Integer Programming for Phylogenetic and Population-Genetic Problems with Complex Data (update 2014)
University of Oxford, July 2014

D. Gusfield
University of California, Davis

There are many important phylogeny problems with “complex data” caused by:

- Missing entries
- Data generated by complex biology, such as recombination or recurrent mutation
- Genotype (conflated) sequences, rather than simpler haplotype sequences

Most of these problems are NP-hard, although some elegant poly-time solutions exist (and are well-known) for simpler data.

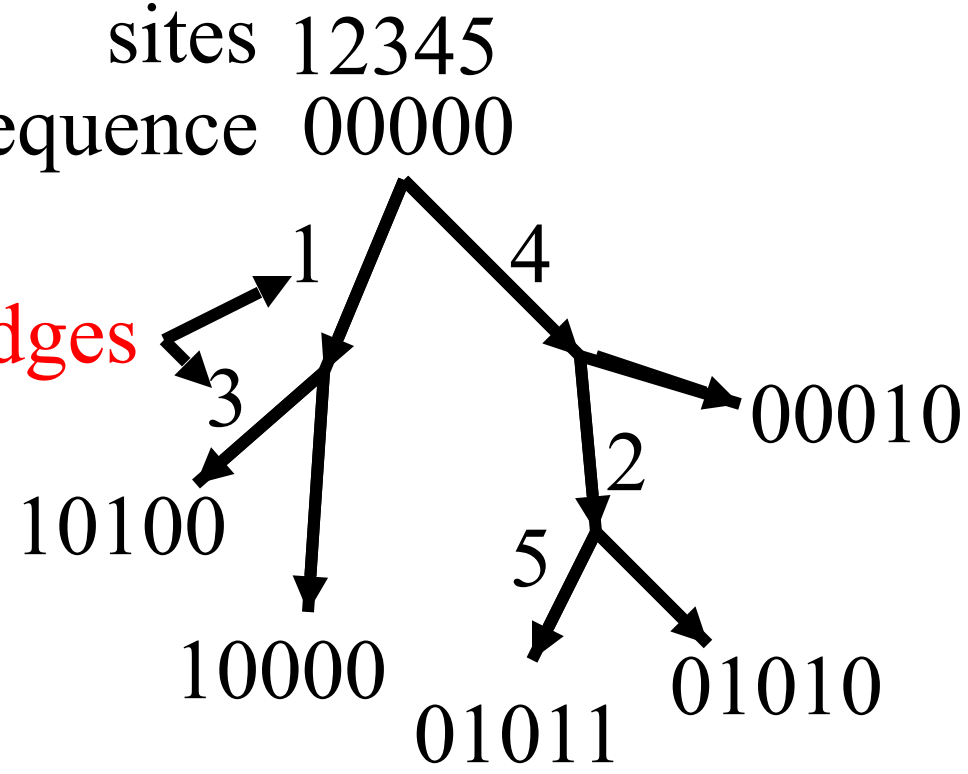
Question

Can Integer Programming efficiently solve these problems in practice on ranges of complex data of current interest in biology?

We have recently developed ILPs for over ten such problems and intensively studied their performance (speed, size and biological utility).

Starting Model: Perfect Phylogeny (infinite sites) model for binary sequences

Only one mutation per site allowed.



Site mutations on edges

The tree derives the set M:

10100
10000
01011
01010
00010

Extant sequences at the leaves

When can a set of sequences be derived on a perfect phylogeny?

Classic NASC: Arrange the sequences in a matrix. Then (with no duplicate columns), the sequences can be generated on a unique perfect phylogeny if and only if no two columns (sites) contain all four pairs:

0,0 and 0,1 and 1,0 and 1,1

This is the 4-Gamete Test

Each binary pair is called a gamete.

A pair of sites that has all four gametes is called incompatible, otherwise is called compatible.

For M of dimension n by m, the existence of a perfect phylogeny for M can be tested in O(nm) time and a tree built in that time, if there is one.

Problem M1: Missing data

Given ternary sequences (0s, 1s, ?s), change the ?s to 0s and 1s in order to minimize the resulting number of incompatible pairs of sites.

(Special case) Perfect Phylogeny with Missing Data:

Determine if the ?s can be set so that there are no resulting incompatibilities. NP-hard in general, but if the root of the required perfect phylogeny is specified, then the problem has an elegant poly-time solution (Pe'er, Sharan, Shamir).

Simple ILP for the Missing Data problem

Create a binary variable Y(i,p) for a ? in cell (i,p), indicating whether the cell will be set to 0 or to 1.

For each pair of sites p, q that could be made incompatible, let D(p,q) be the set of missing or deficient gametes in site pair p,q.

