

# Near-Perfect Phylogenies and the Human Genome

Russell Schwartz

Departments of Biological Sciences and  
Computer Science  
Carnegie Mellon University

joint work with Guy Blelloch, Eran Halperin, Fumei Lam,  
Kedar Dhamdhere, R. Ravi, and Srinath Sridhar

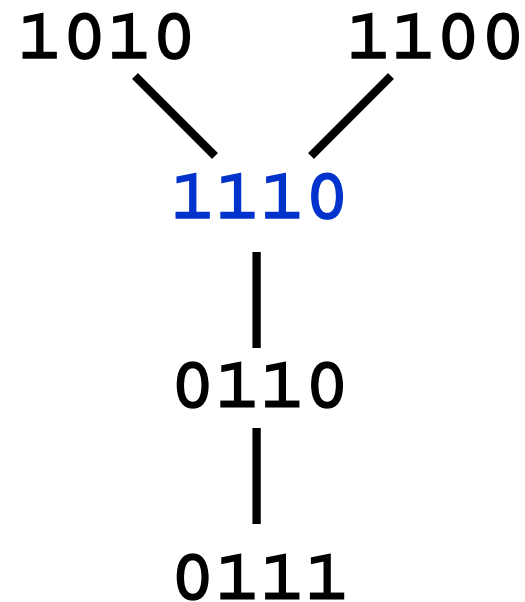
**Carnegie Mellon**

# The Phylogeny Problem

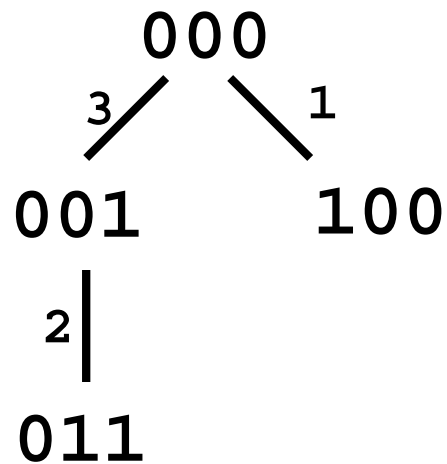
Given some set of variations in different individuals ...

... optimally explain the variants by mutation from a common ancestor

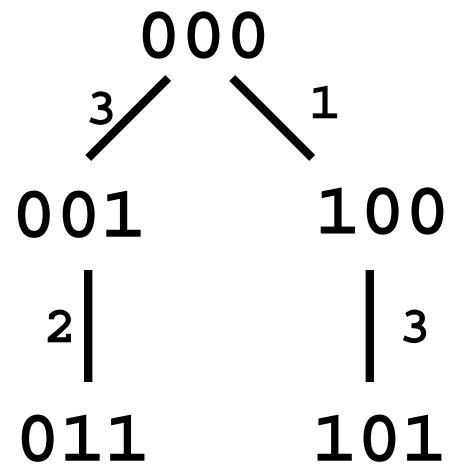
1010  
1100  
0110  
0111



# Perfect vs. "Near-perfect"



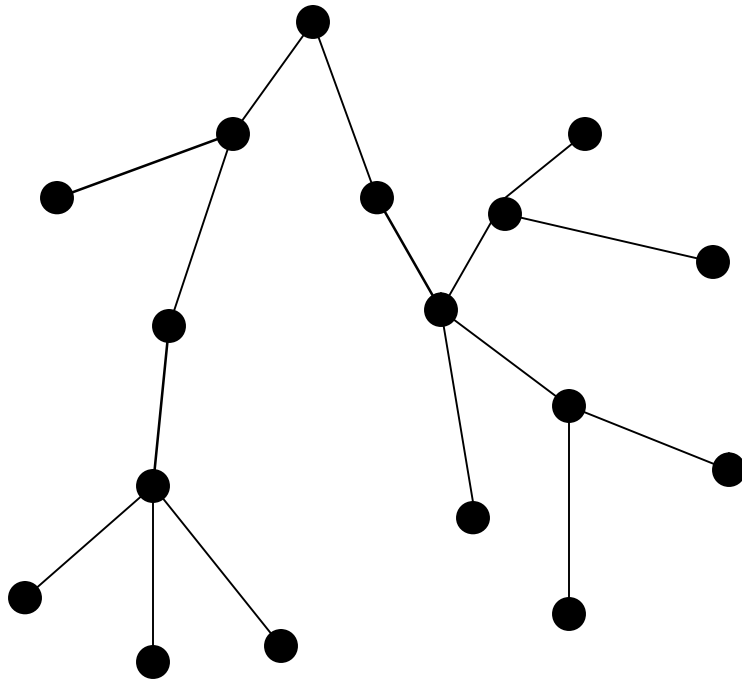
perfect



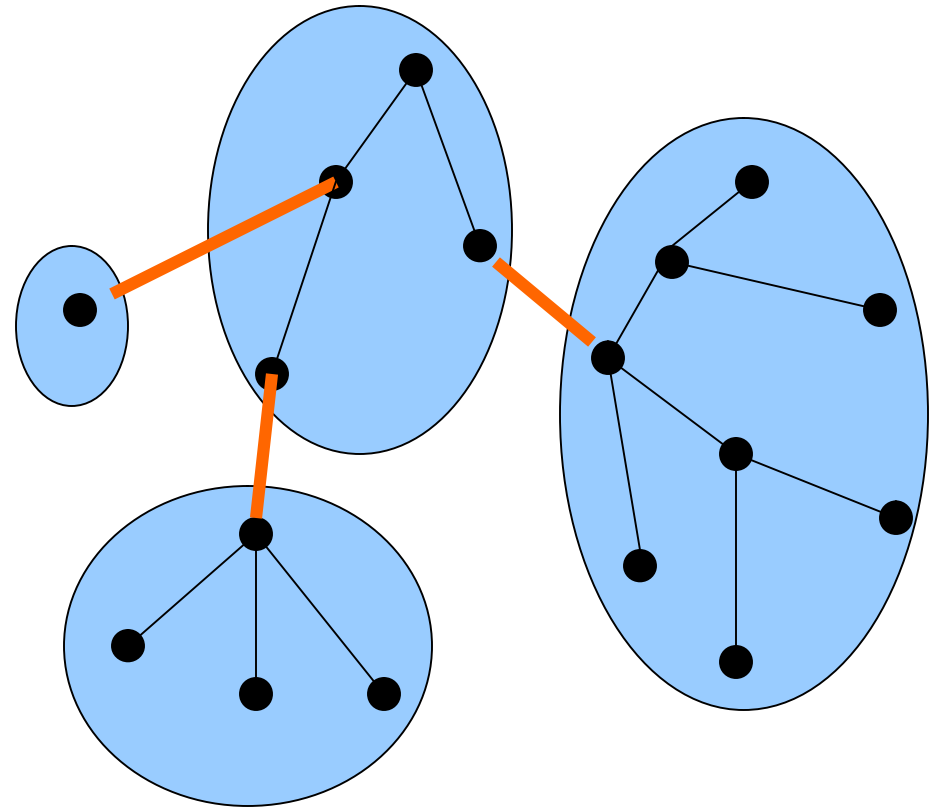
"near-perfect"

# Finding $q$ -Near-Perfect Phylogenies

There must be some optimal tree, which we want to find

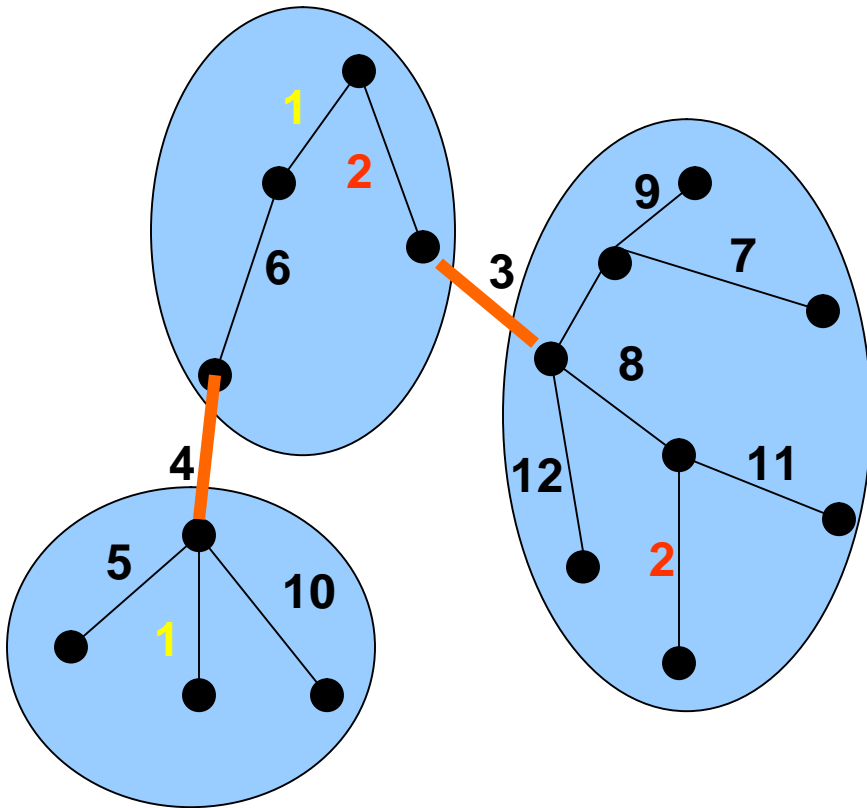


We will try to find it by splitting the tree into “super-nodes” each containing a perfect phylogeny



