, \ldots, n\}$. Now, we label some vertices of $T_{(A)}$ as follows:

If the last 1 entry in row r occurs in column j, then we label vertex j of $T_{(A)}$ with r. Gusfield[12] proved the following lemma.

Lemma 1. Every leaf of $T_{(A)}$ is labeled, and every vertex is labeled at most once.

Necessary and sufficient conditions for PPM

Let A be a matrix and $O_v = \{r/a_{rv} = 1\}$. According to Estabrook et al[14] and Meacham,[15] A is PPM if and only if for every two columns u and v, $O_u \cap O_v = \phi$ or $O_u \subset O_v$ or $O_v \subset O_u$.

Lemma 2. Let r be a row in A and v be a vertex in $T_{(A)}$ with label r; then $a_{ru} = 1$ if and only if u is located in a path from root to v.

Proof. Let u be located in a path from root to v. So u is an ancestor of v. By the way that the tree was constructed from canonical form of matrix A, we have $O_v \subset O_u$. So, if $a_{rv} = 1$ then $a_{ru} = 1$.

Now, let $a_{ru} = 1$. Since v is labeled by r, the last nonzero entry of r occurs in v. So, for every child of v such as w, we have $a_{rw} = 0$. Therefore, $a_{ru} = 1$ implies that u is not a child of v. As $a_{ru} = a_{rv} = 1$, $O_u \cap O_v \neq \phi$. Thus, $O_v \subset O_u$ or $O_u \subset O_v$. Since u is not a child of v, O_u is not a subset of O_v. So, we have $O_v \subset O_u$, and u is an ancestor of v. Therefore, u is located in a path from root to v.

Constructing a matrix from the tree

Definition 6. A rooted tree T is called a (k,n)-*complete tree* if it satisfies the following conditions:

1. $V(T) = \{0,1,2, \ldots, n\}$.
2. For every $1 \le i \le k$, there exists exactly 1 vertex with label i.
3. Every leaf is labeled.
4. Every vertex has at most 1 label.

From Lemma 1 and the way we construct $T_{(A)}$, we obtain that $T_{(A)}$ is (k,n)-complete tree where $A_{k\times n}$ is a PPM. Now, for every (k,n)-complete tree T, we construct a PPM $A^T_{k\times n}$ with nonidentical rows as follows:

Let c be an arbitrary vertex of T and T_c be the subtree of T with root c. We construct $A^T = [a_{rc}]$ by $a_{rc} = 1$ if and only if there exists a vertex in T_c with label r. Gusfeild (1991) showed that A^T is a PPM. Let r be a row of A, which is the

label of u in T. Similar to Lemma 2, $a_{rw} = 1$ if and only if w is located in a path from root to u. Since labels are different and there is a unique path between the root and the vertices of T, rows of A are nonidentical.

It is obvious that $A^{T_{(A)}} = A$ and $T_{(A^T)} = T$. Therefore, there is a one-to-one correspondence between the set of all PPM with k nonidentical rows and n columns and the set of all (k,n)-complete tree.

Extended tree

Let $A_{k\times n}$ be an (m,a,b)-extendable matrix, and let the (k,a,b)-PPM matrix $B_{m\times n}$ be its extension and $T_{(A)}$ be its corresponding tree. Let w be the repeating time function defined on the labeled vertices of $T_{(A)}$ as $w(v) = t$ if and only if v is labeled by r and row r is repeated t times in B. We call $(T_{(A)},w)$ the (m,a,b)-*extended tree* of A and $w(v)$ the repeating label of v.

Lemma 3. The MAF of column c in B is the sum of the repeating label of the vertices in T_c.

Proof. Let $a_{rc} = 1$. Then, by Lemma 3.2, c is located in a path from root to vertex v with label r. So, v is a vertex of T_c. Let $w(v) = t$. Corresponding to t repeats of r, we have t ones in column c. Therefore, the MAF of column c is equal to the sum of repeating labels in T_c. □

Example 3. For matrices in Example 2, repeating labels and MAFs are shown in Figure 3. Let $A_{k\times n}$ be an (m,a,b)-extendable matrix and $(T_{(A)},w)$ be the (m,a,b)-extended tree of A. Now, by the previous lemmas, we have the following observations.

Observations:

O_1: Let u be the ancestor of v in $(T_{(A)},w)$; then the MAF of column u is greater than or equal to the MAF of column v.

