

# On the Complexity of SNP Block Partitioning Under the Perfect Phylogeny Model

Jens Gramm<sup>1</sup>   Tzvika Hartman<sup>2</sup>   Till Nierhoff<sup>3</sup>  
Roded Sharan<sup>4</sup>   Till Tantau<sup>5</sup>

<sup>1</sup>Universität Tübingen, Germany

<sup>2</sup>Bar-Ilan University, Ramat-Gan, Israel

<sup>3</sup>International Computer Science Institute, Berkeley, USA

<sup>4</sup>Tel-Aviv University, Israel

<sup>5</sup>Universität zu Lübeck, Germany

Workshop on Algorithms in Bioinformatics, 2006

## Outline

## What is haplotyping and why is it important?

You hopefully know this after the previous three talks. . .

## General formalization of haplotyping.

### Inputs

- A **genotype matrix**  $G$ .
- The **rows** of the matrix are **taxa / individuals**.
- The **columns** of the matrix are **SNP sites / characters**.

### Outputs

- A **haplotype matrix**  $H$ .
- Pairs of rows in  $H$  **explain** the rows of  $G$ .
- The haplotypes in  $H$  are **biologically plausible**.

## Our formalization of haplotyping.

### Inputs

- A genotype matrix  $G$ .
- The rows of the matrix are individuals / taxa.
- The columns of the matrix are SNP sites / characters.
- The problem is directed: one haplotype is known.
- The input is biallelic: there are only two homozygous states (0 and 1) and one heterozygous state (2).

### Outputs

- A haplotype matrix  $H$ .
- Pairs of rows in  $H$  explain the rows of  $G$ .
- The haplotypes in  $H$  form a perfect phylogeny.

## We can do perfect phylogeny haplotyping efficiently, but ...

### 1 Data may be missing.

- This makes the problem NP-complete ...
- ... even for very restricted cases.

#### Solutions:

- Additional assumption like the rich data hypothesis.

### 2 No perfect phylogeny is possible.

- This can be caused by chromosomal crossing-over effects.
- This can be caused by incorrect data.
- This can be caused by multiple mutations at the same sites.

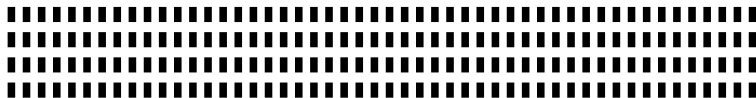
#### Solutions:

- Look for phylogenetic networks.
- Correct data.
- Find blocks where a perfect phylogeny is possible.

## How blocks help in perfect phylogeny haplotyping.

- 1 Partition the site set into overlapping contiguous blocks.
- 2 Compute a perfect phylogeny for each block and combine them.
- 3 Use dynamic programming for finding the partition.

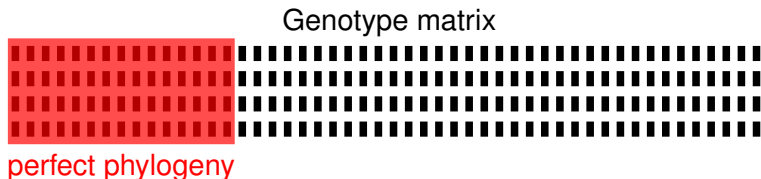
Genotype matrix



no perfect phylogeny

## How blocks help in perfect phylogeny haplotyping.

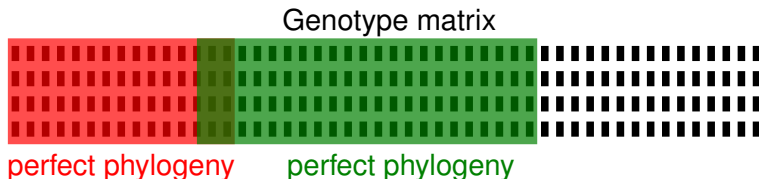
- 1 Partition the site set into overlapping contiguous blocks.
- 2 Compute a perfect phylogeny for each block and combine them.
- 3 Use dynamic programming for finding the partition.





