An Integer Linear Programming Solution for the Most Parsimonious Reconciliation Problem under the Duplication-Loss-Coalescence Model

Morgan Carothers*
mcarothers@hmc.edu
Department of Computer Science
Harvey Mudd College
Claremont, California

Joseph Gardi
jgardi@hmc.edu
Department of Computer Science
Harvey Mudd College

Gianluca Gross
ggross@seas.upenn.edu
Department of Computer Science
University of Pennsylvania
Philadelphia, Pennsylvania

Tatsuki Kuze
tkuze@hmc.edu
Department of Computer Science
Harvey Mudd College

Nuo Liu
ivliu@hmc.edu
Department of Computer Science
Harvey Mudd College

Fiona Plunkett
fplunkett@hmc.edu
Department of Computer Science,
Harvey Mudd College

Julia Qian
jqian@hmc.edu
Department of Computer Science
Harvey Mudd College

Yi-Chieh Wu
yjw@cs.hmc.edu
Department of Computer Science
Harvey Mudd College

ABSTRACT

Given a gene tree, a species tree, and an association between their leaves, the maximum parsimony reconciliation (MPR) problem seeks to find a mapping of the gene tree to the species tree that explains their incongruity using a biological model of evolutionary events. Unfortunately, when simultaneously accounting for gene duplication, gene loss, and coalescence, the MPR problem is NP-hard. While an exact algorithm exists, it can be problematic to use in practice due to time and memory requirements. In this work, we present an integer linear programming (ILP) formulation for solving the MPR problem when considering duplications, losses, and coalescence. Our experimental results on a simulated data set of 12 *Drosophila* species shows that our new algorithm is both accurate and scalable. Furthermore, in contrast to the existing exact algorithm, our formulation allows users to limit the maximum runtime and thus trade-off accuracy and scalability, making it an attractive choice for phylogenetic pipelines.

KEYWORDS

phylogenetics, reconciliation, gene duplication and loss, deep coalescence, integer linear programming

*Authors except the PI in alphabetical order.

BCB '20, September 21–24, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-7964-9/20/09.
https://doi.org/10.1145/3388440.3412474

ACM Reference Format:
Morgan Carothers, Joseph Gardi, Gianluca Gross, Tatsuki Kuze, Nuo Liu, Fiona Plunkett, Julia Qian, and Yi-Chieh Wu. 2020. An Integer Linear Programming Solution for the Most Parsimonious Reconciliation Problem under the Duplication-Loss-Coalescence Model. In *Proceedings of the 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics (BCB '20), September 21–24, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3388440.3412474

1 INTRODUCTION

Phylogenetic studies of gene and genome evolution often rely on understanding the relationship between two types of trees: a *species tree* that depicts the evolutionary history of a set of species, and a *gene tree* that depicts the evolutionary history of a set of genes sampled from these species. When the two trees are congruent, the gene tree topology can be explained through speciation events alone. However, gene trees can often differ from species trees due to several evolutionary processes [2]. For example, macro-evolutionary events such as gene duplication and gene loss can create a new locus or remove an existing locus from a genome, horizontal gene transfer can introduce a new locus from a contemporary species, and gene conversion can replace an existing homologous locus. Within a population, multiple lineages could fail to coalesce, leading to a phenomenon known as incomplete lineage sorting (ILS). Discordance can also arise due to species hybridization, gene fission and fusion, and recombination. Given a gene tree, a species tree, and an association between their leaves, the *reconciliation* problem seeks find a mapping of the gene tree "inside" the species tree that accounts for topological incongruence with respect to a given model of evolution.

Historically, for eukaryotic species, the two most popular reconciliation methods relied on either the duplication-loss [10, 21] or multispecies coalescent [16] models. In the last decade, new evolutionary models and associated reconciliation methods have

been introduced that simultaneously account for duplication, loss, and coalescence [3, 22, 25]. Though a few models also allow for transfers [3, 23], little evidence has been found for horizontal gene transfer in eukaryotes [6], making *DLC models* suitable for capturing eukaryotic evolution. Evaluations of the reconciliation methods show improved inference of orthologs, paralogs, duplications, and losses, particularly as the frequency of ILS increases [22, 27].

In this work, we rely on the DLCoal model [22], which is based explicitly on the popular duplication-loss [10, 21] and multispecies coalescent [16] models for evolution. Two approaches exist for reconciliation under this model: DLCoalRecon [22] finds a maximum *a posteriori* reconciliation and DLCpar [27] finds a maximum parsimony reconciliation (MPR). Because the probabilistic approach relies on a heuristic search and requires several biological parameters that may be difficult to estimate accurately, the MPR approach is more broadly applicable.

However, the MPR problem for the DLC model is NP-hard and even hard to approximate (APX-hard) [1]. While the current dynamic programming algorithm provides an exact solution and is fixed-parameter tractable (when parameterized by the number of genes that map to any given species)[1] [7], it has worst-case exponential runtime and can be problematic to use in practice due to time and memory requirements.

For insight, we can look to other complex phylogenetic problems. For example, given a collection of gene trees, the gene duplication problem, which seeks a species tree that implies the minimum number of gene duplication events, is NP-hard [15]. Given a gene tree and species tree, the problem of finding temporally feasible MPRs under the duplication-transfer-loss model is NP-complete [13, 20, 24]. And finding a reconciliation of minimum cost that simultaneously models the evolution of domains, genes, and species is NP-hard [12]. To overcome the problem complexity, heuristic algorithms are often employed, which do not guarantee optimality and may result in biologically unrealistic evolutionary scenarios. As an alternative, several exact approaches based on integer linear programming (ILP) formulations have been proposed and shown to be effective [4, 11, 13, 26].

In this work, we present an ILP formulation for solving the MPR problem under the DLC model. The corresponding tool is part of the DLCpar software, which is freely available for download at http://www.cs.hmc.edu/~yjw/software/dlcpar.

We demonstrate the utility of our ILP formulation by applying it to a simulated data set of 12 *Drosophila* species and comparing results with our previous dynamic programming (DP) algorithm. For every gene tree, ILP found a reconciliation with cost equal to that of DP. While ILP often has a longer runtime than DP, as tree sizes or gene tree-species tree incongruence increases, ILP becomes the more efficient approach. Furthermore, users can sacrifice accuracy by limiting the maximum runtime of the ILP solver, making it an attractive choice for phylogenetic pipelines.

[1]Let m denote the number of leaves in the species tree, n denote the number of leaves in the gene tree, and k denote the maximum number of nodes at the top or bottom of any species branch. The value of k is induced by the LCA species map. The worst-case running time of the DLCpar algorithm is $\mathbf{O}(m(f(k)+n))$, where $f(k) = B_k 2^{2k}(2k)!k^2$ and B_k denotes the k^{th} Bell number.

2 BACKGROUND

We start by reviewing prior work that formalizes the concept of reconciliations and maximum parsimony reconciliations under the DLC model [7, 18, 27].

2.1 Preliminaries

Throughout this work, the term *tree* refers to a rooted binary tree. Given a tree T, let $V(T)$ denote its node set and $E(T)$ denote its branch set. Let $L(T) \subset V(T)$ denote its leaf set, $I(T) = V(T) \setminus L(T)$ denote its set of internal nodes, and $r(T) \in I(T)$ denote its root node. For node $v \in V(T)$, let $c(v)$ denote its set of children, $p(v)$ denote its parent, and $e(v)$ denote the branch $(p(v), v)$. Define $\leq_T$ ($<_T$) to be the partial order on $V(T)$, where given two nodes u and v of T, $u \leq_T v$ ($u <_T v$) if and only if u is on the unique path between $r(T)$ and v (and $u \neq v$). The partial order $\geq_T$ ($>_T$) is defined analogously. In such a case, u is said to be a (strict) *ancestor* of v and v a (strict) *descendant* of u. We say that u and v are *comparable* if either $u \leq_T v$ or $v \leq_T u$, and u and v are *incomparable* otherwise.

Let a *species tree* S depict the evolutionary history of a set of species, and let a *gene tree* G depict the evolutionary history of a set of genes sampled from these species. To compare a gene tree with a species tree, let a *leaf map* $Le\colon L(G) \to L(S)$ label each leaf of the gene tree with the leaf of the species tree from which the gene was sampled.

Definition 2.1 (Species Map). Given G, S, and Le, let a *species map* $\mathcal{M} : V(G) \to V(S)$ map each node of G to the a node of S subject to the following constraints:

(1) If $g \in L(G)$, then $\mathcal{M}(g) = Le(g)$.
(2) If $g \in I(G)$, then for each $g' \in c(g)$, $\mathcal{M}(g) \leq_S \mathcal{M}(g')$.