O_2: Let v be a leaf with label i in $(T_{(A)},w)$. By proof of Lemma 3.1, column v in $A_{k\times n}$ has only 1 nonzero entry in row i. Since the MAF of column v should be at least a, $w(v) \geq a$.

O_3: Let T_{u_i} have l_i leaves and c_i labeled internal vertices. By using Lemma 4.1 and O_2, the MAF of column u_i in B is at least $al_i + c_i$.

O_4: Let T_{u_i} have l_i leaves and c_i labeled internal vertices. Since the MAF of each column in B is at most b, O_3 implies that $l_i \leq b/a$.

O_5: Let $x_1, x_2, \ldots, x_d$ be the children of root r of $(T_{(A)},w)$. Then, using O_1 for each labeled vertex x_i, we have $w(x_i) \leq b$ if and only if $w(x_i) \leq b$ for every labeled vertex of $(T_{(A)},w)$.

In the following theorem, we show that we can always assume that the desired matrix has at least one all-zero row. By the MAF of vertex v in $T_{(A)}$, we mean the MAF of column v in A^T.

Theorem 1. There is an (m,a,b)-extendable matrix $A_{k\times n}$ if and only if there is an (m,a,b)-extendable matrix $A'_{k\times n}$ that has a zero row.

Proof. It is obvious that root r of $T_{(A)}$ is labeled when A has a zero row.

Let A has no zero rows and r is not labeled. We consider two cases:

1. There exists an internal node u, which is labeled by p. In this case, we consider the labeled tree T' by removing label p of u and giving p to r.
2. Let only the leaves of $T_{(A)}$ be labeled. Consider vertex x such that degree x in T_x is at least 2, and in T_x, every vertex of T_x–x has at most 1 child (as $k \geq 2$ and there is no labeled internal node and $T_{(A)}$ has at least two leaves, there exists such x). Let u and v be two leaves of T_x and z be the ancestor of v and the child of x. (If v is a child of x let $z = v$.) Since u is a leaf of $T_{(A)}$, it has a label such as p. By removing edge xz from $T_{(A)}$, labeling p from u, adding edge uz, and giving p to r, we obtain a new labeled tree T' (Figure 4). In both cases, we define repeating time function w' on the labeled vertices of T' by

$$w'(v) = \begin{cases} w(v) & v \neq u \\ 0 & v = u \\ w(u) & v = r \end{cases}$$

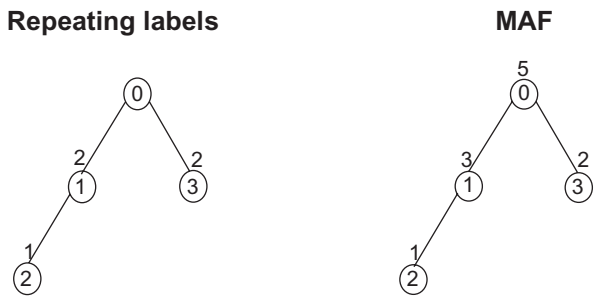


Figure 3 Repeating labels and minimum allele frequency (MAF).

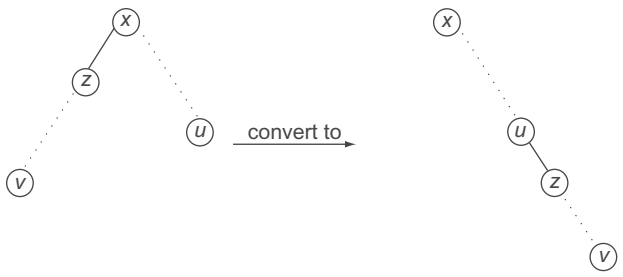


Figure 4 New labeled tree T' obtained from T.

It is obvious that T' is a (k,n)-complete tree, which has a zero row. Now, by using $O_1, \ldots, O_5$ and Lemma 3, we conclude that $A^{T'}$ is (m,a,b)-extendable.

Necessary conditions

In this section, we describe some necessary conditions for the existence of an (m,a,b)-tree. By applying Theorem 1, we find the conditions necessary for the existence of an (m,a,b)-tree in which the root has been labeled. First, we introduce some necessary conditions, and then in the next section, we will show that these conditions are also sufficient.

Theorem 2. Assume that T is a (k,n)-complete tree and (T,w) is an (m,a,b)-tree in which the root has been labeled. Let r be the root of T, $deg_T(r) = d$, and T has l leaves. Then

1. $l \leq k-1$.
2. $\dfrac{k+(a-1)l-1}{b} \leq d \leq l$.
3. $l\,(a-1) + km \leq$.