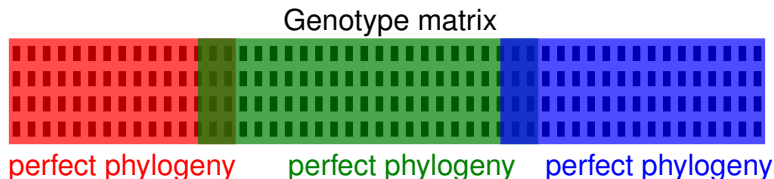
## How blocks help in perfect phylogeny haplotyping.

- 1 Partition the site set into overlapping contiguous blocks.
- 2 Compute a perfect phylogeny for each block and combine them.
- 3 Use dynamic programming for finding the partition.



## How blocks help in perfect phylogeny haplotyping.

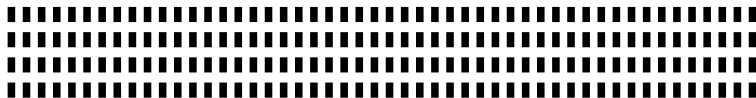
- 1 Partition the site set into overlapping contiguous blocks.
- 2 Compute a perfect phylogeny for each block and combine them.
- 3 Use dynamic programming for finding the partition.



## Objective of the integrated approach.

- 1 Partition the site set into **noncontiguous** blocks.
- 2 Compute a perfect phylogeny for each block and combine them.
- 3 **Compute partition while computing perfect phylogenies.**

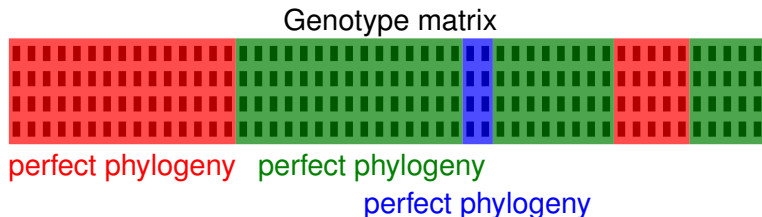
Genotype matrix



no perfect phylogeny

## Objective of the integrated approach.

- 1 Partition the site set into **noncontiguous** blocks.
- 2 Compute a perfect phylogeny for each block and combine them.
- 3 **Compute partition while computing perfect phylogenies.**



## The formal computational problem.

We are interested in the computational complexity of the function  $\chi_{PP}$ :

- It gets genotype matrices as input.
- It maps them to a number  $k$ .
- This number is minimal such that the sites can be covered by  $k$  sets, each admitting a perfect phylogeny.  
(We call this a **pp-partition**.)

## Finding pp-partitions of haplotype matrices.

We start with a special case:

- The inputs  $M$  are **already haplotype matrices**.
- The inputs  $M$  **do not allow a perfect phylogeny**.
- What is  $\chi_{PP}(M)$ ?

### Example

$M$ :

0	0	0	1
0	1	0	0
1	0	0	0
0	1	0	0
1	0	0	0
0	1	0	1
1	1	0	0
0	0	1	0
1	0	1	0

No perfect phylogeny is possible.

# Finding pp-partitions of haplotype matrices.

We start with a special case:

- The inputs  $M$  are **already haplotype matrices**.
- The inputs  $M$  **do not allow a perfect phylogeny**.
- What is  $\chi_{PP}(M)$ ?

## Example

$M$ :

0	0	0	1
0	1	0	0
1	0	0	0
0	1	0	0
1	0	0	0
0	1	0	1
1	1	0	0
0	0	1	0
1	0	1	0

Perfect phylogeny

Perfect phylogeny

$$\chi_{PP}(M) = 2.$$

## Bad news about pp-partitions of haplotype matrices.

### Theorem

Finding *optimal pp-partition of haplotype matrices* is equivalent to finding *optimal graph colorings*.