Constraint 1 asserts that $\mathcal{M}$ extends the leaf map Le. Constraint 2 asserts that $\mathcal{M}$ satisfies the temporal constraints implied by S.

Given a species map $\mathcal{M}$, implied nodes may need to be added to each gene branch that spans multiple branches of the species tree (Supplemental Section S1.1).

Next, we define some useful sets. Given a species node $s \in V(S)$ and a species map $\mathcal{M}$, let $nodes(s)$ denote the set of gene nodes that map to s:

$$nodes(s) = \{\, g \mid g \in V(G);\ \mathcal{M}(g) = s \,\};$$

$bottoms(s)$ denote the subset of $nodes(s)$ that are leaves or whose children map to descendants of s:

$$bottoms(s) = \{g \mid g \in nodes(s);\ g \in L(G) \vee \forall g' \in c(g), g' \notin nodes(s)\};$$

and $tops(s)$ is equal to the bottom nodes of the parent species or contains the root of the gene tree:

$$tops(s) = \begin{cases} bottoms(p(s)) & \text{if } s \neq r(S) \\ \{r(G)\} & \text{otherwise.} \end{cases}$$

Note that $bottoms(s)$ and $tops(s)$ can be viewed as the set of gene nodes at the "bottom" or "top" of species branch $e(s)$, respectively.

2.2 The Labeled Coalescent Tree

The *labeled coalescent tree* (LCT, Figure 1) formalizes the notion of a reconciliation in the DLC model.

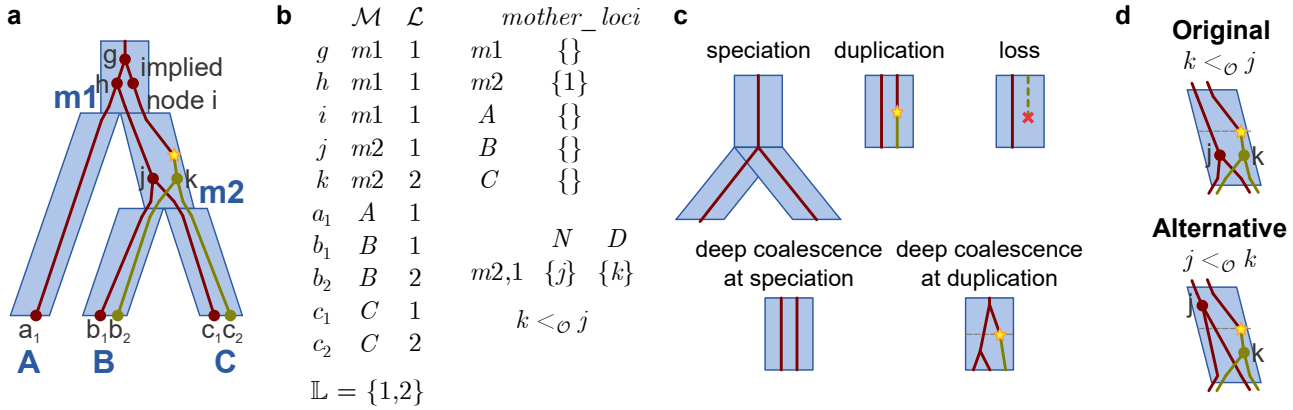

Figure 1: The labeled coalescent tree. (*a*) Evolution is represented using the LCT. In this example, a duplication (yellow star) creates a new locus, "locus 2" (yellow), from the original locus, "locus 1" (red), and lineages j and k fail to coalesce within species $m2$. (*b*) The LCT consists of a species map $\mathcal{M}$, a locus map $\mathcal{L}$, and a partial order $\mathcal{O}$. Sets $mother\_loci(\cdot)$ of loci and $N(\cdot,\cdot)$ and $D(\cdot,\cdot)$ of nodes necessary for the partial order are also shown. (*c*) Evolutionary events are depicted in the LCT. Except for speciation, evolution within a single species tree branch is shown. (*d*) An alternative scenario is presented for evolution in species $m2$. The new partial order induces an extra lineage at the time of the duplication. [Figure and caption adapted with permission from Mawhorter et al. [18] and Wu et al. [27].]

Definition 2.2 (Labeled Coalescent Tree). Given G, S, and Le, a *labeled coalescent tree (LCT)* for $\langle G, S, Le\rangle$ is a tuple $\langle \mathcal{M}, \mathbb{L}, \mathcal{L}, \mathcal{O}\rangle$, where

- $\mathcal{M}\colon V(G) \to V(S)$ is a **species map** that maps each node of G to a node of S.
- $\mathbb{L} \subset \mathbb{N}$ is a **locus set**, a finite set of natural numbers, each representing a locus that has evolved within the gene family.
- $\mathcal{L}\colon V(G) \to \mathbb{L}$ is a **locus map** that maps each node of G to a locus in $\mathbb{L}$.
- $\mathcal{O}$ is a **partial order** on $V(G)$ that represents the relative times of nodes. For each species node $s \in V(S)$, let $mother\_loci(s) \subset \mathbb{L}$ be the set of loci that yield a new locus in species s:

$$mother\_loci(s) = \{\, \mathcal{L}(g) \mid g \in I(G);$$
$$\exists g' \in c(g) : \mathcal{M}(g') = s, \mathcal{L}(g') \neq \mathcal{L}(g)\,\}.$$

  Then for each species node $s \in V(S)$ and each locus $l \in mother\_loci(s)$, consider the set of gene nodes $O(s,l) = N(s,l) \cup D(s,l)$, where $N(s,l)$ contains "original" gene nodes that map to species s and locus l, descend from locus l, and have multiple children:

$$N(s,l) = \{\, g \mid g \in V(G) \setminus \{r(G)\};\ \mathcal{M}(g) = s;$$
$$\mathcal{L}(g) = l;\ \mathcal{L}(p(g)) = l;\ |c(g)| > 1\,\},$$

  and $D(s,l)$ contains "duplication" gene nodes that map to species s and not locus l but immediately descend from locus l:

$$D(s,l) = \{\, g \mid g \in V(G) \setminus \{r(G)\};\ \mathcal{M}(g) = s;$$
$$\mathcal{L}(g) \neq l;\ \mathcal{L}(p(g)) = l\,\}.$$

  Note that the sets $N(s,l)$ and $D(s,l)$ are disjoint. Now consider a total order on $D(s,l)$; this order introduces $|D(s,l)|+1$ bins in which each node in $N(s,l)$ may occur. The total order on $D(s,l)$ and the partition of $N(s,l)$ represent the relative times of duplication nodes as well as the relative times of original nodes with respect to duplication nodes. Define $<_{\mathcal{O}}$ to be the partial order on $O(s,l)$, where given two nodes $g, g' \in O(s,l)$ where $g \neq g'$, then $g <_{\mathcal{O}} g'$ if and only if g precedes g' in time. Note that no order is induced on nodes of $N(s,l)$ in the same bin.

The LCT is subject to the following constraints:

(1) For each $g, g' \in L(G)$ where $g \neq g'$, if $\mathcal{M}(g) = \mathcal{M}(g')$, then $\mathcal{L}(g) \neq \mathcal{L}(g')$.
(2) For each $l \in \mathbb{L}$, there exists a $g \in V(G)$ such that $\mathcal{L}(g) = l$.
(3) For each $l \in \mathbb{L}$, there exists exactly one $g \in V(G)$ such that $L(g) = l$ and either $g = r(G)$ or $\mathcal{L}(p(g)) \neq l$.
(4) For each $s \in V(S)$, each $l \in mother\_loci(s)$, and each $g, g' \in O(s,l)$ where $g \neq g'$, if $g <_{\mathcal{O}} g'$, then $g \not\geq_G g'$.
(5) For each $s \in V(S)$, each $l \in mother\_loci(s)$, each $g \in D(s,l)$, and each $g' \in N(s,l)$, if $g' \in L(G)$, then $g <_{\mathcal{O}} g'$.

Constraint 1 asserts that extant genes (leaves) mapped to the same extant species (leaves) are mapped to different loci. Constraint 2 asserts that $\mathbb{L}$ includes only loci used by at least one gene. Constraint 3 asserts that every locus is created only once. Constraint 4 asserts that $\mathcal{O}$ satisfies the temporal constraints implied by G. Constraint 5 ensures that extant genes (leaves) are ordered in the last bin.[2]

Because the locus set $\mathbb{L}$ is defined by the locus map $\mathcal{L}$, we often represent an LCT using the reduced tuple $\langle \mathcal{M}, \mathcal{L}, \mathcal{O}\rangle$. Note that the species map $\mathcal{M}$ is defined first, then implied speciation nodes are added to G, then $\mathcal{L}$ is defined, and finally $\mathcal{O}$ is defined.