# How to Find the Super-nodes?

- Try to “guess” splitting edges that sit between duplicate edges



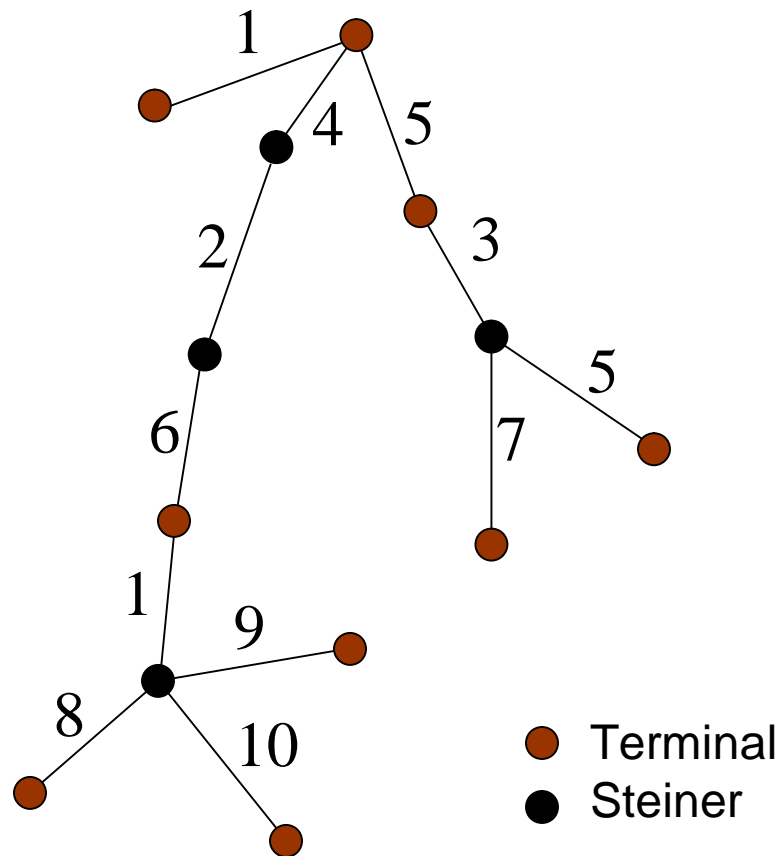
You can use 4-gamete constraints to make sure that guesses are correct at least half the time

# Walking Through the Algorithm

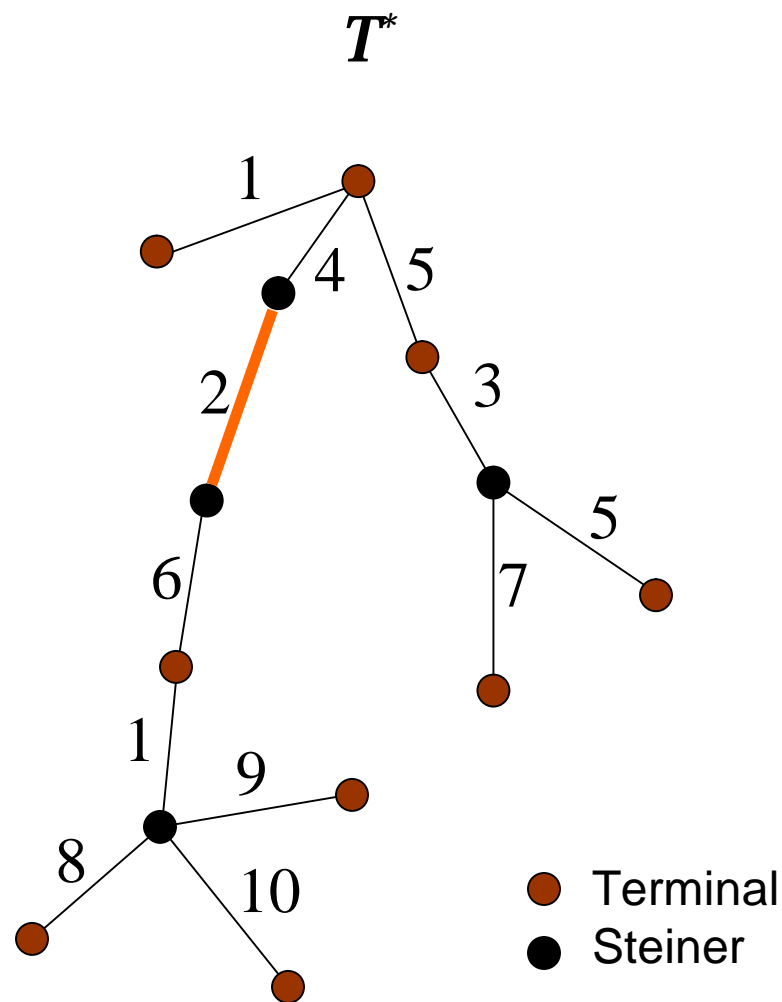
$T^*$

**buildTree(input M)**

1. Check if you can make a perfect phylogeny; if so, return it



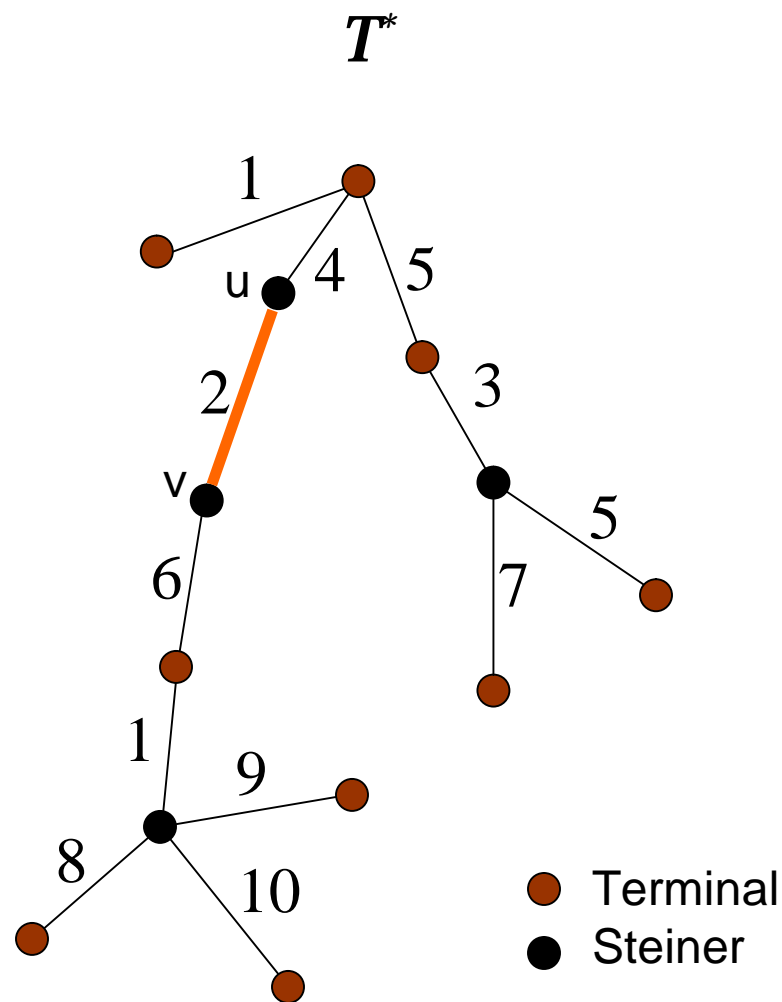
# Walking Through the Algorithm



**buildTree(input M)**

1. Check if you can make a perfect phylogeny; if so, return it
2. Guess a character that mutates exactly once and is in a four-gamete constraint