### Proof sketch for first direction.

- 1 Let  $G$  be a graph.
- 2 Build a matrix with a column for each vertex of  $G$ .
- 3 For each edge of  $G$  add four rows inducing the submatrix  $\begin{pmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}$ .
- 4 The submatrix enforces that the columns lie in different perfect phylogenies.





## Implications for pp-partitions of haplotype matrices.

### Corollary

*If  $\chi_{PP}(M) = 2$  for a haplotype matrix  $M$ , we can find an optimal pp-partition in polynomial time.*

### Corollary

*Computing  $\chi_{PP}$  for haplotype matrices is*

- *NP-hard,*
- *not fixed-parameter tractable, unless  $P = NP$ ,*
- *very hard to approximate.*

# Finding pp-partitions of genotype matrices.

Now comes the general case:

- The inputs  $M$  are **genotype matrices**.
- The inputs  $M$  **do not allow a perfect phylogeny**.
- What is  $\chi_{PP}(M)$ ?

## Example

$M$ :

2	2	2	2
1	0	0	0
0	0	0	1
0	0	1	0
0	2	2	0
1	1	0	0

No perfect phylogeny is possible.

# Finding pp-partitions of genotype matrices.

Now comes the general case:

- The inputs  $M$  are **genotype matrices**.
- The inputs  $M$  **do not allow a perfect phylogeny**.
- What is  $\chi_{PP}(M)$ ?

## Example

$M$ :

2	2	2	2
1	0	0	0
0	0	0	1
0	0	1	0
0	2	2	0
1	1	0	0

Perfect phylogeny

Perfect phylogeny

$$\chi_{PP}(M) = 2.$$

## Bad news about pp-partitions of haplotype matrices.

### Theorem

Finding *optimal pp-partition of genotype matrices* is at least as hard as finding *optimal colorings of 3-uniform hypergraphs*.

### Proof sketch.

- 1 Let  $G$  be a 3-uniform hypergraph.
- 2 Build a matrix with a column for each vertex of  $G$ .
- 3 For each hyperedge of  $G$  add four rows inducing the submatrix  $\begin{pmatrix} 2 & 2 & 2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ .
- 4 The submatrix enforces that the three columns do not all lie in the same perfect phylogeny. □

## Implications for pp-partitions of genotype matrices.

### Corollary

*Even if we know  $\chi_{PP}(M) = 2$  for a genotype matrix  $M$ , finding a pp-partition of any fixed size is still*

- NP-hard,
- not fixed-parameter tractable, unless  $P = NP$ ,
- very hard to approximate.

## Automatic optimal pp-partitioning is hopeless, but...

- The hardness results are **worst-case** results for **highly artificial inputs**.
- **Real biological data** might have special properties that make the problem **tractable**.
- One such property is that perfect phylogenies are often perfect **path** phylogenies:  
In HapMap data, in 70% of the blocks where a perfect phylogeny is possible a perfect path phylogeny is also possible.

# Example of a perfect path phylogeny.

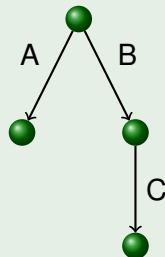
## Genotype matrix

	A	B	C
$G:$	2	2	2
	0	2	0
	2	0	0
	0	2	2

## Haplotype matrix

	A	B	C
$H:$	1	0	0
	0	1	1
	0	0	0
	0	1	0
	0	0	0
	1	0	0
	0	0	0
	0	1	1

## Perfect path phylogeny



## The modified formal computational problem.

We are interested in the computational complexity of the function  $\chi_{\text{PPP}}$ :

- It gets genotype matrices as input.
- It maps them to a number  $k$ .
- This number is minimal such that the sites can be covered by  $k$  sets, each admitting a perfect **path** phylogeny. (We call this a ppp-partition.)



## Good news about ppp-partitions of genotype matrices.

### Theorem

*Optimal ppp-partitions of genotype matrices can be computed in polynomial time.*

### Algorithm

- 1 Build the following partial order:
  - Can one column be above the other in a phylogeny?
  - Can the columns be the two children of the root of a perfect path phylogeny?
- 2 Cover the partial order with as few compatible chain pairs as possible.  
For this, a maximal matching in a special graph needs to be computed.