The LCT allows for several evolutionary events (Figure 1c, Supplemental Section S1.2). A *speciation* event corresponds to a locus present at the bottom of a species branch continuing at the same locus in at least one child species. As a speciation in the LCT reflects a speciation in the species tree, it is considered a null event.

[2]This constraint was not explicitly stated in previous work but was implicitly satisfied in simulations or enforced by properties of MPR solutions.

A *duplication* event corresponds to the creation of a new locus along a gene branch, which occurs when a gene node and its parent are mapped to different loci; such a gene branch is said to *have* a duplication. A *loss* event corresponds to a locus present at either the top of a species branch, or created via a duplication within the species branch, being no longer present at the bottom of the species branch. A *coalescence* event is, in fact, a *deep coalescence*, in which two or more lineages fail to coalesce; such failure can result in multiple lineages at speciations or duplications. Note that inferring speciation, duplication, loss, and coalescence at speciation events requires only the species map and locus map while inferring coalescence at duplication events also requires the partial order (Figure 1d).

2.3 Maximum Parsimony Reconciliations

Let C_D, C_L, C_C, and C_K denote the positive real-number costs associated with duplication, loss, and extra lineages at coalescence at speciation and coalescence at duplication events, respectively. The cost of reconciling G and S according to LCT $\langle \mathcal{M}, \mathcal{L}, O \rangle$ is defined as follows:

Definition 2.3 (Reconciliation Cost). Given G, S, Le, C_D, C_L, C_C, and C_K, the *reconciliation cost* of an LCT $\langle \mathcal{M}, \mathcal{L}, O \rangle$ for $\langle G, S, Le \rangle$ with d duplication events, ℓ loss events, c extra lineages due to coalescence at speciation events, and k extra lineages due to coalescence at duplication events is $\mathcal{R}_{\langle \mathcal{M}, \mathcal{L}, O \rangle} = C_D \cdot d + C_L \cdot \ell + C_C \cdot c + C_K \cdot k$.

Given G, S, Le, C_D, C_L, C_C, and C_K, the objective of the most parsimonious reconciliation (MPR) problem is to find an LCT for $\langle G, S, Le \rangle$ with minimum reconciliation cost (Supplemental Section S1.3). The solution to this problem is not necessarily unique.

Next, we define optimality of LCT components.

Definition 2.4 (Optimal LCT Components). A species map $\mathcal{M}^*$ is said to be optimal if there exists a locus map $\mathcal{L}$ and a partial order O such that $\langle \mathcal{M}^*, \mathcal{L}, O \rangle$ solves the MPR problem. Given a species map $\mathcal{M}$, a locus map $\mathcal{L}^*$ is said to be optimal if there exists a partial order O such that $\langle \mathcal{M}, \mathcal{L}^*, O \rangle$ solves the MPR problem. Given a species map $\mathcal{M}$ and locus map $\mathcal{L}$, a partial order O^* is said to be optimal if $\langle \mathcal{M}, \mathcal{L}, O^* \rangle$ solves the MPR problem.

Henceforth, the term MPR refers to an LCT that solves the MPR problem. We previously showed that the species map $\mathcal{M}^*$ is optimal if and only if $\mathcal{M}^*$ is the lowest common ancestor (LCA) map [7, 27].

3 AN ILP FORMULATION

In this section, we present an integer linear programming formulation for solving the MPR problem under the DLC model. Table 1 summarizes the definitions of ILP variables and provides a key to the notation used in this section.

3.1 Preliminaries

To formulate the ILP problem, we start by setting the optimal species map $\mathcal{M}^*$ to be the LCA map, then prune the species tree to the subtree rooted at $\mathcal{M}^*(r(G))$ and add implied speciation nodes to the gene tree. Let S' denote the pruned gene tree and G' denote the gene tree with implied speciation nodes. The goal of the ILP formulation is to find an optimal locus map $\mathcal{L}^*$ and an optimal order O^*. To simplify notation, throughout the remainder of Section 3, we use S and G to denote S' and G' and $\mathcal{M}$ to denote $\mathcal{M}^*$.

To prevent double-counting of gene lineages, we require an ordering of some nodes. Specifically, we arbitrarily order nodes $tops(s)$ and $bottoms(s)$ by preorder traversal of the gene tree. Given two nodes g and g' in a set, we say that $g' \prec g$ if g' precedes g in this order.

To quantify the number of variables and constraints in the ILP formulation, let m denote the number of leaves in the (input) species tree, h denote the height of the (input) species tree, and n denote the number of leaves in the (input) gene tree. Note that the gene tree has $2n-1$ total nodes and $2n-2$ branches. The pruned species tree has height $h' \leq h$, and adding implied speciation nodes creates at most $h'+1$ nodes along any gene branch. Let n' be the number of nodes in the gene tree with implied speciation nodes. Then $n' \leq (2n-1)+(2n-2)(h'+1)$. Given a set of n distinct elements, let $C(n,k)$ and $P(n,k)$ denote the number of combinations and permutations, respectively, of a subset of k elements.

3.2 Variables and Constraints

3.2.1 LCT Representation. We first define variables to represent the LCT and ensure its constraints are satisfied. Given a locus map, we can define the locus set and thus satisfy LCT constraint 2. For convenience, rather than specify the locus map directly, we specify the placement of duplications along gene branches. Via a trivial extension of Theorem 1 of Bork et al. [1], these two characterizations are equivalent, and thus, we can use locus maps and duplication placements interchangeably. By requiring a duplication to create a new locus, a duplication placement automatically satisfies LCT constraint 3. Thus, we seek an ILP formulation that can represent an LCT as duplications along gene branches and a partial ordering of nodes subject to LCT constraints 1, 4, and 5.

Duplication Variables For each gene branch $e(g) \in E(G)$, we define a binary variable d_g that denotes whether $e(g)$ has a duplication. If $e(g)$ has a duplication, then $d_g = 1$, and otherwise $d_g = 0$.

LCT constraint 1 requires that two gene leaves mapped to the same species belong to different loci, or equivalently, there must be at least one duplication on the path between any two gene leaves mapped to the same species. Given a pair of gene nodes $g_1, g_2 \in V(G)$, let $P_{g_1 g_2}$ denote the edges on the unique path from g_1 to g_2. Then for each pair of gene leaves $g_1, g_2 \in L(G)$ such that $g_1 \neq g_2$ and $\mathcal{M}(g_1) = \mathcal{M}(g_2)$,

$$\sum_{e(g) \in P_{g_1 g_2}} d_g \geq 1.$$

Since one duplication variable is created for each gene branch, there are $n'-1$ variables. The number of duplication constraints is upper bounded by $C(n', 2)$, the number of pairs of gene leaves.

Order Variables The LCT requires only a partial order on gene nodes in the same species and locus. However, because we are simultaneously inferring the locus map by placing duplications, we must instead infer a total ordering on nodes in the same species.

Variables	Definitions
LCT	
$d_g \in \{0,1\}$	$d_g = 1$ iff gene branch $e(g) \in E(G)$ has a duplication
$o_{g_1g_2} \in \{0,1\}$	$o_{g_1g_2} = 1$ iff gene node $g_1 \in V(G)$ precedes some other gene node $g_2 \in V(G)$ in time
Events	
$\ell_{sg} \in \{0,1\}$	$\ell_{sg} = 1$ iff gene node $g \in tops(s)$ becomes lost in species branch $e(s) \in E(S)$
$c_{sg} \in \{0,1\}$	$c_{sg} = 1$ iff for gene node $g \in survived(s)$ with child node $g' \in nodes(s)$, gene branch $e(g') = (g, g')$ is an extra lineage at the top of species branch $e(s) \in E(S)$
$k_g \in \mathbb{N}_0$	$k_g = 0$ if gene branch $e(g) \in E(G)$ has no duplication; otherwise k_g is the number of extra lineages at the time of duplication along $e(g)$
Auxiliary	
$\omega_{g_1g_2} \in \{0,1\}$	$\omega_{g_1g_2} = 1$ iff for species node $s \in V(S)$, gene node $g_1 \in bottoms(s)$ and some other gene node $g_2 \in nodes(s)$, gene branch $e(g_1)$ does not have a duplication and gene branch $e(g_2)$ has a duplication
$p_{g_1g_2} \in \{0,1\}$	$p_{g_1g_2} = 1$ iff at least one edge along the path from gene node $g_1 \in V(G)$ to some other gene node $g_2 \in V(G)$ has a duplication
$\lambda_{sg} \in \{0,1\}$	$\lambda_{sg} = 1$ iff for species node $s \in V(S)$, gene node $g \in tops(s)$ is mapped to the same locus as some other gene node $g' \in tops(s)$ that precedes g
$\kappa_{g_1g_2} \in \{0,1\}$	$\kappa_{g_1g_2} = 1$ iff for species node $s \in V(S)$, gene node $g_1 \in nodes(s) \cup bottom\_children(s)$ and some other incomparable gene node $g_2 \in nodes(s)$ such that both g_1 and g_2 have parents, $e(g_2)$ has a duplication and $e(g_1)$ is a contemporary lineage at the time of duplication along $e(g_2)$

Table 1: Variables and definitions used in the ILP formulation.