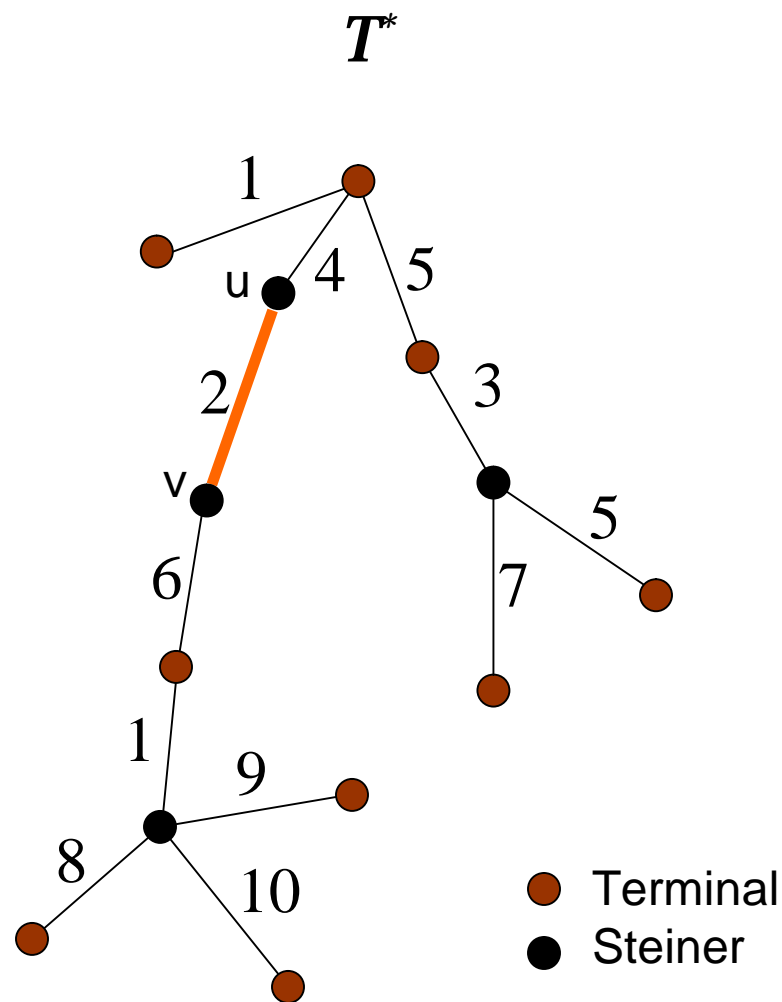
# Walking Through the Algorithm



**buildTree(input M)**

1. Check if you can make a perfect phylogeny; if so, return it
2. Guess a character that mutates exactly once and is in a four-gamete constraint
3. Guess the taxa adjoining the character

# Walking Through the Algorithm

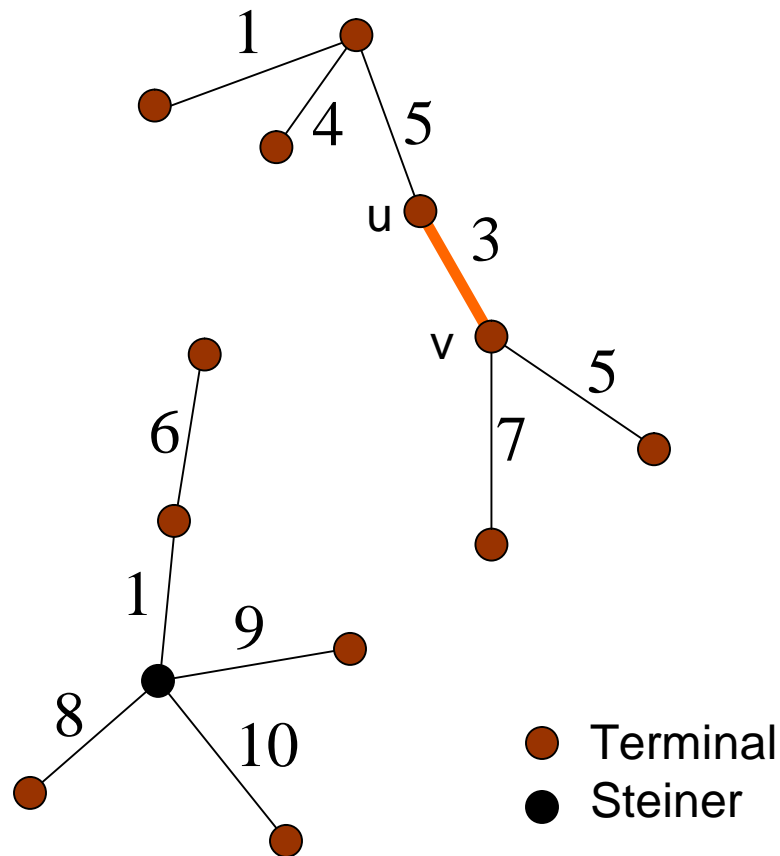


**buildTree(input M)**

1. Check if you can make a perfect phylogeny; if so, return it
2. Guess a character that mutates exactly once and is in a four-gamete constraint
3. Guess the taxa adjoining the character
4. Partition into two sub-problems
5. Solve the sub-problems

# Walking Through the Algorithm

$T^*$



**buildTree(input M)**

1. Check if you can make a perfect phylogeny; if so, return it
2. Guess a character that mutates exactly once and is in a four-gamete constraint
3. Guess the taxa adjoining the character
4. Partition into two sub-problems
5. Solve the sub-problems

# Runtime Analysis

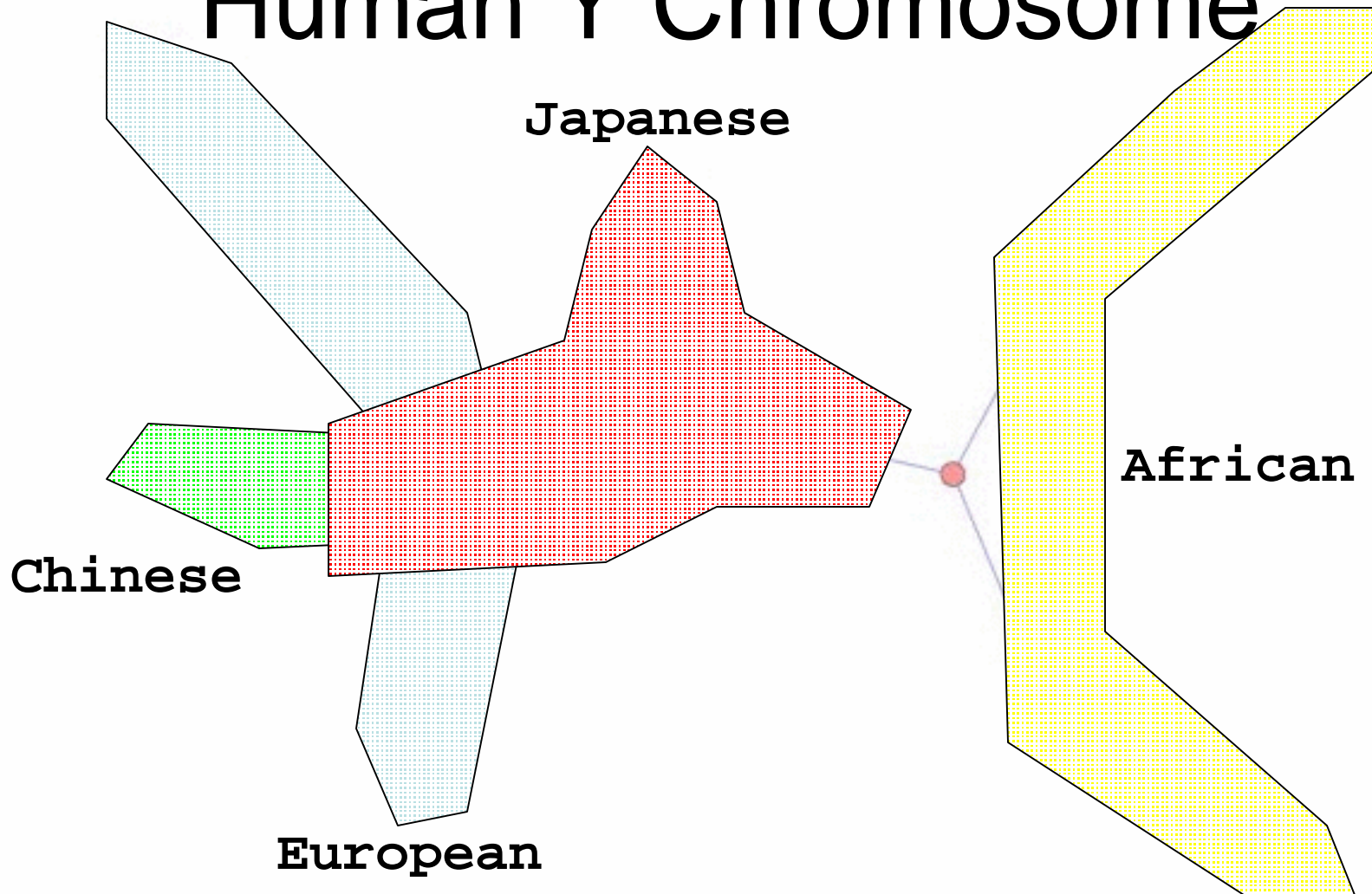
- Each guessed edge has  $\geq 50\%$  probability of success
- Each guess reduces imperfection by at least 1
- At most  $q$  extra guesses to find interfaces between super-nodes
- Implies  $\Pr[\text{finds an optimal tree}] \geq 0.25^q$