For each gamete a,b in D(p,q), create the binary variable B(p,q,a,b),

and create inequalities to set it to 1 if the Y variables for cells for sites p,q are set so that gamete a,b is created in some row for sites p,q.

Example

p q
---
0 0
? 1
1 0
? ?
? 0
0 ?

D(p,q) = {1,1; 0,1}

To set the B variables, the ILP will have inequalities
for each a,b in D(p,q), one for each row where a,b could be created at site p,q.

For example, for a,b = 1,1 the ILP has:
Y(2,p) <= B(p,q,1,1) for row 2
Y(4,p) + Y(4,q) -- B(p,q,1,1) <= 1 for row 4

Example continued

p	q
0	0
?	1
1	0
?	?
?	0
0	?

D(p,q) = {1,1; 0,1}

For a,b = 0,1 the ILP has:

Y(2,p) + B(p,q,0,1) => 1 for row 2
Y(4,q) -- Y(4,p) -- B(p,q,0,1) <= 0 for row 4
Y(6,q) -- B(p,q,0,1) <= 0 for row 6

The ILP also has a variable C(p,q) which is set to 1 if every gamete in D(p,q) is created at site-pair p,q.

In the example:

$$B(p, q, 1, 1) + B(p, q, 0, 1) -- C(p,q) <= 1$$

So, C(p,q) is set to 1 if (but not only if) the Y variables for sites p, q (missing entries in columns p, q) are
set so that sites p and q become incompatible.

If M is an n by m matrix, then we have at most nm Y variables; $2m^2$ B variables; $m^2/2$ C variables; and $O(nm^2)$ inequalities in worst-case.

Finally, we have the objective function:

$$\text{Minimize} \sum_{(p,q) \text{ in } P} C(p, q)$$

Where P is the set of site-pairs that could be made to be incompatible.

Empirically these ILPs solve very quickly, in fractions of seconds or seconds for n and m up to 100 and with up to 30% missing data (tested in
2007 with CPLEX 9 - now CPLEX 12 is available and much faster.)

The ILPs can also be used to impute the values of missing entries. Generally, the imputation is completely correct with up to 5% missing values, and the percentage increases as nm increases. This exploits huge redundancy in the haplotypes.

Problems related to M1

- Site-Removal Problem for complete data: Remove the minimum number of sites from the data, so that no incompatibilities remain. This is a common approach to incompatible data in phylogenetics. NP-hard.
- Site-Removal Problem with missing data (S1): Impute values for the missing entries to minimize the solution to the resulting Site-Removal Problem for complete data.

ILP for S1 - a simple extension to M1

- For each site i, let D(i) be a variable set to 1 if and only if site i is removed.
- For each site-pair p,q in P, add the inequality D(p) + D(q) -- C(p,q) => 0

  to the M1 formulation.

The objective function is now

Minimize Sum D(i)

Genotypes and Haplotypes

Each individual has two “copies” of each chromosome.

At each site, each chromosome has one of two alleles (states) denoted by 0 and 1 (motivated by SNPs)

0 1 1 1 0 0 1 1 0

1 1 0 1 0 0 1 0 0

Two haplotypes per individual

Merge the haplotypes

2 1 2 1 0 0 1 2 0 Genotype for the individual

Haplotyping (Phasing) Problem

- Biological Problem: For disease association studies, haplotype data is more valuable than genotype data, but haplotype data is hard to collect. Genotype data is easy to collect.

- Computational Problem: Given a set of n genotypes, determine the original set of n haplotype pairs that generated the n genotypes. This is hopeless without a genetic model or objective function that reflects the model. Many such models have been studied.

PPH model and objective

Given a set of genotypes, find (if possible) an explaining set of haplotypes (one pair for each genotype) that passes the ``four gamete test".

The PPH problem can be solved in linear time by a very complex algorithm. But it is simple to formulate an ILP for the PPH problem.

A Natural Extension of the PPH model

MinIncompat Problem (HM1): Haplotype to minimize the resulting number of incompatible pairs of sites.

NP-hard problem, but solved efficiently in practice by an ILP which is a simple modification of the ILP for problem M1.

The MinIncompat ILP becomes an ILP for PPH with the addition of a constraint that requires the solution to have value 0. The resulting ILP is feasible if and only if there is a PPH solution.