Consider an ordered pair of gene nodes $g_1, g_2 \in V(G)$ where $g_1 \neq g_2$. We define a binary variable o_{g1g2} that denotes whether g_1 precedes g_2 in time. If g_1 precedes g_2 in time, then $o_{g_1g_2} = 1$, and otherwise $o_{g_1g_2} = 0$.

Next, we introduce the pairs of gene nodes over which $o_{g_1g_2}$ is defined, which can be separated into several cases:

(1) The nodes are in the same species and comparable, so their order is given by the topology of G. That is, for each species node $s \in V(S)$ and each pair of gene nodes $g_1, g_2 \in nodes(s)$ such that $g_1 \neq g_2$ and $g_1 <_G g_2$, we define two variables $o_{g_1g_2}$ and $o_{g_2g_1}$. We constrain these variables to have values $o_{g_1g_2} = 1$ and $o_{g_2g_1} = 0$.
(2) The nodes are in different species that are comparable, so their order is given by the species tree S and the species map $\mathcal{M}$. That is, for each species node $s \in V(S)$, we consider each child $s' \in c(s)$. For each pair of gene nodes $g_1 \in nodes(s)$ and $g_2 \in nodes(s')$, we define two variables $o_{g_1g_2}$ and $o_{g_2g_1}$. We constrain these variables to have values $o_{g_1g_2} = 1$ and $o_{g_2g_1} = 0$.
(3) Otherwise, for each species node $s \in V(S)$ and each pair of gene nodes $g_1, g_2 \in nodes(s)$ where $g_1 \neq g_2$, we define two variables $o_{g_1g_2}$ and $o_{g_2g_1}$.

LCT constraint 4 requires that the order satisfies the temporal constraints implied by G. This constraint is satisfied by our definitions in case 1. However, a valid set of order variables must satisfy additional constraints. Specifically, an ordering of g_1 relative to g_2 implies an ordering of g_2 relative to g_1, as expressed by

$$o_{g_1g_2} = 1 - o_{g_2g_1}.$$

Additionally, an ordering must satisfy the transitive property of inequality: if g_1 precedes g_2 and g_2 precedes g_3, then it must be that g_1 precedes g_3. This constraint is expressed by

$$o_{g_1g_3} \geq o_{g_1g_2} + o_{g_2g_3} - 1.$$

Because we create at most two order variables for each pair of gene nodes, the number of order variables is upper bounded by $P(n', 2) = 2C(n', 2)$. The number of order constraints is upper bounded $P(n', 2) + P(n', 3)$.

Lastly, we recall that, in defining a partial order, the timing of duplications introduces bins in which nodes in the same species and with the same mother locus may occur, and LCT constraint 5 requires that extant genes are ordered in the last bin. We introduce auxiliary variables and constraints to ensure this constraint is satisfied. For each species leaf $s \in Le(S)$, consider each two gene nodes $g_1 \in bottoms(s)$ and $g_2 \in nodes(s)$ where $g_1 \neq g_2$. We define a binary variable $\omega_{g_1g_2}$ that denotes which branches $e(g_1)$ and $e(g_2)$ have duplications. If $e(g_1)$ does not have a duplication and $e(g_2)$ does have a duplication, then $\omega_{g_1g_2} = 1$, and otherwise $\omega_{g_1g_2} = 0$. We note that variable $\omega_{g_1g_2}$ can be expressed using a logical AND operation with two operands, represented in the following constraints:

$$\omega_{g_1g_2} \leq 1 - d_{g_1}$$
$$\omega_{g_1g_2} \leq d_{g_2}$$
$$\omega_{g_1g_2} \geq d_{g_2} + (1 - d_{g_1}) - 1$$

By LCT constraint 5, if $e(g_1)$ does not have a duplication but $e(g_2)$ does, then g_2 must precede g_1 in time. Equivalently, if $\omega_{g_1g_2} = 1$, then $o_{g_2g_1} = 1$:

$$o_{g_2g_1} \geq \omega_{g_1g_2}$$

The number of ω variables is upper bounded by $P(n', 2)$, and the number of constraints is upper bounded by $4P(n', 2)$.

3.2.2 Counting Events. Next, we define how to count the evolutionary events induced by a LCT as encoded in ILP variables.

Number of Duplications The number of duplications is straightforward using the duplication variables and is given by

$$d = \sum_{e(g) \in E(G)} d_g.$$

Number of Losses In a LCT, loci are lost rather than genes. However, just as with order variables, because we are simultaneously inferring the locus map in our ILP formulation, we must have a different way of representing losses so that we can define variables without knowing the locus map. We note that in an MPR, a locus is never created via a duplication and then lost in the same species, as it is always more parsimonious to have never created the locus in the first place. Then, for a species $s \in V(S)$, if a locus $l \in \mathbb{L}$ is lost in the species, then the first gene node $g \in tops(s)$ mapped to locus l is said to *be* lost. We note that g cannot be lost if there exists some other gene $g' \in tops(s)$ such that g' is mapped to l and $g' \prec g$. Thus, each locus can contribute at most once to the number of losses.

For each species node $s \in V(S)$ and each gene node $g \in tops(s)$, we define a binary variable ℓ_{sg} that denotes whether g becomes lost in species s. If g becomes lost, then $\ell_{sg} = 1$, and otherwise $\ell_{sg} = 0$. Then, the number of losses is given by

$$\ell = \sum_{s \in V(S)} \sum_{g \in tops(s)} \ell_{sg}.$$

Before defining constraints for ℓ_{sg}, we introduce some auxiliary variables.

For each ordered pair of gene nodes $g_1, g_2 \in V(G)$ where $g_1 \neq g_2$, we define a binary variable $p_{g_1 g_2}$ that denotes whether at least one edge on the path from g_1 to g_2 has a duplication. If so, then $p_{g_1 g_2} = 1$, and otherwise $p_{g_1 g_2} = 0$. Path variables must be consistent with duplication variables, as expressed by the following constraints:

$$p_{g_1 g_2} \leq \sum_{e(g) \in P_{g_1 g_2}} d_g$$

$$\sum_{e(g) \in P_{g_1 g_2}} d_g \leq |P_{g_1 g_2}|\, p_{g_1 g_2}$$

The first constraint ensures that if there are no duplications on the path between g_1 and g_2, then $p_{g_1 g_2} = 0$. The second constraint ensures that, if there there is at least one duplication on the path between g_1 and g_2, then $p_{g_1 g_2} = 1$. There are $P(n', 2)$ path variables and $2P(n', 2)$ constraints.

For each species node $s \in V(S)$ and each gene node $g \in tops(s)$, let λ_{sg} be a binary variable that denotes whether g is mapped to the same locus as some other gene node $g' \in tops(s)$ that precedes g. If so, then $\lambda_{sg} = 1$, and otherwise $\lambda_{sg} = 0$. Note that if $\lambda_{sg} = 1$, the existence of g' means that g cannot be lost. λ variables must satisfy the following constraints:

$$\lambda_{sg} \leq \sum_{g' \in tops(s): g' \prec g} \left(1 - p_{gg'}\right)$$

$$\sum_{g' \in tops(s): g' \prec g} \left(1 - p_{gg'}\right) \leq |tops(s)|\, \lambda_{sg}$$

The first constraint ensures that if, for each gene node $g' \in tops(s)$ such that $g' \prec g$, there is at least one duplication along every path between g and g' so that $p_{gg'} = 1$, then $\lambda_{sg} = 0$. That is, g cannot be mapped to the same locus as any preceding g'. The second constraint ensures that if there exists a gene node $g' \in tops(s)$ such that $g' \prec g$ and there is no duplication on the path between g and g' so that $p_{gg'} = 0$, then $\lambda_{sg} = 1$. That is, g is mapped to the same locus as a preceding g'. Since an internal gene node can be at the top of two species, it can result in at most two λ variables. Thus, the number of λ variables is upper bounded by $2(n' - n)$, and the number of constraints is upper bounded by $4(n' - n)$.