Proof.

1. Since r and the leaves of T are labeled with nonidentical rows, 1 holds.
2. It is obvious that $d \leq l$. By O_2 for each leaf v, $w(v) \geq a$. Suppose $x_1, x_2, \ldots, x_d$ are the children of r, and T_{x_i} contains l_i leaves and c_i internal labeled vertices. Then by O_3, we have

$$l_i a + c_i \leq b, \qquad 1 \leq i \leq d.$$

Therefore

$$\sum_{i=1}^{d} (al_i + c_i) \# bd$$

$$al + k - l - 1 \leq bd$$

So,

$$\frac{k+(a-1)l-1}{b} \# d$$

3. By O_2 for each leaf v, we have $w(v) \geq a$, and for each labeled vertex u, we have $w(u) \geq 1$.

Then, the number of rows in a (k,a,b)-PPM matrix $B_{m\times n}$, which is an extension of extended A_T, is at least

$$al + (k - l).$$

We categorize the children of r, $x_1, x_2, \ldots, x_d$, into 3 groups:

- $A_1 = \{\xi_t | \chi_t = 0 \text{ and } \lambda_t = 1\}$
- $A_2 = \{\xi_t | \chi_t = 0 \text{ and } \lambda_t \neq 1\}$
- $A_3 = \{\xi_t | \chi_t \neq 0\}$

Let $\alpha = |A_1|$ and $\beta = |A_2|$.

Theorem 3. Let (T,w) be the same as in Theorem 2 and $B_{m\times n}$ is its corresponding (k,a,b)-PPM. Then

1. $l_i \leq b/a$
2. $d + l - n \leq a$
3. Let a/b (a be a divisor of b) and $a \neq b$; then the number of x_i in which T_{x_i} has b/a leaves is less than or equal to $n - k + 1$.

Proof.

1. It results from the observation O_4.
2. Let n_i be the number of vertices of T_{x_i}. Then, obviously

$$n_i \; \$ \begin{cases} c_i + l_i & c_i \geq 1 \\ l_i + 1 & c_i = 0 \text{ and } l_i \neq 1 \\ 1 & c_i = 0 \text{ and } l_i \neq 1 \end{cases}$$

So, we have

$$\begin{aligned} n &= \sum_{i=1}^{d} n_i \\ &= \sum_{c_i \geq 1} n_i + \sum_{c_i = 0, l_i \neq 1} n_i + \sum_{c_i = 0, l_i = 1} n_i \\ &\geq \sum_{c_i \geq 1} (c_i + l_i) + \sum_{c_i = 0, l_i \neq 1} (l_i + 1) + \sum_{c_i = 0, l_i = 1} 1 \\ &= \sum c_i + \sum l_i + \beta \end{aligned}$$

Now, we find the upper and lower bounds.

- *Upper bound*

$$\begin{aligned} n &\geq \sum c_i + \sum l_i + \beta \\ &= k - l - 1 + l + \beta \\ &\Rightarrow \beta \leq n - k + 12 \end{aligned} \tag{1}$$

- Lower bound

We have $|A_1 \cup A_2| = d - a$. Since the number of internal vertices of T which have labels is $k - l - 1$,

$$\begin{aligned} \beta &\geq (d - \alpha) - (k - 1 - 1) \\ &= d - \alpha + l - k + 1 \end{aligned}$$

Now, we have

$$d - \alpha - k + l - 1 \leq n - k + 1$$

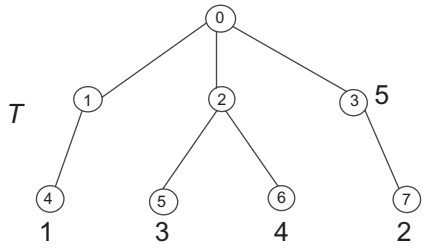
$$d + l - n \leq \alpha$$

and part 2 is thus proved.

3. Suppose a/b and $a \neq b$. Let $S = \{x_i | l_i = b/a\}$. By observation O_3, for each $x_i \in S$, we have $c_i = 0$. So, $S \subset A_2$, and by inequality 1, we have $|S| \leq n - k + 1$.