## Summary

- Finding optimal pp-partitions is **intractable**.
- It is even intractable to find a pp-partition when **just two noncontiguous blocks are known to suffice**.
- For perfect **path** phylogenies, optimal partitions can be computed **in polynomial time**.

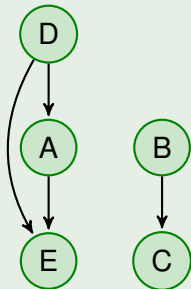
## The algorithm in action.

Computation of the partial order.

### Genotype matrix

	A	B	C	D	E
G:	2	2	2	2	2
	0	1	2	1	0
	1	0	0	1	2
	0	2	2	0	0

### Partial order



Partial order: →

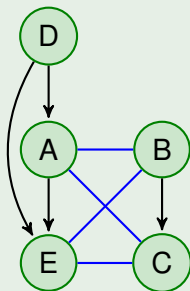
## The algorithm in action.

### Computation of the partial order.

#### Genotype matrix

	A	B	C	D	E
	2	2	2	2	2
G:	0	1	2	1	0
	1	0	0	1	2
	0	2	2	0	0

#### Partial order



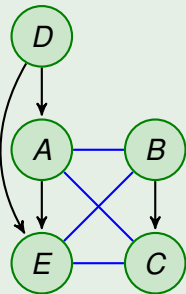
Partial order: →

Compatible as children of root: —

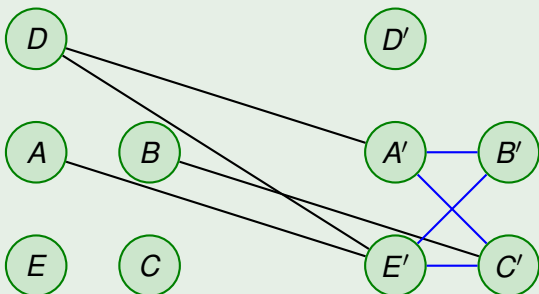
## The algorithm in action.

The matching in the special graph.

Partial order



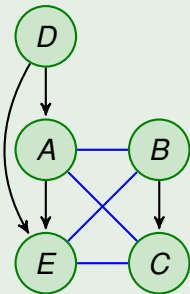
Matching graph



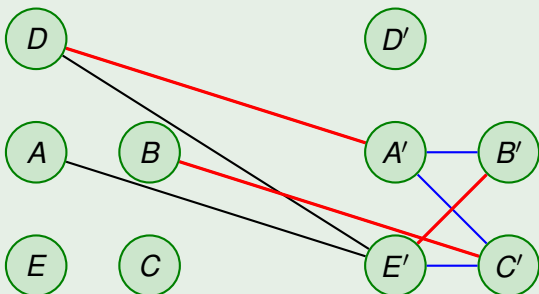
## The algorithm in action.

### The matching in the special graph.

#### Partial order



#### Matching graph

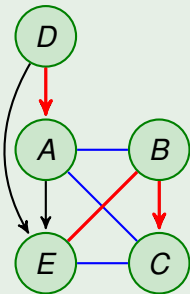


A **maximal matching** in the matching graph

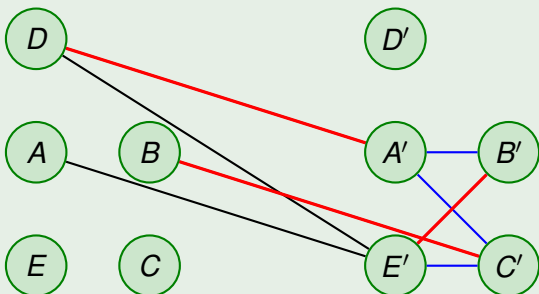
## The algorithm in action.

The matching in the special graph.

Partial order



Matching graph



A **maximal matching** in the matching graph induces **perfect path phylogenies**.