Finally, we define the constraints on the loss variables ℓ_{sg} using these auxiliary variables:

$$\ell_{sg} \geq 1 - \lambda_{sg} + \left(\sum_{g' \in bottoms(s)} \left(p_{gg'}\right) - |bottoms(s)| \right)$$

That is, g becomes lost ($\ell_{sg} = 1$) if two conditions are satisfied. First, as noted before, to prevent double-counting of losses, we require that $\lambda_{sg} = 0$ so that no preceding gene node in $tops(s)$ is mapped to the same locus. Second, we require that for each $g' \in bottoms(s)$, there exists at least one duplication on the path between g and g' so that the locus of g no longer exists at the bottom of the species branch. We further note that, because we are minimizing the number of losses, $\ell_{sg} = 0$ unless these conditions are satisfied. Similar to λ variables, the number of loss variables and constraints is each upper bounded by $2(n' - n)$.

Number of Extra Lineages due to Coalescence at Speciation For each species node $s \in V(S)$, let $survived(s)$ denote the subset of $tops(s)$ with a child in $nodes(s)$:

$$survived(s) = \{g \mid g \in tops(s);\ \exists g' \in c(g) : g' \in nodes(s)\}$$

Note that $survived(s)$ can be viewed as the set of gene nodes that "survived" in species s. A gene node g can be in $tops(s)$ but not $survived(s)$ if g survived in the sibling species s' of s but was lost in s. Furthermore, because the gene tree includes implied speciation nodes, for each $g \in survived(s)$, there exists at most one child $g' \in nodes(s)$. As with $tops(s)$, we define an arbitrary ordering of nodes in $survived(s)$ via preorder traversal of the gene tree.

For each species node $s \in V(S)$ and each gene node $g \in survived(s)$ with child node $g' \in nodes(s)$, we define a binary variable c_{sg} that denotes whether $e(g')$ is an extra lineage at the top of species branch $e(s)$. If $e(g')$ is an extra lineage, then $c_{sg} = 1$, and otherwise $c_{sg} = 0$. Then, the number of extra lineages due to (deep) coalescence at speciation is given by

$$c = \sum_{s \in V(S)} \sum_{g \in survived(s)} c_{sg}.$$

Next, we note that, to count extra lineages, we look at the loci at the top of a species branch, that is, at the locus of g, not the locus of g', which may be different if $e(g')$ has a duplication. $e(g')$ is an extra lineage if g is mapped to the same locus as some other gene node $g'' \in survived(s)$ that precedes g. Thus, c_{sg} is constrained similarly to λ_{sg}:

$$\sum_{g'' \in survived(s): g'' \prec g} \left(1 - p_{gg''}\right) \leq |survived(s)|\, c_{sg}$$

While λ_{sg} has two constraints, one upper bound and one lower bound, c_{sg} is only constrained by lower bound. Because we are minimizing the number of extra lineages, $c_{sg} = 0$ unless there exists a constraint $c_{sg} \geq 1$. So an upper bound is not needed.

Similar to λ variables, the number of coalescence at speciation variables and constraints is each upper bounded by $2(n' - n)$.

Number of Extra Lineages due to Coalescence at Duplication For each gene node $g \in V(G)$, we define a non-negative integer variable k_g. If $e(g)$ has no duplication, then $k_g = 0$. Otherwise, k_g denotes the number of extra lineages at the time of duplication along $e(g)$. Then, the number of extra lineages due to (deep) coalescence at duplication is given by

$$k = \sum_{g \in V(G)} k_g.$$

Before defining constraints for k_g, we introduce some auxiliary sets and variables.

For each species node $s \in V(S)$, let us consider the set of gene lineages in species branch $e(s)$. Clearly, this set must include $e(g)$ for all $g \in \mathit{nodes}(s)$. However, it must also include $e(g)$ where g is a child of a bottom node $p(g) \in \mathit{bottoms}(s)$. To understand why it is important to consider this second set of nodes, we consider the case in which a bottom node $g \in \mathit{bottoms}(s)$ precedes some other node $g' \in \mathit{nodes}(s)$. Then the children branches of g might exist at the time of g' and contribute to the number of extra lineages. Formally, for each species node $s \in V(S)$, let $\mathit{bottom\_children}(s)$ denote the union of the set of children nodes of $\mathit{bottoms}(s)$:

$$\mathit{bottom\_children}(s) = \bigcup_{g \in \mathit{bottoms}(s)} c(g).$$

For each species node $s \in V(S)$, consider a pair of gene nodes $g_1 \in \mathit{nodes}(s) \cup \mathit{bottom\_children}(s)$ and $g_2 \in \mathit{nodes}(s)$ where $g_1 \neq g_2$, both g_1 and g_2 have parents, and g_1 and g_2 are incomparable. We define a binary variable $\kappa_{g_1 g_2}$ that denotes whether $e(g_2)$ has a duplication, $p(g_1)$ and $p(g_2)$ are mapped to the same locus, $p(g_1)$ precedes g_2 in time, and g_2 precedes g_1 in time. If so, then $\kappa_{g_1 g_2} = 1$, and otherwise $\kappa_{g_1 g_2} = 0$. In other words, $\kappa_{g_1 g_2} = 1$ if and only if gene lineage $e(g_1)$ exists at the time of duplication along $e(g_2)$ and the locus of lineage $e(g_1)$ at the time of duplication is equal to the mother locus of the duplication. The variable $\kappa_{g_1 g_2}$ can be expressed using a logical AND operation with four operands, represented in the following constraints:

$$\kappa_{g_1 g_2} \leq d_{g_2}$$
$$\kappa_{g_1 g_2} \leq 1 - p_{p(g_1)p(g_2)}$$
$$\kappa_{g_1 g_2} \leq o_{p(g_1)g_2}$$
$$\kappa_{g_1 g_2} \leq o_{g_2 g_1}$$
$$\kappa_{g_1 g_2} \geq d_{g_2} + (1 - p_{p(g_1)p(g_2)}) + o_{p(g_1)g_2} + o_{g_2 g_1} - 3$$

The number of κ variables is upper bounded by $P(n', 2)$, and the number of constraints is upper bounded by $5P(n', 2)$.

Finally, we define the constraints on the coalescence at duplication variables k_g using these auxiliary variables:

$$k_{g_2} = \left(\sum_{g_1 \in A} \kappa_{g_1 g_2} \right) - 1$$

where

$$A = \{ g \mid s = \mathcal{M}(g_2);\ g \in \mathit{nodes}(s) \cup \mathit{bottom\_children}(s) \}.$$

Since one coalescence at duplication variable is created for each gene branch, the number of variables and constraints is each equal to $2n' - 2$.

3.3 Objective Function

The objective is to minimize the reconciliation cost, which is repeated here for completeness:

$$\mathcal{R} = C_D \cdot d + C_L \cdot \ell + C_C \cdot c + C_K \cdot k$$

Once a solution is found, it is straightforward to convert the duplication and order variables into a locus map and order. Together with the LCA species map, these components constitute an MPR. Lastly, it is worth noting that multiple feasible solutions to the ILP may correspond to a single LCT due to equivalent orderings set by the order variables.

Correctness

The proof of correctness of the ILP formulation is straightforward. Every LCT constraint has a corresponding set of constraints in the ILP, and the objective function of the MPR problem is captured exactly in the objective function of the ILP.

Complexity

Recall that we let h denote the height of the species tree and n denote the number of leaves in the gene tree. Using the LCA map as the optimal species map, the pruned species tree has height $h' \leq h$, and the gene tree with implied speciation nodes has $n' \leq (2n - 1) + (2n - 2)(h' + 1)$ nodes, which is bounded by $O(nh)$. In the ILP formulation, the number of variables of each type is bounded by some constant factor of n', $C(n', 2)$, or $P(n', 2)$, so there exist $O((nh)^2)$ variables. The number of constraints of each type is bounded by some constant factor of n', $C(n', 2)$, $P(n', 2)$, or $P(n', 3)$, so there exist $O((nh)^3)$ constraints. However, we note that though the total number of variables and constraints grows polynomially, no known polynomial-time algorithm exists to solve ILPs.

Implementation

We implemented our ILP formulation in Python. The program uses the PuLP package [19] to setup the ILP problem then launches one of several ILP solvers. The software currently allows users to select the publicly available CBC solver from COIN-OR [14] or the more powerful commercial solver CPLEX, which is free for academic use. Both solvers offer multithreaded parallel optimizers.

4 RESULTS

To assess the impact of our ILP software, we compared the results of our previous dynamic programming (DP) approach and our new ILP approach with CPLEX 12.9 [5]. Briefly, the DP approach sets the optimal species map to be the LCA map, then exhaustively enumerates "tiles" for each species, with each tile consisting of a locus map and a partial order for nodes in the species, and finally uses dynamic programming to combine tiles so that loci of nodes shared across species match. In contrast, our ILP approach relies on an ILP

solver to efficiently search the space of feasible solutions. However, we note that the time and space complexity of both approaches depends not only on the sizes of gene and species trees but also on the level of discordance between them. For both methods, we used the default event costs of $C_D = C_L = 1$ and $C_C = C_K = 0.5$.