Derandomizes to a final provable bound of  
 $O(21^q + 8^q nm^2)$

# Some Haplotype Phylogeny Examples

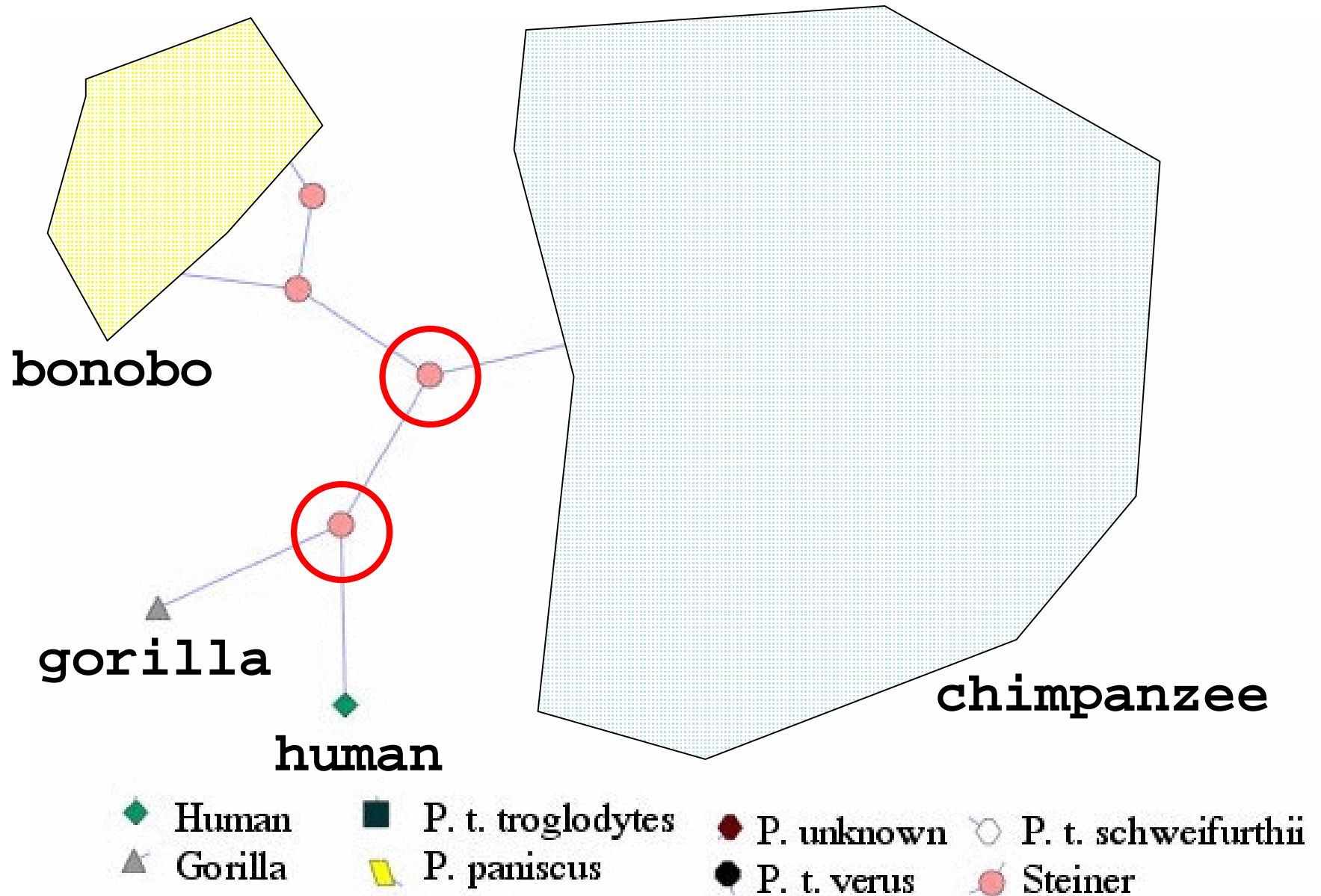
input	rows x cols ( $n \times m$ )	$q$	parsimony
human Y chrom. (Intl HapMap Consortium 2003)	150 x 49	1	16
chimp Y chrom. (Stone et al. 2002)	15 x 98	1	99
chimp mtDNA (Stone et al. 2002)	24 x 1041	2	63
human mtDNA 1 (Wirth et al. 2004)	13 x 48	3	30
human mtDNA 2 (Wirth et al. 2004)	26 x 48	7	43
<i>Pseudomonas</i> bacterial DNA (Merimaa et al. 2005)	17 x 1510	7	96

# Human Y Chromosome



- ◆ CEPH
- ▲ Japanese
- ◻ Yoruba - Africans
- Steiner
- ◻ Chinese
- ▲ Chinese/  
Japanese
- ◆ CEPH/Chinese/  
Japanese

# Primate mitochondrial DNA

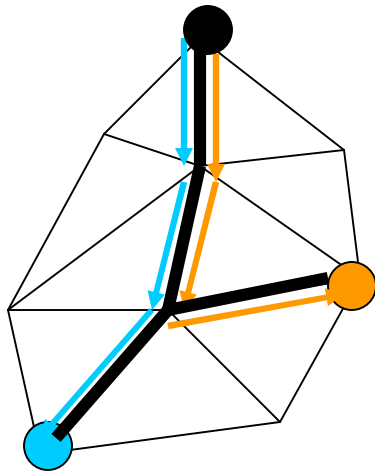


# Can We Speed It Up Without Losing Provable Optimality?

- Yes, with an integer linear program (ILP):
- An ILP is defined by
  - a set of variables (either integer or real)
    - e.g.,  $x_1 \in \{0, 1\}$ ,  $x_2 \in \{0, 1\}$ ,  $x_3 \in \mathcal{R}$
  - a set of linear constraints
    - e.g.,  $x_1 + 2x_2 - x_3 \geq 10$
  - a linear objective function
    - e.g., minimize  $2x_1 - 3x_2 + x_3$

# An ILP for Phylogeny Inference

- Create a graph of all possible nodes and edges
- Create a “flow” to each input sequence from an arbitrary root
- Minimize edges needed to accommodate all flows



Input: edge weights ( $d_{u,v}$ )

Variables:

edge selections ( $s_{u,v}$ ), flows ( $f_{u,v}^t$ )

Constraints:

$$\sum_v f_{u,v}^t = \sum_v f_{v,u}^t, \quad u \neq t, u \neq 0$$

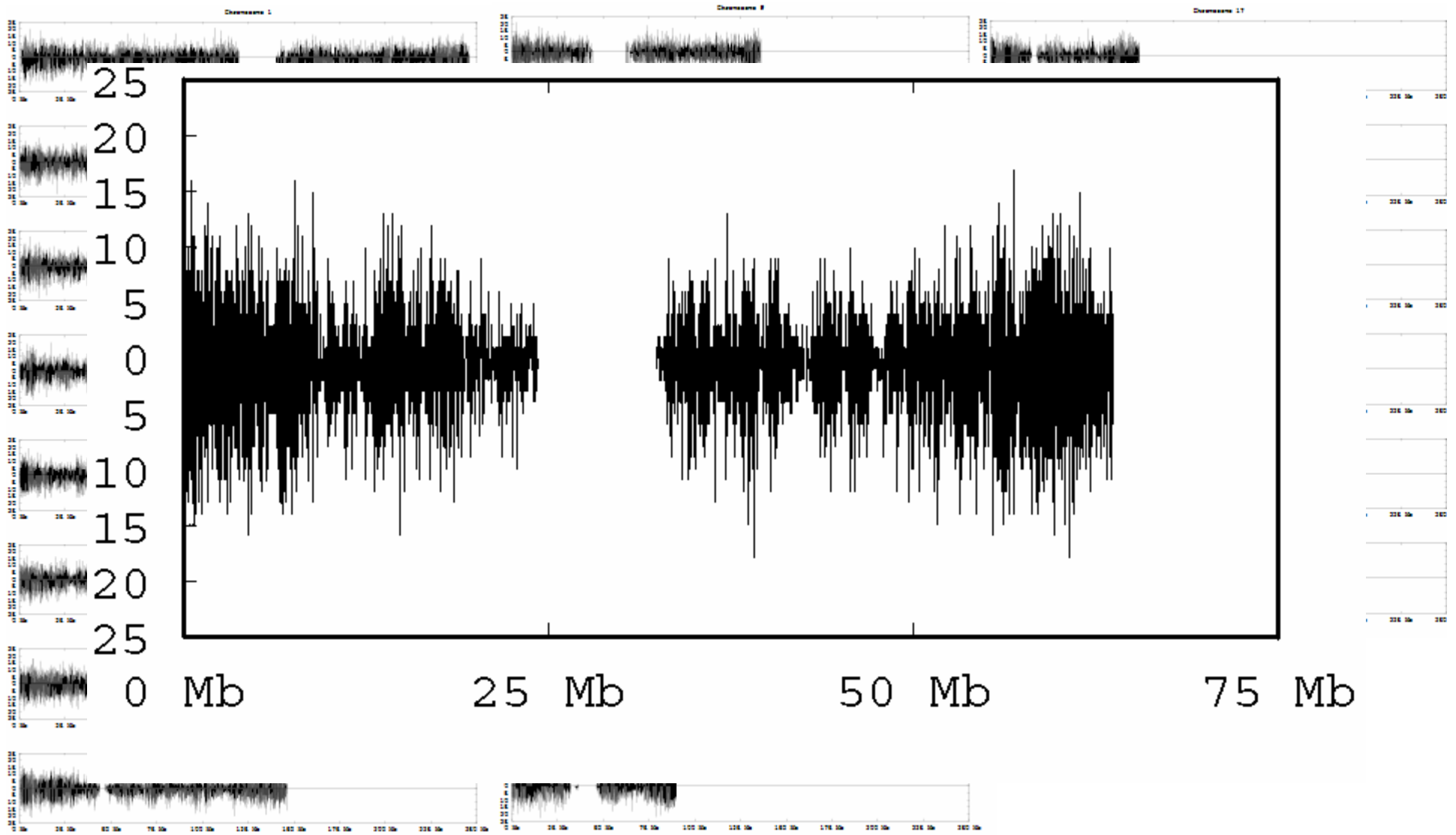
$$\sum_v f_{v,t}^t = 1, \quad \sum_v f_{t,v}^t = 0$$