Sufficient conditions

In the previous section, we obtained some necessary conditions for the existence of an (m,a,b)-extendable tree whose root has been labeled. In this section, we show that these



$$A^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad (A^T, w) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 5 In this figure, we have $w(1) = w(2) = w(5) = 2$ and $w(3) = w(4) = 1$.

conditions are sufficient too. For this purpose, let $l_1, l_1, \ldots, l_d$ satisfy the conditions of Theorems 2 and 3. Then, we introduce an algorithm for constructing the rooted (m,a,b)-extendable tree T with root r; $x_1, \ldots, x_d$ are the children of r and T_{x_i} has l_i leaves. First, we determine the number of labeled internal nodes in each T_{x_i}.

Determination of c_i

We categorize the children of r into three groups:

1. G_1: Children with $l_i = 1$. By Theorem 3, part 2, we have $|G_1| \geq d + l - n$.
2. G_2: Children with $l_i = b/a$ and $l_i \neq 1$.
3. G_3: Children with $l_i = b/a$ for $b \neq a$. By Theorem 3, part 3, we have $|G_3| \geq n - k + 1$.

According to G_1,G_2, and G_3, we determine c_i s as follows:

1. For each $x_i \in G_2$ and $d + l - n$ elements of G_1, we set $c_i = 0$.
2. Let $|G_2| \leq k - l + 1$. Then, for each $x_i \in G_2$, we assign $c_i = 1$. Now, we distribute the value $k - l + 1 - |G_2|$ among the members of G_2 and those of G_1 for which $c_i \neq 0$ or c_i is not determined in step 1. Now, suppose $|G_2| \geq k - l + 1$. Let F be a subset of G_2 of size $k - l + 1$. For each $x_i \in F$, set $c_i = 1$. For all the remaining vertices such as x_j, which is not considered in the above steps, we assign $c_j = 0$. By this procedure, part 2 of the Theorem 3, $(|A_1| \geq d + l - n)$ holds.

Algorithm for determination of c_i

1. Categorize d children of the root
2. *for* $i = 1$ *to* d
3. *if* $l_i = 1$ *then* put this child in G_1
4. *if* $l_i \neq b/a$ and $l_i \neq 1$ *then* put this child in G_2
5. *if* $l_i = b/a$ and $b \neq$ a *then* put this child in G_3
6. determine c_i (related to each x_i) according to G
7. *if* $x_i \in G_3$ *or* $d + l - n \in G_1$
8. $c_i = 0$
9. *else if* $|G_2| \leq k - l + 1$
10. *if* $x_i \in G_2$ *then*
11. $c_i = 1$
12. Distribute $k - l + 1 - |G_2|$ among members of G_2 and those of G_1 for which $c_i \neq 0$
13. *else if* $|G_2| > k - l + 1$
14. $F \leftarrow$ (subset of G_2 of size $k - l + 1$)
15. *if* $x_i \in F$ *then* $c_i = 1$
16. *else* $c_1 = 0$.

Determination of n_i

- We first initialize n_i for each x_i as follows:

$$n_i = \begin{cases} c_i + l_i & c_i \geq 1 \\ l_i + 1 & c_i = 0 \text{ and } l_i \neq 1 \\ 1 & c_i = 0 \text{ and } l_i = 1 \end{cases}$$

- We distribute $n - \sum n_i$ between all T_{x_i} at random.

To show that these steps are possible, it is enough to show that $\sum n_i \leq n$. By the proof of part 2 of Theorem 3, it is enough to show that $\beta \leq n - k + 1$. The number of vertices for which $l_i \neq 1$ and $c_i = 0$, β, is as follows:

1. If $|G_2| < k - 1 + 1$, then $\beta = |G_3|$.
2. If $|G_2| \geq k - l + 1$, then $\beta = |G_3| + |G_2| - k + l - 1$.

Since $|G_3| + |G_2| = d - |G_1|$ and $|G_1| \geq d + l - n$, $|G_3| + |G_2| \leq n - l$, in both cases, $\beta \leq n - k + 1$.

Algorithm for the determination of n_i

1. Initialize n_i for each x_i
2. *if* $c_i \geq 1$
3. *then* $n_i = c_i + l_i$
4. *else if* $c_i = 0$ and $l_i \neq 1$
5. *then* $n_i = l_i + 1$
6. *else if* $c_i = 0$ and $l_i = 1$
7. *then* $n_i = 1$
8. Distribute $n - \sum n_i$ between trees related to children of the root (T_{x_i})

Constructing (*m,a,b*)-extendable tree

In this subsection, we will construct a rooted tree T with root r. $x_1, \ldots, x_d$ are its children, and T_{x_i} has n_i vertices, l_i leaves, and c_i labeled internal vertices for which c_is and n_is are determined as described earlier. The following algorithm constructs T_{x_i}, $1 \leq i \leq d$.