4.1 Simulated gene trees across 12 flies

We used the simulated *Drosophila* data set of Rasmussen and Kellis [22], which was previously used to evaluate reconciliations under the DLC model [22, 27]. Briefly, this data set modeled real data by using estimated species trees and parameters (divergence times, duplication and loss rates, effective population sizes, generation times) for a clade of 12 *Drosophila* species, then used the DLCoal model to simulate gene trees. To allow more extreme incongruence between the gene tree and species tree, the authors used a wide range of effective population sizes ($1 - 500$ million individuals) and a range of duplication-loss rates (1×, 2×, and 4× as fast as the estimated real rate). For each of the 24 parameter settings, 500 gene trees were simulated. In our experiments, we used a single core on a Linux-based Mac with a a 3.0 GHz Intel Core i5-7400 processor and 8 GB of RAM.

As both DP and ILP are exact approaches, we first verified that for every gene family, both programs return equally optimal solutions in terms of reconciliation cost.

Next, we compared the runtime of DP against ILP (Figure 2). Our ILP approach can be up to an order of magnitude slower to several orders of magnitude faster than the existing DP approach. ILP tends to require more consistent amounts of time, with a noticeable speedup over DP for gene trees simulated with high duplication-loss rates or large effective population sizes. These findings suggest that ILP may be more useful for larger, more incongruent gene trees, which are likely to become more prevalent as we sequence denser clades.

We also compared the time to setup versus solve the ILP problem (Figure 3). We found that the time to generate the variables and constraints often exceeds the time required by the solver itself. Unfortunately, these results suggest that advancements in ILP solvers may only moderately improve overall runtime.

4.2 Simulated species trees and gene trees

We are also interested in the performance of our approach for a variety of species trees. Thus, we simulated species trees and gene trees using SimPhy [17] with a speciation rate of 0.2 events/species/myr, an extinction rate of 0.18 events/species/myr, duplication and loss rates of 0.05 events/gene/myr, a generation time of 1 yr/generation, and an effective population size of 1 million. We used a range of species tree sizes (10-50 taxa) and created 50 gene tree-species tree pairs for each size. In our experiments, we used a 64-core cluster consisting of four AMD Opteron 6276 CPUs, each with 16 cores at 2.3 GHz, and a total of 512 GB of DDR3-1600 RAM. We restricted each run to 2 days and 20 GB of RAM. For ILP, we also took advantage of CPLEX's parallel optimizer and used 8 threads per run.

As with the flies dataset, when comparing total runtime (Figure 4), neither the ILP nor DP approach is consistently faster. However, in contrast to the flies dataset, our analysis of time to setup and solve the ILP problem (Figure 5) suggests that more efficient ILP solvers can substantially decrease overall runtime. We believe that increased heights of the species trees result in larger ILP problems that are more difficult for solvers, an effect not seen in the smaller fly species tree.

5 DISCUSSION

In this work, we have presented an ILP formulation and corresponding tool for solving the NP-hard maximum parsimony reconciliation problem under the duplication-loss-coalescence model. Our ILP formulation provides an exact solution, with the total number of variables and constraints polynomial in the size of the gene tree and the height of the species tree. Our analysis of simulated data sets shows that ILP yields reconciliations with cost equal to the existing DP algorithm. By relying on ILP solvers, we do not need to exhaustively search the space of reconciliations as DP does, making our ILP formulation more scalable to large data sets with high gene tree-species tree incongruence.

Our ILP formulation has a further advantage over DP in that it allows users to limit the maximum runtime of the ILP solver. Previously, the only alternative to an exhaustive search relied on heuristics such as limiting the number of duplications and losses per species or searching through hill climbing [27]. These heuristics both limited the accuracy of resulting reconciliations, and in the latter case, it could be unclear how many steps were necessary to achieve a reasonable reconciliation. In contrast, despite being NP-hard, ILP problems benefit from solvers that can yield approximate solutions, though more research is needed to characterize the trade-off between accuracy and scalability. In this vein, future work might also investigate the effect of linear programming relaxations or heuristic methods on resulting ILP solutions.

However, we note that our ILP formulation is not without its disadvantages. For example, there can exist multiple optimal reconciliations between a gene tree and species tree for a fixed assignment of event costs [7]. Because the number of solutions tends to increase exponentially with gene tree size, enumeration is infeasible, but the DP approach can count the number of solutions and return one sampled uniformly at random. In contrast, because ILP variables over-specify a reconciliation (in that multiple settings of the variables could yield the same LCT), it is unclear how one could count or sample solutions using ILP.

Our ILP implementation currently relies on Python's PuLP package to setup the ILP problem, which has the advantage of allowing multiple solvers but at the cost of added overhead. An alternative approach that would likely speed up the setup would rely on the Python or C++ interfaces provided by CPLEX. Furthermore, some variables must take on binary or integer values because they are constrained by other variables. Relaxing integrality constraints for such variables may speed up the solver.

We envision our ILP formulation as one step towards more complex analyses of gene family evolution. For example, DLCpar is used to reconcile gene trees and species trees as part of the OrthoFinder method [8, 9] for inferring orthologs. Because the full DP search is too time-intensive, OrthoFinder uses DLCpar with a heuristic search instead. By providing a more thorough search and allowing users to bound the runtime, our ILP formulation already has several

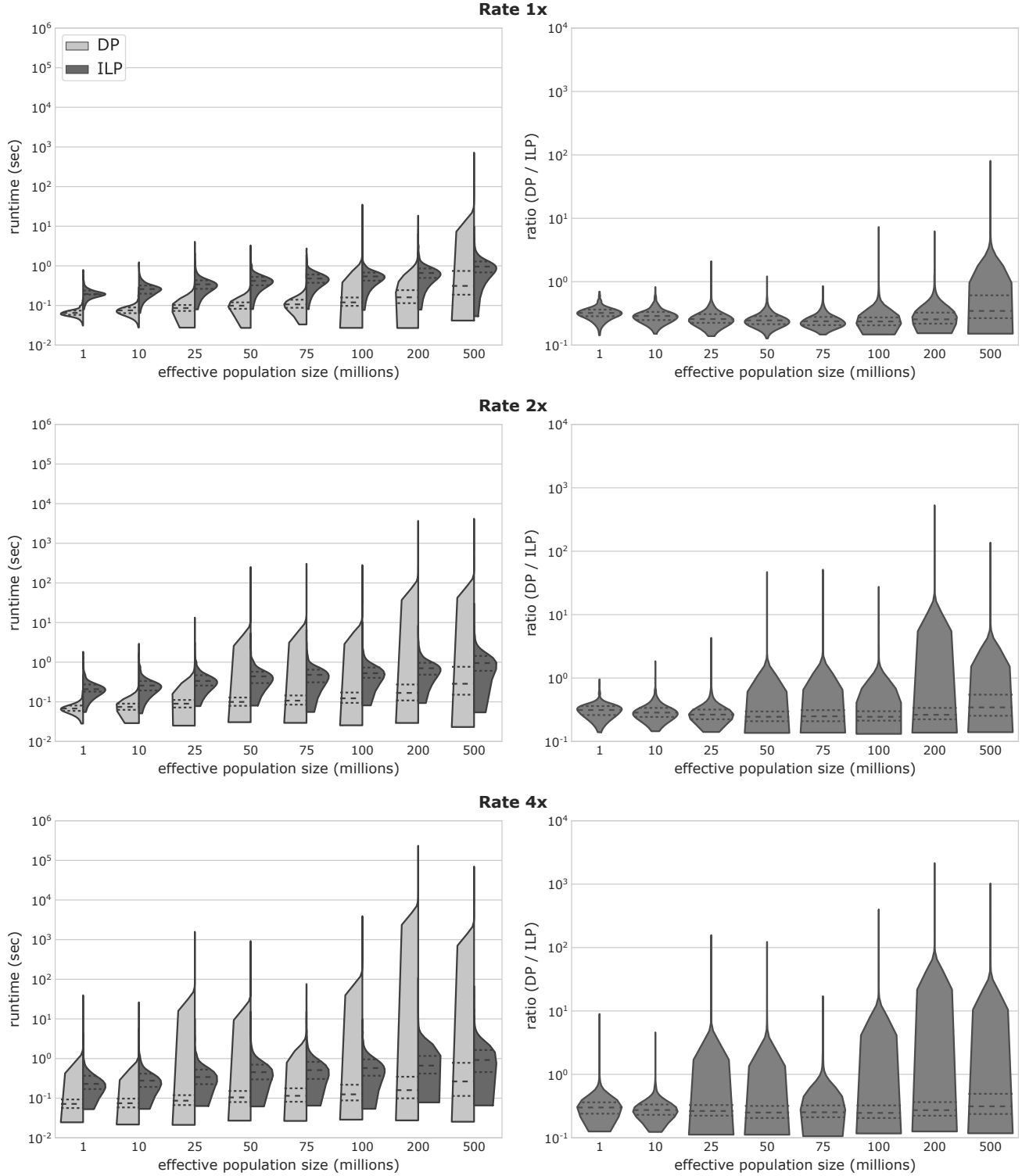

Figure 2: Runtimes of DP and ILP approaches on simulated fly gene trees. Gene trees were simulated using duplication and loss rates the same as (1×), twice (2×), and four times (4×) the rate estimated in real data and a wide range of effective population sizes, with 500 gene trees simulated per parameter setting. We reconciled the gene trees using both our existing dynamic programming (DP) approach and our new integer linear programming (ILP) approach. Distributions and quartiles are shown.