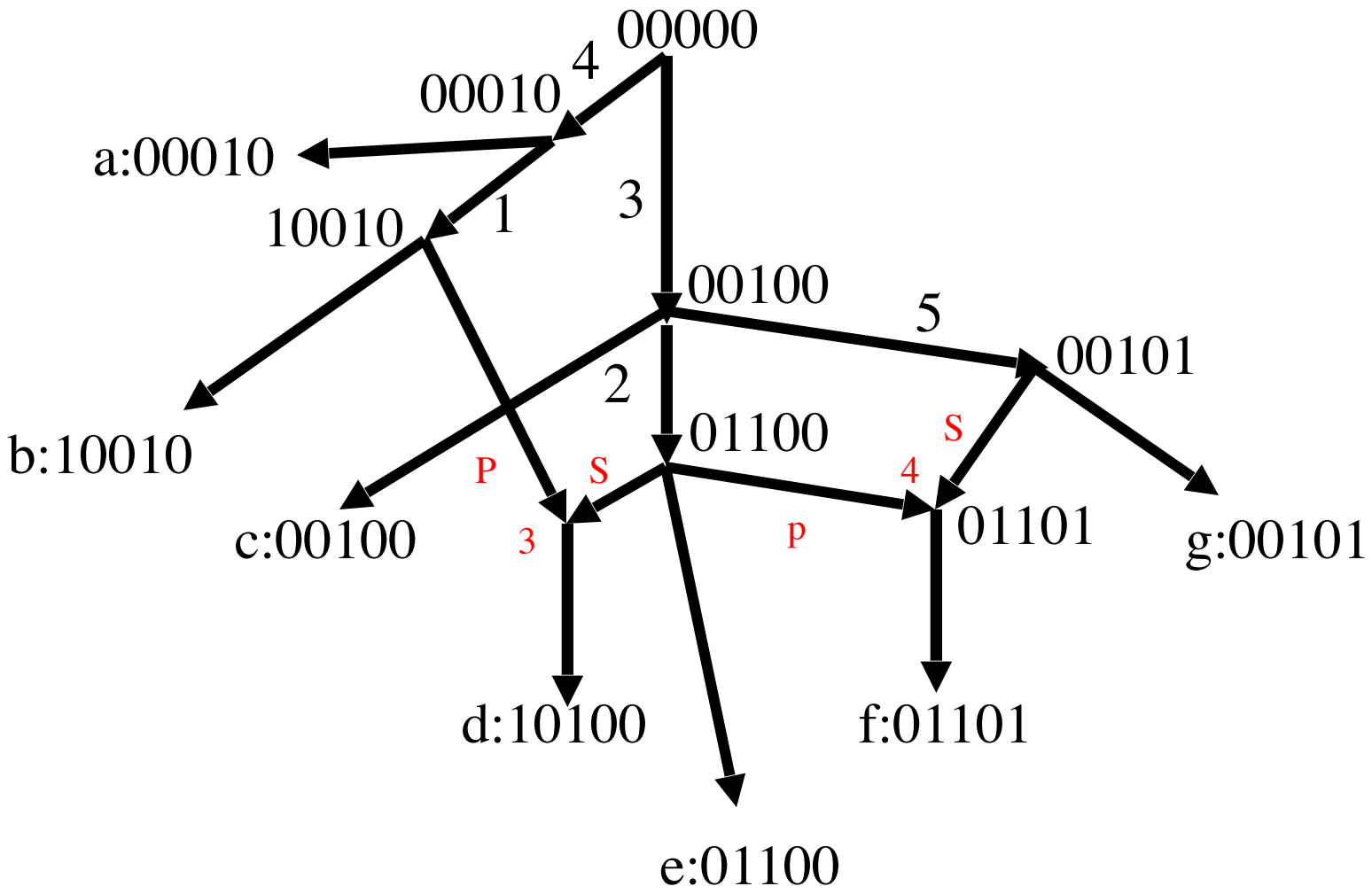
Allowing Limited Recombination: Galled-Trees

- An ARG where no recombination cycles share an edge is called a galled tree.
- A cycle in a galled-tree is called a gall.