$$\sum_v f_{0,v}^t = 1$$

$$0 \leq f_{u,v}^t \leq s_{u,v}$$

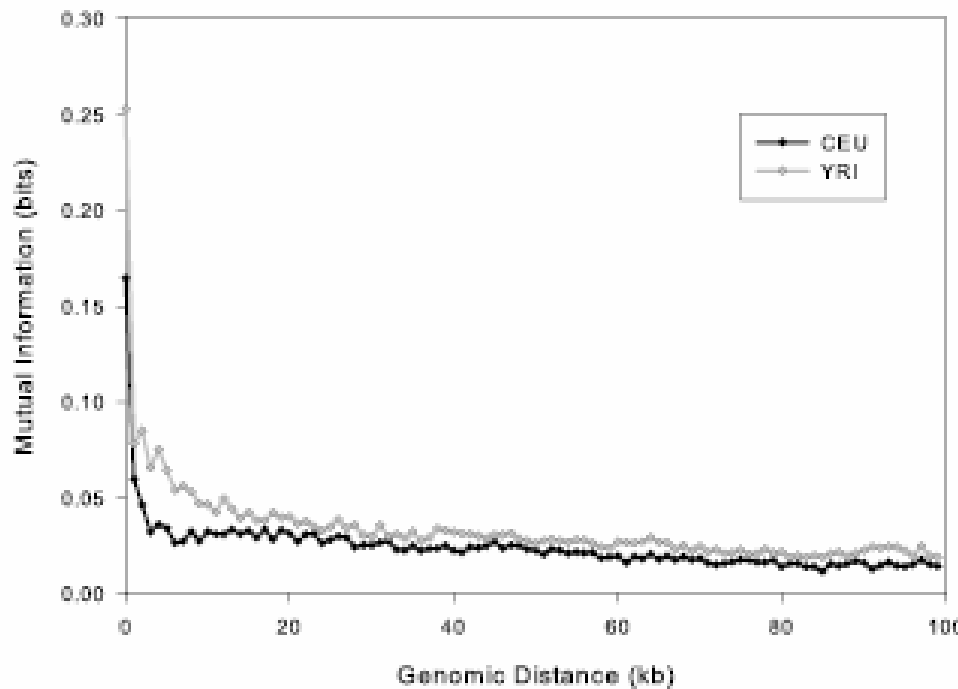
Objective:  $\min \sum_v d_{u,v} s_{u,v}$