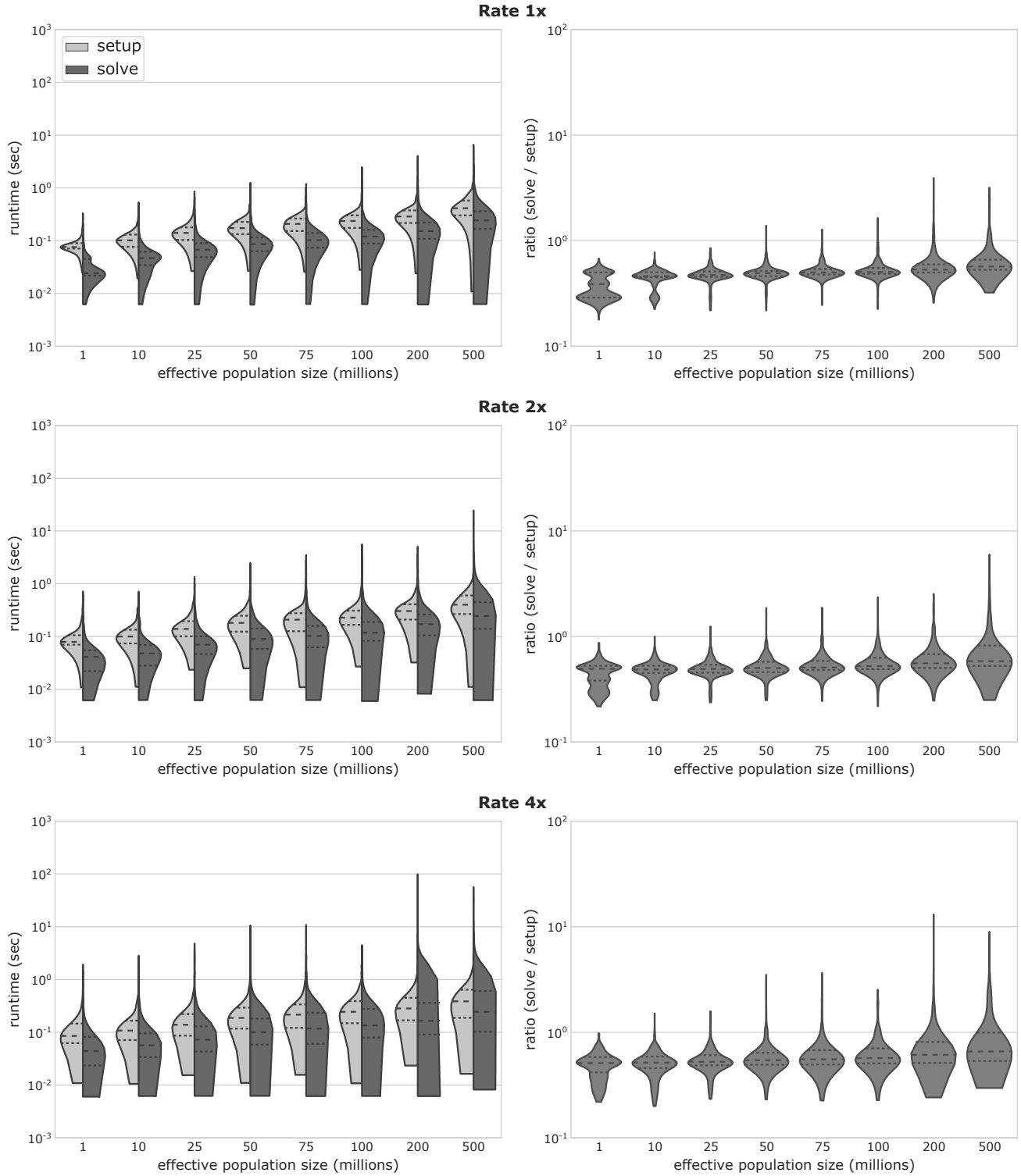

Figure 3: Runtimes to setup and solve the ILP formulation on simulated fly gene trees. See Figure 2 for a description of the dataset.

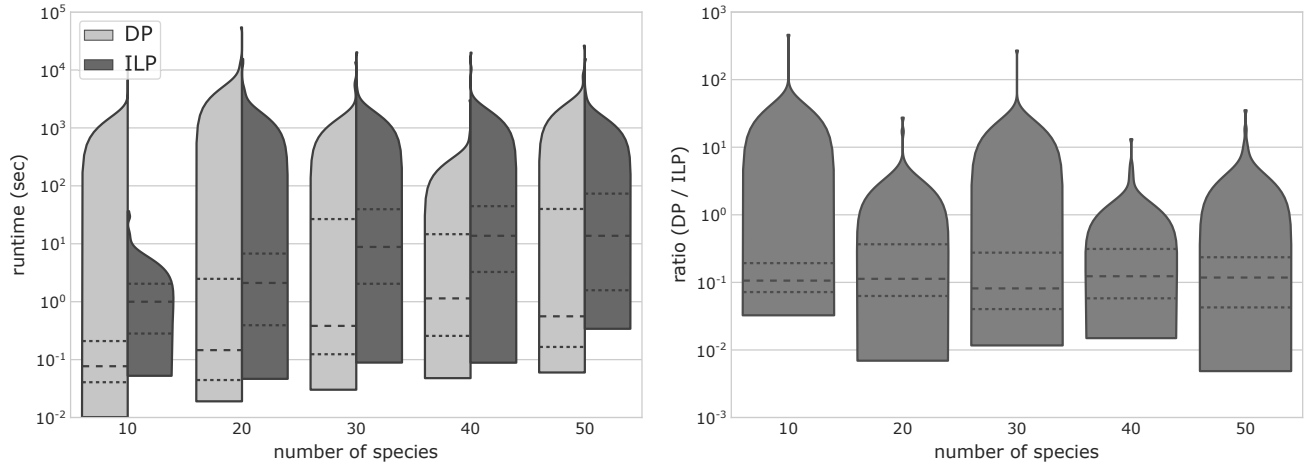

Figure 4: Runtimes of DP and ILP approaches on simulated species trees and gene trees. Species trees and gene trees were simulated using SimPhy [17] using a range of species tree sizes, with 50 gene tree-species tree pairs for each size. Distributions and quartiles are shown.

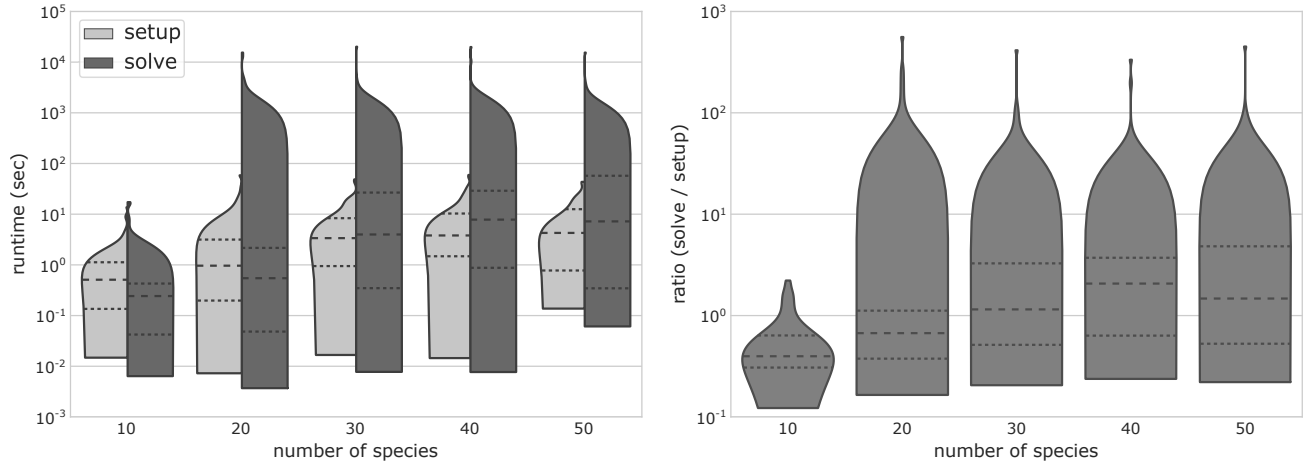

Figure 5: Runtimes to setup and solve the ILP formulation on simulated species trees and gene trees. See Figure 4 for a description of the dataset.

advantages over the heuristic method and may therefore be a better alternative.