Algorithm for the construction of $T_i(n_i, l_i, c_i)$

1. Let *LS* be the set of leaf vertices
2. Let *IS* be the set of internal vertices
3. $IS \leftarrow x_i$
4. $LS \leftarrow \phi$
5. *for* $j \leftarrow 2$ *to* n_i
6. *do if sizeof* $(LS) = l_i$
7. then $PS = LS$
8. *else if* l_i – *sizeof* $(LS) = n_i - j + 1$
9. *then* $PS = IS$
10. *else* $PS = LS \cup IS$
11. select vertex v from PS randomly and put the new vertex, w, as v′s child
12. $LS = LS \cup w$
13. *if* $v \in LS$
14. *then* $LS \leftarrow LS - v$
15. $IS \leftarrow IS \cup v$
16. Mark c_i vertices from *IS*.

Now, we add the root r and edges rx_i, $1 \leq i \leq d$. The desired tree T is constructed. To label the vertices of T_{x_i}, $1 \leq i \leq d$ and root r and the leaves of T, we assign $\{1,2, \ldots, k\}$ to the labeled vertices of $\bigcup_i T_{x_i}$ randomly.

In this algorithm, we first construct all ordered pairs (l,d) that satisfy the conditions of Theorems 2 and 3. Then, we choose some of these pairs randomly and construct all d-tuples $(l_1, \ldots, l_d)$, satisfying the conditions of Theorems 2 and 3 and $l = l_1 + l_2 + \ldots + l_d$. Now, one of the d-tuples is chosen randomly. Then, we classify x_i according to l_i. Using this classification, we consider a primary class for c_i. Now, for the remaining vertices for which we have not assigned any c_i, we choose a c_i randomly. For calculating n_i, we first assign an initial value for each vertex x_i and then distribute $n - \sum n_i$ randomly. It should be noted again that by randomness, we mean distribution according to a uniform random variable.

We also define a function w on the labeled vertices of the (k,n)-complete tree, T, such that (T,w) becomes an (m,a,b)-tree (Figure 5). We obtain w from the following recursive algorithm:

$$w_o(u) = \begin{cases} a & \text{if } u \text{ is a leaf of } T \\ 1 & \text{if } u \text{ is a labled vertex} \end{cases}$$

We define w_{i+1} recursively from w_i as follows: If there exists x_j such that the MAF of x_j in (T,w_i) is less than b, we choose an arbitrarily labeled vertex from Tx_i, such as u, and define $w_{i+1}(u) = w_i(u) + 1$. We continue this procedure until we obtain the function w_j such that $\sum_{u \in T} w_j(u) = m$ or the MAF of $x_i = b$ for all $1 \leq i \leq d$. Now, w is defined by $w = w_j$ if $\sum_{u \in T} w_j(u) = m$. For the case that $\sum_{u \in T} w_j(u) < m$, we consider w by

$$w(u) = \begin{cases} m - \sum_{u \in T} w(u) & \text{if } u = r \\ w(u) & \text{if } u \neq r \end{cases}$$

Let the vertex u of A_T have labels r_0 and $w(u) = 0$. We repeat row r_0 of A_T, t times. Let B be the matrix obtained from A_T by repeating the procedure. It is obvious that B has m rows, n columns, and k nonidentical rows. Let u be a column of B and a vertex of (T,w). Consider T_u and one of its leaf, such as u_0, with label r_0. We know that $w(u_0) \geq a$. The entry of A_T in column u_0 and row r_0 is 1. Therefore, by Lemma 2, the entry of A_T in column u and row r_0 is also 1. Since we repeat row r_0 in B at least a times, the MAF of column u in B is at least a. On the other hand, let x_i be the ancestor of u. Since $w(u) \leq w(x_i)$ and $MAF(x_i) \leq b$ by observation O_1, $MAF(u) \leq b$. Therefore, $B_{m \times n}$ is a (k,a,b)-PPM.

Running time of the algorithm

This algorithm has five steps. In the first step, the algorithm finds d, l, and $l_1, \ldots, l_d$, which satisfy the necessary conditions in $O(n^2)$. In the second step, the algorithm finds the n_i and c_i for $1 \leq i \leq d$. In the third step, the algorithm constructs Tx_i, $1 \leq i \leq d$, with n_i internal vertices, c_i labeled internal

vertices, and l_i in $O(n_i^2)$. As $\sum_{i=1}^{d} n_i \leq n$ and $\sum_{i=1}^{d} n_i^2 \leq n^2$, the running time of the algorithm in this part is $O(n^2)$. In the next step, the algorithm defines a function on labeled vertices and finds its value recursively. As this function is called at most m times and in each call updating MAF of x_i's takes $O(n)$, the running time of the algorithm in this part is $O(mn)$. In the last step, the algorithm constructs the desired matrix from the tree in $O(mn)$. Thus, the total running time of the algorithm is $O(\max(n^2, nm))$.