# Application: Whole Genome Imperfection

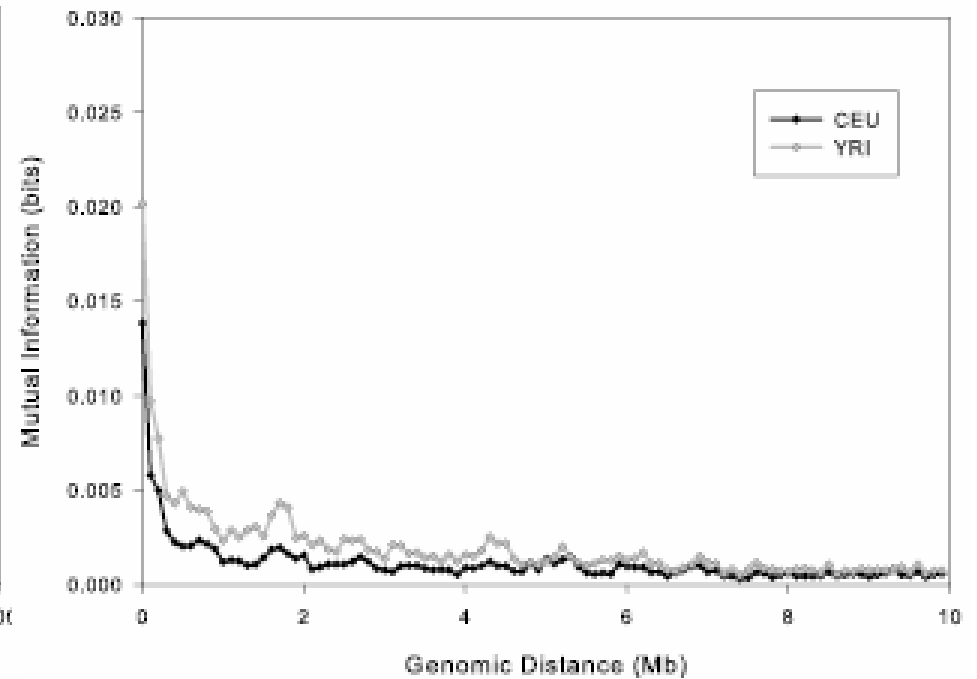


# Distance Dependence of Mutual Information

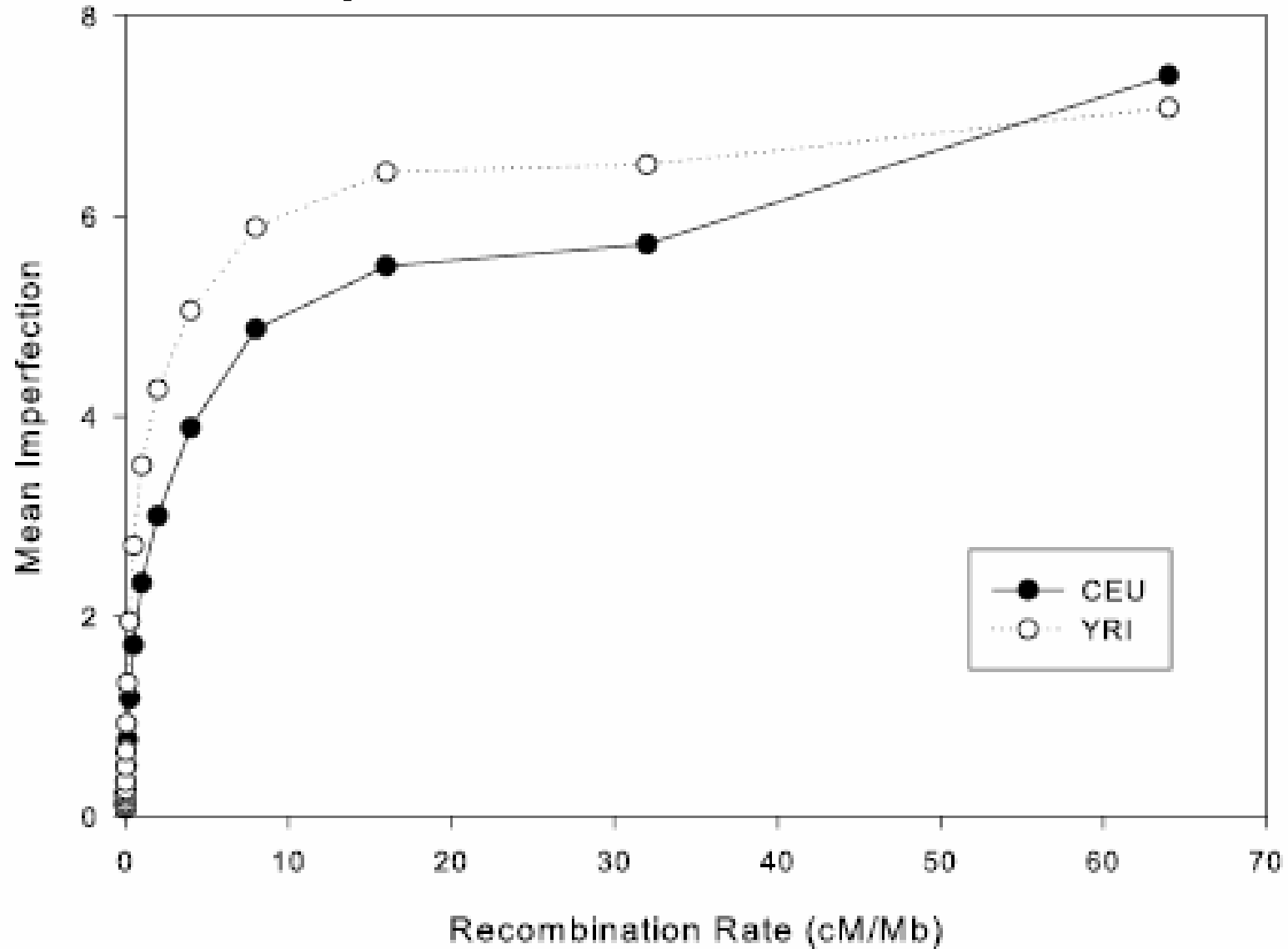
**kilobase scale**



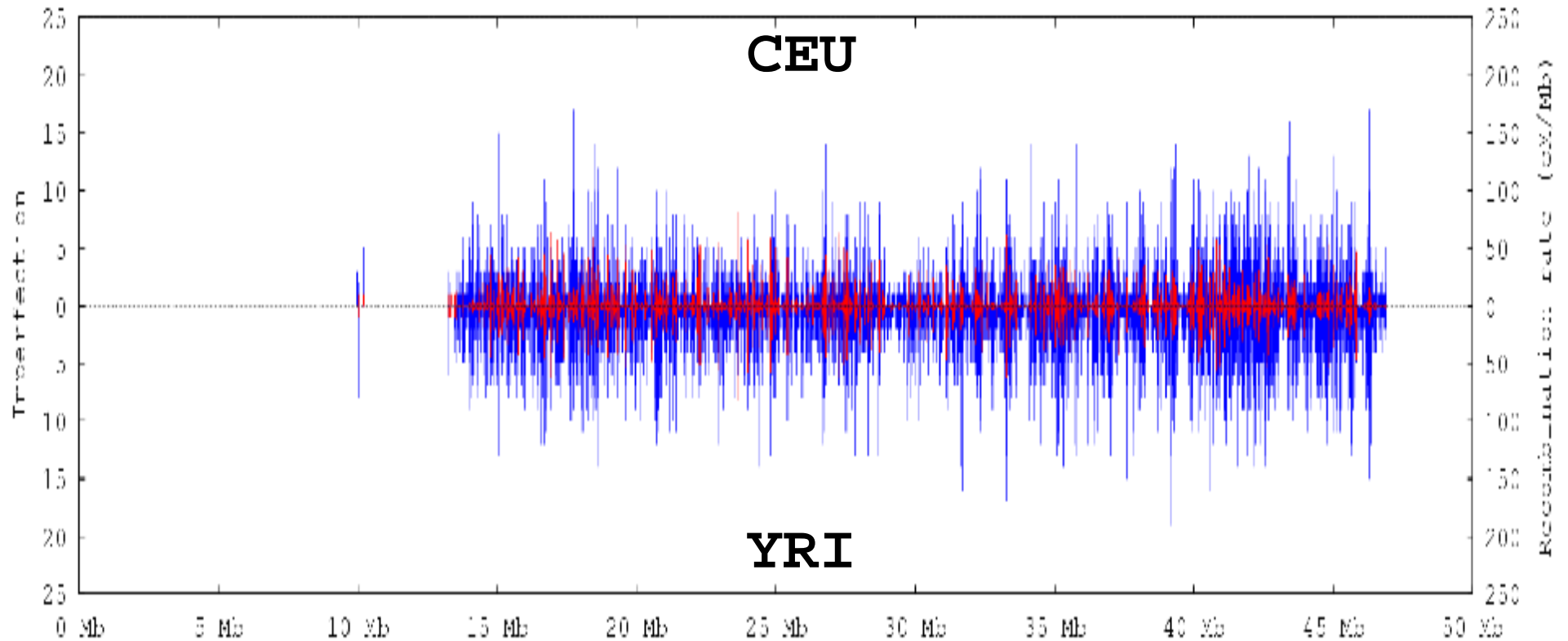
**megabase scale**



# Relationship to Recombination Rate



# Population Conservation



**imperfection**

**Chromosome 21 recombination rate**

**Correlation(CEU imperfection,CEU recombination)=0.489**

**Correlation(YRI imperfection,YRI recombination)=0.435**

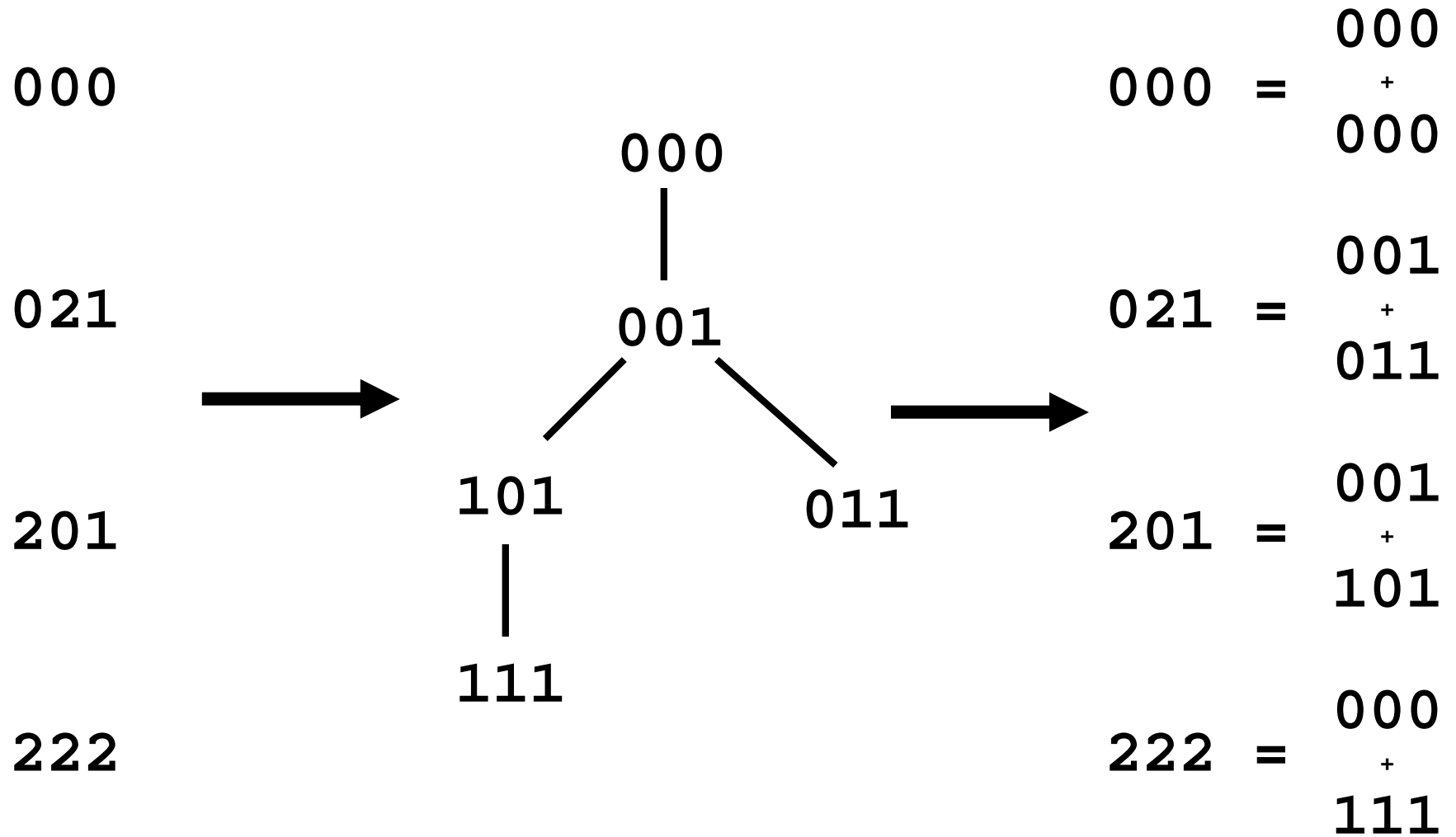
**Correlation(CEU imperfection,YRI imperfection)=0.603**

**Carnegie Mellon**

# Finding “Interesting” Sites in the Human Genome

central SNP	chr.	pos.	imp. (CEU)	imp. (YRI)	gene	variation
rs2305816	7	137,726,612	13	11	hypothetical protein (LOC136306)	C233F
rs2245220	20	4,700,718	7	16	prion protein 2 (PRND)	T174M
rs2294015	8	124,653,455	16	7	annexin A13 (ANXA13)	I272V (isoform a), I313V (isoform b)
rs7761137	15	44,198,345	15	8	calpain 11 (CAPN11)	N691S
rs2616490	8	89,038,253	11	12	matrix metalloproteinase 16 (MMP16)	V415I
rs4774	16	10,967,280	10	12	MHC class II trans activator (CIITA)	A500G
rs1051488	6	31,427,103	11	10	major histocompatibility complex class I, B (HLA-B)	A329T
rs3764879	17	4,929,683	9	12	phospholipase D2 (PLD2)	S821G
rs1760897	14	18,866,381	11	10	telomerase-associated protein 1 (TEP1)	P116S

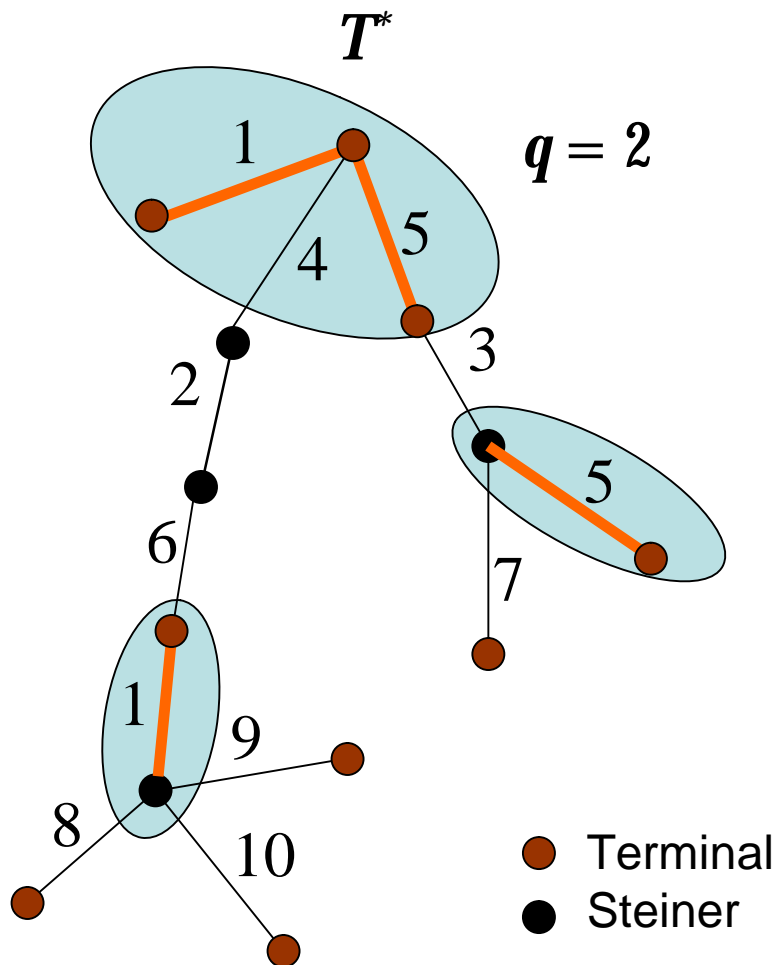
# What if We Only Have Genotypes?