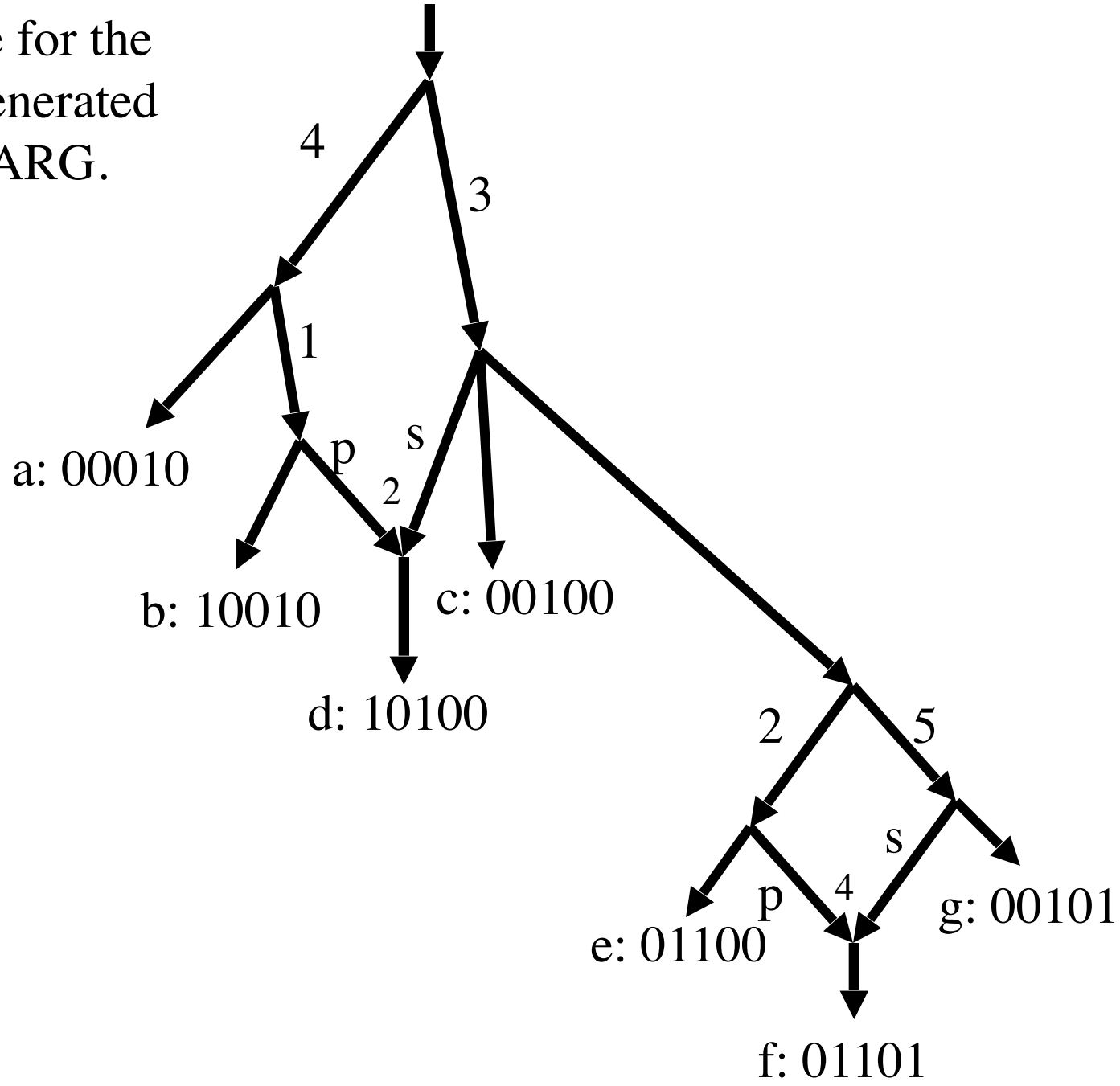
Recombination Cycles

- In an ARG, with a recombination node x, if we trace two paths backwards from x, then the paths will eventually meet.
- The cycle specified by those two paths is called a ``recombination cycle”.

ARG


00000
4
00010
a:00010
1
10010
3
00100
5
00101
2
b:10010
01100
S
P
S
4
c:00100
3
p
01101
g:00101
d:10100
f:01101
e:01100

A galled-tree for the sequences generated by the prior ARG.



Results about galled-trees

- Theorem: Efficient (provably polynomial-time) algorithm to determine whether or not any sequence set M can be derived on a galled-tree.
- Theorem: A galled-tree (if one exists) for M, produced by the algorithm, is a MinARG, i.e., it minimizes the number of recombinations used over all possible ARGs for M.
- Theorem: If M can be derived on a galled tree, then the Galled-Tree is ``nearly unique". This is important for biological conclusions derived from the galled-tree.

Allowing Limited Recombination

- Haplotyping and imputing missing data under the galled-tree model:
- Given a set of genotypes, can they be phased so that the resulting haplotypes can be generated on a galled-tree with up to k recombinations on k edge-disjoint cycles (i.e., galls)?

- The case of k = 0 is just the PPH problem.
- An ILP formulation exists for k > 0 based on a NASC for the existence of a galled-tree for haplotypes (due to Y. Song). It has been tested for k = 1, 2 and solves efficiently for n and m up to ??? (40 x 20).
- Not possible in 2007 (CPLEX 9), but works with CPLEX 12.6 and Gurobi.

Software

Perl script to generate the ILPs (for input to Cplex, Gurobi or other ILP solvers) can be found at:

wwwcsif.cs.ucdavis.edu/~gusfield