Discussion

In this article, we have presented a new model for perfect phylogeny matrices. Our goal was to construct a random perfect phylogeny matrix with k different haplotypes, n SNPs, and a population size of m for which the MAF of each SNP is between two specific numbers a and b. Our new approach allows us to find the necessary and sufficient conditions for the existence of such a matrix. As the solution matrix is a binary matrix with m rows and n columns, any algorithm for this problem is $\Omega(nm)$. We developed an $O(\max(n^2, nm))$ time algorithm based on this model to solve this problem.

We used the available methods to construct the perfect phylogeny matrix without taking MAF into account, and we then eliminated those columns that do not satisfy the MAF condition. It should be noted that there are two problems concerning this approach. First, we need to use an algorithm that is able to calculate the MAF of each column automatically and eliminate it if the conditions are not satisfied. Second, it is very probable that the number of columns that should be removed is very high. So, we will obtain matrices with few columns, and we have to run the algorithm several times to obtain a matrix with n columns and the required MAF. Therefore, an algorithm that could construct such matrices is of much interest. We have developed software, RAPPER, for implementing this algorithm, which is available at http://bioinf.cs.ipm.ir/softwares/RAPPER.

Acknowledgment

Changiz Eslahchi thanks Shahid Beheshtshi University for their support. This research is in part supported by a grant from IPM.

Disclosure

The authors report no conflicts of interest in this work.

References

1. Gusfield D. Haplotyping as perfect phylogeny: conceptual framework and efficient solutions. In: Myers G, Hannenhalli S, San-koff D, et al. editors. *Proceedings of RECOMB 2002*. New York, NY: ACM Press; 2002:165–175.
2. Bafna V, Gusfield D, Lancia G, Yooseph S. Haplotyping as perfect phylogeny: a direct approach. *J Comput Biol.* 2003;10:323–340.
3. Eskin E, Halperin E, Karp R. Efficient reconstruction of haplotype structure via perfect phylogeny. *J Bioinform Comput Biol.* 2003;1:1–20.
4. Excoffier L, Novembre J, Schneider S. SIMCOAL: a general coalescent program for the simulation of molecular data in interconnected populations with arbitrary demography. *J Heredity.* 2000;91:506–509.
5. Spencer CC, Coop G. SelSim: a program to simulate population genetic data with natural selection and recombination. *Bioinformatics*. 2004;20:3673–3675.
6. Mailund T, Schierup MH, Pedersen CN, Mechlenborg PJ, Madsen JN, Schauser L. CoaSim: a flexible environment for simulating genetic data under coalescent models. *BMC Bioinformatics.* 2005;6:252.
7. Marjoram P, Wall JD. Fast "coalescent" simulation. *BMC Genet*. 2006;7:16.
8. Hudson RR. Generating samples under a Wright-Fisher neutral model of genetic variation. *Bioinformatics*. 2002;18:337–338.
9. Hellenthal G, Stephens M. msHOT: modifying Hudson's ms simulator to incorporate crossover and gene conversion hotspots. *Bioinformatics.* 2007;23:520–521.
10. Posada D, Wiuf C. Simulating haplotype blocks in the human genome. *Bioinformatics.* 2003;19:289–290.
11. Montanna G. HapSim: a simulation tool for generating haplotype data with pre specified allele frequencies and LD coefficients. *Bioinformatics*. 2003;21:4309–4311.
12. Gusfield D. Efficient algorithms for inferring evolutionary trees. *Networks.* 1991;21:19–28.
13. Pe'er I, Pupko T, Shamir R, Sharan R. Incomplete directed perfect phylogeny. *SIAM J Comput*. 2004;33:597–607.
14. Estabrook GF, Johnson CS Jr, McMorris FR. An idealized concept of the true cladistic character. *Math Biosci.* 1975;23:263–272.
15. Meacham C. Theoretical and computational considerations of the compatibility of qualitative taxonomic characters. Numerical Taxonomy. *Nato ASI Series.* 1983;G1:304–314.