# Algorithm Overview

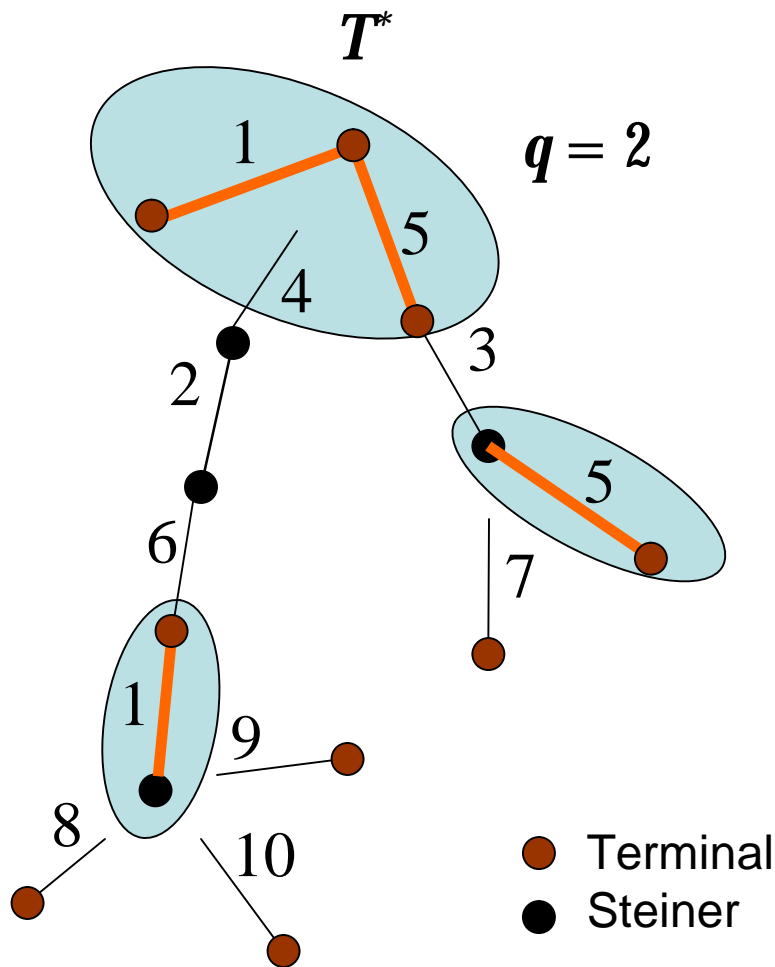
1. Guess the set of multiple mutants,  $Q$
  2. Remove  $Q$  from the set of genotypes  $G$
  3. Perform perfect phylogeny haplotyping on  $G$
  4. Identify other edges in conflict with  $Q$ , by enumerating insertion sites of  $Q$ -edges
  5. Phase  $Q$  columns using non-conflicting edges
- Run time:  $nm^{O(q)}$

# Algorithm Walk-Through



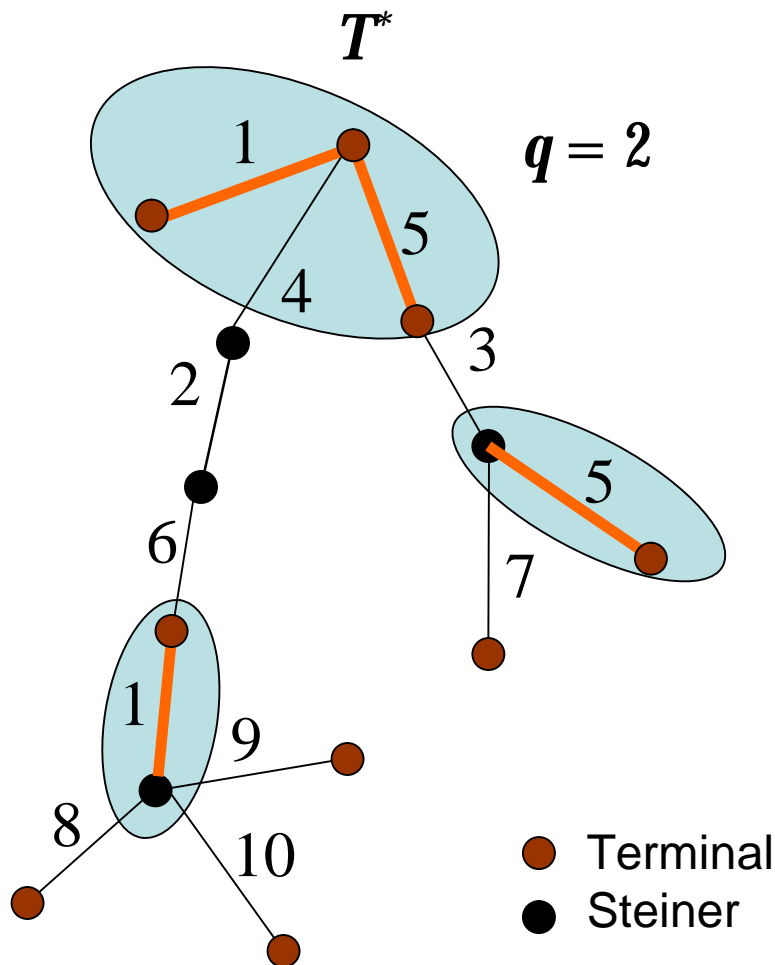
- Remove  $Q$  from  $G$
- Construct perfect phylogeny
- Add edges within super-nodes by brute force
- Link trees using Conflict Graph (+extra info)

# Algorithm Walk-Through



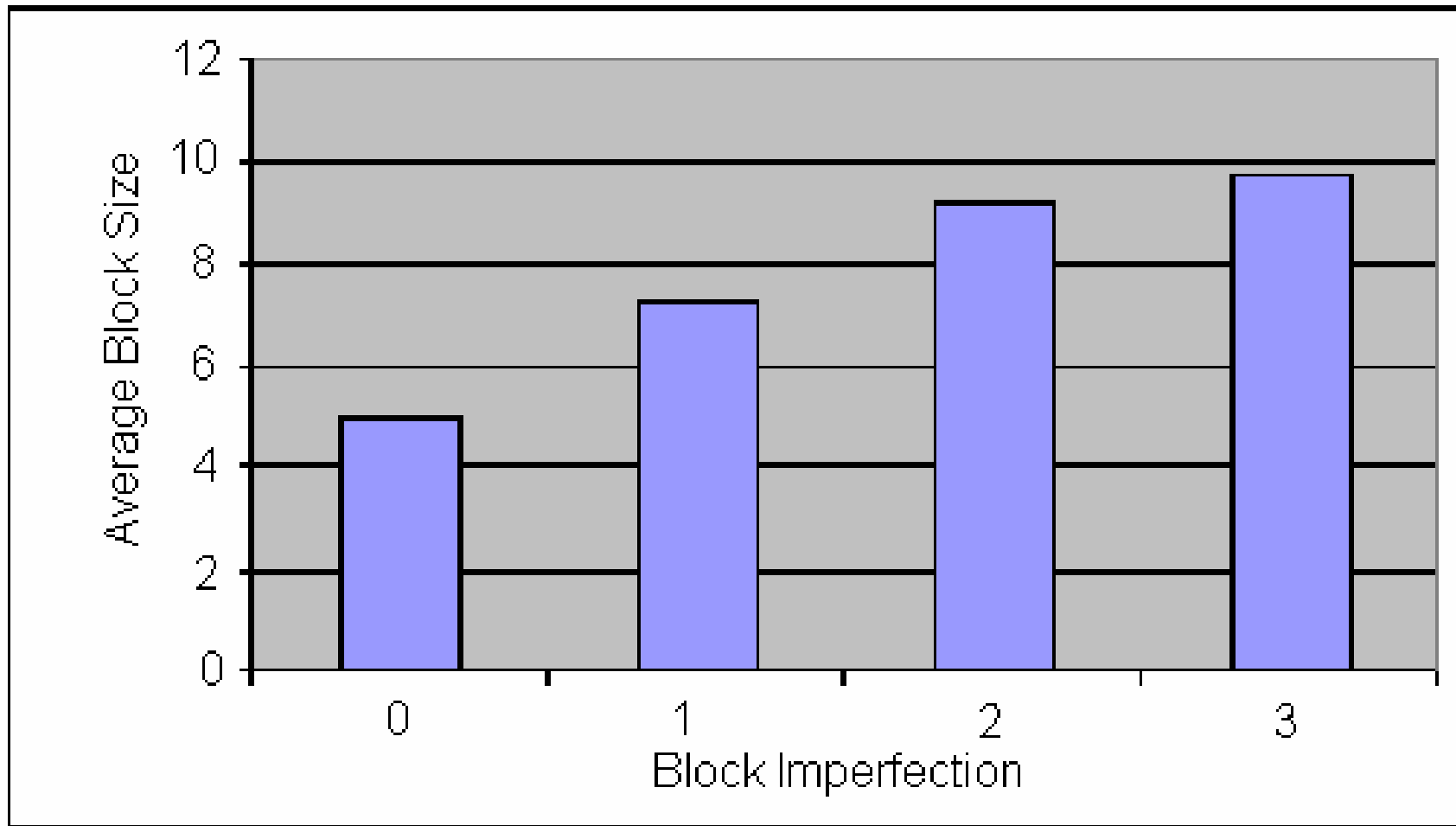
- Remove  $Q$  from  $G$
- Construct perfect phylogeny
- Add edges within super-nodes by brute force
- Link trees using Conflict Graph (+extra info)