Lastly, though the DLC model is more complex than the standard duplication-loss model and multispecies coalescent model, it still fails to account for several evolutionary processes, including species hybridization, horizontal gene transfer, copy number hemiplasy, and genetic linkage. Inferring reconciliations under such model extensions may be difficult if not impossible using the DP algorithm. However, by allowing us to focus on encoding constraints rather than solving for solutions under the constraints, our ILP approach opens up several avenues for further research.

ACKNOWLEDGMENTS

The authors thank Eliot Bush for use of his cluster for preliminary experiments, Susan Martonosi for helpful discussions on ILP formulations and feedback on this manuscript, and Ran Libeskind-Hadas for helpful discussions on reconciliations and feedback on this manuscript.

This work was supported by the Department of Computer Science and the Dean of Faculty of Harvey Mudd College. This material is based upon work supported by the National Science Foundation under Grant No. IIS-1751399 to YW.

REFERENCES

[1] Daniel Bork, Ricson Cheng, Jincheng Wang, Jean Sung, and Ran Libeskind-Hadas. 2017. On the computational complexity of the maximum parsimony reconciliation problem in the duplication-loss-coalescence model. *Algorithm Mol Biol* 12, 6 (2017). https://doi.org/10.1186/s13015-017-0098-8

[2] Bastien Boussau and Celine Scornavacca. 2020. *Phylogenetics in the Genomic Era.* No commercial publisher | Authors open access book, Chapter Reconciling Gene trees with Species Trees, 3.2:1–3.2:23.

[3] Yao-ban Chan, Vincent Ranwez, and Céline Scornavacca. 2017. Inferring incomplete lineage sorting, duplications, transfers and losses with reconciliations. *J*

Theor Biol 432 (2017), 1–13. http://www.sciencedirect.com/science/article/pii/S0022519317303740

[4] Wen-Chieh Chang, Gordon J. Burleigh, David F. Fernández-Baca, and Oliver Eulenstein. 2011. An ILP solution for the gene duplication problem. *BMC Bioinf* 12, 1 (Feb. 2011), S14. https://doi.org/10.1186/1471-2105-12-S1-S14

[5] International Business Machines Corporation. 2019. *IBM ILOG CPLEX Optimization Studio, CPLEX User's Manual, Version 12 Release 9.*

[6] Etienne G. J. Danchin. 2016. Lateral gene transfer in eukaryotes: tip of the iceberg or of the ice cube? *BMC Biology* 14 (Nov. 18 2016). https://doi.org/10.1186/s12915-016-0330-x

[7] Haoxing Du, Yi Sheng Ong, Marina Knittel, Ross Mawhorter, Nuo Liu, Gianluca Gross, Reiko Tojo, Ran Libeskind-Hadas, and Yi-Chieh Wu. 2019. Multiple Optimal Reconciliations under the Duplication-Loss-Coalescence Model. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* (2019), 1–1. https://doi.org/10.1109/TCBB.2019.2922337

[8] David M. Emms and Steven Kelly. 2015. OrthoFinder: solving fundamental biases in whole genome comparisons dramatically improves orthogroup inference accuracy. *Genome Biology* 16, 1 (Aug. 2015), 157. https://doi.org/10.1186/s13059-015-0721-2

[9] David M. Emms and Steven Kelly. 2019. OrthoFinder: phylogenetic orthology inference for comparative genomics. *Genome Biology* 20, 1 (Nov. 2019), 238. https://doi.org/10.1186/s13059-019-1832-y

[10] Morris Goodman, John Czelusniak, G. William Moore, A.E. Romero-Herrera, and Genji Matsuda. 1979. Fitting the Gene Lineage into its Species Lineage, a Parsimony Strategy Illustrated by Cladograms Constructed from Globin Sequences. *Syst Zool* 28, 2 (1979), 132–163.

[11] Lei Li and Mukul S. Bansal. 2018. An Integer Linear Programming Solution for the Domain-Gene-Species Reconciliation Problem. In *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*. Association for Computing Machinery, Washington, DC, USA, 386âĂŞ397. https://doi.org/10.1145/3233547.3233603

[12] L. Li and M. S. Bansal. 2019. An Integrated Reconciliation Framework for Domain, Gene, and Species Level Evolution. *IEEE/ACM Trans Comput Biol Bioinform* 16, 1 (2019), 63–76.

[13] R Libeskind-Hadas and M Charleston. 2009. On the Computational Complexity of the Reticulate Cophylogeny Reconstruction Problem. *J Comput Biol* 16 (2009), 105–117. https://doi.org/10.1089/cmb.2008.0084

[14] R. Lougee-Heimer. 2003. The Common Optimization INterface for Operations Research: Promoting open-source software in the operations research community. *IBM Journal of Research and Development* 47, 1 (Jan. 2003), 57–66.

[15] Bin Ma, Ming Li, and Louxin Zhang. 2000. From Gene Trees to Species Trees. *SIAM J Comput* 30 (Aug. 2000), 729–752.

[16] Wayne P. Maddison. 1997. Gene Trees in Species Trees. *Syst Biol* 46, 3 (Sept. 1997), 523–536.

[17] Diego Mallo, Leonardo De Oliveira Martins, and David Posada. 2016. SimPhy: Phylogenomic Simulation of Gene, Locus, and Species Trees. *Syst Biol* 65, 2 (March 2016), 334–344. https://doi.org/10.1093/sysbio/syv082

[18] Ross Mawhorter, Nuo Liu, Ran Libeskind-Hadas, and Yi-Chieh Wu. 2019. Inferring Pareto-Optimal Reconciliations across Multiple Event Costs under the Duplication-Loss-Coalescence Model. *BMC Bioinformatics* 20, 20 (Dec. 2019), 639. https://doi.org/10.1186/s12859-019-3206-6

[19] Stuart Mitchell, Michael O'Sullivan, and Iain Dunning. 2011. PuLP: A Linear Programming Toolkit for Python. (Sept. 5 2011). http://www.optimization-online.org/DB_FILE/2011/09/3178.pdf

[20] Y. Ovadia, D. Fielder, C. Conow, and R. Libeskind-Hadas. 2011. The Cophylogeny Reconstruction Problem Is NP-Complete. *J Comput Biol* 18, 1 (2011), 59–65.

[21] Roderic D.M. Page. 1994. Maps Between Trees and Cladistic Analysis of Historical Associations among Genes,Organisms, and Areas. *Syst Biol* 43, 1 (March 1994), 58–77. https://doi.org/10.1093/sysbio/43.1.58

[22] Matthew D. Rasmussen and Manolis Kellis. 2012. Unified modeling of gene duplication, loss, and coalescence using a locus tree. *Genome Res* 22 (2012), 755–765. https://doi.org/10.1101/gr.123901.111

[23] Maureen Stolzer, Han Lai, Minli Xu, Deepa Sathaye, Benjamin Vernot, and Dannie Durand. 2012. Inferring duplications, losses, transfers and incomplete lineage sorting with nonbinary species trees. *Bioinformatics* 28, 18 (2012), 409–415.

[24] Ali Tofigh, Michael Hallett, and Jens Lagergren. 2011. Simultaneous Identification of Duplications and Lateral Gene Transfers. *IEEE/ACM Trans Comput Biol Bioinform* 8, 2 (March 2011), 517–535. https://doi.org/10.1109/TCBB.2010.14

[25] Benjamin Vernot, Maureen Stolzer, Aiton Goldman, and Dannie Durand. 2008. Reconciliation with Non-Binary Species Trees. *J Comput Biol* 15, 8 (Sept. 2008), 981–1006. https://doi.org/10.1089/cmb.2008.0092

[26] N. Wieseke, T. Hartmann, M. Bernt, and M. Middendorf. 2015. Cophylogenetic Reconciliation with ILP. *IEEE/ACM Trans Comput Biol Bioinform* 12, 6 (Nov. 2015), 1227–1235.

[27] Yi-Chieh Wu, Matthew D. Rasmussen, Mukul S. Bansal, and Manolis Kellis. 2014. Most Parsimonious Reconciliation in the Presence of Gene Duplication, Loss, and Deep Coalescence using Labeled Coalescent Trees. *Genome Research* 24, 3 (March 2014), 475–486. https://doi.org/10.1101/gr.161968.113