# Algorithm Walk-Through



- Remove  $Q$  from  $G$
- Construct perfect phylogeny
- Add edges within super-nodes by brute force
- Link trees using Conflict Graph (+extra info)

# Finding Recombination-Free Blocks



mean block sizes versus allowed imperfection  
from unphased data (Encode 7q21 region)

# Improves Phasing Accuracy on Imperfect Data

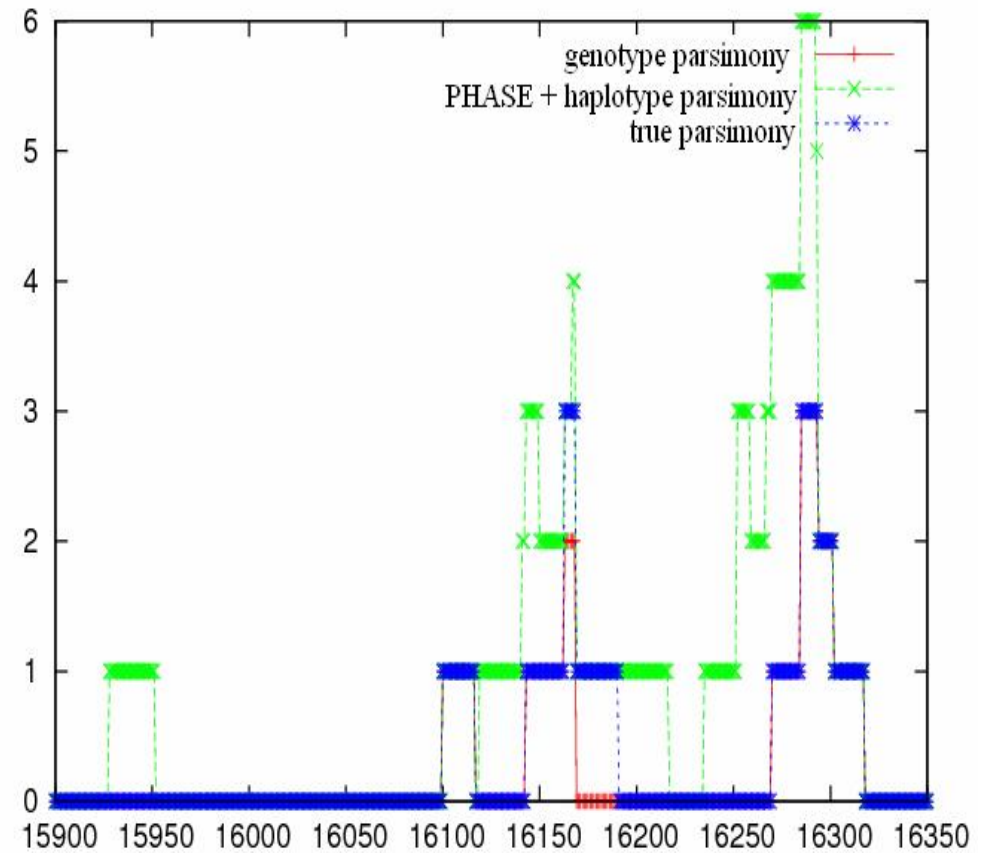
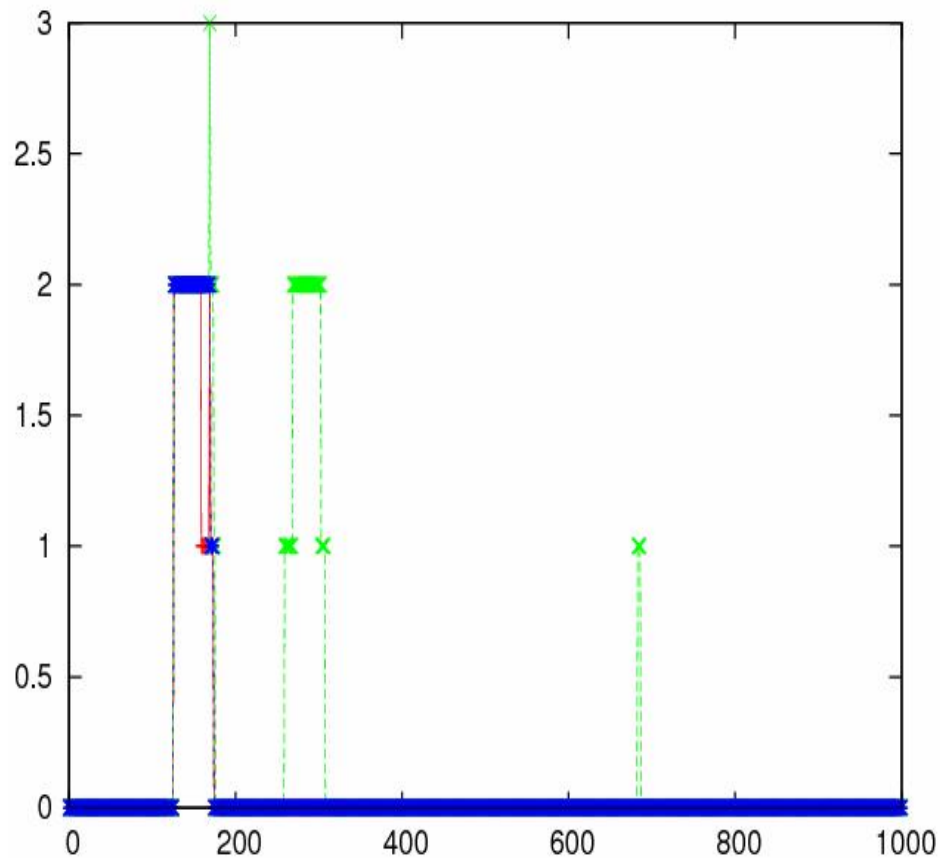
Imperfect ion	# Blocks	# SNPS	Error Rate			Run time (s)		
			PHASE	Haplotyper	Our Method	PHASE	Haplotyper	Our method
0	3497	20816	0.11	0.11	0.17	3521.33	337.18	17.21
1	461	4211	0.53	0.47	0.35	805.62	80.62	8.77
2	93	1266	0.83	0.68	0.55	268.02	59.28	1111.18

Patil et. al. (Chromosome 21) phase known data set

# Using ILP to Speed Up Genotype Phylogenies

1. Enumerate possible phases of the genotypes
2. Enumerate possible Steiner nodes for those phases
3. Use the haplotype ILP on the full set of possible nodes to find optimal phylogenies

# Unphased Phylogeny vs. Phase + Haplotype Phylogeny



# Conclusions

- It is possible to find provably optimal solutions to many problems previously solved only by heuristics
- Genotype phylogenies are harder than haplotype, but show significant improvement over phasing+phylogeny
- These techniques have many applications to characterizing patterns of recurrent mutation in the human genome

# References

- Dhamdhere, Sridhar, Blelloch, Halperin, Ravi, and Schwartz (2005) “A new algorithm for near-perfect binary phylogenetic trees.” CMU CS Tech Report CMU-CS-05-119.
- Sridhar, Dhamdhere, Blelloch, Halperin, Ravi, and Schwartz (2005) “Simple reconstruction of binary near-perfect phylogenetic trees.” *Intl Workshop on Bioinformatics Research and Applications*.
- Blelloch, Dhamdhere, Halperin, Ravi, Schwartz, and Sridhar (2006) “Fixed-parameter tractability of binary near-perfect phylogenetic tree reconstruction.” *Intl Colloq. on Automata, Languages, and Programming*.
- Sridhar, Dhamdhere, Blelloch, Halperin, Ravi, and Schwartz (2006) “Algorithms for efficient near-perfect phylogenetic tree reconstruction in theory and practice.” *IEEE Transactions on Computational Biology and Bioinformatics*.
- Sridhar, Blelloch, Ravi, and Schwartz (2006) “Optimal imperfect phylogeny reconstruction and haplotyping.” *Computational Systems Bioinformatics Conference*.